DETECTION OF TRAFFIC RULE VIOLATION USING AUTONOMOUS DRONE

PROJECT (PHASE-I) REPORT

submitted by

HEMANTH MOHAN

TVE21ECRA07

to

APJ Abdul Kalam Technological University

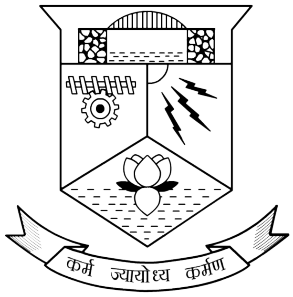
in partial fulfilment of the requirements for the award of the Degree

of

Master of Technology

in

Robotics and Automation



DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

College of Engineering Trivanandrum

Thiruvananthapuram

December 7, 2022

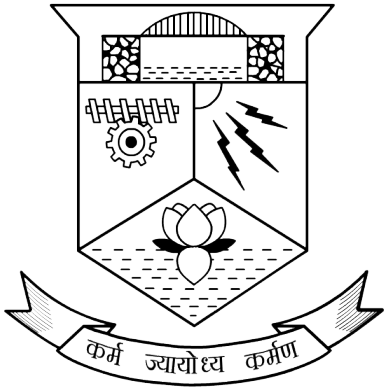
DECLARATION

I undersigned hereby declare that the project report ("DETECTION OF TRAFFIC RULE VIOLATION USING AUTONOMOUS DRONE") submitted for partial fulfillment of the requirements for the award of degree of Master of Technology of the APJ Abdul Kalam Technological University, Kerala, is a bonafide work done by me under supervision of Dr. Deepak S, Assistant Professor, Department of Electronics and Communication Engineering. This submission represents my ideas in my own words and where ideas or words of others have been included, I have adequately and accurately cited and referenced the original sources. I also declare that I have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma or similar title of any other University

Trivandrum

December 7, 2022 HEMANTH MOHAN

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
COLLEGE OF ENGINEERING TRIVANDRUM
THIRUVANANTHAPURAM - 16



CERTIFICATE

This is to certify that this project report entitled **"DETECTION OF TRAFFIC RULE VIOLATION USING AUTONOMOUS DRONE"** is a bonafide record of the work done by **HEMANTH MOHAN (TVE21ECRA07)**, under our guidance towards partial fulfilment of the requirements for the award of the Degree of **Master of Technology in Electronics and Communication with specialization in Robotics And Automation**, of the A P J Abdul Kalam Technological University during the year 2021-2023.

Dr. Deepak S
Assistant Professor
Dept. of ECE
(Project Guide)

Dr. Binu L S
Professor
Dept. of ECE
(Project Coordinator)

Dr. Sreelatha G
Professor
Dept. of ECE
(PG Coordinator)

Dr. Ajayan K R
Professor
Dept. of ECE
(Head of the Department)

ACKNOWLEDGEMENT

A lot of efforts and hard work has been put into this project in course of its development. However, it would not have been possible without the kind support and help of many individuals and other sources. I would like to extend my sincere thanks to all of them.

I thank God Almighty for guiding me throughout the project. I am highly thankful to **Dr. Suresh Babu V**, Principal, College of Engineering, Trivandrum, **Dr. Ajayan K R**, Professor, Head of the Department of Electronics and Communication Engineering, College of Engineering, Trivandrum and **Dr. Sreelatha G**, PG coordinator, Department of Electronics and Communication Engineering for providing me an opportunity to present the project at its best.

I would like to express my sincere gratitude to my project guide **Dr. Deepak S**, Assistant Professor, Electronics and communication Engineering, project coordinators **Dr. Binu L S**, Professor, Department of Electronics and Communication Engineering, and PhD scholar **Sreedivya R S** for all necessary help extended to me in the fulfilment of this work.

Above all I would like to thank my parents, without whose blessings I would not have been able to accomplish my goal. Above all, I thank the Almighty for the immense grace and blessings.

HEMANTH MOHAN

ABSTRACT

In many nations, the number of bike riders has been rising quickly over time. Due to a variety of factors, including their economic importance, motorcycles are preferred by residents from all social classes. According to the standard, wearing a helmet is required, although the vast majority don't. The protection of the riders is one of the helmet's main objectives. Seatbelt use in cars promotes passenger safety in a similar manner to the use of helmets, yet many people choose not to do so. Along with not wearing safety equipment, people these days also break other traffic laws including cutting across lanes, driving too fast, and overtaking in hospital and school zones, among others. There is a need for automatic detection of traffic rule violations because the process is cumbersome and manual, even though there are police forces that support the detection by issuing traffic violation tickets. Therefore, the project's main goal is to develop and use autonomous drone technology for the automatic identification of traffic rule violations.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

