SmartFlow: Integrated Emergency Vehicle Detection and Traffic Rule Monitoring System

Submitted to the
Department of Master of Computer Applications
in partial fulfilment of the requirements
for the Mini Project (MCAP1)

by

Komal S Kallagoudar
1MS22MC016

Under the guidance of

Abhishek K L
Assistant Professor

Department of Master of Computer Applications

RAMAIAH INSTITUTE OF TECHNOLOGY

(Autonomous Institute, Affiliated to VTU)
Accredited by National Board of Accreditation & NAAC with 'A+' Grade
MSR Nagar, MSRIT Post, Bangalore-560054
www.msrit.edu

2024

RAMAIAH
Institute of Technology

DEPARTMENT OF MASTER OF COMPUTER APPLICATIONS

CERTIFICATE

This is to certify that the project entitled SmartFlow: Integrated Emergency Vehicle Detection and Traffic Rule Monitoring System is carried out by

Student Name	USN
Komal S Kallanagoudar	1MS22MC016

students of 3rd semester, in partial fulfillment for the Mini Project (MCAP1), during the academic year 2023-2024.

Guide	Head of the Department
Abhishek K L Assistant Professor	

Name of Examiners	Signature with Date
1.	
2.	

ACKNOWLEDGEMENTS

I would like to express our gratitude to all those who gave us the opportunity to complete this project. A special thanks to our project head, Mr. Abhishek K. L, Asst. Professor, Department of MCA, whose help, stimulating suggestion and encouragement, helped us to coordinate our project.

I would also like to acknowledge with much gratitude the crucial role of staff of Department of MCA Lab, who gave the permission to use all required components and the necessary material to complete the project We would also like to express our gratitude to the Head of the Department.

DECLARATION

I hereby declare that the project report entitled "SmartFlow: Integrated Emergency Vehicle Detection and Traffic Rule Monitoring System" based on study undertaken by me, towards the partial fulfilment for the Mini Project (MCAP1) carried out during the 3rd semester, has been compiled purely from the academic point of view and is, therefore, presented in a true and sincere academic spirit. Contents of this report are based on my original study and findings in relation there to are neither copied nor manipulated from other reports or similar documents, either in part or in full, and it has not been submitted earlier to any University/College/Academic institution for the award of any Degree/Diploma/Fellowship or similar titles or prizes and that the work has not been published in any specific or popular magazines.

Place: Bangalore **Komal S Kallanagoudar**

Date: **1MS22MC016**

ABSTRACT

This project explores the advancement of Smart Traffic Management System using the Internet of Things (IoT). It works as middleware on the foundation of the IoT and augments the idea of the smart city through the traffic light control, congestion control, and emergency vehicle detection. The main objective of this project is to design an adaptive traffic light signals using the camera module and to design an effective method to overcome the ambulance delay problem. Traffic light controlling becomes major issue with increase in automobiles which causes congestion and it also became a major reason for the ambulance delay. This requires a smart system to handle traffic signals and to reduce ambulance delay. Using this system development at traffic junction we need not to worry about handing the traffic manually and also consumes less time as compared to the conventional traffic system. It also reduces the ambulance delay time.

Table of Contents

1. Introduction 1

1.1 Overview 1

1.2 Problem Definition 1

2. Literature Survey 3

3. Hardware and Software Requirements 3

3.1 Hardware Requirements 5

3.2 Software Requirements 5

4. Software Requirements Specification 6

4.1 System Features 6

4.1.1 Traffic Analysis 6

4.1.2 Object Detection and Classification 6

4.1.3 Emergency Vehicle Detection 6

5. System Design Description (SDD) 7

5.1 System Overview 7

5.2 Database Design/Data Set Description 7

5.3 Functional Design 8

5.3.1 Describe the functionalities of the system: 8

5.3.2 Behavioral design: 10

6. Implementation 11

7. Testing 18

7.1 Description of Testing 18

7.2 Test Cases 19

8. Results and Discussion 21

9. Conclusion 22

10. Scope for Further Enhancement 23

Bibliography 24

Table of Figures

1	System Architecture……..	7
2	Use case Diagram……	8
3	Behavioural Design……	10
4	Vehicle Detections …….	21
5	Congestion Control…….	21

1. Introduction

1.1 Overview

- The traffic management system of a metropolitan city is a keystone for urban mobility. With the rise of the population, the demand for vehicles grows up and hence the requirement of transportation has also increased. Infrastructural development becomes an indispensable part of complementing the population growth to augment urban mobility. But the traditional traffic management system is shown not only ineffective for accompanying the increased number of vehicles with the use of police control and traffic light system but also incompetent enough to handle this growth of traffic on road systems. This traffic congestion consequentially consumes precious working time for being incapable of handling extensive traffic congestion and eventually leads to the environmental pollution for an extended period of vehicle emission. Adequate pre-measures and proper planning can help to reduce the number of traffic problems and manage an increased number of vehicles on the road. Traffic system utilize the concept of automation with IoT is called as "Smart Traffic". Smart Traffic Management System is an advanced and integrated solution designed to optimize traffic flow, reduce congestion, enhance road safety, and improve overall transportation efficiency within urban or metropolitan areas. This system relies on various sensors placed strategically throughout the road network to monitor traffic conditions
- This system will monitor the traffic using camera.
- The system can control traffic signals at intersections dynamically based on real-time traffic data.
- Adaptive traffic signal systems adjust signal timings to minimize waiting times and reduce idling
- Reducing congestion and energy consumption at intersection.
- Ensuring immediate clearance for emergency vehicles. Facilitating safer and shorter commute time.
- The emergency vehicle is detected, which gives ambulances priority to pass through traffic lights.

