Road Guard System

Mai Hamed Salim AL-Kalbani

2840080

Dr. Shermina Jeba

April 2023

Dissertation Outline

Computing Science and Mathematics
University of Stirling

Abstract

Problem: Over the past few years, a steady increase in the number of traffic accident victims caused by stray animals has been observed across Oman, where the rate of accidents involving stray animals is 41 deaths annually [1]. These stray animals move from their places of residence to the cities in search of food, and often ending in brutal traffic accidents. The Royal Oman Police is still imposing penalties and harshly worded statements for the owners of these animals; however, these animals still gather in cities, especially the roads [2].

Objectives: The proposed project aims to solve the problem of accidents caused by animals through computer vision data analysis and data mining. This project aims to develop an application that controls a CCTV camera and an alert panel for drivers. The camera can detect and identify animals crossing the road, through which the alert panel lights up for drivers to alert them to the presence of a large animal on the road. For this to be possible it would require:

1- An accurate algorithm to be trained to recognize animals.
2- An application to connect the camera to capture and process live images.
3- A simulator to alert drivers that receives alerts from the application after the animal is detected. The dataset will then be segmented to make predictions to determine road accidents caused by these animals.

This application makes easier for the authorities to monitor animals that crossroads and cause accidents, till the authorities after that can enact laws to control speed in some roads in which animals frequently pass. This application was created to serve the community to avoid fatal accidents caused by these animals.

Methodology: Create an application to reduce the number of traffic accidents caused by stray animals that crossroads in Oman. The proposed method uses the YOLO model to detect objects, the model is set to detect main three types of animals that cross the roads (camels, goats, donkeys). Once detected, an alert is notified to drivers and authorities. A design of the proposed solution was made, in which we will use an integrated system directly connected to the CCTV camera with driver's sign. Application was created using Google Colab, NetBeans, PyCharm, Jupyter. And two languages: Python, Java.

Achievements: The project was successful, weight and CFG files were able to detect animals. The logistic regression model is also able to predict accidents with excellent accuracy. All mentioned objectives have been successfully completed. The main objectives that have been identified:

1- Creating an algorithm to identify animals.
2- Create an application that connects both the CCTV camera and the driver alert panel.
3- Notify Admin and drivers.
4- Create data analysis.
5- Create a model that can predict accidents.

Attestation

I understand the nature of plagiarism, and I am aware of the University's academic integrity policy. I certify that this dissertation reports original work by me during my University project except for the following.

Signature:  Date: 19/04/2023

Table of Contents

Abstract i

Attestation ii

Acknowledgements iii

Table of Contents iv

List of Figures v

1 Introduction 11

1.1 Background and Context 12

1.2 Scope and Objectives 12

1.3 Achievements 13

1.4 Overview of Dissertation 13

2 State of The Art 14

3 Methodology 19

3.1 Agile Methodology 19

3.2 Requirements Analysis 20

3.2.1 Requirement capture 20

3.2.2 Use Case Diagram 21

3.2.3 Problem analysis 21

3.2.3.1 CCTV Camera 22

3.2.3.2 Animals Detection 22

3.2.3.3 The warning signs 23

3.2.3.4 Warning the authorities 24

3.2.3.5 Object Detection 24

3.3 Design 27

3.3.1 System Architecture 27

3.3.2 Process flow diagram 28

3.3.3 User interface Design 28

3.3.3.1 User interface Wireframes 29

3.3.3.2 Main interface Wireframes 30

3.3.3.3 Choose video 30

3.3.3.4 Choose Detect File 31

3.3.3.5 Data Analysis 31

3.3.3.6 API Button 31

3.4 Development Platform and Tools 32

3.4.1 Software 32

3.4.2 Hardware 34
3.5 System Design 35
4 Implementation 36
4.1 IDE's and Languages 36
4.2 Deep Neural Networks 36
4.2.1 OpenCV 36
4.3 Implementation of Core Components 36
4.3.1 Model training 36
4.3.2 Model Result 39
4.3.3 Model Precision 41
4.3.4 Precision Results 43
4.3.5 Implementing Animals Recognition 45
4.3.6 Camera & signs Connection and Live Feed 47
4.3.7 Implementing Model to prediction accidents 48
4.4 Implementing the UI 53
4.4.1 The Main GUI 53
4.4.2 Light up for driver & Receive notification 59
4.5 Project scenario 60
4.6 Testing and Result Discussion 61
4.6.1 Personal Testing 61
4.6.2 Professional Testing 64
5 Conclusion 65
5.1 Summary 65
5.2 Critical Evaluation 65
5.3 Limitations and Future Work 68
6 References 69

List of Figures

Figure 1. Number of accidents in Oman 6
Figure 2. YOLO NCNN Mobile Application Detection 9
Figure 3. Animals Detection System 10
Figure 4. Animals GUI Detection 10
Figure 5. Animal Intrusion Detection system 11
Figure 6. Detection animals in the backyard 12
Figure 7. The architecture of the TraCon System 12
Figure 8. Agile Diagram 13
Figure 9. Use case diagram 15
Figure 10. Road Guard System 16
Figure 11. Distance Calculation of the Detected Animals 17
Figure 12. Residual blocks 18
Figure 13. Bounding Box of object 18
Figure 14. IoU 19
Figure 15. IOU Equation 20
Figure 16. RetinaNet architecture 20
Figure 17. Road Guard System Architecture 21
Figure 18. Flowchart 22
Figure 19. Road Guard user interface 23
Figure 20. Main page Wireframe 24
Figure 21. Main page Wireframe 24
Figure 22. Detect page Wireframe 24
Figure 23. Data analysis Wireframe 25
Figure 24. API Wireframe 25
Figure 25. The vector recognizing optical features 26
Figure 26. The hierarchy of Java Swing Components 27
Figure 28. Labelling the object 36
Figure 29. Text file of object position 36
Figure 30. Create labels file using python script 36
Figure 31. Start the model training 37

Figure 32. Weights files created 37
Figure 33. Graph of training loss representation in Yolov4 38
Figure 34. Number of Iteration in Yolov4 35
Figure 35. Schematic of IoU loss 39
Figure 36. Accuracy of mAP and weight file 40
Figure 37. YOLOv4 architecture 41
Figure 38. Yolov4 layers of the darknet 41
Figure 39. Pre class sigmoid 42
Figure 40. Predicted and Ground truth bounding box 43
Figure 41. AP calculation 44
Figure 42. mAP calculation 44
Figure 43. Comparative precision results 'Yolov4' and 'Yolov4-tiny' 44
Figure 44. Test model 45
Figure 45. Results of Yolov4 with Yolov4-tiny 45
Figure 46. FP and TP detections 46
Figure 47. Assembly scheme of the embedded system 47
Figure 48. The ultimate Road Guard virtualization 47
Figure 49. Data Samples 48
Figure 50. ROC graph in Logistic Regression 49
Figure 51. ROC code 51
Figure 52. Calculate AUC, CA Logistic Regression 51
Figure 53. Best decision Logistic Regression 52
Figure 54. Confusion matrix for Logistic Regression 52
Figure 55. Graph in Logistic Regression 53
Figure 56. Download pickle prediction file 53
Figure 57. Java and Python Road Guard Controller 54
Figure 58. Input Button Java & Python 54
Figure 59. Console screen after detected animals in Java 55
Figure 60. Detect Button in Java & Python 55
Figure 61. API connect with controller using IP 56
Figure 62. Create communication between controller and Driver's in Java 57
Figure 63. Create communication between controller and Driver's in Python 57
Figure 64. Camel code number in localhost using Java 57
Figure 65. Camel code number in localhost using Python 57
Figure 66. Enter wrong IP and Wrong port 58
Figure 67. Bring Data animals from Database 58

Figure 68. Connect to MongoDB in Java and Python 59
Figure 69. Connect successfully to DB after detected animals in Java..... 59
Figure 70. Animals Data displayed MongoDB 60
Figure 71. Connect Driver sign board with Road Guard controller..... 60
Figure 72. Connect Driver sign with controller using IP, Port number in Python 61
Figure 73. Request and Response between Python controller & Java driver sign..... 61
Figure 74. Issue to connect OpenCV in Java..... 66
Figure 75. Road Guard controller in Java and Python..... 67
Figure 76. Data Analysis python file in java..... 68
Figure 77. Output detected Python Vs Java 68

List of Tables

Table 1. Number of traffic accidents because of animals in Oman 11

Table 2. Estimated cost 35

Table 3. Class integer number 37

Table 4. YOLOv4 Model calculation 39

Table 5. Animals code 56

Table 6. variables supplied 49

Table 7. Class integer number 63

Table 8. Server Model Test 39

Table 9. Receiving warning controller Test 64

Table 10. Sending warning from driver board Test 64

Table 11. Store Animals Info in DB Test 65

Table 12. Display Data from DB to Dashboard Test 65

Table 13. Difference between Java and Python 67

Acknowledgements

I would like to thank my supervisor, Dr. Sharmina Jeba, who was patient and made many valuable suggestions while writing the dissertation, as she took the time to help me make this report more readable than it was. And I would like to thank Mr. Talal and his team to take time to review this project.

1. Introduction

Recently, great progress has been observed in the field of technology, as it directly affected various fields, these areas include artificial intelligence, the Internet of Things, and quantum computing. These technologies have the potential to revolutionize the industry and solve problems that are considered obstacles. Artificial intelligence techniques have developed significantly over the past years, with the ability to add equipment in these technologies. One of these technologies is the object detection technology, which has witnessed a great development during the recent period as it has exceeded human capabilities in accuracy, and this technology is still in constant progress, thanks to advances in neural networks and machine learning. In a simplified and accurate manner, neural networks are known as simplified models of the developed human brain, where neural networks can train using a huge amount of data to identify the elements with high accuracy and speed. Different and useful practical applications can be developed, but this project will focus on the possibility of reducing the problem of traffic accidents caused by large animals crossing the roads of the Sultanate of Oman. In addition, to other problems as destruction of public and private properties such as farms, parks.

According to reports gave by the National Center for Statistics and Information, the number of traffic accidents has increased significantly in the past years due to the presence of animals on the road specifically (camels, donkeys, sheep). Animal collision with vehicles is the second cause of death due to accidents in Oman [*https://www.ncsi.gov.om/Pages/AllIndicators.aspx*].

Year	Number of accidents	Human		Type of animals		
		death	injury	Camel	Goats	Others…
2017	3845	466	341	92	18	70
2018	2802	155	684	131	21	26
2019	2120	420	792	127	65	37
2020	1341	371	1365	115	23	51
2021	3000	440	1621	157	94	66

Figure 1. Number of accidents in Oman

When starting to develop this system to identify large animals that cross the roads, we must focus on the principle of safety and accuracy of model. We need to take into consideration the distance for warning signs before the driver reaches the area where a large animal is detected so that the driver can avoid the accident. Also, when we want to detect large animals that may cause damage to public property, animals may not always in visible from the camera. As well as challenges associated with identifying animals such as the timing of the animal's passage, whether it is in the morning or evening, which affects accuracy. This project aims to create a system linked to a camera capable of identifying large animals that contribute to horrific accidents, or causing damage to public property, as well as warning panels that light up when these animals pass to alert drivers.

1.1 Background and Context

Loose animals are one of the main problems that cause accidents in Oman. Lives can be saved using CCTV cameras connected to the application to detect animals and inform the people and the authorities.

Taking things from the beginning, the worst accident caused by stray animals occurred in Al Buraimi, nine people died after the bus driver tried to avoid hitting a herd of camels. In 2016, two police officer died when a police car collided with a wild camel in AL Sharquaa Governorate. Less than 6 years ago an Indian schoolchild and a delivery man died after the vehicle collided with the wild camel. Just two months after this accident, another car collided with a wild camel in which a 15-year-old Omani student died. And accidents occurred after this year, but the most prominent of them was a bus of girl's students coming from Muscat to Ibri, in which students died because of the bus colliding with a camel stray. "Despite the presence of a fence on some main roads, it does not perform its intended purpose, either because of poor implementation or because it has become worn out due to natural factors, or the presence of some gaps and openings, which facilitates the crossing of these camels and their entry into the width of the public streets"[3] .

Drivers on public streets become afraid when crossing these areas, because animals may be suddenly present on the roads, which leads to brutal accidents, for this reason, the project will be extremely useful to help drivers avoid these accidents. And notify the authority like municipality of the presence of these animals in these areas.

The main use of this system is for the Ministry of Municipality or ROP, drivers, and insurance companies. The camera will start to detect large objects that cross the roads, the camera will stream feed to the system and then will send alert to the authorities and drivers sign.

1.2 Scope and Objectives

The main objective of the project is to detect the animals on the road, then sends an alert to the admin and drivers sign. In addition to the ability of the program to create data analysis for animals that were discovered on the road.

Through the previous discussion, the project inspired by the idea of animal detection from the signs of animals on the roads to avoid accidents. Animals' detection systems split into three main types: road-based, vehicle-based, and animal-based systems. Through these systems, it found that they significantly reduce the number of accidents, but there are defects on systems, the disadvantages of animal-based systems are each animal needs one device, which means an increase in cost. This device also needs to repair each two months, and this expensive and tiring. In this project an effective and less costly program will be developed.

Scope

The project will focus on the areas of the Oman where the spread of animals on the roads has been observed, such as: Muscat, AL-Buraimi, Ibri, AL-Batinah, AL-Sharqiyah, Dhofar, AL-Wistaa, Musandam.

Objectives

The main objective of this project is to establish a system that identifies the loose animals crossing the road or entering public and private property in Oman and informing authorities, warning people and data analysis of detected animals. This system will link to a high-resolution camera, which will identify the object and then an alert will send to authorities and drivers sign. Here are the main objectives of the project:

1- To develop road guard controller application.

2- To detect the animals a currently.

3- To develop driver warning to inform ROP.

4- To analysis the data of animals crossing.

5- To develop an AI model that can predict accidents caused by animals.

1.3 Achievements

The desired goal of this project has been achieved; through this thesis we developed a better understanding of machine learning with a special focus on the YOLO algorithm. Furthermore, we learned about the Python programming language, data mining and predicting. Based on this knowledge, we developed the Road Guard application. This application can connect to the camera and the warning board then send notifications to the authorities when an animal is detected on the road or near the road. It will also light a road sign on the street to alert drivers. Addition, it presents a general analysis of the animals that were discovered as well as predicting the accident. Although the general function of the project has been completed, there is room for adding more improvements and developments and implementing some new functions.

1.4 Overview of Dissertation

This dissertation consists of 5 chapters, each chapter explains a specific aspect of the project, these chapters are organized as follows:

Chapter 2- Literature review: This chapter looks at similar projects to provide some overall analysis and evaluation of these projects and to see areas for improvement.

Chapter 3- Methodology: This chapter addresses the methodology of the project different problems and challenges that the project is likely to face, captures problems with animal identification and problems with the project in general. Moreover, shows more technical details of Road Guard System and tests section shows that the application is working properly, including tests (personal testing, such as testing the user interface and coding, and external testing, which represents the opinions of stakeholders).

Chapter 5- Conclusion: This chapter summarizes the achievements that have been achieved from the project with mentioning the critical and practical evaluations of the project. And also, with the developments that can be developed in the future if the project is to continue.

2. State-of-The-Art

The idea of creating a system that monitors the roads is one of the modern areas, through which this system can monitor animals that threaten the lives of drivers, so there are not many projects that clearly speak of these ideas. In this section, it will be discussed and analysed with the available examples through which will highlight the advantages and disadvantages that can improved in this project. After completing a critical analysis of these examples of projects, algorithms will be reviewed to give a conclusion of my choices.

a. YOLO NCNN Mobile Application Detection

The application of discovering wild camels designed to discover all kinds of wild camels that may cross the road. The app is working fine but it has false detections, so the app needs more database for the string element. The feature of adding images via photo studio worked perfectly, the application was able to detect sentences by displaying random images. The application requires sufficient space in the phone before use.

The user interface was straightforward and easy to use, with displays of image details such as size, detection time and more. As for the live photo mode, it was too distracting for the live camera at the bottom and centre of the screen as we could not focus on a particular point. There is a button at the top left of the screen called (Detect), which allows you to add an image via the studio, which selects the object directly.

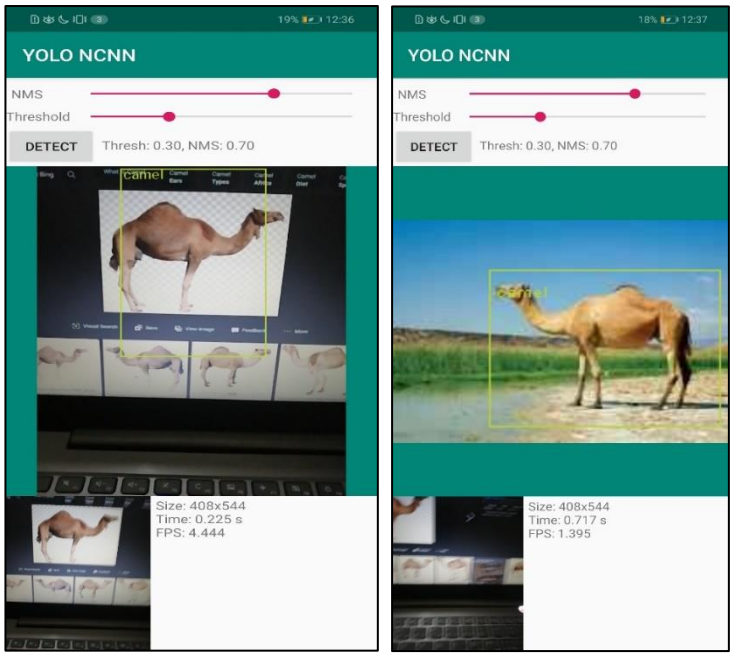


Figure 2. YOLO NCNN Mobile Application detection

Conclusion, after trying the application, the strengths of the user interface were that it was easy to use and direct, the only thing that should be added is to separate the two screens (the two cameras) and make them in one screen only so that the user can focus accurately. Camel's tracking worked well but needs to add more database to improve it. Regarding the project, proposed solutions will add by reducing the error rate as an increase in the database and dispensing with the mobile application with a GUI.

b. Automated animal detection System and Prevention

The animal detection application works with the Raspberry Pi camera, the device works by ultrasound and IR sensor, when detecting the animal, which works well, as the user can detect the animal and receive an emergency call directly on the phone. It noted that the detection of animals is general and not limited, as the user cannot find the type of animal, whether it is large or not. On the other hand, an increase in the percentage of false detection is seen when there are fierce winds and vehicles next to the sensor [4].

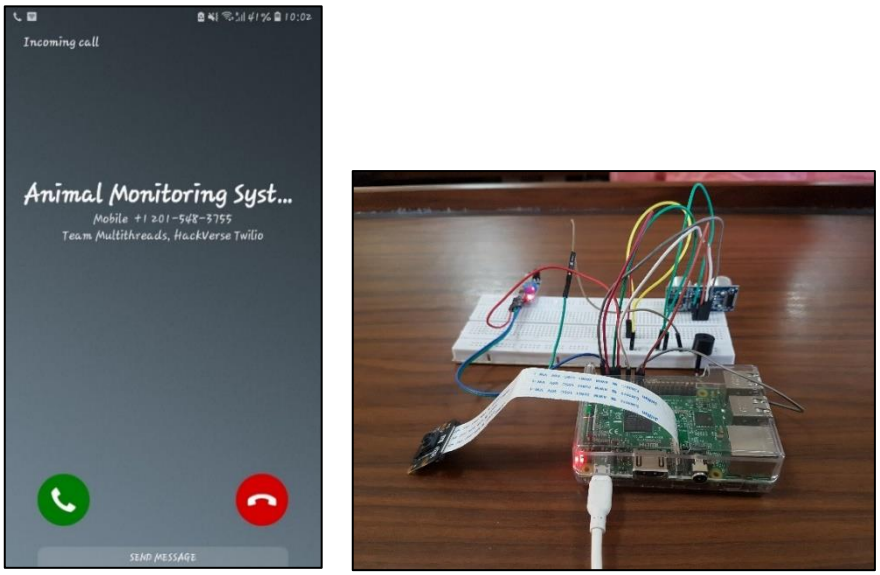


Figure 3. Animals' detection system

Conclusion: Although the system works very well, but it needs modifications that may improve the quality and effectiveness of the system, the system needs to replace the infrared sensor because it increases the false detection rate, the recommendation will be to use the raspberry camera only. There is another problem related to connecting the system to the mobile phone, as it is possible that the region does not have a good network to receive the call, so we see it as a not a promising idea, and the proposed alternative is to use warning panels. We liked the system as it was also able to send notifications to the user if there was no animal around. For the project's improvements, the goal will be to use big warning panels instead of a mobile phone.

c. Automated Forest Surveillance Safety

Unlike previous apps, this Python-developed app linked to a Raspberry Pi Camera with a GUI screen that displays data analysis for animals. The system can store video, identify, and track animals in the forest, send pictures by e-mail with alerts via mobile phone in emergency situations, control through the Windows application and identify animals detected by the camera. The GUI is simple and understandable, but at the same time the GUI crowded, which considered a bad use of the space utilization of the GUI. It noticed that when the camera not connected to the system, it produces error messages without clarifying the concept of the error, the programmer made error dialog message without writing the type of the error. For the project's improvements, the goal will be distribution of the screen option in GUI screens connected to one GUI to avoid cluttering of options [5].