CNN	Convolutional Neural Network
YOLO	You Only Look Once
DL	Deep Learning
R-CNN	Region Based Convolutional Neural Network
OCR	Optical Character Recognition
DNN	Deep Neural Network
RRT	Rapidly Exploring Random Tree
RL	Reinforcement Learning
ANPR	Automatic Number Plate Recognition

CHAPTER 1
INTRODUCTION

Instead of obeying traffic laws and wearing safety gear like helmets and seatbelts, they only concentrate on speeding and passing other vehicles on the highway. The number of bike riders has been increasing quickly over time in various countries. Motorcyclists of all socioeconomic strata prefer them for a number of reasons, including their economic significance. The majority of people don't wear helmets, even though it is mandated by the standard. One of the helmet's primary goals is to protect the wearer. Similar to wearing a helmet, wearing a seatbelt in a car increases passenger safety, yet many people opt not to.The primary causes of fatalities and catastrophic injuries include overtaking, speeding, and regular behaviour of those who do not use safety equipment. These causes account for almost 90% of all accident-related fatalities.

Due to this reasons Surveillance cameras are placed at specific places, especially near accident-prone areas, school zone , hospital zone etc. Since they are placed permanently at specified places people often try to fake the act of obeying the traffic rules. That is, people wear helmets or seatbelts as they approach near the surveillance camera or they try to evade the cameras by finding other routes or slow down the speed of the vehicle as they approach near the cameras. So there is a need for detection of traffic rule at random places. So the main idea of the project is to design and implement an drone that can detect traffic rule violations at random places automatically.

1.1 PROBLEM IDENTIFICATION

The existing Surveillance Detection system has several disadvantages. Especially in countries like India there are main roads as well as several byroads, so implementing survillence systems on all roads and byroads are not cost effective. Another reason is that since these systems are fixed permenantly at specific places people often has the exact knowledge of the positions of the survillence systems and they often fake the act of wearing safety equipments or obey traffic rules only when they are in close proximity. Thus there is a need for detection of these traffic violations at random places automatically.

1.2 OBJECTIVES

Since the main focus is on detection of traffic violation at random places automatically, the main goal of the project is to design and implement an autonomous drone that can travel to random locations automatically. This autonomous drone plans a path for its travel to random locations and is incorporated with safety equipment detection algorithm to detect the traffic violators during its travel.

1.3 THESIS OUTLINE

A detailed literature survey on different deep learning approaches for detction of traffic rule violation are included in Chapter 2. Chapter 3 explains the methodology of the proposed system. Implementation and experimental details of the proposed system is presented in Chapter 4. The experimental results are included in Chapter 5. The report is concluded in Chapter 6.

CHAPTER 2
LITERATURE SURVEY

This chapter describes a detailed review of research work in the area of different methodologies in detection of traffic rule violation and different path planning algorithm for automating the drone control.

2.1 RELATED WORKS

For traffic rule detection Aniruddha tonge, Usman Khan et al. [1] proposed a helmet and crosswalk detection system. The proposed system takes video input from CCTV Cameras.Vehicle detection from the image frame was carried out using the YOLO technique. For the detection of helmet violations, a CNN-based classifier was used, whereas Mask RCNN was used to identify crosswalk violations. The key benefit of the suggested system was that it could detect more violations than a system requiring human intervention and that it offered an entirely autonomous system. The primary flaw in this method is the difficulty in detecting and identifying the licence number from fancy number plates.

Adhikari, Himanshu et al. [2] proposed the development of a system that can detect moving objects, classify vehicles, detection of helmets and identifies license plates based on machine learning algorithms such as SVM and DNN. The suggested technique provides better detection outcomes than SVM. By giving more training data during training, the low detection accuracy was enhanced. However, the suggested system is unable to recognise many vehicles at the scene, and it also has very poor accuracy when attempting to extract the licence plate number.

Convolutional Neural Network-based image classification and pre-trained real-time object detection models was used by Rohit, C.A. et al. [3] to distinguish between people who are wearing and not wearing helmets in traffic camera video sequences. Caffe was the pre-trained model that was employed. The caffe model's primary benefit is its quick detection. Due to the model's emphasis on speed, the detection accuracy was only about 84%, and there was no image text classifier to identify the number plates.

Madhuri, Praveen Kumar et al. [4] proposed a CCTV with speed braker system.

The footage can be recorded because CCTVs are placed at specified checkpoints. Speed limiters are used at these checkpoints to provide drivers adequate time to capture and identify the vehicle on camera. Only helmets were targeted for detection. The suggested system employs the YOLO detection algorithm, which is quicker and more effective than CNN and has a detection accuracy of above 90%. The algorithm's tendency for significant localization errors was this approach's principal flaw.