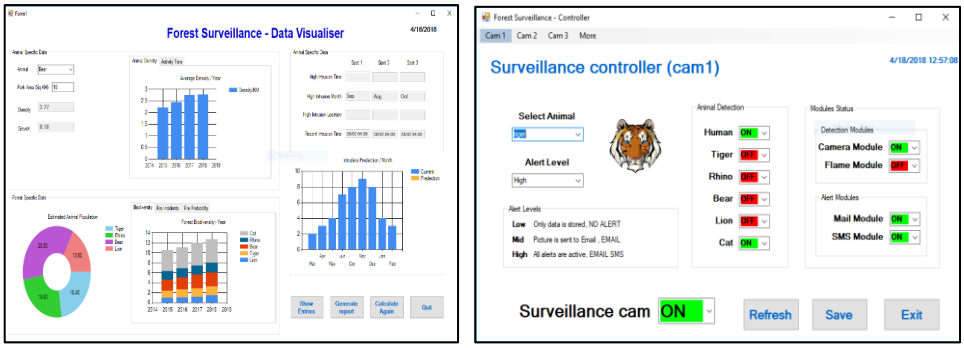


Figure 4. Animals' GUI detection

Conclusion after thoroughly exploring the application, however, it needs some extensive modifications in the design of the user interface, and avoiding colours that can cause some confusion, as well as moving away from adding all the features in one screen only. Regarding the project, separate windows will be added from Main window and stay away from distractions such as loud colours.

d. Farm Animal Tracking Project

The Animal Detection System is a model of deep learning developed on YOLO; this application runs on the computer powered by a webcam. This application focuses on discovering animals only, the app works by asking the user to open webcam. After the experiment, it noticed that the system can quickly identify animals, as the detection error rate is extremely low, and the reason is that the database is large, which enables the system to accurately detect animals. Also, the application works on the Yolo algorithm, which is characterized by better detection accuracy and speed. The system was not practical as it took a long time to figure out how to use it. For the project's improvements, the goal will be to using a large database to avoid wrong detection, as well as the program will be connected to the Raspberry pi camera instead of webcam [6].

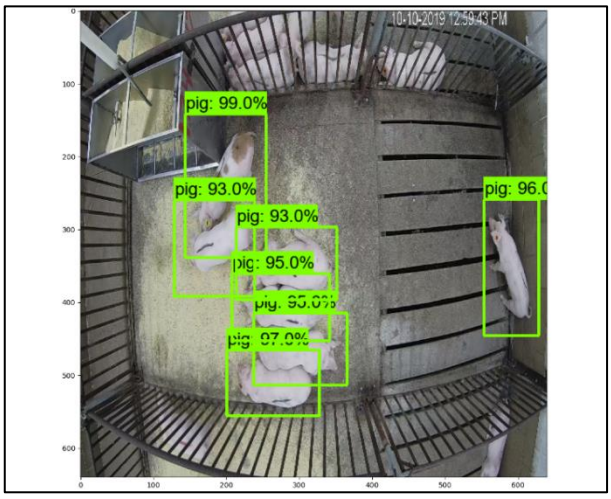


Figure 5. Animals' Intrusion detection

Conclusion although the main objective of the program achieved the expected requirement; however, the program is not practical, for the project will be added to the large dataset as possible to avoid wrong detection.

e. Detecting animals in the backyard

The animal detection system works by the camera at night. The system requires a connection to the Telegram app in addition to a camera linked to the program in order to detect and capture video from the camera to the Telegram app. When the camera detects any animal, it will send a video recording of this animal through the Telegram program. Through experience, it seems that the program works well, as it was able to identify with the same accuracy after several attemps of using it. For the project's improvements, the goal will be to connect the road guard system for notification messages to the Ministry of municipality [7].

Figure 6. Detection animals in the backyard

Conclusion, after trying the application, it was noticed that the application works perfectly, in addition to the presence of an excellent feature, which is to notify the user with a video clip. So, in the project this feature will be added by alert the authorities in controller instead text massages.

f. Closed-loop feedback optogenetic system.

The software detects the object using the object detection module, which streams the images through the webcam to the computer, which send feedback control signals to the optogenetic devices through Arduino Microcontroller board. Through this system, the project can verify the behavior of the organism in genetics, for example the system can:open field test, elevated plus maze test, Morris water maze, social preference test etc, and then create a statistical result for the work. This system is mainly based on real time object detection using Python language, the system works on one type of animal, rats.

By previewing the source code, the program seems to be working fine, it was tested using video clips from YoutTube, the program was able to detect rats moving quickly and easily [8].

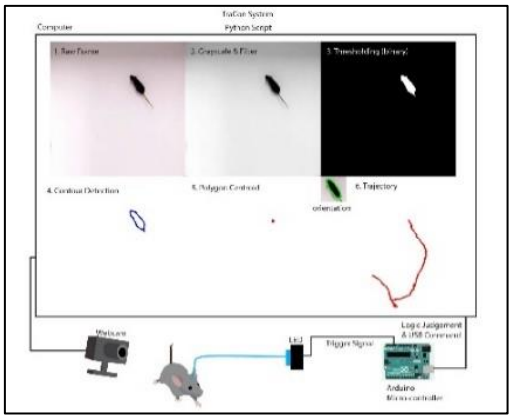


Figure 7. Architecture of the TraCon system

Conclusion. After fully reviewing the program, the program was excellently able to detect mice moving quickly and easily. Through the project, this feature will be added via the camera, through the camera, animals will be detected in real time.

3. Methodology

3.1 Agile Methodology

We begin our design process by Agile Diagram of the Road Guard System, in a simplified form, the Agile method is a simplified method for managing software projects by dividing the project into stages. This method forms a continuous cooperation between the programmer and stakeholders to improve the project at each stage. Therefore, this method considered an excellent way to apply it in the current project. Dividing the project into sections helps reduce project risks as well as reduce requirements. Each iteration we work with throughout the project development lifecycle includes planning, requirements analysis, design, coding, and finally testing before project proposals presented [9],[10].

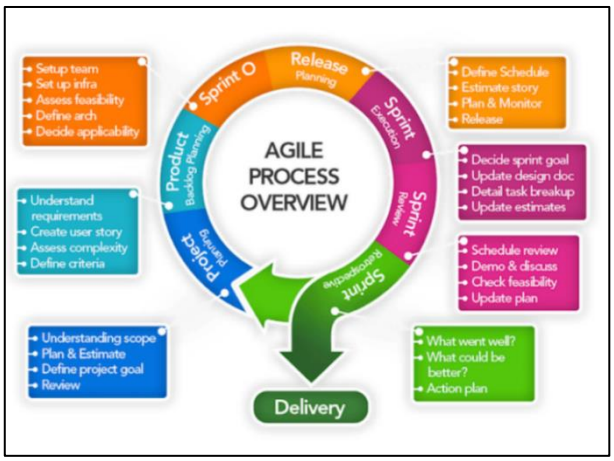


Figure 8. Agile diagram

There following are the stages of the project by applying the Agile model, Figure 8 illustrates Agile life cycle, with each iteration adding new features to the project, which leads to the project's growth gradually. With verification and validation of the features early in the development process, so the chance of providing a failed product is much less. So let's examine the project in each stages of the Agile lifecycle in detail.

Plan requirements: In this step, the main aspects of the project should be identified, it was discussed in detail in the section 1.2. the main objective of this project is to discover wild animals that can cause problems, and an alert is sent to stakeholders.

Development: The project must be designed and developed according to approved guidelines. It was discussed in the sections 3.1.

Test: When the project is completed, a test of the project must be completed and the results documented before delivery. The test is done to ensure the quality of the project, as it must be verified that the coding is clean, and errors are searched for and addressed, as well as a trial run to ensure the effectiveness of the program.

Deliver: In this step, the project or program is presented to stakeholders or clients that have a relationship with this project.

Assess: Take feedback from stakeholders and collect them for inclusion in the project. In other words, all stakeholders must be constantly and directly informed of the project objectives and updates, and provide an opportunity for stakeholders to review the project and provide comments periodically.

Conclusion: Agile methodology is a good option to implement in this project, as it allows taking feedback from clients and stakeholders before the project is presented, which contributes to reducing errors in the project.

3.2 Requirements Analysis

3.2.1 Requirement capture

When checking the project objectives mentioned earlier, there are requirements that must added in the GUI.

1- The application must give the user the ability to connect to the camera and driver sign.

2- The application should display a live feed of the camera with the results of animal detection clearly visible, meaning that the bounding box should appear when the animal detected with the animal's name clearly visible.

3- The admin must be able to receive warning when animals are detected.

4- The admin must be able to view an analysis of the animal data that has been detected.

5- The driver must be able to see the alert when the animal has crossing.

After analysis of these requirements, it found that the application will include a home page (Controller) that acts as a central hub and through which feed will be displayed, the controller also will showing the current time. In addition, buttons that have pop screens to display the more features such as: choose the type of feed, whether it is CCTV or MP4 video, the feed is controlled through two buttons: open to proceed and stop to pause the feed. The files that the admin wants to add to detector the object (it will be optional). And there will be another page to display the data analysis connected with database. Also, controller screen will connect with driver's sign to warning the driver's after detecting animals using API.

The 'Input' and 'Detect' page should allow saving the input data in text file to facilitate the saving process for the user meaning that when the application closed and run again, the user should not add selection files and add the camera from the beginning, unless the user wants to change. It was necessary to add this feature so that it would not be tiring to add the same options every time the user enters the application.

These requirements were determined after talking to some of the stakeholders, as well as research in internet was conducted to find out what features can be added in the project.

3.2.2 Use Case Diagram

System can analysis video and IP camera stream, then alert admin for animals detected in X area as well as alert driver sign. System will store the detected info from each frame in database. The system administrator has permissions to perform operations such as adding detectors files and displaying the data analysis and connected to driver's sign.

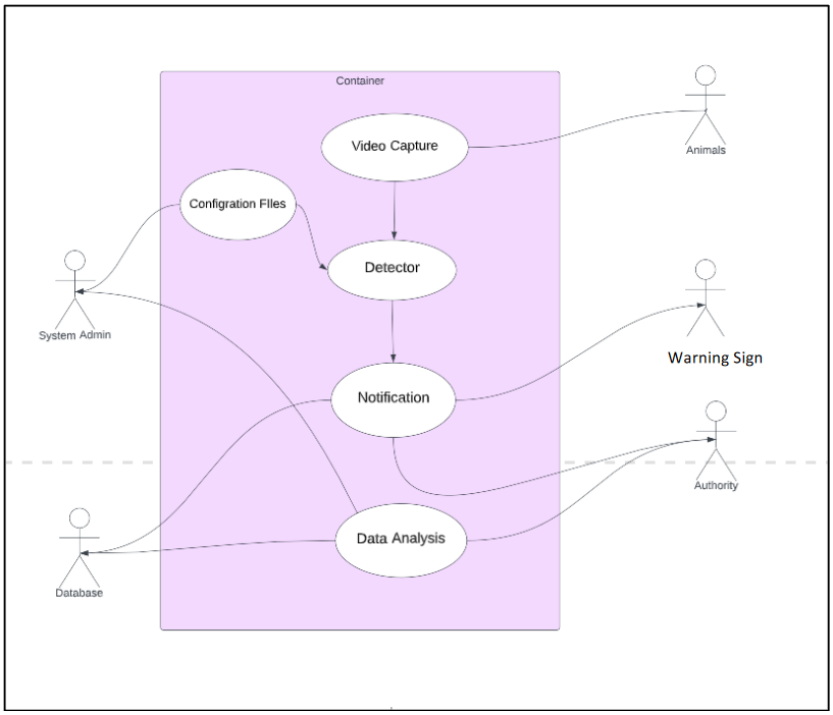


Figure 9. Use case diagram

3.2.3 Problem analysis

As discussed previously, the main objective of the project is to develop a system that controls the CCTV camera, and drivers sign. Where the camera used in the live feed to detect animals and then send alert to the drivers sign. From here, we can identify the components of the project through four things:

1- Application that connected between CCTV camera, and the drivers sign.

2- Detectors files that identify the animals.

3- Send an alert to the authorities/ admin.

4- Data analysis for the detected animals.

In this section these components will analysed and discussed.

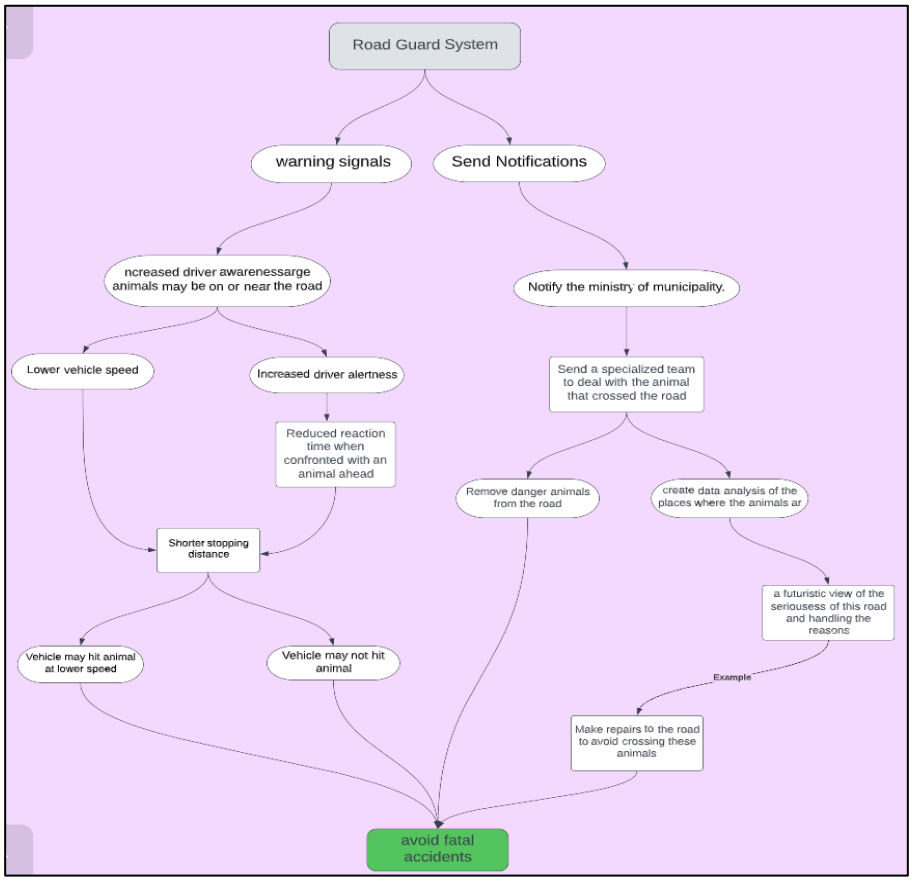


Figure 10. Road guard diagram

3.2.3.1 CCTV Camera

As mentioned earlier, in this project a camera will be used to sends a live feed to the road guard controller, the type of camera chosen in this project is CCTV camera. Which is a type of video surveillance system, it is good type for different lighting, such as night vision and others, and it also programmable by USB port. The use of this type of camera has advantages and in return there are some limitations, these benefits are that this large field of view, and that will help us to detect the animals in large place. Also, this type of camera has high resolution that can help to avoid some wrong detection happen because group of animals suddenly appear in same stage, also it is have good night vision to detect the animals in night. The only limitation for this type of camera is high cost which can be barrier. because this type of camera is not common in computer vision, sometimes there are errors there happening suddenly when connecting of which is that the computer cannot recognize the camera, or that the camera connected correctly but does not work. If we are going to use this type of camera it is important to take these points into consideration.

A. Ensure that the CCTV camera and computer vision system are compatible: This can involve checking the specifications of both devices to make sure they use the same communication protocol.

B. Use high-quality cabling.

3.2.3.2 Animals Detection

There are challenges that may affect the accuracy of animal detection, and solutions provided is using the CCTV camera with heigh quality provided with IR, the CCTV camera has an IR detection feature that enables you to detect objects at night. It is important to know this because you do not need to detect animals that cross the roads only during the morning, therefore before evaluating the project, it is necessary to focus on lighting, because it is a prerequisite.

The errors that occur due to the detection of animals can be reduced by increase the size of samples according to the mentioned categories (camels, goats, donkeys).

3.2.3.3 The warning signs

A digital billboard will be using to warning drivers of collision with wild animals that cross that roads early helps each driver to reduce speed and avoid accidents that may occur and lead to deaths, this panel will connect to computer has drivers sign warning emulator. So, it is necessary to know the safe distance in which the billboard should place on the road, after research we found that the best way to solve this problem is by calculating the distance to the animal discovered mathematically.

Calculate the expected distance for the billboard

We will assume that the billboard will be placed 200 meters before the CCTV on the highway. The question here, is 200m sufficient to warn a driver driving on the highway?

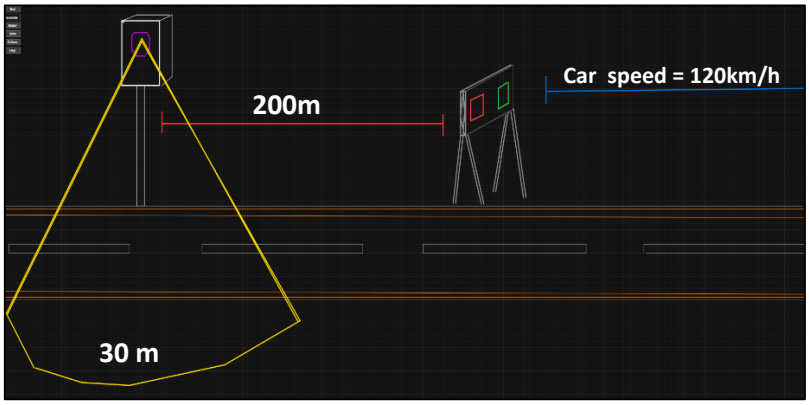


Figure 11. Distance calculation of detect system

The natural driver's reaction time is 1.5 seconds. To find the distance covered during this time:
The total distance the driver to stop = distance between the warning sign and the animal + the distance that the car travels during the driver's reaction time.

1- Calculate the time it will take the driver to reach the animal= $\frac{200}{120} = 1.67$ minuts

2- Stopping distance = Stopping Distance = driver has a deceleration of 6 m/s^2.

Stopping Distance = $\frac{120\ m/s^2}{2\times 6\ m/s^2} = 1200\ meters$

Since the Stopping Distance (1200 m) is greater than the Distance to the animal (200 m), the driver will be able to stop before hitting the animal. So, the 200 m is prefect distance to put the driver sign.

3.2.3.4 Warning the authorities

A road guard controller will be used to send notifications if animals are detected on the roads or near residential neighbourhoods that may threaten public property or cause fatal accidents. The camera will be used to send the captures to the controller and then send alert directly to the authorities, also, accurate data will be sent such as (camera location, camera ID, date, time, type of animals). Through which the official can locate this animal and send a special team to deal with it. Also, the admin can reveal the data analysis, that is mean when the data analysis is presented the official can then create solutions for the road in cooperation with the ROP, through which laws will be enacted such as reducing speed in that area.

3.2.3.5 Object Detection

Object recognition is not a new-born term; there are algorithms that use object recognition. This section describes the techniques used to detection, and highlight the advantages and disadvantages, with providing conclusion.

YOLO algorithm

YOLO algorithm divided into three techniques:

1- Residual blocks:

In the YOLO technique, the image divided into different networks, each network contains dimensions of S x S, the image below shows how to divide the input image into multiple networks:

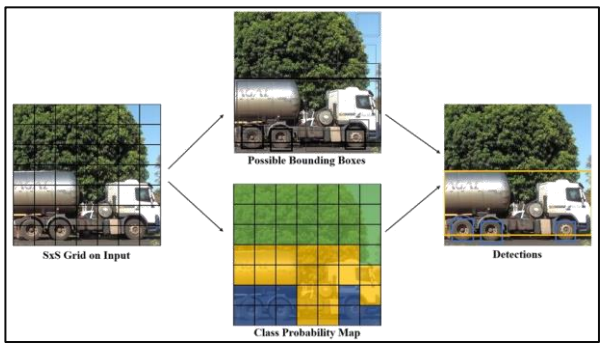


Figure 12. Residual blocks

In the previous picture, there are grid cells containing equal dimensions, each cell will detect the object that will appear inside it, for example if an object appears inside one of these cells, this cell will be responsible for discovering that object.

2- Bounding box regression:

Briefly, the bounding box is primarily responsible for highlighting the outline of the object in the pictures, as each bounding box consists of several attributes: width, which is symbolized by (bw), height symbolized by (bh), and defining the category, for example, animals, which are symbolized by the symbol C. And most importantly It is the centre of the bounding box (bx, by) [11].

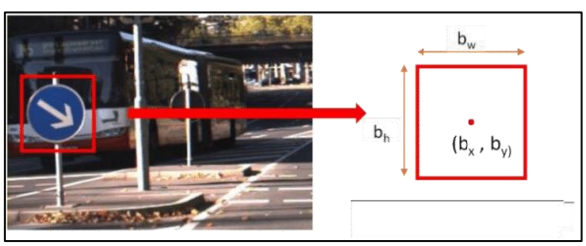


Figure 13. Bounding box of object

The YOLO algorithm uses bounding box regression to predict the object's height and width as well as its position and class type. Figure 13 shows the probability of an object appearing in the bounding box.

3- Intersection Over Union (IOU):

It is the overlap area between the bounding box and the expected bounding box. The following image shows how IOU works.

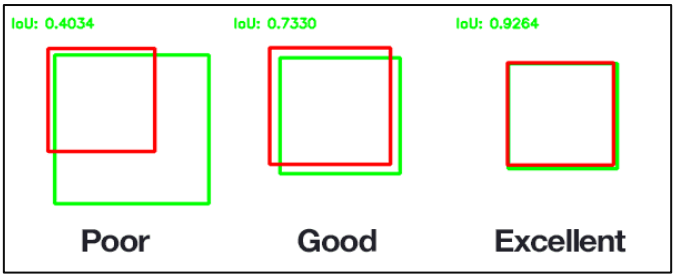


Figure 14. IoU

The previous picture shows two bounding squares, each containing a different colour, the green square represents the prediction, while the red square represents the truth. So, in this algorithm, the two bounding squares must be equal to be detected.

This technique helps to measure the accuracy of the prediction in the model, where we can know the interference of the expected bounding square with the original square. If the value of the IoU height, the better the detection [12]. So, we have two bounding boxes, then, IOU defined as

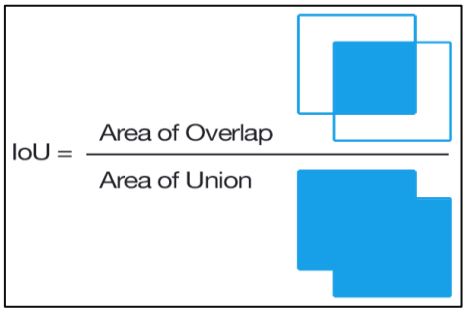


Figure 15. IoU equation

- **Advantages and Features**

The biggest advantage of YOLO is its high speed in identifying objects in images, as it can process more than forty-four frames in just one second.

On the other hand, during the process of training the objects, the algorithm can perceive all the image information, so the false positive rate will decrease in the background. Also, this algorithm can detect all objects of unnatural images faster and more accurately [13].

- **Disadvantages**

There was a weakness in the detection of small organisms and lower recall [14].

Retina Net

In short, Retina Net algorithm it is a single-stage detector, it has two features that the YOLO algorithm lacks:

1-Focal Loss

2- Feature Pyramid Network

These features help match detection performance in devices without losing speed. It is worth noting that this algorithm uses entropy to solve potential problems, called focal loss.

$$FL(p,y) = -\sum_{t} y_t (1-p_t)^{\gamma} \log p_t$$

T: class index.

Yt: class label.

Pt: the expected probability if the object is of class t [15].

This algorithm is used by an infrastructure called Feature Pyramid Network; this network computes the convolutional feature maps of the entire images.

The architecture of this algorithm consists of four main elements: (bottom-up pathway, top-down pathway, classification subnetwork, regression subnetwork).

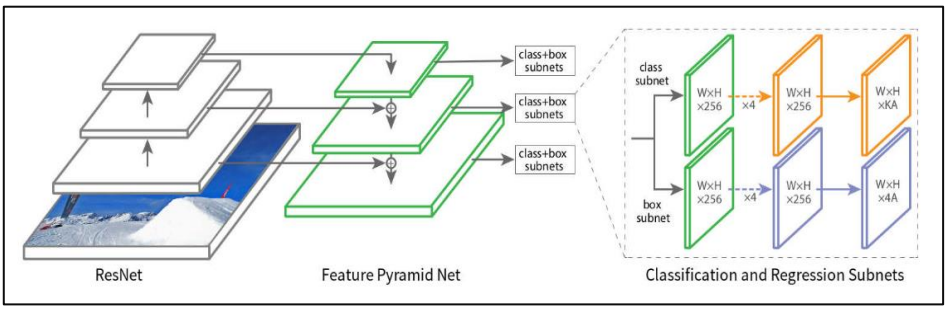


Figure 16. RetinaNet architecture

In the figure 16, this algorithm consists of:

ResNet: A path from bottom to top contains a backbone network called (Feature Pyramid Network), through which multiple feature maps calculated [15].

Feature Pyramid Network: A top-to-bottom path, where it collects feature maps from the layers of the upper pyramid, and then connects layers from top till the bottom and then again from bottom till the top of the same size through the side links.

Classification Subnetwork: Object classification.

Regression Subnetwork: refinement the bounding box.

Advantages

- Detection of small organisms [14].

Disadvantages

- Retina Net does not work with the real-time requirements, which limits its scope applications because the detection accuracy is not high [14].

Conclusion

There are ways of learning to discover things, although Retina Net has now proven effective that there are flaws in this technique that make it unsuitable for the project which are accuracy and speed. The best algorithm for real-time object detection is the most accurate,

according to average accuracy (MAP) the best algorithm for object detection in 2021 is YOLO. On the other hand, the fastest algorithm to detect objects in real time through the inference time scale, which is expressed in milliseconds, the lower is the better, as yolo algorithm is the fastest algorithm to detect objects in 12 milliseconds [14].

3.3 Design

3.3.1 System Architecture

We can design an initial system architecture to derive an overall picture of how the components of a road guard application work.

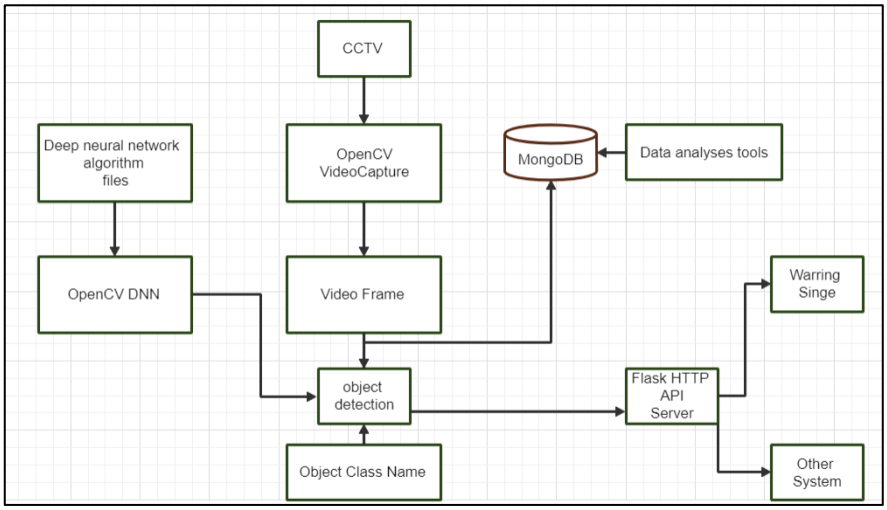


Figure 17. Road guard system architecture

Figure 17 shows road guard system architecture. The diagram begins with the start symbol, followed by the direction of flow in the flowchart, where after starting the road guard the OpenCV video capture process will take feed from CCTV stream and break it to frame. Then the object detector will process the frame based on loaded OpenCV DNN Neural Network which loaded with algorithm files to detect object defined by Object class name .If Object is detected it will be logged in database and Real-time notification will be sent to API interface and connected system such (Warring singe) will display the notification including object name and location .Also the Analysing tools will connect to database to process the records and with function to filter and plot the data.

3.3.2 Process flow diagram

The diagram begins with the start symbol, followed by the direction of flow in the flowchart, where after starting the road guard the camera takes the input images. Then will send the images it has collected to the CCTV, then sends tires and analyses images by the model, if the animal is (Camel, goat, donkey) directly will sends notifications to authorities and billboard warning. If not, the process will be started from the beginning. The process goes on if the animal is not found.

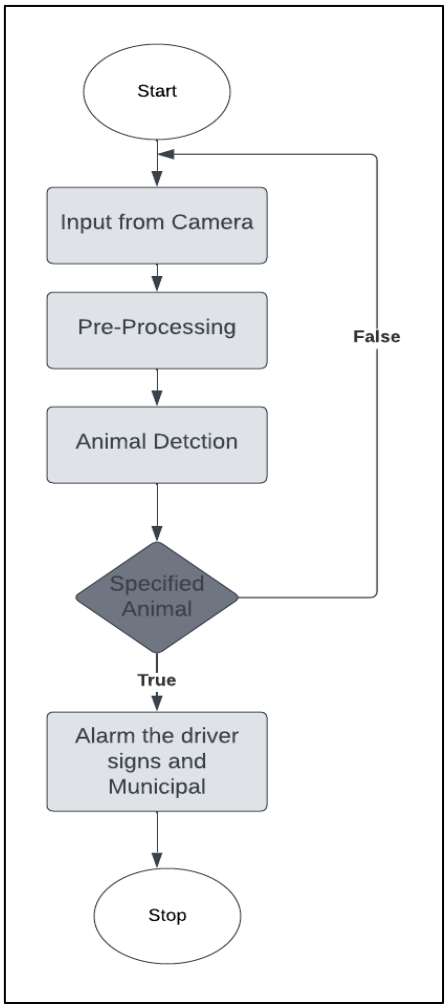


Figure 18. Flowchart

3.3.3 User interface Design

The next step after completing the development of the basic components of the program is to design the user interface with useful functions. So, before diving directly into the design of the program there is a need for a design outline to give a clearer picture of the application. This section will talk about requirements and main design.

3.3.3.1 User interface Wireframes

This section will talk about the initial design of the user interface for the road guard, we will talk about what the application will look like and how the elements will interact with each other.

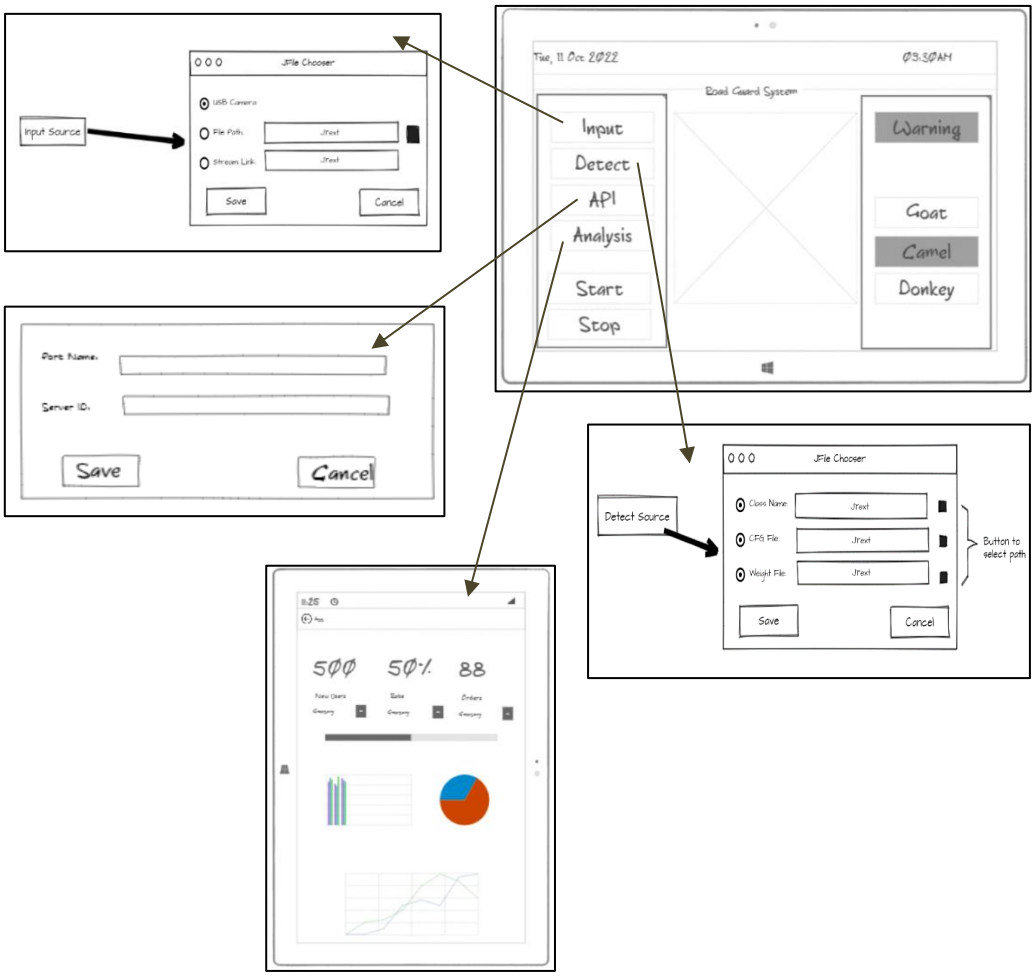


Figure 19. Road guard user interface

3.3.3.2 Main interface Wireframes

Figure 20 shows the design of the user interface home page. The design is an easy and simple interface without any factors that may cause the user to distract. The date and time displayed in the upper space, the buttons are on the left side, and the video is in the middle, in addition to the warning signs on the right side. The buttons supply the following functions:

Input: Add video, whether live or via the desktop.
Detect: Add file (CFG, weight).
API: Connected to driver sign.
Analysis: Data analysis for animals.
Start: Open video.
Stop: Stop video.

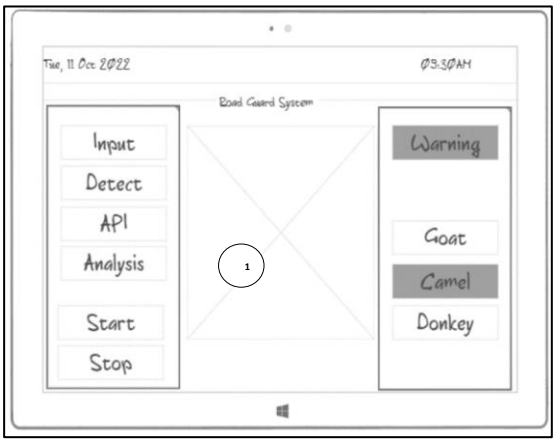


Figure 20. Main page wireframe

(1): It will display a camera feed with live results for finding this animal within the bounding box, and the box will also display the name of the animal that detected to display the result more clearly.

3.3.3.3 Choose video

The figure 21 shows the user interface design for the video add page. Through this page, the user can find the desired video, whether through the camera, or the PC. This window must open separately from the main window. This window will allow you to enter the path of the video if it is on the desktop or the IP of the camera to streamed.

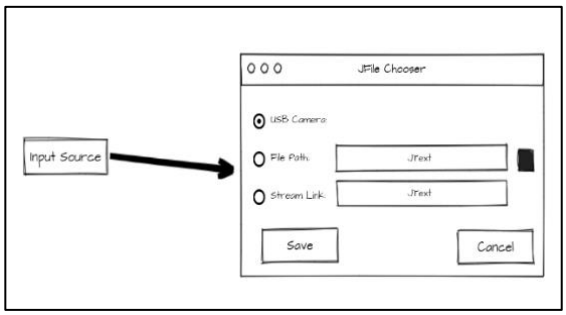


Figure 21. Video page wireframe

3.3.3.4 Choose Detect File

Figure 22 shows the design of the upload files page. Following this page like the adding videos page through the design, allowing the user to choose the detectors files need to detect, for example the user can add detection files for animals, faces, or vehicles. When adding these files, road guard system can figure out what the object needs to detect.

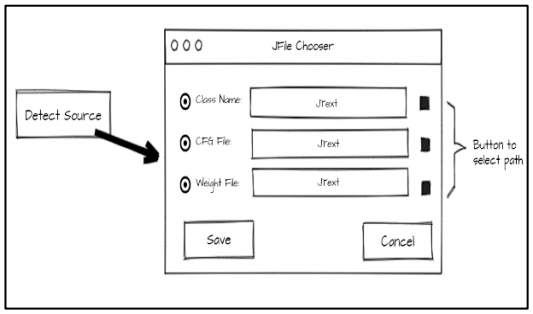


Figure 22. Detect page wireframe

3.3.3.5 Data Analysis

Figure 23 shows the design of the data for the animals that are found by using the stored database displayed, the user can know all data for this animal information like type of animals, how many animals detected in the road, through this page.

Figure 23. Data analysis wireframe

3.3.3.6 API Button

In Figure 24, this file appears when the API button pressed. This button used to configure the IP and port so another system can access the application data.

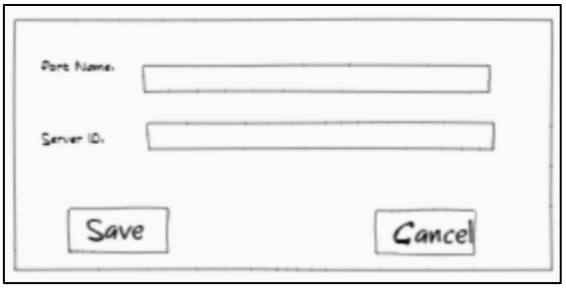


Figure 24. API wireframe

3.4 Development Platform and Tools

In this section, we will discuss the techniques that help in creating the application. A prerequisite for maintaining excellent real-time performance of the algorithm is a GPU. And a computer with this following:

3.4.1 Software

OpenCV

OpenCV [16] is an open-source machine learning software library that focuses entirely on image processing and real-time object discovery. The OpenCV library uses algorithms to compare digital or live images stored in a huge database. Object recognition used by deep learning algorithms to compare live images with animal data in the database. The OpenCV library built in C language so that any device can run this programming language, which means that it can run the OpenCV library. It can be use on (windows, Linux, macOS) and mobile devices (android, IOS).

The OpenCV library used to develop a wide range of systems and applications for various uses, for example, identifying animals that cause damage to farms using a live video camera that alerts farm owners if there is any animal approaching private farms. For this system to work, there are three main steps:

- **Animals Detection**

Once the animal photo captured using the live camera, that photo cropped and then send alert to the controller and driver sign.

- **Create a vector to identify the animal**

After capturing the animal's image, this algorithm uses the Dlib function to create a vector 128 that uses the animal's attributes.

- **The algorithm compares the image in the database**

This algorithm compares the vector to the images in the database by using the Euclidean distance to see if that animal matches the images in the database.

Conclusion, the OpenCV library is a powerful library that can support applications, which makes this library suitable for the project due to its ability to process images and video.

Figure 25. The vector recognizing optical features

NoSQL (MonogoDB)

NoSQL Databases [31] are used in big data and real time in addition to web applications, because this type of database is highly scalable. NoSQL databases are stored in more understandable ways. Therefore, this type of database was chosen because it is flexible and supports different sets of data models. Therefore, in this project, MonogoDB was chosen because it allows us to deal with large data, in addition to its great efficiency, as it is expandable and flexible [17].

YOLO

As discussed in Section 3.2.3.5, the Yolo algorithm is used to detect objects in real time. This algorithm will be used in the project to detect animals due to its speed and accuracy in detection.

JAVA

It is one of the programming languages [18], this language is used for general purposes, as it depends on the class. This language is classified as fast, secure, and reliable. This language is used in developing applications in computers, data centres as well as game consoles and many more, as java contains many and varied libraries.

- **Swing library**: it is used to design the Java GUI; Swing provides a wide range of tools and packages.

• **Java Platform**: t is a group of programs that help programmers develop Java applications. It contains:

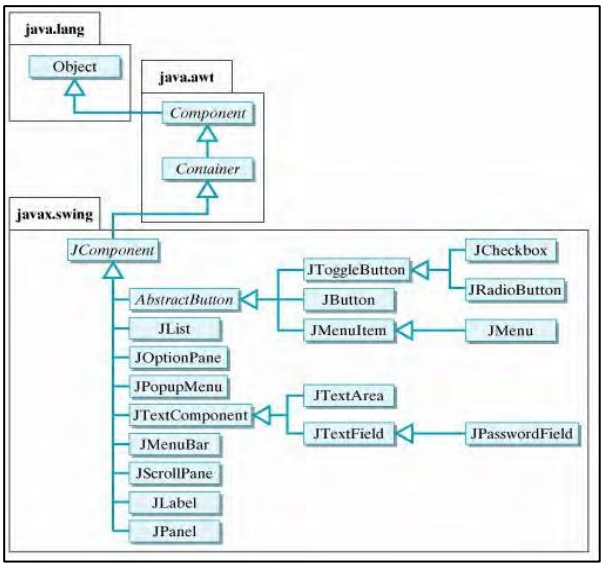


Figure 26. The hierarchy of java swing components

Conclusion Java is a safe and powerful language that can quickly create graphical user interfaces by using large and diverse libraries, which makes this language suitable for the project because it is a secure language.

Python

It is one of the programming languages often used to create websites, programs, and data analysis procedures. Python is used to perform data analysis by using specialized analysis libraries, for example the data analysis library [19].

- **Pandas**: It is a software library developed using the Python language. This library can process and analyse data quickly. This library provides many tools, including the representation of complex data structures.
- **Scikit-Learn**: They are advanced analysis tools in Python, also used in complex machine learning, which leads to the creation of many complex models in addition to the creation of complex regressions.