Meenu R, Sinta Raju et al. [5] proposes a framework for real time detection of traffic rule violators using the video obtained from the CCTV footage. The R-CNN algorithm served as the foundation for the system, which used Open-ALPR to recognise licence plates and detect people without wearing helmets. The accuracy attained during the detection was high (92%) and this approach produces a faster detection rate than other deep networks like CNN and YOLO.

For autonomous navigation of UAV, a mission-oriented path planning algorithm built on a Reinforcement Learning framework was suggested by Islam, Shafkat, et al. [6] to guide the drone to its destinations. The A* algorithm is used for path planning. Instead of employing traditional function approximation techniques, the approach resolves RL problems in continuous space. The key benefit of this method is that UAVs can avoid collisions with fixed and moving obstacles in their route and can also reposition themselves as the location of the target region changes. The algorithm also enables UAVs to avoid overlapping with other UAVs in the monitoring zone by guiding them to the best spot along the monitoring region's border.

Kothari, Mangal et al. proposes another path planning algorithm for autonomous navigation of drones. The Path planning algorithm employing RRTs creates paths for many unmanned aerial vehicles (UAVs) in real time, from provided beginning locations to goal locations, in the presence of static, pop-up, and dynamic obstacles. In the presence of static, pop-up, and dynamic obstacles, this offers collision-free routes. The RRT algorithm's primary benefit is its ability to quickly construct an optimal path.

2.2 CONCLUSION FROM LITERATURE REVIEW

The review of previous research works explains different methods for the detection of traffic rule violators. From this it is clear that the use of YOLO algorithm for detection yields better result and is more accurate since YOLO is fast and efficient than other algorithms such as R-CNN, Fast CNN etc. Eventhough A* algorithm is faster than RRT algorithm for finding optimal path, the use of RRT is more practical since RRT algorithm can generate the path for automating the drone in a short amount of time.

CHAPTER 3
METHODOLOGY

The aim of this project is to create an autonomous drone that uses deep learning to identify persons who are not wearing proper safety equipment like seat belts or helmets. This involves detection using deep learning networks like YOLO, R-CNN, DNN etc. YOLO detection algorithm is used for detecting those not wearing helmets. Then YOLO detection algorithm along with Automatic Number Plate Recognition using OCR is used to detect the number plate of these traffic violators. An automatic path planning algorithm is developed which enables the drone to travel to random locations automatically.

3.1 Helmet and Seatbelt Detection using Deep Learning

For seat belt and helmet recognition, a variety of deep learning methods are available. These deep learning algorithms include YOLO, R-CNN, Fast CNN, and others. The YOLO algorithm is utilised for detection in this project. Yolo, short for You Only Look Once, has improved prediction accuracy and bounding box intersection over union. The main benefit of YOLO is its quickness. Compared to its competitors, YOLO is a lot faster algorithm.

With 24 convolutional layers, four max-pooling layers, and two fully linked layers, the YOLO architecture is similar to GoogleNet. The method divides the image into N grids, each with a region of equal dimension (SxS). The detection and localization of the object contained within each of these N grids is their respective responsibility. Collecting the collection of images of persons wearing and not wearing helmets and seatbelts is the first stage in implementation. B y creating boundary boxes labelling the helmet and no helmet classes, this dataset is used to train the machine. Using this model, we can then use real-time predictions to determine if the person in the video is wearing a helmet or not.

3.2 Automatic Number Plate Recognition using OCR

After identifying the traffic violators not wearing safety equipments like helmet or seatbelt the next step is to detect their vechile number plate. For Automatic number

plate recognition Yolo Detection based on transfer learning approach is used. Transfer learning focuses on storing knowledge gained while solving one problem and applying it to a different but related problem. That is a pre-tained model is used for detection based on Yolo. This approach make use of dataset of vechiles with number plates to train the model. Then model is used to detect the number plate and the number plate from the image/ video is cropped and saved into a detection folder. Then using Easy-OCR engine to detect the text from the numberplate and convert it into a text format. Thus the corresponding image path along with the number plate text is saved in excel sheet as csv format. Thus identification of the traffic violator along with their vechile number plate is obtained.

3.3 3D Mapping and Motion Planning

For making the drone autonomous, automatic navigation is the key requirement. 3D mapping and motion path planning are responsible for autonomous navigation. First the system creates a 3D map of the environment using the camera equipped on the drone. Then using a path planning algorithm suuch as A*, RRT , D* etc to find and develop an optimal path for the travel.