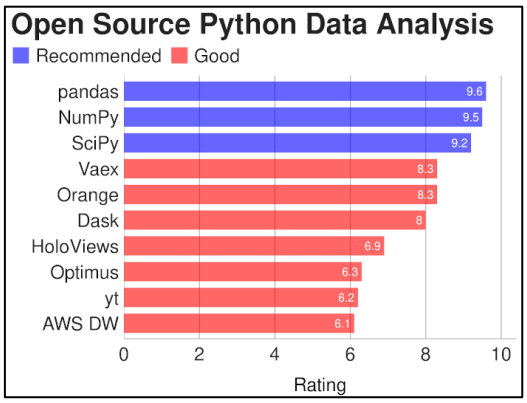


Figure 27. Open-source python data analysis

The figure 27 shows recommendations for conducting data analysis using the Python language, as pandas are the first choice due to their speed and accuracy [25].

Conclusion The Python language contains many open-source libraries such as the Panda library, which allows the creation of data analysis quickly and accurately, which makes this library suitable for the project.

Darknet

It is an open-source neural network framework, Darknet is very fast and easy to install as well as accurate. It supports both GPU and CPU computing.

3.4.2 Hardware

System parts	Price	Purpose
CCTV Camera	N/A	Capture the footage and send it to Guard Road system
Billboard warning	N/A	To warning the driver

Table 2. Estimated cost

3.5 System Design

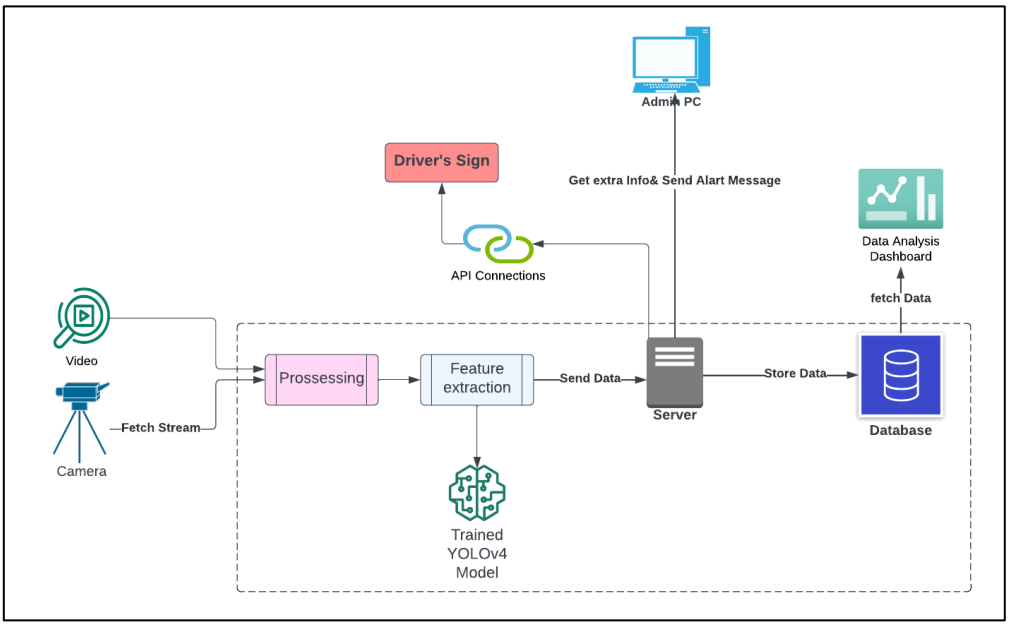


Figure 28. Road guard system diagram

From the above diagram system Figure 28 will be as following:

Data Source:

Data stream it will be for road guard system will be in camera or mp4 video.

Trained YOLOv4 Model:

The model will be the part responsible for analysing the video stream and detecting the type of object that has been detected such as camel, goat, donkey, etc. After the animal is detected in the stage, in every minute the system will stream. If it contains an animal or not, if the answer is yes, the system will send it to the back-end server to store it in the database, after that the user can view data analysis. The system will use the API to connect the system with the drivers alert panel, meaning if the system detects an animal, it will send a notification to the drivers' alert panel. So, in short, the administrator will connect the driver board with the control by inputting (IP and port) so that the driver can receive the alerts. Then, when the camera is connected to the road guard system, it will start streaming video and detecting animals in and then sending an alert to the control panel and the driver's panel. During this time the system will store the information detected from the camera in the database so that the system can send the information to the backend server which It will work in data analysis and presentation.

4. Implementation

The objectives and potential problems that may arise in this project have been identified, however, there are some points to clarify more from a technical point of view to provide some critical analysis and understand the actions that are being taken. In this section, the technical and technical descriptions used in the Road Guard System project will be covered.

4.1 IDE's and Languages

As mentioned earlier, the main languages that will be used in this project are Java and Python. As for the chosen development environment, the Road Guard System was developed using NetBeans and Google Colab, and PyCharm which is a commonly used Python-based environment. On the other hand, the reason we chose NetBeans is because this IDE is easy for me and very popular.

4.2 Deep Neural Networks

A neural network [20] designed to work just like the human brain does solving problems such as computer vision, quantitative estimation, and others, for example determining whether the image in the Road Guard System is of a camel or a goat. Therefore, this task requires processing pixel values in images or video, so there are some groups that help achieve this goal, including OpenCV.

4.2.1 OpenCV

As mentioned earlier in section 3.4.1, this project used the OpenCV library because it is better and easy to implement. As someone who does not have enough experience in machine learning, the OpenCV library is a good choice for its speed and flexibility.

4.3 Implementation of Core Components

This section explores the application whose design was discussed in the previous section, and now it is time to move on to implementation. This section will discuss a detailed description of the basic mechanics and how to implement the main components of the project.

4.3.1 Model training

As mentioned earlier, the project uses the YOLO algorithm to identify animals in the roads. To train this network requires a large dataset, so it will take time to train, it took 26 hours to train and collect the dataset. At this stage when configuring the network for the application, we must have a dataset for all the objects that the application will recognize as we mentioned earlier. To create this dataset, each animal must have at least 100 images were downloaded from Google Images. All images were labelled with bounding boxes by using (*https://www.makesense.ai/*), with added Camel class name for camel's pictures, Donkey class name for donkey's pictures and Goat class name for goat's pictures. The main idea is to label each object to identify the pixels in the image and to add a class name to these images. In labelling we should classify animals from class 0 to class 2, so we have classified the animals as:

Number of animals	Animals name
0	Camel
1	Donkey
2	Goat

Table 3. Class integer number

After completing the creating of bounding box around the object for each image, the program creates '.txt' files for each uploaded image. This file contains the pixel coordinates of the object. Figure 29 shows the images labelling process as well as the content of the generated text file.

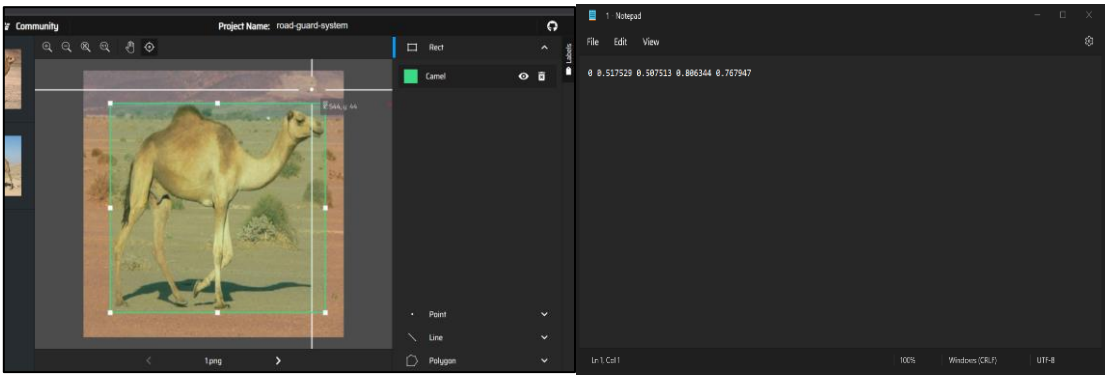


Figure 29. Labelling the object and text file

After we have finished naming the images, we must create a training model, so we use Google Colab because it is freely available in addition to being open source. Two groups of data were divided, the first group, which is for training: (n = 512), and the second group, which is for validation: (n = 208). The version of the YOLO algorithm was chosen, which is Yolo4. Darknet framework was used to prepare the training. Two training models have been created, one is a version called full: ''yolov4'', and the other version called tiny: ''yolov4-tiny''. In the beginning, we created a general survey of the most famous animals that cause accidents in Oman, and it was found that donkeys, camels, and goats are the main cause of animal accidents in Oman. We trained 3 different types of animals: camels, goats, and donkeys Figure 30.(To create training model, we have followed: https://github.com/pjreddie/darknet).

```
[ ]  labels_path = '/content/TrainData/obj.names'
     #make a list of your labels
     labels = ['Camel, Donkey, Goat']


     with open(labels_path, 'w') as f:

         f.write('\n'.join(labels))

     #check that the labels file is correct
     !cat /content/TrainData/obj.names

     Camel, Donkey, Goat
```

Figure 30. Create labels file using python

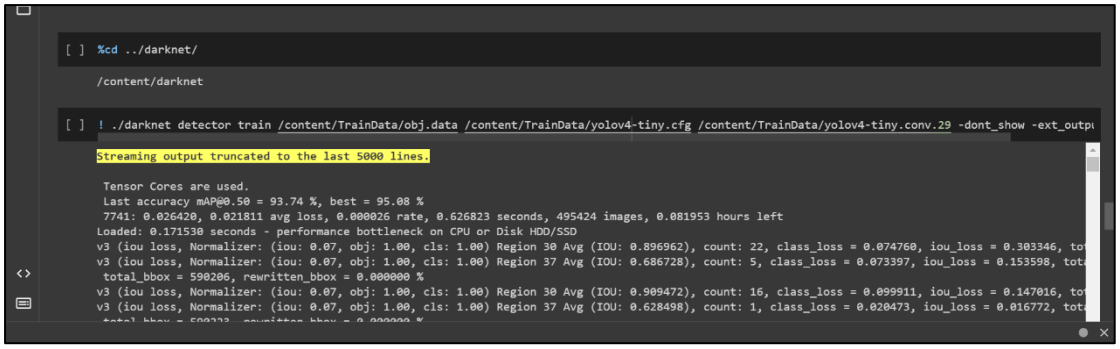


Figure 31. Start the model training

As Figure 32 shows, the weight files were created successfully.

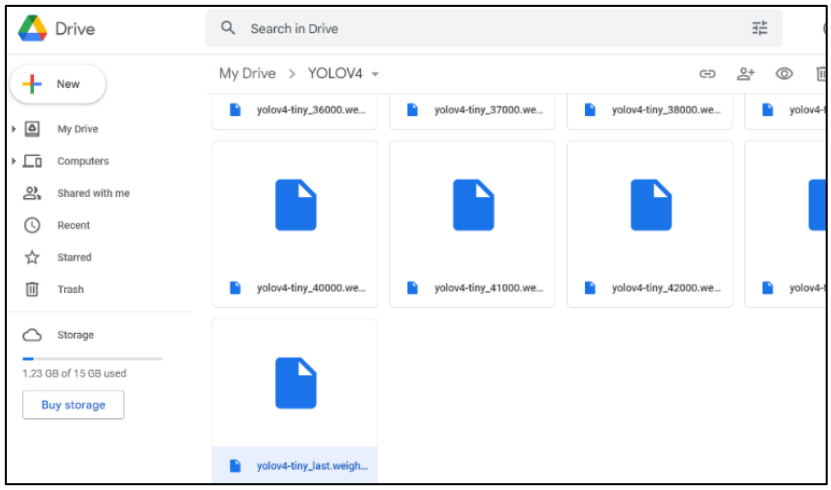


Figure 32. Weight file created

In the following table shows the values for creating the YOLOv4 file:

	Explain	Results
Number of classes	(Camel, Donkey, Goat)	3 classes
Number of datasets	(Camel = 200 pic, Donkey = 200 pic, Goat = 180 pic)	512 samples
Size of pictures (width, height)	Width= 620 pixels Height= 620 pixels	620* 620 pixels
Number of Batch	64	64
Max batches (Iteration)	8000	8000
Subdivisions	64 ÷ 8 = 8	8
Mini- Batch size	64 ÷ 8 = 8	8
Epochs	512 = 1 epoch, So, 512 ÷ 8 = 64	64

Table 4. YOLO model calculation

Batch size: Number of samples, that the algorithm will process in one pass only, the higher the Batch size, the more memory space we need [21].

Subdivision is used to reduce the RAM usage of the GPU at a time, subdivision represents the number of mini-batches present in one batch, so subdivision is required in our project through which the mini- batch will be calculated which GPU will process in one batch.

The learning rate [22] it is a hyper-parameter that controls the value of weight in network, on the other hand it is determines how quickly weights can change.

4.3.2 Model Result

The training results are summarized in following table, which shows: The value of the mean accuracy (mAP) where the best weights that have the highest value of the map were saved, in addition to the training time that the model took. The value of IoU = 0.5 and is presented as mAP@0.5.

mAP: (mean Average Precision) a measure of the model's accuracy in detecting objects. The higher the mAP score, the better the model's accuracy in detecting objects [23].

Model	Best mAP@0.5	Time of training
	93%	23 h

Table 5. Training result of model after 8,000 steps

The blue curve in Figure 33 represents the training loss in the training dataset for our project, while the red line represents mAP values. With repeated training of the samples, the loss percentage red line will decrease dramatically and tend to zero. Y-axis shows loss of our project and X-axis Iteration or epochs number. The loss is lowered as the numbers of epoch are increased.

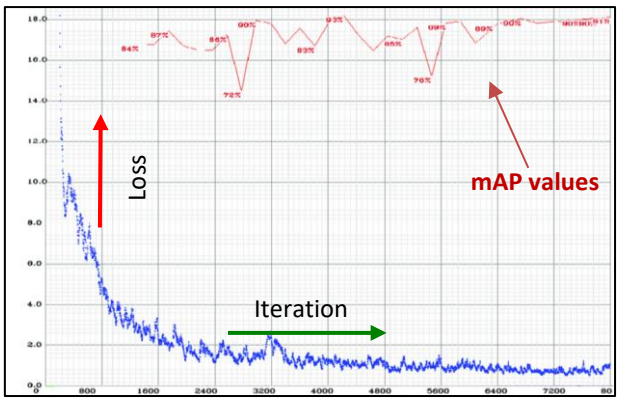


Figure 33. Graph of training loss representation in YOLOv4

The mAP value starts at 84% and start increase to the 87% level, then loss value starts to decrease after 1600 iterations of model collection, with the pattern value starting at 86% and flipping around 72% after 2800 iterations. At 4200 iterations, the best mAP value was recorded, reaching 93%, during which the loss value was stable at the minimum value of 0.5.

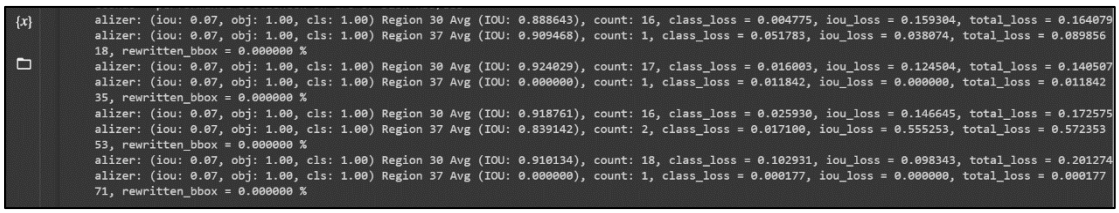


Figure 34. Number of iterations in YOLOv4

Figure 34 represents one batch of images for our project, which are divided according to our subdivisions that are shown in Table 4. Note that the subdivision is calculated as 8. Given Figure 34, the training is repeated in 8 groups of 8 images.

IoU_loss: It is the maximum overlap between the predicted bounded square and the basic truth square. It focuses on both the bypass area as well as the distance between centers [24]. The loss IoU calculated as follows:

$$L_{IoU} = 1 - IoU + \frac{\rho^2(b, b_{gt})}{C^2} + \alpha u$$

$$u = \frac{4}{\pi^2}\left(tan^{-1}\frac{\omega_{gt}}{h_{gt}} - tan^{-1}\frac{\omega}{h}\right)^2 \; and \; \alpha = \frac{u}{(1 - IoU) + u'}$$

ω_{gt}, h_{gt} = widths and heights of the ground truth box.
h , ω = widths, heights of prediction bounding box.
u = aspect ratio.
u' = first derivative.
α = positive trade-off based on distance of IoU loss.

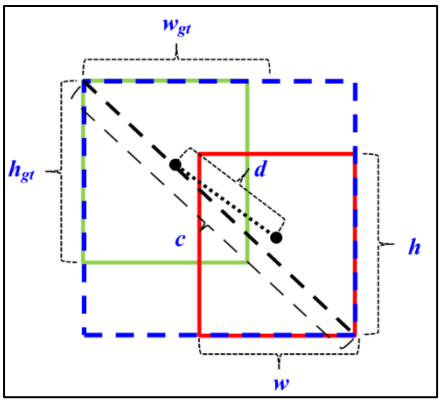


Figure 35. Schematic of IoU loss

Total_loss: To Calculate the total loss function L we can use following function:

$$L = 1 - IoU + \frac{\rho^2(b, b_{gt})}{C^2} + \alpha u^{-}$$

$$\sum_{i=1}^{S^2}\sum_{j=1}^{B} I_{ij}^{obj}\left[\hat{C}_i \log(C_i) + \left(1 - \hat{C}_i\right)\log(1 - C_i)\right] -$$

$$\nprec_{noobj} \sum_{i=1}^{S^2}\sum_{j=1}^{B} I_{ij}^{obj}\left[\hat{C}_i \log(C_i) + \left(1 - \hat{C}_i\right)\log(1 - C_i)\right] -$$

$$\sum_{i=1}^{S^2}\sum_{j=1}^{B} I_{ij}^{obj} \sum_{c \in clases}\left[\hat{P}_i(cl)\log\left(P_i(cl)\right) - \left(1 - \hat{P}_i(cl)\right)\log\left(1 - P_i(cl)\right)\right]$$

Explanation:

S^2: It is number of grid points in the image.
B : It is number of bounding boxes that are connected to each grid.
$\prec_{noobj}$: it is a weight associated with the confidence loss.
I_{ij}^{obj}: Determines if there is an object in cell i.
j: Predict the bounding box in cell i.
C_i: true confidence in object detection.
$\hat{C}_i$: confidence scores of the prediction.

$I_{ij}^{obj} = 1$ The target group is expressed in the bounding box, which is represented by the letter 'j', as it was created by network, which is expressed by the letter 'i'.

else =

$$I_{ij}^{obj} = 0$$

cl: class associated to target detection.
$P_i(cl)$: True possibility of making a detection of the class cl object in grid i.
$\hat{P}_i(cl)$: possibility score prediction.

```
Tensor Cores are used.
Last accuracy mAP@0.50 = 93.74 %, best = 95.08 %
8000: 0.008539, 0.023164 avg loss, 0.000026 rate, 0.579031 seconds, 512000 images, 0.022748 hours left
```

Figure 36. Accuracy of mAP and weight file

`8000`: Iteration
`0.008539`: Total loss
`0.023164 avg loss`: It expressed the average loss error, as it should be the lowest possible number. Training can be terminated when the average loss is less than 0.060730.
`0.000026 rate`: Current learning rate
`0.579031 seconds`: The total it took to process this batch.
`512000 images`: Total images used during training.

Then, through using the dataset, 3 files are created using the Python language, the first file (coco name) that contains a list of all animal's names, the second and third files (CFG & wight) work to identify these animals. As these files work like a brain that will recognize all animals previously added in the dataset.

4.3.3 Model Precision

In this project, we focused on using yolov4, as this version is characterized by more accuracy compared Yolov4-tiny, but it takes more space than the size of the yolov4-tiny model because in yolov4-tiny has reduced the detection grid to only 7 of layer traditional convolution.

Convolution: is an algorithm that performs a filtering process, where the image is filtered and transformed using the kernel to obtain certain information [25].

Kernel: It is a matrix containing values, the size of these values is determined by the transformation effect of the convolution operation [26].

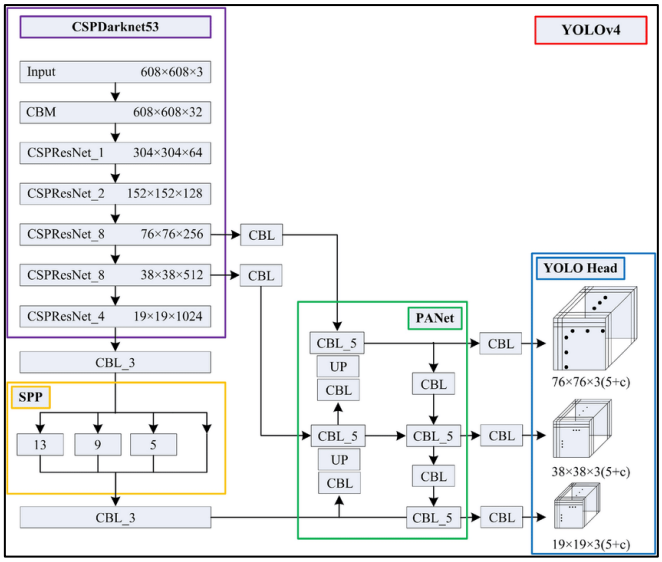


Figure 37. YOLOv4 architecture

YOLOv4 algorithm contains of:

1. **Feature Extractor:**

The feature extractor in the Yolov4 model consists of only 53 separate layers out of a total of 106 layers. The following table shows the Yolov4 layers of the darknet:

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Figure 38. YOLOv4 layers of the darknet

Through the previous table, we can divide the total set of frameworks into a connected block with a stride of two layers of convolution layer, where we can convert the feature extractor directly into a classifier by adding separate layers, which should be 3 layers only, namely:(Avgpool, Connected and Softmax).