3.4 Overall Framework

The starting and the ending position of a route contains drone stations where the drone automatically position themselves to charge. First it starts from the charging station and hovers around the locations where the system is assigned for detection (near school zones, hospital zones or accident prone areas). Thus the system creates a 3 dimensional map of the surroundings. After 3D mapping the drone returns to its initial position and hovers. Note that the 3D mapping is done only once when introduced to a new surroundings. The system then uses any of the path planning algorithm to determine the optimal path between the starting point and the goal. For this project RRT algorithm is used since it find shortest and optimal path in a very short amount of time. During its travel the drone detects the traffic violators and detect and extract their number plates. Here the goal position may not be a single position. Depending on the path of travel multiple goal position may be set. The detection takes place during the travel as well as on each goal position where the drone hovers for some amount of time. The algorithm continously monitor and detect the traffic violators during its travel.

The flowchart of overall working of the proposed model is shown below:

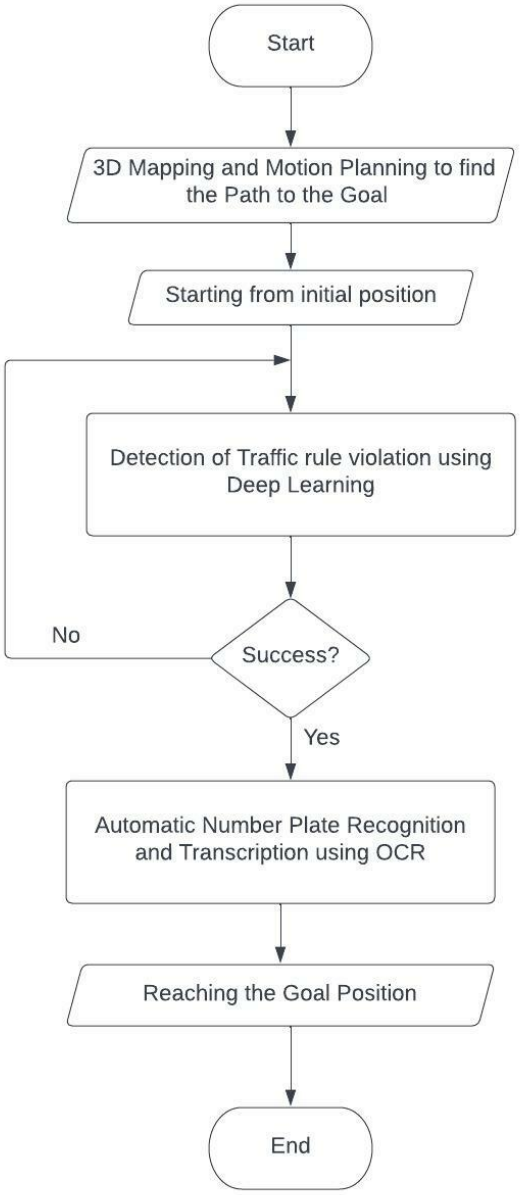


Figure 3.1: Flowchart of the proposed method

CHAPTER 4
IMPLEMENTATION AND EXPERIMENTAL DETAILS

4.1 Datasets

The datasets used for this project was from Kaggle dataset. For the detection of seatbelts and helmet a dataset consisting of 764 images were used. But the problem was that after training the model overfits. So used custom images with their labels along with the dataset thus having a total of 914 images. The dataset used for number plate recognition consist of 543 images.

4.2 Software

Python is the coding language used. In order to write and execute the python and to use tensorflow and computer vision library Anaconda Python is used. The code is executed in jupyter notebook.

4.3 Experimental setup

For seatbelt and helmet detection using YOlO a dataset consisting of 764 images were used, from which 80% of the images were used for training and 20% for testing. The training was done for 3000 iteration steps and the accuracy obtained was low and the model overfits. To resolve this custom images with their labels is added to the dataset thus having a total of 914 images. The training was again done for 10000 iteration steps and the accuracy obtained was around 89% and loss function reduced to a small value. The model was then used for detection. For Automatic number plate recognition dataset consisting of 543 images were used from which 90% is used for training and 10% of the images for testing. The training was done for 5000 iteration steps which provides an accuracy around 85%. The model is then used to detect number plate from images as well as from video captured in real-time from laptop camera. For transcribing the text written on the number plate EasyOCR engine is used. The python code is executed on anaconda python software with all the deep learning libraries installed.

4.4 System Design

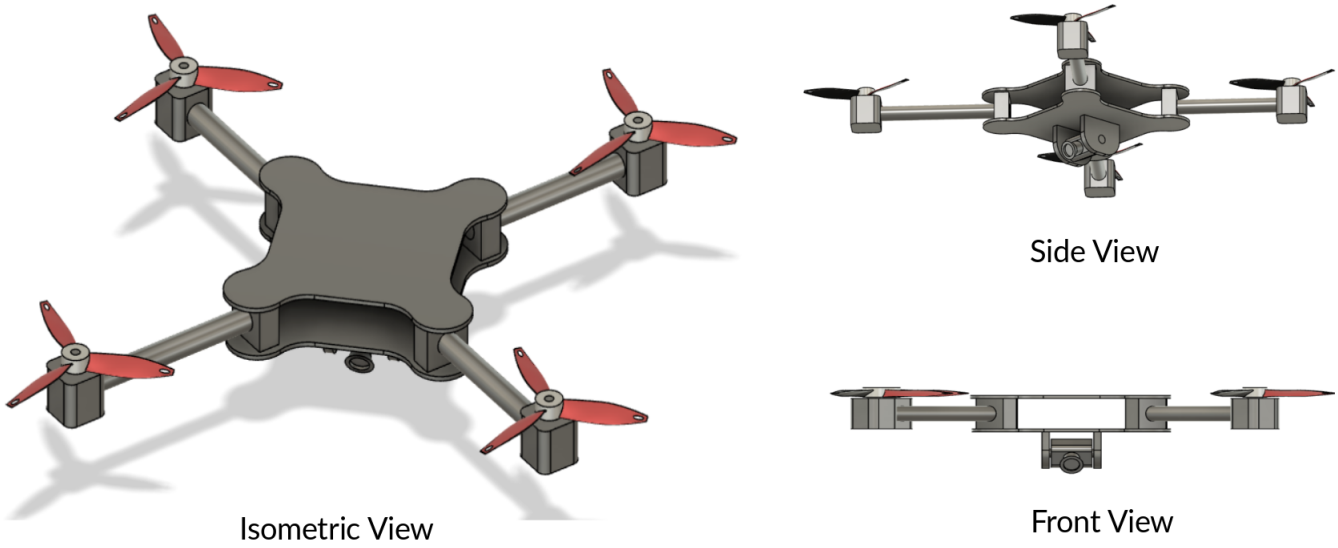


Figure 4.1: 3D CAD model of the proposed system

The proposed the system is designed from scratch using Fusion 360. The base is having a dimension of 270x270 mm and the length of each of the arm is 135mm. Each of these arms are placed at 120 degree angle and the distance between these arms are 170mm. The rotors considered to be using is BLDC 2200 kV with propeller 1045 together each of these rotors can produce a thrust of around 1200gms. Therefore 4 rotors combined thrust of around 4800gms will be sufficient enough to lift the raspberry pi module, control unit , power unit and the Gimbal assembly.

The quadcopter model needs to be converted into URDF format after being created so that ROS may use it for simulation. URDF is similar to the Simulation Description Format (SDF), which is frequently employed to create Gazebo environments built around pre-existing robots. Robots whose kinematics can be described by a tree are the only ones that URDF can represent. The parameter server loaded the robot's URDF model at launch, using the default name robot description.
The steps for generating URDF file is as follows:

Step 1 :Add solid works to urdf plugin in solidworks
Step 2 :Run in solid works.
Step 3 :Click Tools,then choose ADD-INS .
Step 4: In ADD-INS,a drop down list will appear,click on scripts and addsin
Step 5 : A dialog box will appear probably on left side of the window.

step 6: Select URDF Exporter and click RUN.

Step 7: A dialog box will appear probably on left side of the window.

Step 8 :Select a folder and click OK.

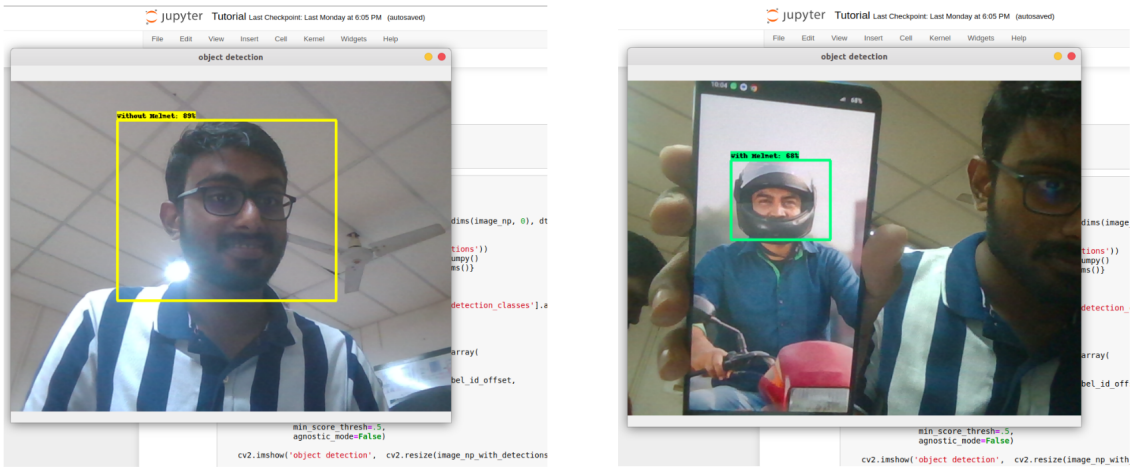
Step 9 :A dialog box will appear with message ”SUCCESSFULLY CREATED URDF FILE”

Step 10 :Finally click OK. The URDF folder will be created in the destination given. Copy the folder to some location in drives so that we can access from Windows. URDF is an XML format for representing a robot model.