Average Pooling: It is process that calculates that average value of patches for the feature of MAP, this process is used to create a feature map [38]. It is often used after the convolutional layer [27].

Connected layer: It is the layers in the neural network in which the input is from one layer only, as this layer is connected to the activation unit of the next layer [28].

Soft-max: It is often used as a final layer for a classification task, which receives the final representation of the samples and outputs the classification prediction [29].

2. **Bounding Box and Output:**

As explained earlier in Section 3.2.3.5, a bounding box in short is a box that surrounds an object to specify its location in the scene and category. Bounding boxes are mainly used in our animal detection project, where the goal is to identify the location and name of the animal in the scene.

The following Parameters are used to describe the bounding box [30]:

- Class: Show the 'object' name inside the bounding box e.g. (Camel, Goat, Donkey).
- (X1, Y1): It is the top-left corner of the bounding box.
- (X2, Y2): It is the bottom-right corner of the bounding box.
- (Xc, Yc): It is the center of the bounding box.
- Width: width of the bounding box.
- Height: height of the bounding box.
- Confidence: It is about the possibility of the presence of an animal in this location, for example, e.g., the high confidence represents 0.8 that there is an 80% percentage in the scene for the presence of a camel animal in the square.

3. **Per-Class Sigmoid:**

It checks whether there is a probability of finding the animal in bounding box. This is a simple binary classification task, since for each bounding box and for each object score there is its own logistic regression model.

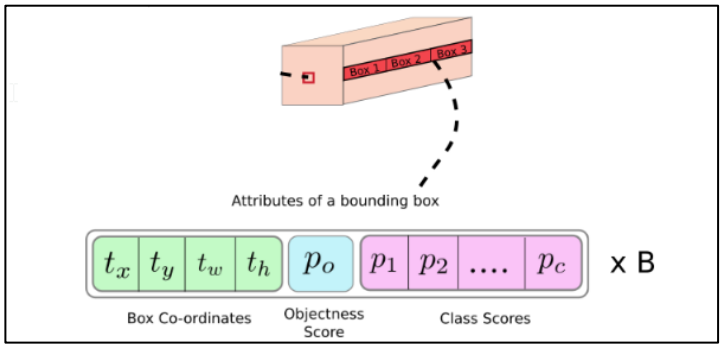


Figure 39. Pre class sigmoid

4.3.4 Precision Results

The average accuracy of both AP and mAP in addition to F-1 is used to measure the model's performance for object detection. The sample data for the object can be divided into four classes according to the original and projected data:

True negative (TN): The model predicts a correct outcome but for the negative class.

True positive (TP): It is outcome that is correctly predicted in the model for the positive class.

False negative (FN): The model predicts an incorrect outcome but for a negative class.

False positive (FP): Tit is predicting an incorrect outcome but for a positive class, there are no wild animals, but the test predicts their presence.

Precision: It is a percentage that identifies samples of positive cases from all samples that are expected to belong to that category [31].

Recall: It is the percentage that is determined by the machine learning model and that is classified as (positive category) of the total samples [31].

Ground truth: The term ground truth is used in statistics and machine learning, and it is intended to verify the results of machine learning to ensure the accuracy of the results compared to the real world. That is, the ground fact is the process of collecting data that is appropriate and demonstrable for this test [32].

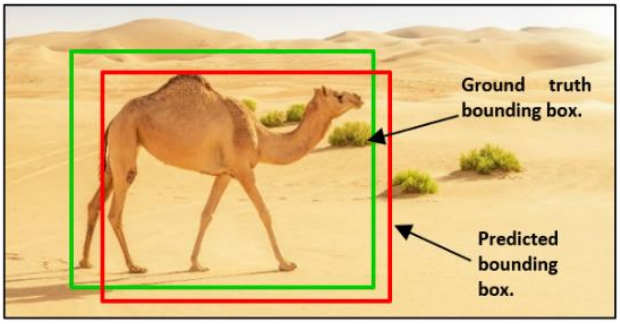


Figure 40. Ground truth with predict bounding box

We can define both Precision = P and Recall = R as follows:

$$P = \frac{TP}{TP + FP} \qquad P = \frac{TP}{TP + FN}$$

So, the result indicates the precision of object detection:

$$F1 = \frac{2PR}{(P + R)}$$

In Figure 41, AP represents the area under the P-R curve:

$$AP = \int_0^1 P(R)dR$$

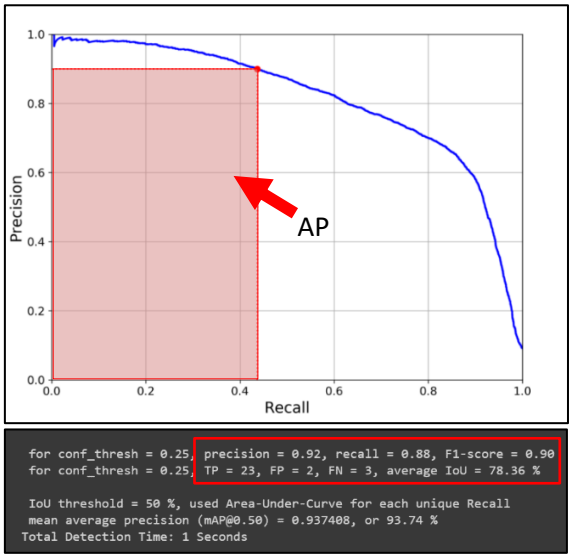


Figure 41. AP calculation

The higher area at the under of curve indicates the higher prediction in the object, on the other hand mAP represents the average among of all APs for given class. So, the higher the

result, the greater the accuracy of the model in the discovery. The mAP is calculated using the following equation:

$$mAP = \frac{1}{n}\sum_{k=1}^{k=n} APK$$

Where APk = AP of class

k, n = representing the number of classes, in dissertation N= 3

The following figure shows the mAP calculation:

```
Tensor Cores are used.
Last accuracy mAP@0.50 = 93.74 %, best = 95.08 %
8000: 0.008539, 0.023164 avg loss, 0.000026 rate, 0.579031 seconds, 512000 images, 0.022748 hours left

calculation mAP (mean average precision)...
Detection layer: 30 - type = 28
Detection layer: 37 - type = 28
20
```

```
mean_average_precision (mAP@0.50) = 0.937408
```

Figure 42. mAP calculation

In comparison of models, the samples showed similar performance with a slight increase in the performance of YOLOv4 compared to the performance of YOLOv4-tiny. The results of the average accuracy of the model by class (AP) as well as the average accuracy of the (mAP) detector for each model are shown in the following figure. The results presented for both groups were satisfactory. The accuracy results for the YOLOv4 class = 93% and 90% for the YOLOv4-tiny. However, in each class there have a same number of images, a larger number of camels(n=200) and donkeys were labelled compared to goats (n=180). This explains all the differences that were previously shown in the detection results between the mentioned classes.

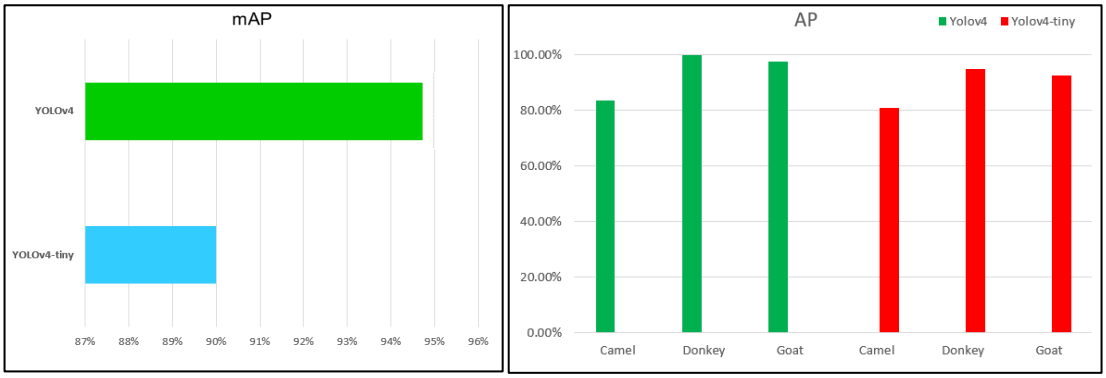


Figure 43. Comparative precision results YOLOv4 and YOLOv4-tiny

After training and extracting the files, the performance of the detector was tested, the observed frame speed of the Yolov4-tiny model was higher than the speed of the Yolov4 model. As the results of the Yolov4-tiny speed = 45 frames per second, unlike the yolov4 model, which was 37 frames per second, but the accuracy of Yolov4-tiny is much less. In our project, we care more about accuracy than speed, so Yolov4 was chosen.

4.3.5 Implementing Animals Recognition

Figure 44 shows the YOLOv4 model to detect both camels, Goat, and donkeys. The model test follows the following steps:

1- As mentioned previously, labelled are put on the test images.

2- The training model receives the photos.

Figure 44. Test model detection

Figure 45 represents the detection results between both yolov4 and yolov4-tiny model. Through the following graph, both donkey and camels presented an increase in the FP result (n = 77), and Donkey (n=42) compared to the Goat class (n=36). This problem is often due to the appearance of a large number of the same animal in the image, but a classification of all these animals was not made in the ground truth data, but in the test, they were detected.

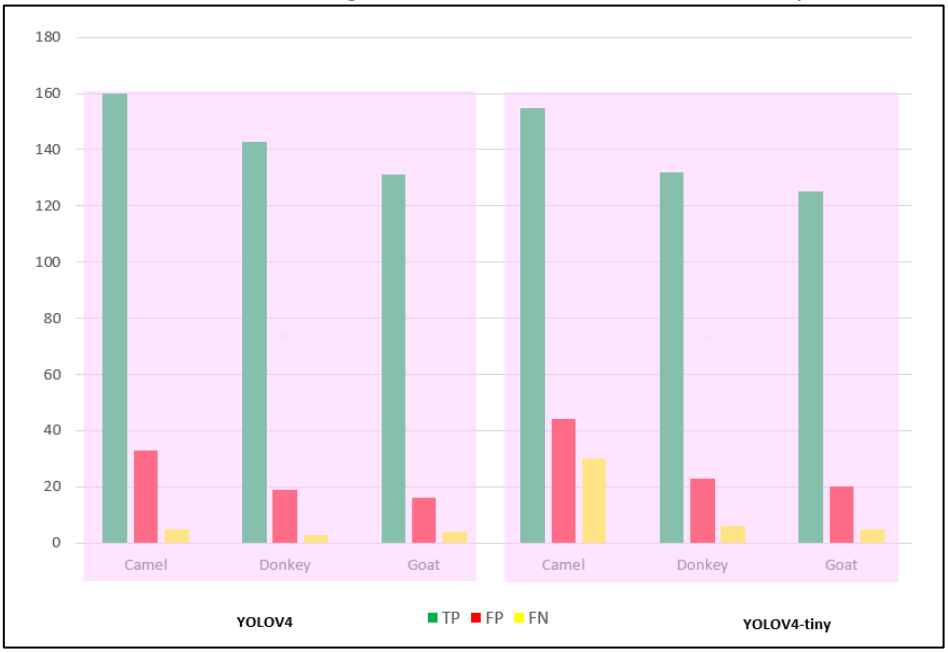


Figure 45. Result of YOLOv4 and YOLOv4-tiny

One of the main problems facing our current system is the animals that the system could not detect yet, which express (FN). Through the previous picture, we note that the camel class show many objects that it did not detect (n = 35), compared goat class (n = 9) and the donkey class (n = 9). Although some of the images may contain animals that have not been detected yet, which are called (false negatives), at the same time there is a (true positive). Meaning that in such cases that occur in real time, it will not become a problem in the reports that may be generated, if Road Guard at least a positive detected is observed in the monitored scene. It is worth noting that most of the (FP) detections that were examined were mainly due to the presence of other discoveries from the TP, and this is a good thing and does not cause any problem. an example of these events is shown by the Road Guard system.

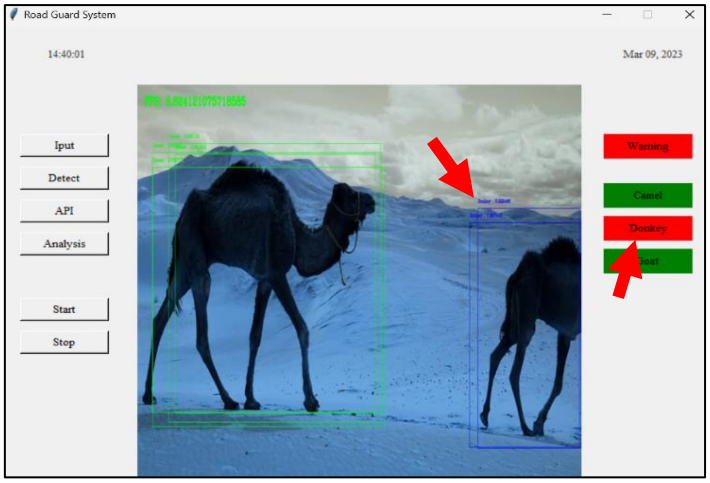
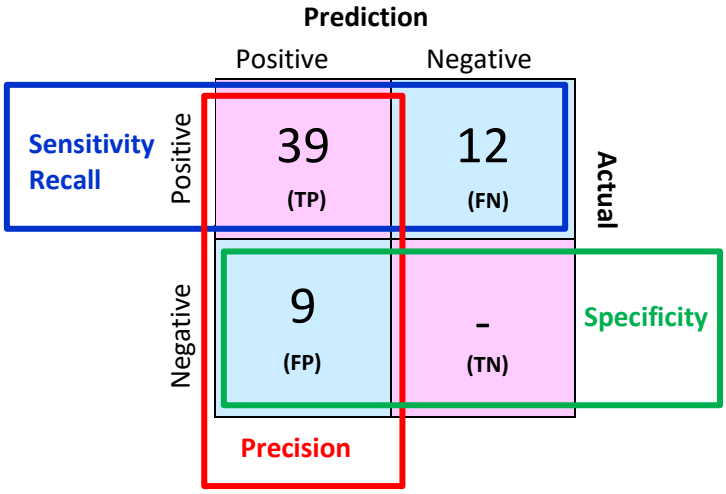


Figure 46. FP and TP detections

There are some challenges that may occur during the detection of animals in real time, as all these challenges that may occur must be dealt with, such as a difference in the lighting of the scene, the timing of the discovery, whatever morning or evening. Through this algorithm, false positive detections (FP) often appear, for several reasons, such as errors in translation, the presence of background noise, or the presence of similar objects that may pass through the scene, in addition to that, a false negative (FN) result can occur for reasons such as the difference in the size of the animal.

Overall evaluation on test dataset:



The confusion matrix

In the confusion matrix, we note that there is no truth and negative value TN, each bounding box. All possible bounding boxes that are not correctly detected are expressed as true negative. Scientifically, this value is uncountable, and for this reason the real negative is ignored and is not used by the metrics.

4.3.6 Camera & signs Connection and Live Feed

In the problem presented in section 3.2.3 we were thinking about the fact that most cars in the Sultanate of Oman do not have a smart camera system, or a special night vision system installed on all animals that threaten the lives of drivers. So, we were focusing on developing

an early warning system of collisions with wild animals. The system consists of a camera placed at the top of the warning sign Figure 47, which are placed on the chosen road.

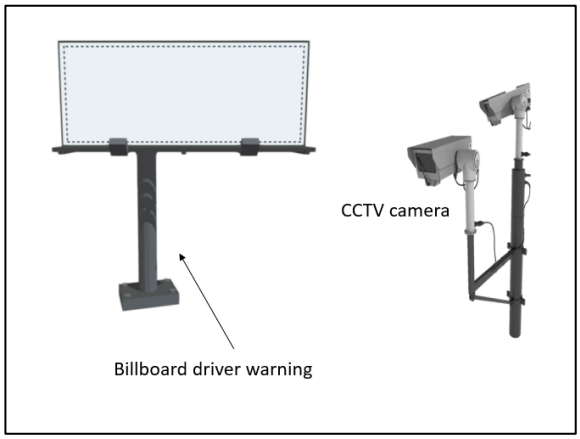


Figure 47. Assembly scheme of the embedded system

A video stream is captured from the camera, then an object detection algorithm is deployed, and a machine learning algorithm is used to classify the moving objects. If the moving objects are classified in the specified area as an animal dangerous to the safety of the driver and the car, a warning will be displayed directly on the controller then an alert will be sent to the driver sign (See Figure 48). To connect to the camera as discussed, real time streaming protocol (RTSP) will be used, it is a real-time streaming protocol that requires network connectivity for data to be transmitted in real time from multimedia to a server, which requires entering the camera's IP.

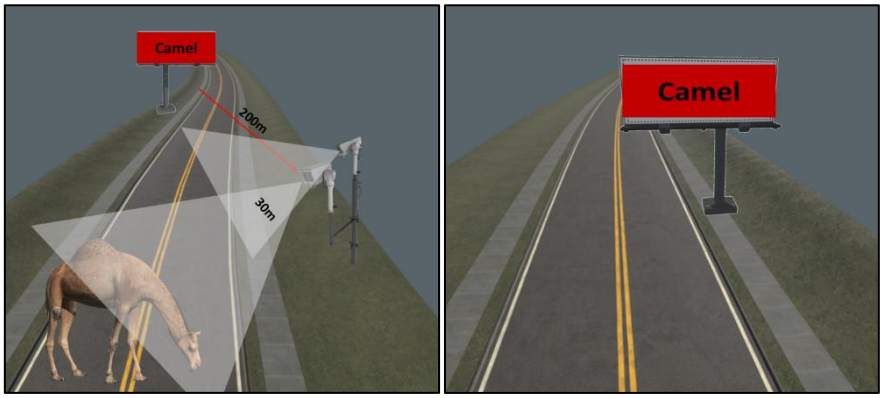


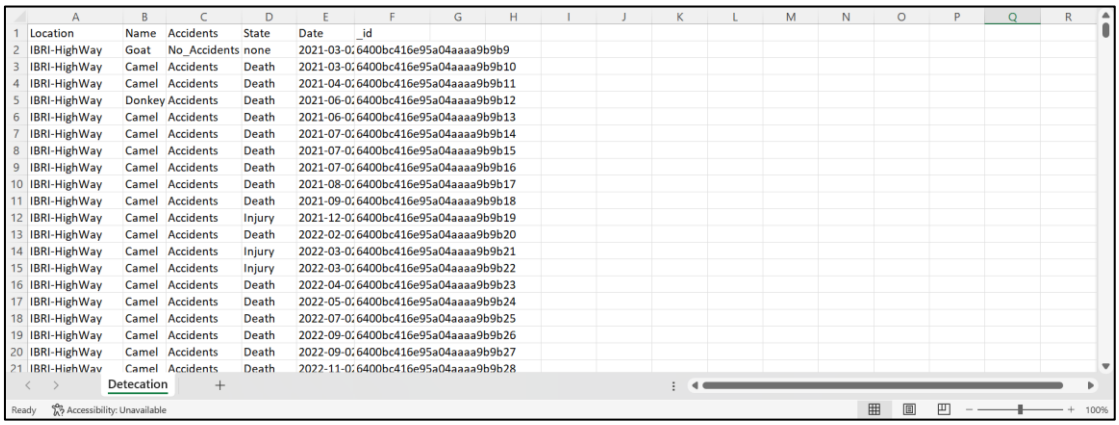
Figure 48. The ultimate road guard virtualization

4.3.7 Implementing Model to prediction accidents

To create the accident prediction process in this project, we will work with a data set that was taken from National center for statistical information. This data refers to accidents caused by animals on the road.

Data

Figure 49 shows the data that will be used in the process of predicting accidents. It is important, before proceeding with the prediction process, to carry out the so-called pre-processing, which is to determine which data will be used and which will be ignored, for the purpose of accuracy of the machine learning model. Bad data may change the model, causing the model to be deceived into using bad data, which will greatly affect the prediction.