CHAPTER 5

RESULTS

5.1 Helmet detection using YOLO-V3 algorithm



Detection without Helmet

Detection with Helmet

Figure 5.1: Detection with and without helmet using webcam

The trained model is loaded into the python environment. The result shown above is from the video captured in real-time from the laptop camera. OpenCV and its corresponding libraries are used to capture the video in real-time. The developed model is able to detect both helmet and no helmet conditions.

5.2 Automatic Number Plate Recognition and Identification Using Transfer Learning and EasyOCR

For Automatic Number Plate Recognition a trained model is used for detection of the number plate location. The program code in written in python and the algorithm is able to detect the number plate location from any input image as well as from the video captured in real-time from laptop camera using OpenCV.

Figure 5.2: Detection of number plate from input image

After detection the cropped image of the number plate is saved into a detection images folder. For transcribing the text from the cropped image of the number plate EasyOCR engine is used. The image path of the number plate along with there number plate text is updated in an excel sheet in csv format.

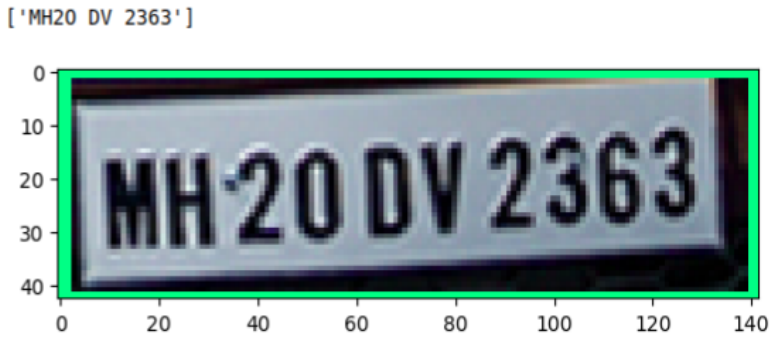


Figure 5.3: Cropped image of the detected number plate

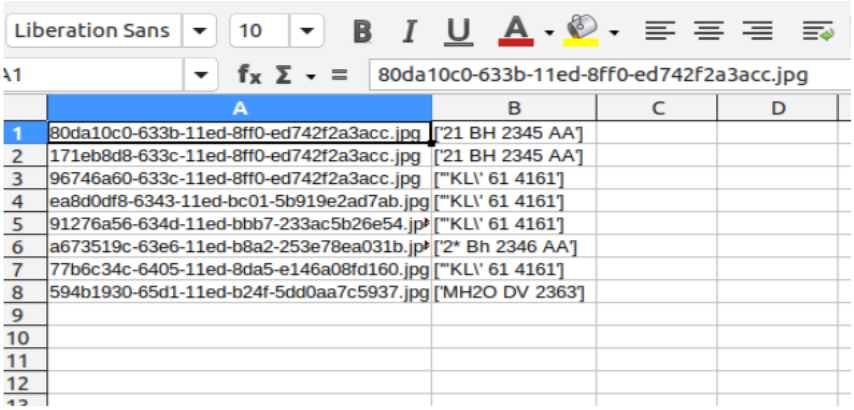


Figure 5.4: Image path and number plate text updating in Excel

5.3 Simulation in ROS

Robot operating system (ROS) is an open-source meta-operating system. It offers all of the features one would anticipate from an operating system, such as hardware abstraction, low-level device control, the implementation of frequently used features, message-passing between processes, and package management. Additionally, it offers programmes and libraries for getting, creating, writing, and running code on various systems.

For visualising robot planning in ROS, there are two applications: Gazebo and Rviz. The Player Project's 3D simulation, called Gazebo, is a component of it. The dynamic environments that a robot might experience are precisely replicated by Gazebo. The libraries in the gazebo simulator can be categorised as simulation, rendering, user interface, communication, and sensor generation.Gazebo can simulate robot in a three dimensional world. It produces both physically plausible object interactions and realistic sensor feedback (it includes an accurate simulation of rigidbody physics).

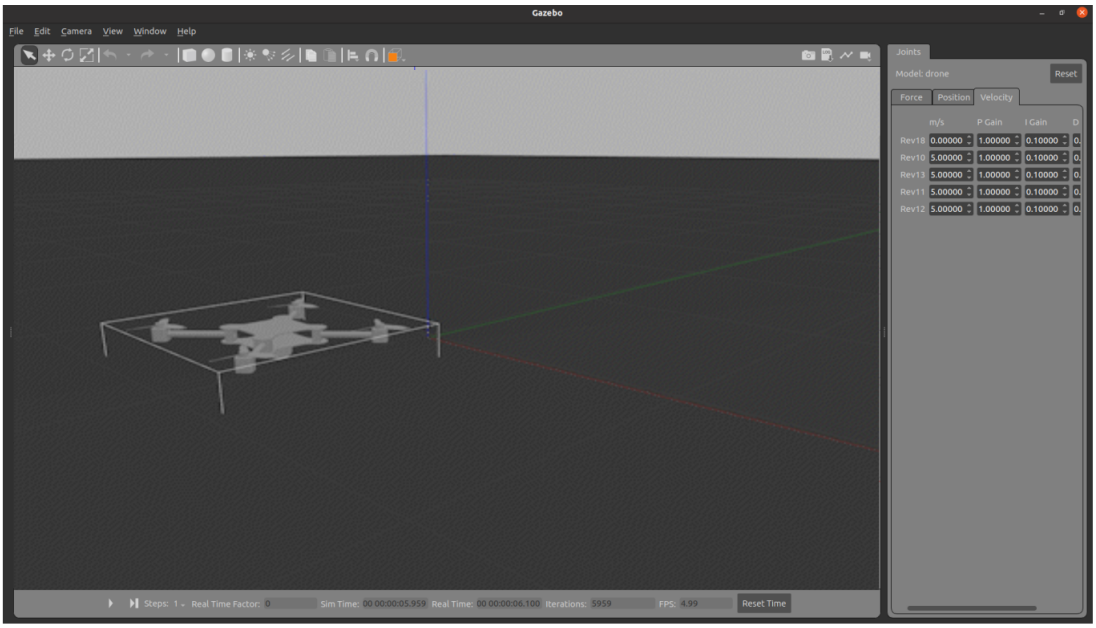


Figure 5.5: Quadcopter in ROS environment

To display the topics communicated between nodes and to allow users to design and alter robots, ROS's RViz visualisation tool is extremely helpful. An easy way to verify that the joint connection between links is correctly loaded is to launch the RViz node and other crucial nodes to create a ROS network that can control the position of each joint. After the robot model has appeared on the window, the joint connection can then be verified.

CHAPTER 6
CONCLUSION

The aim of this project is to implement an autonomous drone that is able to detect people not wearing safety equipments like helmets or seatbelts and to recognize and extract their number plates. The detection of helmet, automatic recognition and transcription of number plate, system design and simulation in ROS is being discussed. However path planning and 3D mapping for autonomous navigation of the drone has yet to be completed.

In future work, the automatic path planning and 3D mapping will be completed and a modification to the existing path planning algorithm will be done. This work can be extended to detect other traffic violations like cross-lane cutiing, detection of the use of mobile phone while driving, overspeeding of vechiles on accident prone areas, school and hospital zones etc.

BIBLIOGRAPHY

[1] Tonge, Aniruddha, et al. "Traffic Rules Violation Detection using Deep Learning," *2020 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA). IEEE, 2020.*

[2] Adhikari, Himanshu, et al. "Helmet Detection on Two-wheeler Riders Using Machine Learning," *"International Journal Of Advances In Electronics And Computer Science, Volume-5, Issue-9, September, 2018.*

[3] Rohith, C. A., et al. "An efficient helmet detection for MVD using deep learning," *2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI). IEEE, 2019.*

[4] Maduri, Praveen Kumar, et al. "Seat Belt And Helmet Detection Using Deep Learning," *2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N). IEEE, 2021.*

[5] Meenu R, Sinta Raju, et al. " Detection of Helmetless Riders Using Faster R-CNN",*International Journal of Innovative Science and Research Technology, Volume-5, Issue-5, May – 2020.*

[6] Islam, Shafkat, and Abolfazl Razi. "A path planning algorithm for collective monitoring using autonomous drones,"*2019 53rd Annual Conference on Information Sciences and Systems (CISS). IEEE, 2019.*

[7] Kothari, Mangal, Ian Postlethwaite, and Da-Wei Gu. "Multi-UAV path planning in obstacle rich environments using rapidly-exploring random trees,"*Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference. IEEE, 2009.*