	A	B	C	D	E	F
1	Location	Name	Accidents	State	Date	_id
2	IBRI-HighWay	Goat	No_Accidents	none	2021-03-0	6400bc416e95a04aaaa9b9b9
3	IBRI-HighWay	Camel	Accidents	Death	2021-03-0	6400bc416e95a04aaaa9b9b10
4	IBRI-HighWay	Camel	Accidents	Death	2021-04-0	6400bc416e95a04aaaa9b9b11
5	IBRI-HighWay	Donkey	Accidents	Death	2021-06-0	6400bc416e95a04aaaa9b9b12
6	IBRI-HighWay	Camel	Accidents	Death	2021-06-0	6400bc416e95a04aaaa9b9b13
7	IBRI-HighWay	Camel	Accidents	Death	2021-07-0	6400bc416e95a04aaaa9b9b14
8	IBRI-HighWay	Camel	Accidents	Death	2021-07-0	6400bc416e95a04aaaa9b9b15
9	IBRI-HighWay	Camel	Accidents	Death	2021-07-0	6400bc416e95a04aaaa9b9b16
10	IBRI-HighWay	Camel	Accidents	Death	2021-08-0	6400bc416e95a04aaaa9b9b17
11	IBRI-HighWay	Camel	Accidents	Death	2021-09-0	6400bc416e95a04aaaa9b9b18
12	IBRI-HighWay	Camel	Accidents	Injury	2021-12-0	6400bc416e95a04aaaa9b9b19
13	IBRI-HighWay	Camel	Accidents	Death	2022-02-0	6400bc416e95a04aaaa9b9b20
14	IBRI-HighWay	Camel	Accidents	Injury	2022-03-0	6400bc416e95a04aaaa9b9b21
15	IBRI-HighWay	Camel	Accidents	Injury	2022-03-0	6400bc416e95a04aaaa9b9b22
16	IBRI-HighWay	Camel	Accidents	Death	2022-04-0	6400bc416e95a04aaaa9b9b23
17	IBRI-HighWay	Camel	Accidents	Death	2022-05-0	6400bc416e95a04aaaa9b9b24
18	IBRI-HighWay	Camel	Accidents	Death	2022-07-0	6400bc416e95a04aaaa9b9b25
19	IBRI-HighWay	Camel	Accidents	Death	2022-09-0	6400bc416e95a04aaaa9b9b26
20	IBRI-HighWay	Camel	Accidents	Death	2022-09-0	6400bc416e95a04aaaa9b9b27
21	IBRI-HighWay	Camel	Accidents	Death	2022-11-0	6400bc416e95a04aaaa9b9b28

Detecation

Ready Accessibility: Unavailable 100%

Figure 49. Data samples

Data Selection

The following table shows the variables contained in the dataset were showed see figure 49.

Number	Variable	Type	Role
1	Location	Nominal / logical	Input
2	Name	Nominal / logical	Input
3	Accidents	Nominal / logical	Input
4	Status	Nominal / logical	Input
5	Date	Numeric	Not used
6	_id	Numeric	Not used

Table 6. Variables supplied

_id: This will not use because it will not provide us with any information in this project.

Name: This will be included because there is a serious discrepancy about the type of animal that will cause the accident, for example the percentage of accidents due to stray camels is expected to increase by 54%, compared to donkeys by 36%.

Date: We decided to remove this as it does not affect to the result.

Accidents: (accidents) contains nominal separate values for (accident) or (No accident) and this variable is useful because the prediction will revolve around whether this place will result in an accident or not.

Location: The variable location contains different and separate values for the type of area the animal is spotted in. This is especially useful because the difference in the location of the animal on the street will affect the result so it will be added.

Status: The condition variable (whether the person died or was injured during the accident) was omitted since this variable does not have a conclusive effect on the prediction process.

Building Logistic Regression Models

The logistic regression model was applied to previous data related to accidents in Oman, which were collected through police records and displayed on the official website of the National center for statistical information. In the data the (Accident) is the dependent variable, so 'accidents' will be a dichotomous variable with two categories, the presence of an 'accident' and 'No accident'. Therefore, animal traffic on the road was classified as having an accident or not. Due to the binary nature of this dependent variable, it was found that the logistic model is suitable for prediction. The results of logistic regression as used in our project are an excellent tool in providing explanations that can be used to reduce accidents due to stray animals.

Figure 50 shows the ROC curve for accidents and the other for non-accidents. The AUC curve helps us to know the ability of the model to distinguish both positive and negative results. The higher the AUC result, the better the model is in classifying the results.

```
y_test = y_test.map({'No_Accidents': 1, 'Accidents': 0}).astype(int)

import matplotlib.pyplot as plt
plt.title('Receiver Operating Characteristic')
plt.plot(fpr, tpr, 'b', label = 'ROC curve (area = %0.2f)' % roc_auc)
plt.legend(loc = 'lower right')
plt.plot([0, 1], [0, 1],'r--')
plt.xlim([0, 1])
plt.ylim([0, 1])
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```

Figure 51. ROC code

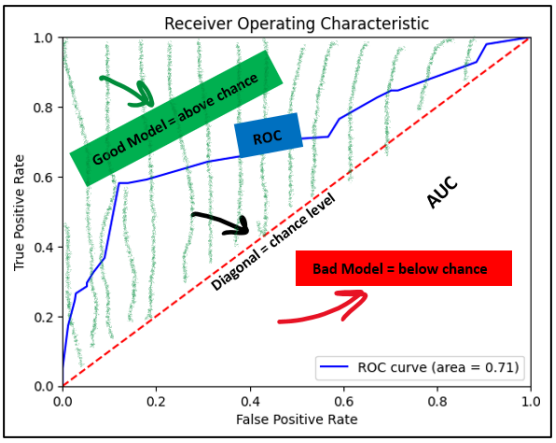


Figure 50. ROC graph in logistic regression

The ROC curve [33] is a graphical used to create a classification model that contains a binary classification, such as logistic regression, where it shows the trade-off between both the TP rate and the FP rate.

The main objective of building the ROC curve is to find out to what extent the classifier can expect to perform well in general. In Figure 50, the inclined line shows the random classifier. On the other hand, the ROC curve should be oriented in the upper left corner as far away from the diagonal line as possible.

To construct the ROC curve for Logistic Regression, we first need to obtain the expected probabilities of the test (Accident, No_Accident), then we change the classification threshold from 0 to 1 and calculate both the Positive rate and the corresponding False Positive at each threshold.

```
from sklearn.metrics import roc_auc_score, accuracy_score
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.datasets import load_breast_cancer

# Make predictions on the test set
y_pred = log_reg.predict(X_test)
y_pred_proba = log_reg.predict_proba(X_test)[:, 1]

# Calculate AUC and CA
auc = roc_auc_score(y_test, y_pred_proba)
ca = accuracy_score(y_test, y_pred)

# Print AUC and CA
print("AUC:", auc)
print("CA:", ca)
```

```
AUC: 0.7111135234235539
CA: 0.8195777351247601
```

Figure 52. Calculate AUC, CA logistic regression

AUC represents the area under the curve, and it is value ranges between 0 to 1. The model whose predictions are 100% wrong has a value of 0.0. On the other hand, the model's predictions are correct if the AUC ratio = 100%, meaning the value = 1.0. Through our project model, we obtained the value of AUC = 0.71, which means that the logistic regression model has good predictive power. On the other hand, our project obtained a classification accuracy (CA) = 0.81, which means that the classification model correctly predicted 81% of the samples in the group.

Best decision Logistic Regression result

```
# Calculate accuracy score
acc_score = accuracy_score(y_test, y_pred)
print("Accuracy Score:", acc_score)
# Calculate precision
precision = precision_score(y_test, y_pred, average='weighted')
print("Precision:", precision)
# Calculate recall
recall = recall_score(y_test, y_pred, average='weighted')
print("Recall:", recall)
# Calculate F1-score
f1 = f1_score(y_test, y_pred, average='weighted')
print("F1-score:", f1)
```

```
Accuracy Score: 0.8195777351247601
Precision: 0.7959827518968333
Recall: 0.8195777351247601
F1-score: 0.8017592409262875
```

```
from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

   Accidents       0.86      0.93      0.89       423
No_Accidents       0.53      0.33      0.41        98

    accuracy                           0.82       521
   macro avg       0.70      0.63      0.65       521
weighted avg       0.80      0.82      0.80       521
```

Figure 53. Best decision logistic regression

Accuracy Score: In general, accuracy is a measure of proximity to the target, where the accuracy percentage of the model was calculated by measuring the percentage of the sum of the true positive and correct negatives from all the predictions that were made. In our model we obtained an accuracy of good 0.81.

Precision: As explained in 4.3.3 section the accuracy measures the accuracy of positive predictions. Through our model, we obtained an accuracy = 0.79.

Recall: as mentioned in section 4.3.3, It helps us to discover the true positives. In our model, we got 0.81, and this means that the model is sinful in identifying positive cases.

F1 Score: F1 represents a measure of the accuracy of the model, as it takes both accuracy and memory into account. In the previous model, we got 0.8, as this analogy represents a better performance.

Confusion matrix

The confusion matrix represents the summary of the performance of the classification algorithm. The confusion matrix shows a clear picture of the performance of the classification model and the types of errors that it may produce if the predictions are correct or incorrect according to each category. The summary is represented in the form of a table.
As discussed in Section 4.3.3, there are 4 types of possible outcomes while evaluating the performance of the rating model, which are: [TP, TN, FP, FN].

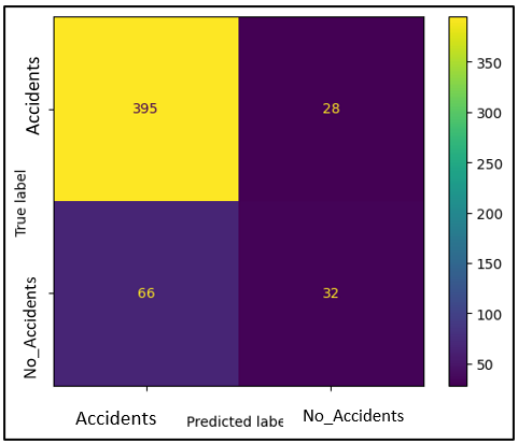


Figure 54. Confusion matrix for logistic regression

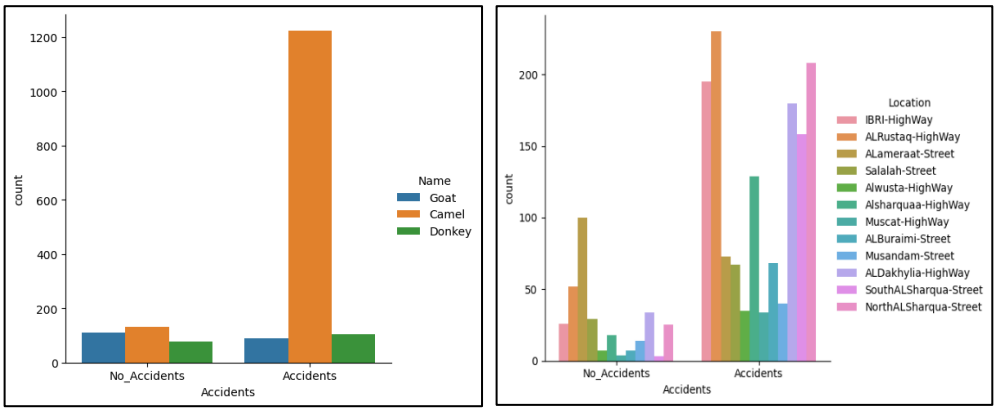


Figure 55. Graph in logistic regression

Figure 55 shows a description of the governorates most affected by animal accidents on the roads and shows the most types of animals that cause accidents in these governorates.