Appendix

6.1 Real Time Helmet Detection Algorithm

```
import cv2
import numpy as np
category_index = label_map_util.create_category
_index_from_labelmap(ANNOTATION_PATH+'/label_map.pbtxt')
cap.release()
cap = cv2.VideoCapture(0)
width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
while True:
    ret, frame = cap.read()
    image_np = np.array(frame)

    input_tensor = tf.convert_to_tensor(np.expand
    _dims(image_np, 0), dtype=tf.float32)
    detections = detect_fn(input_tensor)

    num_detections = int(detections.pop('num_detections'))
    detections = {key: value[0, :num_detections].numpy()
                  for key, value in detections.items()}
    detections['num_detections'] = num_detections

    detections['detection_classes'] = detections
    ['detection_classes'].astype(np.int64)

    label_id_offset = 1
    image_np_with_detections = image_np.copy()
```

```
    viz_utils.visualize_boxes_and_labels_on_image_array(
                image_np_with_detections,
                detections['detection_boxes'],
                detections['detection_classes']+label_id_offset,
                detections['detection_scores'],
                category_index,
                use_normalized_coordinates=True,
                max_boxes_to_draw=5,
                min_score_thresh=.5,
                agnostic_mode=False)

    cv2.imshow('object␣detection', cv2.resize
    (image_np_with_detections, (800, 600)))

    if cv2.waitKey(1) & 0xFF == ord('q'):
        cap.release()
        break
detections = detect_fn(input_tensor)
from matplotlib import pyplot as plt
```

.

6.2 ANPR Algorithm

```
import cv2
cap = cv2.VideoCapture(0)
width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))

while cap.isOpened():
    ret, frame = cap.read()
    image_np = np.array(frame)

    input_tensor = tf.convert_to_tensor(np.expand_
    dims(image_np, 0), dtype=tf.float32)
```

```
detections = detect_fn(input_tensor)

num_detections = int(detections.pop('num_detections'))
detections = {key: value[0, :num_detections].numpy()
               for key, value in detections.items()}
detections['num_detections'] = num_detections

# detection_classes should be ints.
detections['detection_classes'] = detections
['detection_classes'].astype(np.int64)

label_id_offset = 1
image_np_with_detections = image_np.copy()

viz_utils.visualize_boxes_and_labels_on_image_array(
            image_np_with_detections,
            detections['detection_boxes'],
            detections['detection_classes']+label_id_offset,
            detections['detection_scores'],
            category_index,
            use_normalized_coordinates=True,
            max_boxes_to_draw=5,
            min_score_thresh=.8,
            agnostic_mode=False)

try:
    text, region = ocr_it(image_np_
    with_detections, detections, detection_
    threshold, region_threshold)
    save_results(text, region, csv_filename, folder_path)
except:
    pass
```

```
        cv2.imshow('object detection', cv2.resize
        (image_np_with_detections, (800, 600)))

        if cv2.waitKey(10) & 0xFF == ord('q'):
            cap.release()
            cv2.destroyAllWindows()
            break
import easyocr
detection_threshold = 0.7
image = image_np_with_detections
scores = list(filter(lambda x: x> detection_threshold,
detections['detection_scores']))
boxes = detections['detection_boxes'][:len(scores)]
classes = detections['detection_classes'][:len(scores)]
width = image.shape[1]
height = image.shape[0]
for idx, box in enumerate(boxes):
    roi = box*[height, width, height, width]
    region = image[int(roi[0]):int(roi[2]),
    int(roi[1]):int(roi[3])]
    reader = easyocr.Reader(['en'])
    ocr_result = reader.readtext(region)
    print(ocr_result)
    plt.imshow(cv2.cvtColor(region,cv2.COLOR_BGR2RGB))
region_threshold = 0.6
def filter_text(region, ocr_result, region_threshold):
    rectangle_size = region.shape[0]*region.shape[1]

    plate = []
    for result in ocr_result:
        length = np.sum(np.subtract
        (result[0][1], result[0][0]))
        heigth = np.sum(np.subtract
```

```
        (result[0][2], result[0][1]))

        if length*height / rectangle_size >
         region_threshold:
            plate.append(result[1])

    return plate
filter_text(region, ocr_result, region_threshold)
import csv
import uuid
'{}.jpg'.format(uuid.uuid1())
def save_results(text, region, csv_filename, folder_path):
    img_name = '{}.jpg'.format(uuid.uuid1())

    cv2.imwrite(os.path.join(folder_path,
     img_name), region)

    with open(csv_filename, mode='a',
     newline='') as f:
        csv_writer = csv.writer(f, delimiter=',',
         quotechar='"', quoting=csv.QUOTE_MINIMAL)
        csv_writer.writerow([img_name, text])
save_results(text, region, 'detection_results.csv',
 'Detection_Images')
```