```
filename= 'weight_pred_model'
pickle.dump(log_reg, open(filename, 'wb'))
```

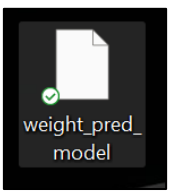


Figure 56. Download pickle prediction file

After completing the training of the logistic regression model, we use the pickle.dump() function to save the trained model in a file called “weight_pred_model”, here the model is ready to use for prediction.

Conclusion

Given that the 'accidents' variable is of a binary nature because it has only two categories (there is an accident or there is no accident), so the logistic regression was used to develop the prediction model for the Road Guard project. The aim was to provide a model that can be used to predict traffic accidents in Oman. From During the analysis of the data, the model revealed that the probability of accidents due to animals in Rustaq is high, as it was observed that the most animal causing accidents was the camel. This discovery may lead to a greater focus on Rustaq Governorate, which helps the ROP to impose a new speed law on that street in addition to adding a number of safety improvements the possibilities contained in this project can be used to determine priorities for reducing the risk of these accidents, for example, due to the possibility of a large number of accidents due to camels on Rustaq Road, two cameras should be added along Rustaq road instead of one camera, and a new law should be enacted to reduce speed on this road and to warn drivers through awareness programs.

4.4 Implementing the UI

In this section, the application discussed in section 3.3.3.1 will be explored, the main mechanics of implementing project components will be discussed and illustrated.

4.4.1 The Main GUI

Road Guard is divided into two main sections, a special section for server, and the other section is the client interface. The server section receives the stream of frames, this part implements computer vision algorithms for animal detection. Then it detects any animals around the road or in the road and then sends its results to the GUI. The main task of the server is to receive the feed from the camera, then send it after processing to the clients and provide them with the camera stream in the administrator user interface. Customers can be from the police, the municipality ministry. It also saves the names and types of animals with the date and time in its database in case the authorities need to display this data and enact some laws on drivers, for example, to adhere to a certain speed.

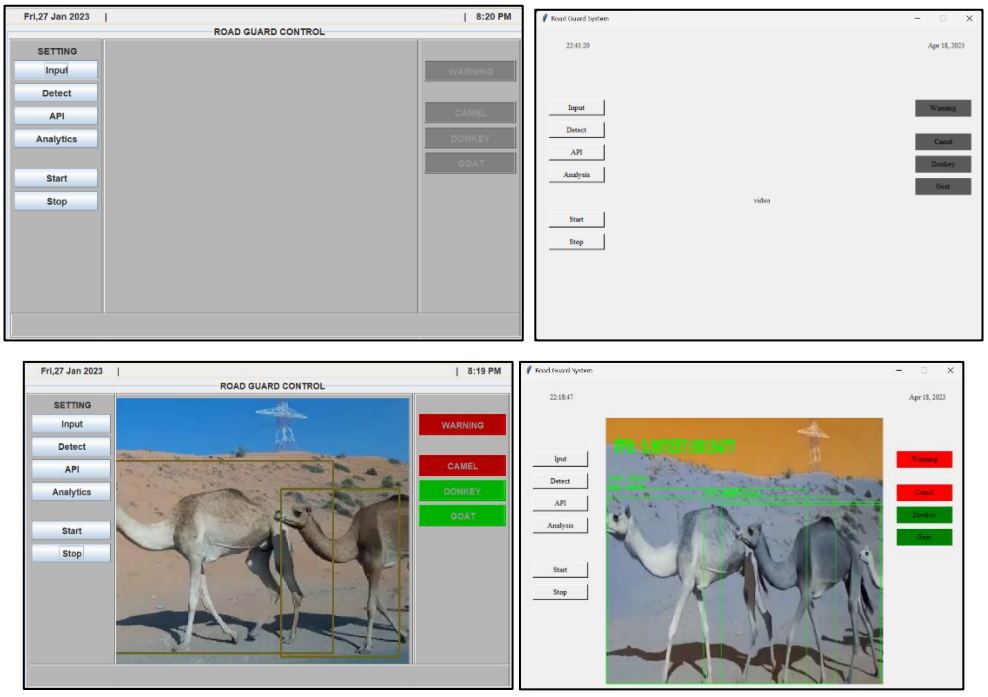


Figure 57. Java and python road guard controller

Figure 57 shows the final state of the main user page of the application, when the application is running, this will be the first page that will appear to the user in Java and Python. Once you press the start button, the camera feed will begin to flow, and once this process begins, the camera will start capturing frames and detecting animals. But for this step to take place, the user will be required to enter the source by pressing the (Input) button located at the top left of the main screen of the application. This option is intended to specify whether the application will detect animals through mp4 video or through a direct camera. Once you press the (Input) button, a screen will pop up allowing the user to enter the video source as shown in Figure 57.

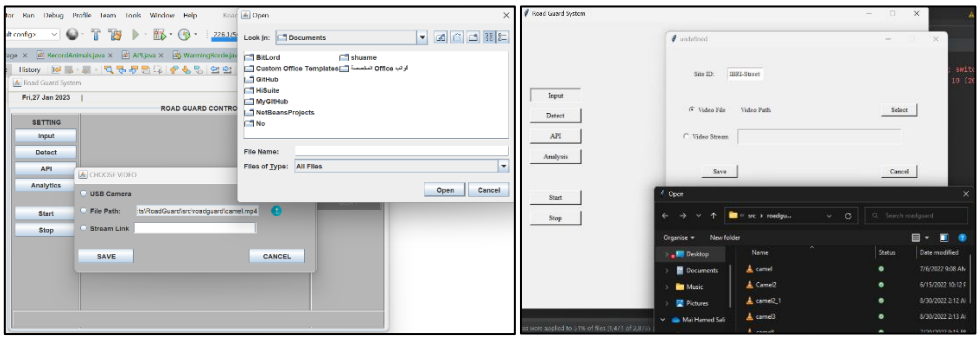


Figure 58. Input button in java and python

The next console screen shows the location of the animal from the video as well as the name of the animal that was detected.

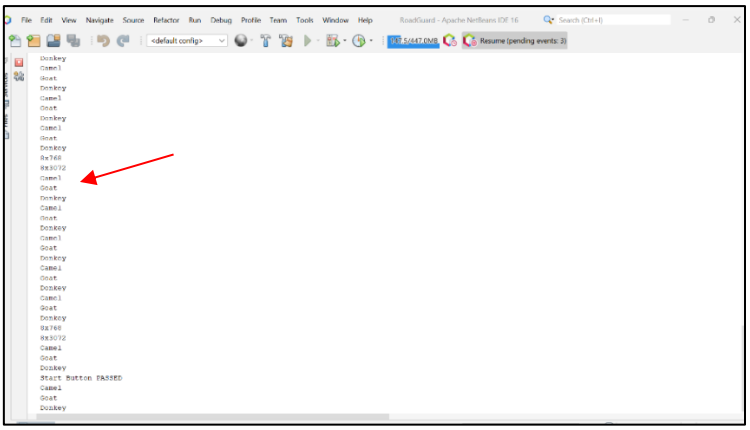


Figure 59. Console screen after detected animals in java

An exceptional feature has been added to the application, where the user can change the detector settings by adding his own discovery files by pressing the (Detect) button. When pressing the (Detect) button, a sub-screen will appear to the user Figure 60. This screen allows the inclusion of his detector's files. When pressing (Save), the application will directly read these files, meaning that the application will discover the files that were entered instead of the main file on which the application is running.

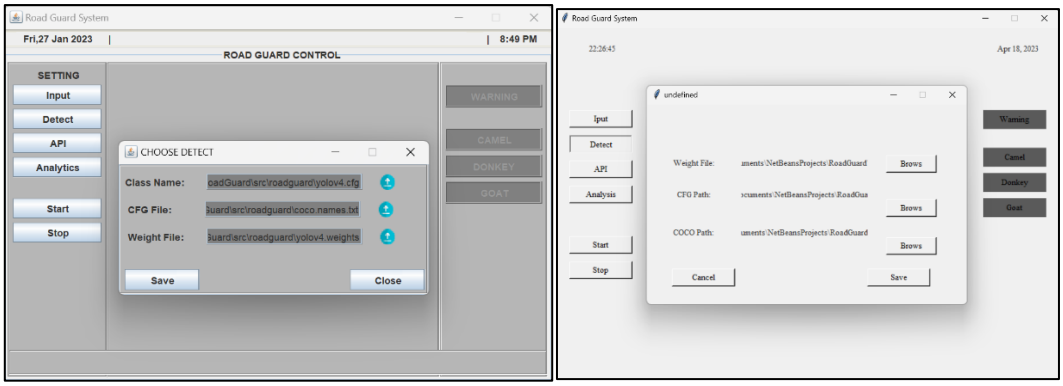


Figure 60. Detect button in java and python

API Connection Button

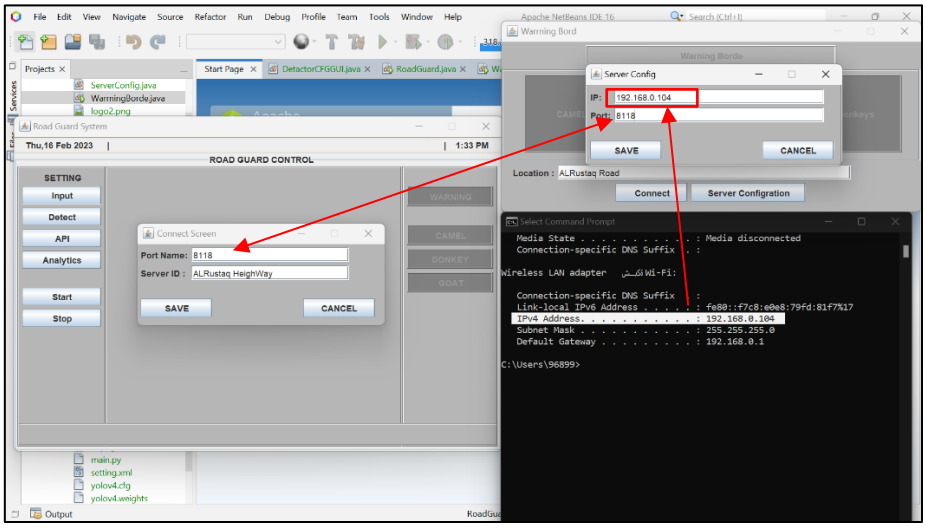


Figure 61. Connect driver sing with controller using IP

Figure 61 shows how the API button works. The API button connects the driver sign screen with the road guard screen, by adding the IP number of the device on which the Road Guard screen works and adding the port number that was added in the Road Guard screen. The administrator here can link the two screens. The Road Guard screen also contains a box for adding the name of the place where the camera will be placed, in contrast, in the driver screen, in the box for the location of the plate, the location of the plate is added (in Java).

Sending warnings from Road Guard screen into Driver signs.

A web server was used to link the driver's screen with the Road Guard screen. Each animal was given its own code to identify it during the linking process:

Animals name	Code
Camel	1
Donkey	4
Goat	2
Camel & Goat	3
Camel & Donkey	5

Goat & Donkey	6
All	7

Table 7. Animals code

When an animal is detected via a live camera or by adding a video, will note here that the symbols of these animals appear through the browser. If the symbol 1 appears, this means that the detected animal is a camel, so the driver's panel will light up with the name of the camel and so on.

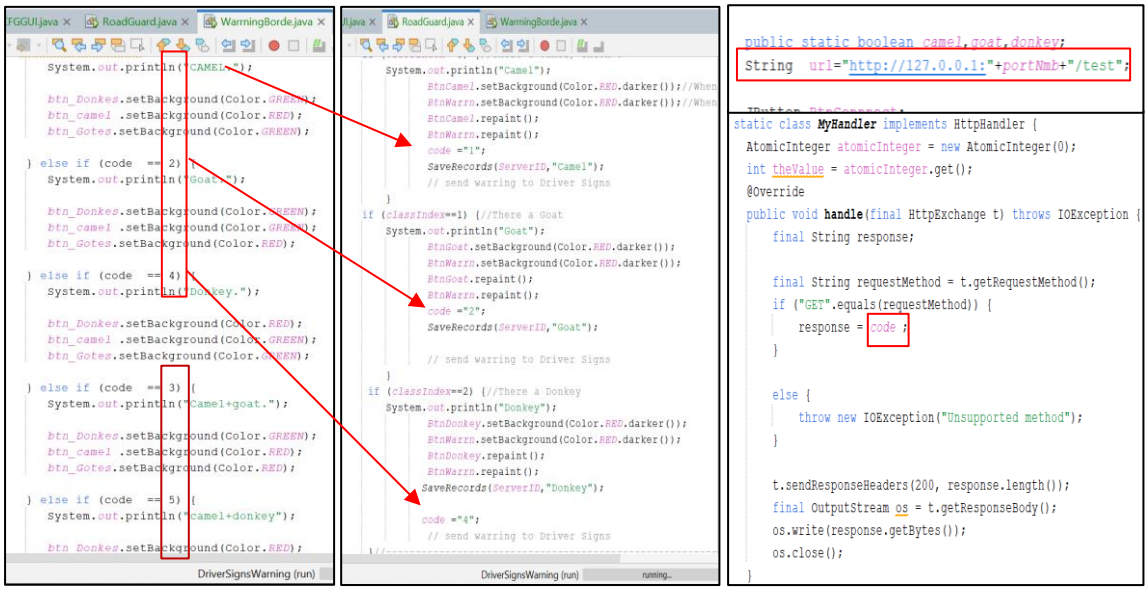


Figure 62. Create communication between controller and driver in java

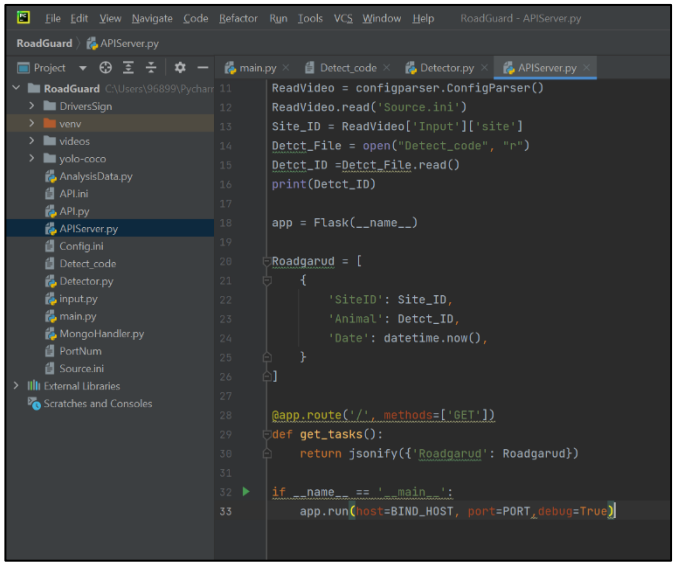


Figure 63. Create communication between controller and drivers in python

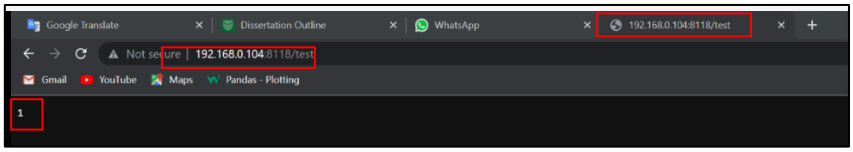


Figure 64. Camel code number showing in localhost using Java

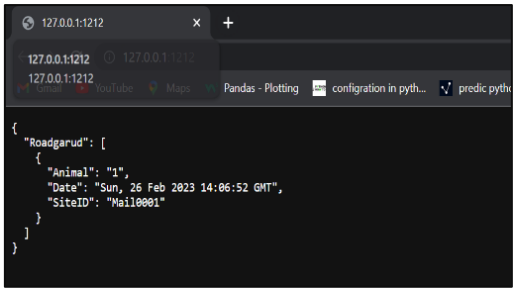


Figure 65. Camel info number showing in localhost using python

The picture on the left shows adding a wrong port to a valid IP, the no-access screen appears, also in the right picture adding a wrong IP to a valid port.

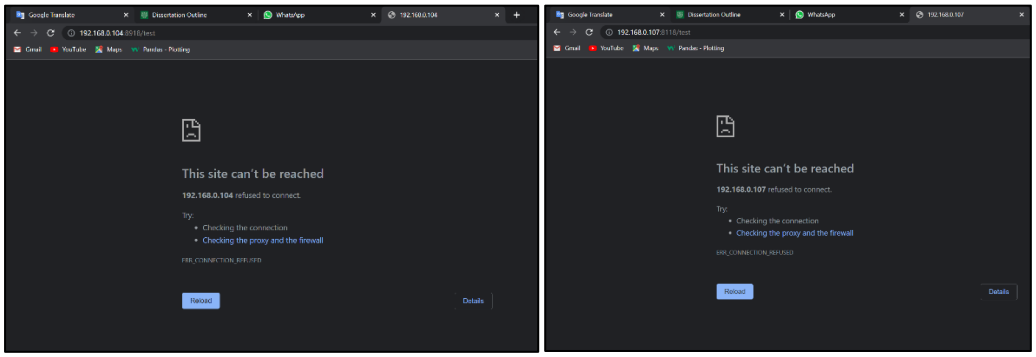


Figure 66. Entered wrong IP, port number

Data Analysis Button

In Figure 67, the data analysis page for the observed animals in the scene is shown. This page shows the distribution of the data according to their number and type. The user can search for a specific type of animal that was spotted in the scene by adding the date and time. Then he must specify whether the aim of the search is to display all data (all) or to display a specific type of animal, using (Option Menu), then the user presses on the (Show) button. The (location, ID, animal name, date, and time) are displayed in white space. At the bottom of the Dashboard, the number of animals is displayed at the bottom, in addition to a graph showing the number of animals that have been monitored in addition to their type. Two types have been added different from the graph as you see in Figure 67.

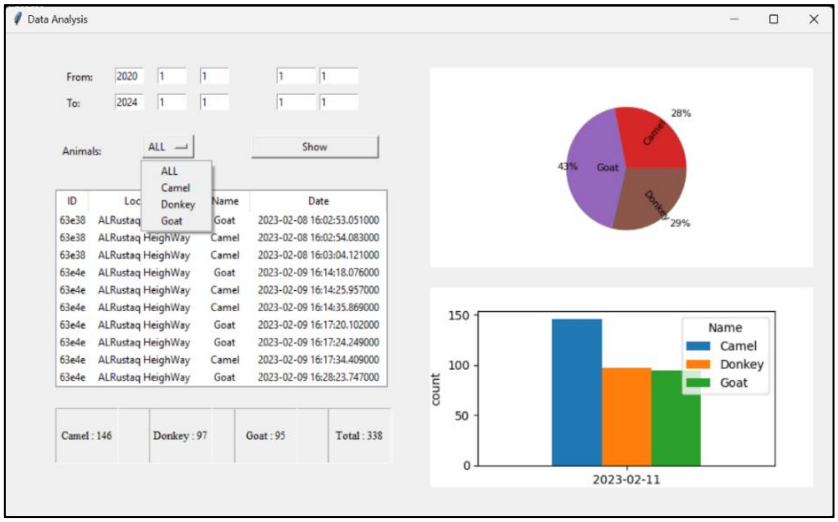


Figure 67. Bring data animals from database

Connected to MongoDB

As mentioned in section 3.4.1, MongoDB was chosen to create the road guard system database. This type of database was chosen because of its features of storing many data, flexible document schemas, and convenient design. Figure 68 shows the code for connecting Java and Python with the database. Through this encoding, both (Collection) and (Document) are created and connected to localhost.

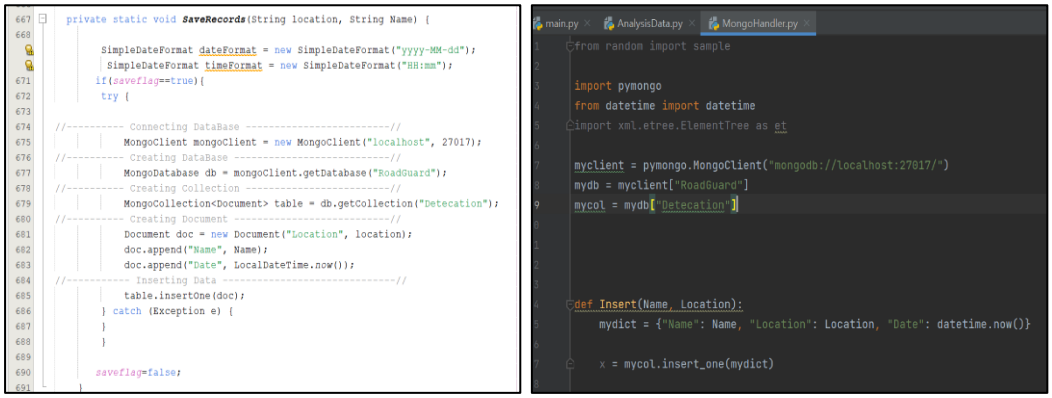


Figure 68. Connect to MongoDB in java and python

The following figure 69 shows the connection of the Road Guard system to the database through the console screen, where the result appears in red indicating that the application has connected to the database correctly.

Figure 69. Connect successfully to DB after detected animals in java

Figure 70 shows the MongoDB, which contains (database name: RoadGuard), (collection name: Detection), in addition to a group of the following files: [_id: unique number of each detection, Name: name of the animal that was detected, Location: location of the camera, Date: the date the animal was detected.

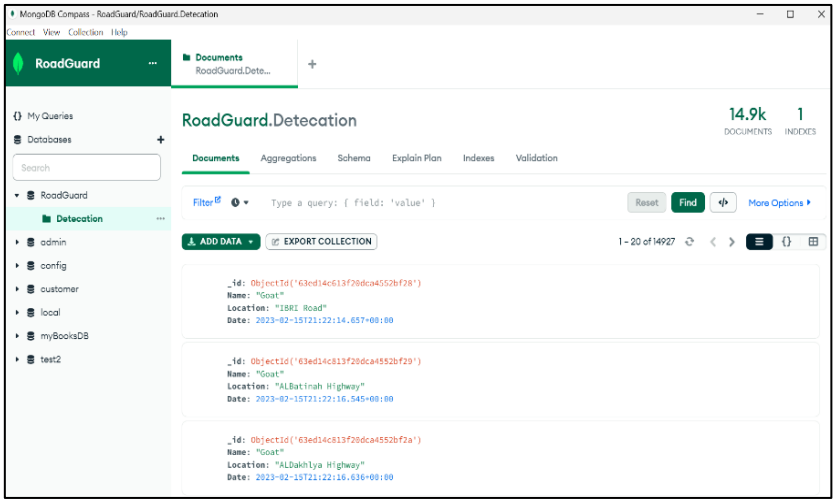


Figure 70. Animals' data displayed MongoDB

4.4.2 Light up for driver & Receive notification

We created an emulator that simulates the function of the driver board. Figure 71 shows the appearance of the emulator in both Java and Python and Java.

Drivers' lights only activate when a stray animal is detected near the road or on the road. On the opposite side, the light of the warning panel is change from red to green when no animal is detected. The figure also shows the output on the console screen, displaying the animal class with code. The animal is examined first, and if it is detected, its code is displayed with the name of the class.

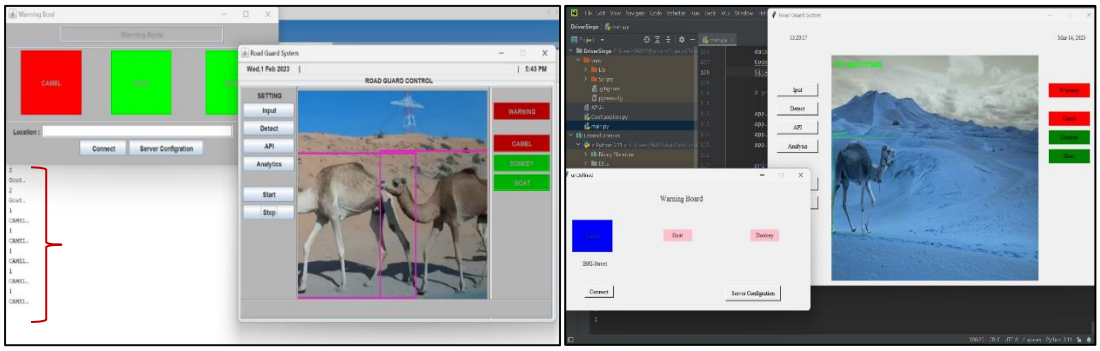


Figure 71. Connect driver sign with controller

Connect Road Guard with Drivers sign

Figure -72 shows the operation of the discovery after entering the port number and the IP number.

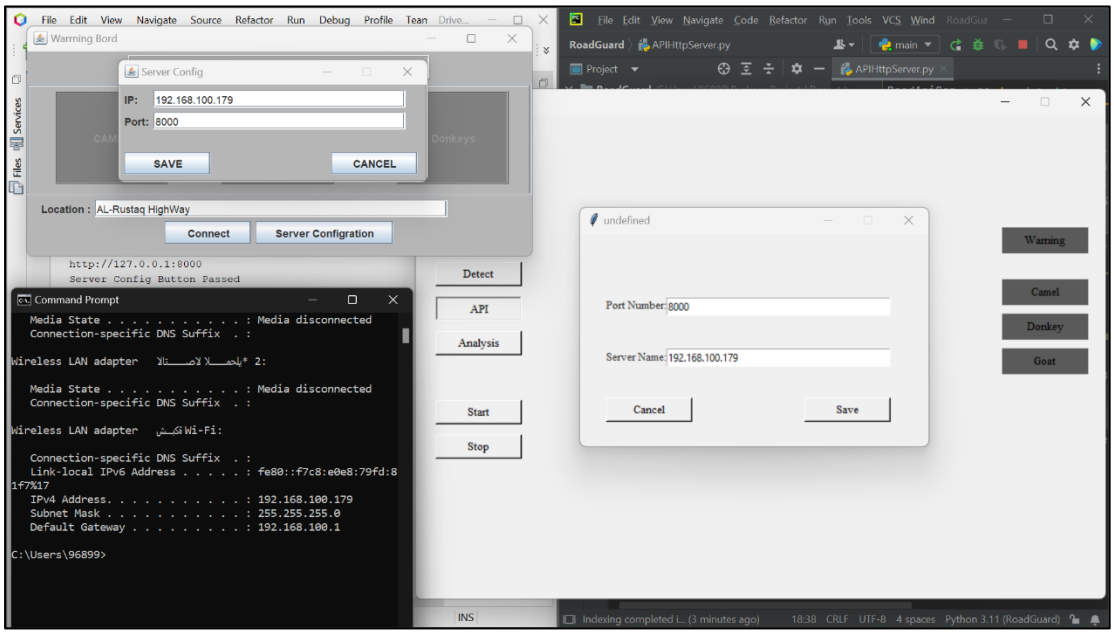


Figure 72. Connect driver sign with controller using IP, port in python

Console screen

Figure 73 shows the request and response by the driver's board and the road guard app. We have linked the java with python to show the request and response more clearly. When running, the driver's dashboard switches to sending requests to the Road Guard application. When an animal is detected, which has been numbered, it is recognized and sent to the driver's dashboard. The console screen shows both the request and the response.

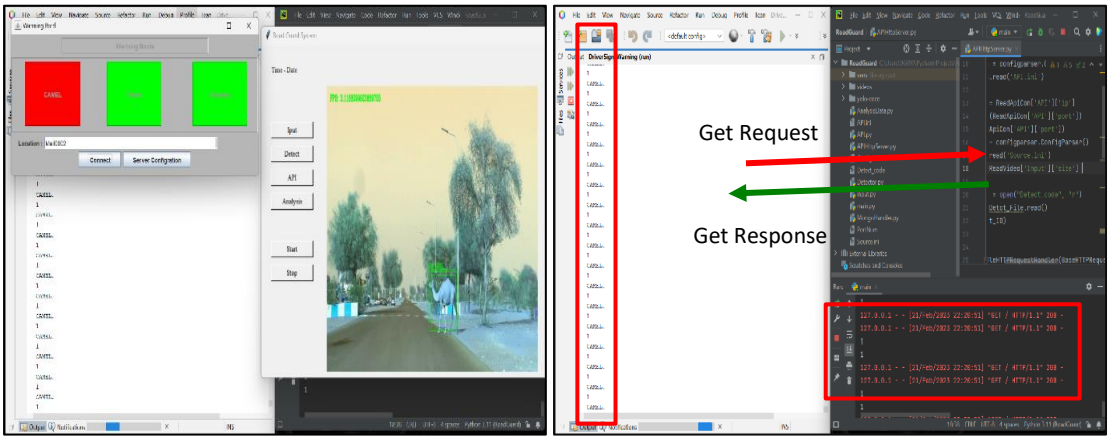


Figure 73. Request and response between python road controller and driver sign in java

4.5 Project Scenario

Example of exploiting the data that the Road Guard System will provide:

If a CCTV camera is placed on Rustaq Highway, and an alert driver sign is 200m before the CCTV, it will be able to detect (Camel, Donkey, Goat), and then the information's will help ROP to do the following:

1- Create a law that limits excessive speed on this road.

2- Create traffic light.

3- Check if there is an accident caused by these animals that maybe has not been reported, it will show by the CCTV stream in the road guard system control screen.

4- Predict the number of accidents.

Another example for road guard system for the admin:

Suppose the Ministry of Municipality needs to know the number of animals crossing on the Ibri highway:

1- The road guard will be able to calculate the total number of animals that have detected.

So that this information is useful for the employee of the ministry of municipal to know the following:

1- Animal type e.g., camel, crossing the road by hour/day/month, etc.., it is possible to know the number of the animal type.

2- Create speed breakers by knowing the area's most vulnerable to the presence of these animals.

Another example:

If the government needs to analyse a CCTV recording of any year in which these stray animals were spotted, the Road Guard system will be able to display the data analysis with graphs to save more time than human analysis.

4.6 Testing and Result Discussion

In this section, the application will be tested on two separate tests: the personal test, in which I will attach all my honest notes, and the external test, which will depend on the responses of specialists. The test is more concerned with the internal evaluation of the application than the external design.

4.6.1 Personal Testing

Before issuing the project and presenting it for user testing, a general test was applied to test the applicability of the project. The project was tested by LAPTOP-EHSLAVNG with core i5. The development of this project focused heavily on making it efficient and sending alerts quickly in addition to other features.

Thanks to my previous experience in creating a phone application to detect camels, I found that the current project is more efficient and has greater features. But nevertheless, I think that the program lacks some minor points, this is because the project must be completed within a certain period, so road guard system did not provide the best accuracy. These problems can be easily solved by adding more data and using a high-resolution camera.

The road guard test was conducted in an agile method, as tests were conducted on each unit after completion of its implementation, and then its functions were tested to verify its effectiveness. The project was verified by testing a variety of animal tests and verifying the accuracy of the road guard in detection.

1. Streaming Test

The stream test was concerned with two main functions:

• Receiving the Camera steam

• Sending the stream to the server

TESTCAM-ID1	Receiving Camera Streaming
purpose	Check the camera if work perfect
Prerequisite	Stream link should be connected to the server
Expected Results	Frame received and showed in GUI
Actual Results	Frame received and showed in GUI correctly
State	Pass
Issue	none
Test Date	19/3/2023

Table 8. Streaming camera test

2. Server module test

The server unit test focuses on testing animal detection in addition to testing sending notifications to the client if an animal is detected in the target area. The tests were as follows:

TESTMODL-ID2	Check detection animals' accuracy
purpose	Checking detecting (Camel, Goat and Donkeys) in the frames
Prerequisite	Detector files should be uploaded (CFG, COCO, weight)
State	Pass
Issue	None
Test Date	19/3/2023

Table 9. Server model test

3. Receiving warning alarm

TESTRG-ID3	Receiving alarm
purpose	Check receiving the warning in road guard screen
Prerequisite	1. Detector files should be uploaded. 2. Port number should be same with port number in driver board. 3. IP should be entered in driver board.
Expected Results	Animals is detected in road guard.
Actual Results	Animals detected and warning received successfully
State	Pass
Issue	none
Test Date	19/3/2023

Table 10. Receiving warning controller test

4. Sending warning alarm to the driver sign

TESTDRIVER-ID4	Sending alarm to the driver signs
purpose	Check sending the alarm from server to driver signs
Prerequisite	1. Detector files should be uploaded. 2. Port number should be same with port number in driver board. 3. IP should be entered in driver board.
Expected Results	Animals is detected in road guard GUI and sent to the driver signs
Actual Results	Alarm is sent to driver signs
State	Pass
Issue	none
Test Date	19/3/2023

Table 11. Sending warning to driver sign

5. Display information's into database.

TSTMODL-ID5	Display (Date, location, name of animals, Id in database)
purpose	Display detection info into database
Prerequisite	none
Expected Results	Display _id, Name, Location, and date
Actual Results	All required information's are displayed successfully
State	Pass
Issue	none
Test Date	19/3/2023

Table 12. Store animals' info in database test

6. Sending the data from database to analysis screen

TSTMODL-ID6	Sending the data from database to analysis screen
purpose	Check brings required output from database to dashboard
Prerequisite	none
Expected Results	Display all required data in dashboard
Actual Results	Display all required output in dashboard successfully.
State	Pass
Issue	none
Test Date	19/3/2023

Table 13. Display data from database to dashboard test

Test final system

For the final system to be tested, all Road Guard components must work together during the test, so the test will be as follows:

A. Upload video stream to road guard controller (video contain all animals that we mentioned before).

B. During the video analysis, the system will detect (Camel || Goat || Donkey), then will sending notification to admin in road controller and drivers.

C. During this time, road guard system will send each frame have animals in back-end server to store it into MongoDB.

D. After that, admin will be able to view show and analysis the data by using matplotlib.

Goals Achievement

From the previous test, we have success to achieve the following goals during working in this project. Some of parts did not success because of bad analysis but it is not critical point.

Goal	Achievement
Create Statistical data summary	Pass
Ability to handle RTSP	Pass
Update animals' type in Driver sign immediately	Failed
Update animals' type in road guard controller immediately	pass
Road guard system don't require a high expensive hardware.	Failed

Table 14. Goals Achievement

4.6.2 Professional Testing

I was able to get the opinion of experts in the field of AI, as I found a company interested in developing artificial intelligence programs in Muscat under the supervision of Mr. Talal. Mr. Talal and his team spent time using the Road Guard system. He provided me with some good points from his experience in artificial intelligence that can be addressed in the project, his overall observation was positive as Mr. Talal added some programming advice to develop the application. Mr. Talal and his team liked the fact that the app was created entirely by just one person with a fresh background in AI and computer vision. Mr. Talal also added that the application in general is smooth and understandable to the user and useful to drivers, and this is the point of establishing the Road Guard system in the first place.

5 Conclusion

5.1 Summary

In this dissertation, a road guard system connected to a video camera and a dashboard for drivers was presented. The system consists of camera modules which are placed next to the chosen roads. The main idea of the Road Guard System was to help drivers to reduce accidents caused by wild animals in Oman. When the system detects an animal near or on the road and classifies this animal as causing accidents, a warning is displayed on the road signs and an alert is sent to the admin to take this animal off the road. The selected models provided excellent detection accuracy, despite the high levels of positive detection in all groups, the Camels group show many undetected object FN compared with goat and donkey class. The main problem related to undiscovered objects is the most important problem that must be considered in the Road Guard system, and accordingly it needs some modifications that can be developed in future. On the other hand, the accuracy results of the goat category are less than the donkey's class, because the number of samples for goat class that used in classification is small. However, the test was conducted on video files.

5.2 Critical Evaluation

My own evaluation: As we mentioned earlier, the main objective of the project is to develop an integrated system capable of alerting drivers and the Ministry of Municipality & ROP, in addition to presenting data analysis and live video presentation. In this regard, we believe that the project has achieved its goals, as it was able to identify animals on the roads, send alerts, and create a special database for analysis. The most difficult challenge we faced while creating this application was that the project must meet all real-time requirements, there was no other choice but to submit to this request, otherwise the project will lose its purpose and core value. We encountered a variety of errors throughout the training period, and as a result we had to collect data continuously as more than 400 data related to these animals were collected. And then a label was added figure 30. After completing the training, we tested this data. An error was noticed because the extensions for all images were different, which forced us to re-collect the images and create a unified .png extension, and we trained again. The model was ready after 13 hours of continuous work, after that the OpenCV library was used to test the training by adding different MP4 video. Not all animals in all videos were accurately detected since it was impossible to collect massive data for these animals personally, so we must collect a lot of data with the same extension and the same size to have excellent accuracy.

It was discovered that an error appeared while training the data, as it was noticed that the mAP calculates 4 class of animals, but only 3 animals were trained! We did not find a solution, so we re-trained and downloaded the data again.

We had some problems in the development process and dealing with discovery files, finding a good weight file was difficult as we consumed a lot from time to time to recreate the discovery files several times, the reason is that when testing these files, some problems appear, such as insufficient images, or a problem with naming images, etc...

We would have liked to have been able to fix the wrong detect but unfortunately there was not enough time to fix this problem. On the other hand, we faced some problems when linking python detectors files in Java. The most prominent problems that represented a major obstacle to moving to the next steps was the inability of Java to read the detector files that were created using the Python language via Google Colab. It took time to discover the solution. To fix this problem, Java needs only a specific version of OpenCV to work, as All versions do not work except version 4.5.3

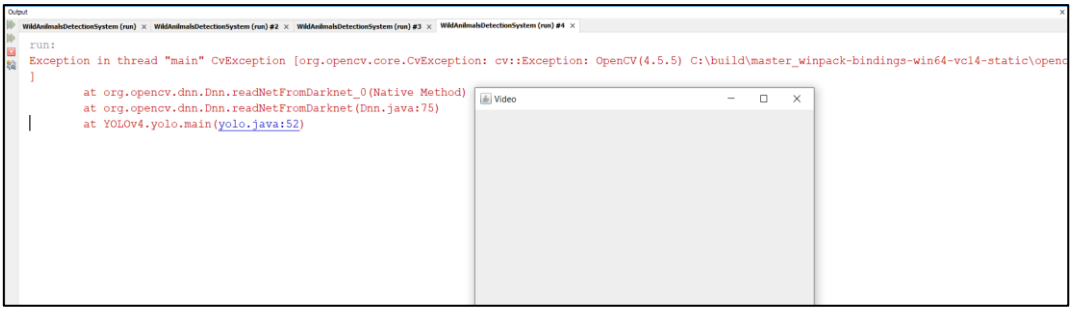


Figure 74. Issue to connect OpenCV in Java

There is a major problem that was discovered during the test, which is the problem of slowing down the video stream, which led to a slowdown in sending alerts to the driver board, as well as an increase in the percentage of unjustified false detection from video. When we tried to solve the problem, we found that the main reason is that the Java language is not good to use it in the Yolo algorithm, because of this problem, it created many problems. To fix this problem, we switched directly to the Python language and created the same project. All problems caused by the Java language have been fixed in python, and the overall accuracy and speed are excellent. As a result, the differences, and problems that we noticed between the two languages to create this project are as follows:

Language	**Python**	**Java**
Packages	Easily to install	We spend time to find perfect version for OpenCV package
Video processing	Faster	Slower
Code	Short	Long
Developing	Easy	Very hard
Complex Codes	Not complex	complex

Table 15. Main deference between Java and python

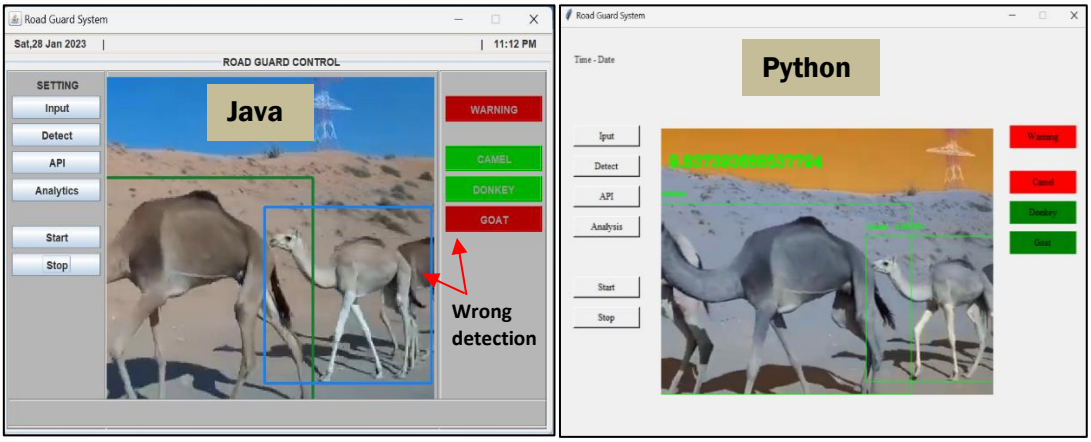


Figure 75. Road guard controller in java and python

In Figure 75, a comparison between the detection of the same scene for both Java and Python, it turns out that there is a (false positive) detection in the Java language compared to a (true positive) detection in the same animal in the Python language.

Also, it was noticed that there was a big slowdown when integrating the Python Data Analysis file with Java. When the Analysis button is pressed in the Java user interface; It takes some time for the analysing page that was built by the Python language to appear, which makes the

application not smooth for the user, so this was one of the reasons for moving from the Java language to the Python language.

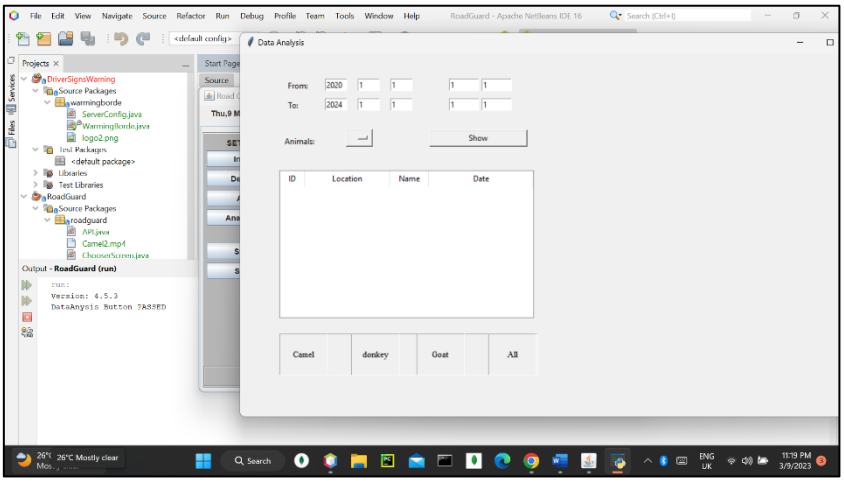


Figure 76. Data analysis python script in java

Another discovery, regarding the driver's board, when connected to Road Guard Control board and entering the port number and the IP number, the animal code will show in output as mentioned in 4.4.1, as these numbers may often be incomprehensible, so, we modified this option to represent the output in JSON format to be clearer.

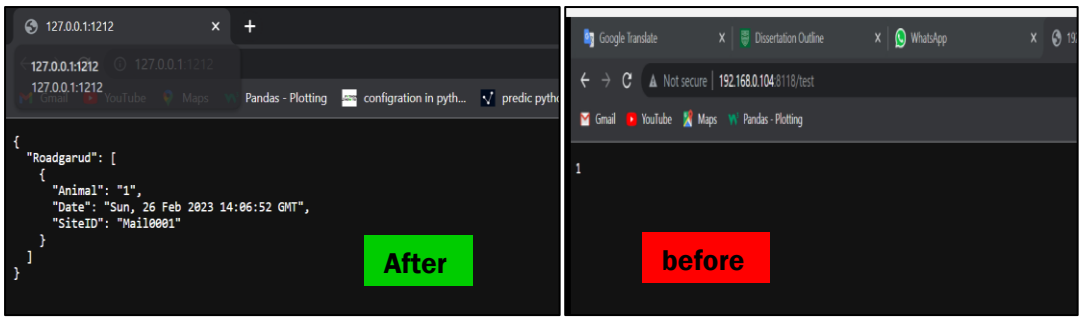


Figure 77. Output detected python Vs Java

Method Evaluation

The evaluation for road guard system will be as following:

- On Road Guard Analysis

In evaluation system, road guard must detect three types of animals: [Camel, Goat, Donkey], animals' detection worked as expected. So, in csv files we must found these animals stored there with details.

- Data Analysis from MongoDB

In this point, the admin must be able to show all the collected data from database except _id and then view it in dashboard. Admin must be able to filter the data. All thing in this point worked as expected.

- Driver's Alert

The driver's alert must have same 'IP, Port' number in road guard controller, so if the road guard detects animals, it will send alert into driver's sign. All thing in this point worked as expected.

Professional Evaluation Mr. Talal and his team liked the idea of the Road Guard system and confirmed that the system is useful. The overall comments were positive and encouraging. However, there were some points about the Road Guard system that Mr. Talal was not sure about, for example: detecting animals during the night. He also added the camera coverage point, the camera must have a large coverage, in addition to measuring the appropriate distance between the camera and the plate, considering the speed of the driver on the road. Mr. Talal also stressed the need to control the discoveries more to avoid any false communication. Mr. Talal also suggested using a type of camera that includes the entire detection area to solve the problem of classification (false positives).

Reflection: In general, I am happy with the result of this project, as I have fulfilled all the requirements that were listed in Section 1.2 despite the limited time, I tried as much as possible to handling all available problems within this specified time. If there is a goal to complete this project, I will try to add more powers and other features and fixes some problems.

5.3 Limitations and Future Work

As a future work, the animal detection model will be improved with a more accurate model linked to CCTV cameras. We may also want to add new features, for example notify the nearest hospital & ROP in the event of an accident because of crossing animals. We can also update the database with accident record information caused by animals to be used in enacting new laws such as reducing speed on the road. The structure of the Road Guard system is extendable and adjustable. Having such a Road Guard System can be of great importance to the authorities and car owners. In the upcoming updates of the road guard, it will be possible to significantly increase the data set by obtaining new images, in addition to enhancing reinforcement learning so that road tests can be conducted. In addition, we can add other types of animals for training, according to local necessity. As mentioned earlier, there are some limitations to the success of the application, which is that there is a wrong identification of the type of animal. This problem should have been addressed in the Problems and Analysis section, but due to lack of time, this problem will be added to future updates to the application.

6.References

[1] S. A. Shaibany, "Glow in the dark camels: keeping you safe on the roads of Oman," *The National*, Aug. 02, 2017. https://www.thenationalnews.com/world/gcc/glow-in-the-dark-camels-keeping-you-safe-on-the-roads-of-oman-1.616414.

[2] Z. A. Nasseri, "Fines up to RO100 on owners of stray animals," Oman Observer, Jul. 26, 2020. https://www.omanobserver.om/article/11685/Main/fines-up-to-ro100-on-owners-of-stray-animals.

[3] Y. Ebuen, "Camels vs vehicles: How many have died in five years?," Oman Observer, May 12, 2020. https://www.omanobserver.om/article/13332/Main/camels-vs-vehicles-how-many-have-died-in-five-years.

[4] D. Chaudhary, "Multithreads-NITK-HackVerse," GitHub, Oct. 20, 2020. https://github.com/dheeraj-2000/Multithreads-NITK-HackVerse.

[5] Mohit, "Autonomous-Forest-Surveillance-Safety-System-using-OpenCV," GitHub, Dec. 05, 2022. https://github.com/mohit9949/Autonomous-Forest-Surveillance-Safety-System-using-OpenCV.

[6] K. Erdem, "Farm Animal Tracking Project," GitHub, Jan. 26, 2023. https://github.com/burnpiro/farm-animal-tracking.

[7] G. Baimuratov, "gaiar/animal-detector," GitHub, Jan. 05, 2023. https://github.com/gaiar/animal-detector.

[8] G.-W. Zhang, "Track-and-Control(TraCon)-Toolbox," GitHub, Feb. 23, 2023. https://github.com/GuangWei-Zhang/TraCon-Toolbox.

[9] K. Team, "What is Agile?," blog.kintone.com. https://blog.kintone.com/business-with-heart/what-is-agile-development.

[10] T. Stober and U. Hansmann, Agile software development : best practices for large software development projects. Berlin: Springer,.

[11] R. S. Shankar, L. V. Srinivas, P. Neelima, and G. Mahesh, "A Framework to Enhance Object Detection Performance by using YOLO Algorithm," IEEE Xplore, Apr. 01, 2022. https://ieeexplore.ieee.org/document/9760859.

[12] "Intersection over Union (IoU) for object detection," PyImageSearch, Nov. 07, 2016. https://pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/.

[13] P. Sharma, "A Practical Guide to Object Detection using the Popular YOLO Framework - Part III (with Python codes)," Analytics Vidhya, Dec. 06, 2018. https://www.analyticsvidhya.com/blog/2018/12/practical-guide-object-detection-yolo-framewor-python/#:~:text=The%20biggest%20advantage%20of%20using.

[14] L. Tan, T. Huangfu, L. Wu, and W. Chen, "Comparison of RetinaNet, SSD, and YOLO v3 for real-time pill identification," BMC Medical Informatics and Decision Making, vol. 21, no. 1, Nov. 2021, doi: https://doi.org/10.1186/s12911-021-01691-8.

[15] Carmelo Maria Torre, Osvaldo Gervasi, Beniamino Murgante, and Sanjay Misra, Computational science and its applications Pt. 2. Berlin Heidelberg Springer, 2013.

[16] G. Bradski and A. Kaehler, Learning OpenCV. "O'Reilly Media, Inc.," 2008.

[17] "Run NoSQL Database in the Cloud or On-Premises," Oracle.com, 2023. https://www.oracle.com/database/nosql/

[18] J. Gosling, B. Joy, G. Bracha, and G. Steele, The Java Language Specification. Addison-Wesley Professional, 2005.

[19] M. Pilgrim, Dive into python : [a guide to the python language for programmers]. United States? Soho Books, 2009.

[20] W. Rowe, "What Is a Neural Network? An Introduction with Examples," BMC Blogs. https://www.bmc.com/blogs/neural-network-introduction/#:~:text=Neural%20networks%20are%20designed%20to

[21] J. Brownlee, "Difference Between a Batch and an Epoch in a Neural Network," Machine Learning Mastery, Aug. 09, 2022. https://machinelearningmastery.com/difference-between-a-batch-and-an-epoch/#:~:text=The%20batch%20size%20is%20a%20number%20of%20samples%20processed%20before

[22] Genesis, "Learning Rate: Neural Network," From The GENESIS, Apr. 12, 2020. https://www.fromthegenesis.com/learning-rate-neural-network/

[23] "Mean Average Precision (mAP) Explained: Everything You Need to Know," www.v7labs.com. https://www.v7labs.com/blog/mean-average-precision#:~:text=Mean%20Average%20Precision(mAP)%20is%20a%20metric%20used%20to%20evaluate

[24] "Hasty.ai," Hasty.ai. https://hasty.ai/docs/mp-wiki/metrics/iou-intersection-over-union#:~:text=To%20define%20the%20term%2C%20in

[25] M. Basavarajaiah, "6 basic things to know about convolution," Medium, 29-Mar-2022. [Online]. Available: https://medium.com/@bdhuma/6-basic-things-to-know-about-convolution-daef5e1bc411.

[26] "Convolution," NVIDIA Developer, 24-Sep-2018. [Online]. Available: https://developer.nvidia.com/discover/convolution#:~:text=The%20convolution%20algorithm%20is%20often,convolution%20operation%20is%20called%20deconvolution.

[27] "Papers with Code - Average Pooling Explained," paperswithcode.com. https://paperswithcode.com/method/average-pooling#:~:text=Average%20Pooling%20is%20a%20pooling

[28] "Fully Connected Layer vs Convolutional Layer: Explained | Built In," builtin.com. https://builtin.com/machine-learning/fully-connected-layer

[29] "Multi-Class Neural Networks: Softmax | Machine Learning Crash Course," Google Developers. https://developers.google.com/machine-learning/crash-course/multi-class-neural-networks/softmax#:~:text=Softmax%20extends%20this%20idea%20into

[30] "Learn about bounding boxes in image processing | Nanonets," Nanonets AI & Machine Learning Blog, Aug. 25, 2022. https://nanonets.com/blog/image-processing-and-bounding-boxes-for-ocr/.

[31] P. Huilgol, "Precision and Recall | Essential Metrics for Data Analysis (Updated 2023)," Analytics Vidhya, Sep. 03, 2020. https://www.analyticsvidhya.com/blog/2020/09/precision-recall-machine-learning/#:~:text=Precision%20and%20recall%20are%20two.

[32] "What is Ground Truth in Machine Learning? | Domino Data Lab," www.dominodatalab.com. https://www.dominodatalab.com/data-science-dictionary/ground-

truth#:~:text=Ground%20truth%20in%20machine%20learning%20refers%20to%20the%20reality%20you.

[33] Google, “Classification: ROC Curve and AUC | Machine Learning Crash Course,” Google Developers, 2019. https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc

End.