1.2 Problem Definition

In contemporary times, the escalating demand for transportation coupled with rapid urbanization places an immense strain on existing infrastructure, particularly in managing road transportation. One of the most pressing challenges stemming from this is the widespread occurrence of traffic jams, predominantly concentrated in urban areas. These jams give rise to a multitude of issues, including heightened levels of noise and air pollution, as well as significant delays in travel time. The current state of congestion poses a substantial threat to various facets of life, including the economy, environment, and overall well-being. A significant contributor to traffic congestion is the malfunctioning of traffic lights and other related infrastructural deficiencies. Such inadequacies result in prolonged red-light delays, exacerbating congestion and its associated negative impacts. This congestion not only squanders valuable productive time but also leads to the wastage of fossil fuels, exacerbates pollution levels, and inflicts substantial economic losses.

The existing traffic signal systems deployed across cities, with their fixed predetermined timings for red and green signals, prove insufficient in addressing the aforementioned challenges. Recognizing this limitation, numerous endeavors have been made to imbue traffic lights with intelligence, enabling them to respond dynamically to the density of vehicles on the road.

The proposed system will solve this problem. The camera’s installed in the traffic signals will send AI to count number of vehicles so that based on number of vehicles the traffic lights delay is adjusted.

2. Literature Survey

The proposed system smart flow enables to control the traffic based on density of the traffic. And also enables the ambulance to avoid the delay due to road congestions. The Infrared Technology, visual sensing, RFIDs/radar system. All these together can help in controlling the traffic whenever any ambulance or some emergency vehicle comes closer. The STMS eliminates the delay time faced by the emergency response vehicles on to the vehicle by turning on the green signal for the emergency vehicle [1]. Basic information of the patient is taken, this information is further used to send it to the hospital. Depending upon the emergency, the driver sends the direction towards which it wants to travel [8].

The camera installed on the traffic junctions will transfer the images to the system. Using Artificial Intelligence (AI) the number of vehicles detected using YOLO algorithm and based on the number of vehicles and type of vehicles the green delay for a particular lane is adjusted, so that congestion is controlled. If the green delay for every lane is equal then where there are more vehicles that lane drives have to wait for long time.

Not only the traffic management but also following the traffic rules is also important. The proposed monitor that, if the vehicle passed through a red-light signal, record the offence, notify it to the police control room, and send a message to the violator with necessary details such as date, time, location, image (if taken), and the penalty levied. The vehicle's number plate as every vehicle has a unique number through which it is easily differentiated from other vehicles. Vehicles in each country have a unique license number, which is written on its license plate. This number distinguishes one vehicle from the other, which is useful especially when both are of same make and model [4]. The YOLOv3 algorithm is used to detect the traffic violation. The violations detected are vehicles jumping red signals, vehicle riding without helmets and vehicle drivers without seat belts [5].

Public transport operators consequently need to provide reliable services in order to minimize disruption events that can affect the vehicles and their drivers, such as breakdowns, accidents or illnesses. The project focuses on the type of events and approaches related with the vehicle drivers and the identification of both their

performance profiles and health condition while in operation. IoT-based system is designed to avoid countless mishaps due to drowsy drivers' behavioural and psychological changes by focusing on driver's eye movements. In addition to monitoring the intensity of the collisions impacts during road accidents, it is also records of the location for taking supportive action by using following technologies and algorithms Face and Eye Detection by Machine Learning (ML) and Deep Learning (DL) Algorithms, FPGA-Based Drowsiness Detection System and Eye Recognition System Based on Wavelet Network Algorithm [2][7]. The alert message helps in locating the location so that the medical services can be provided on time and this way the precious lives can be saved. If in case there is no casualty and assistance is not required then you can terminate the message sending process using the switch provided in the device. The message is transmitted and the location of the accident is identified using the GPS module. With the help of the Accelerometer sensor the accident can be precisely detected [10].

Li-Fi technology, where streetlights transmit area maps to vehicles, aiding navigation without relying on mobile data or Wi-Fi. The received map is displayed on the vehicle's screen, offering a novel road navigation helps ambulance driver to take shortest path and avoid the traffic so reach the hospital as soon as possible. LiFi for highway navigation offers avenue for improving communication, navigation, and overall driving experience on highways. The street lights available on the road are at an average distance of 40 m from each other. The spectrum used for Li-Fi is eco-friendly. Each of the street light will transmit the road map of the area before the current location at a distance of 10 to 20 m. The system does not require the users to pay for the Li-Fi connection unlike the internet connection [6][12].

3. Hardware and Software Requirements

3.1 Hardware Requirements

- Arduino MEGA
- ESP32-cam
- Ultrasonic Sensors
- RFID Module
- Traffic LEDs
- Jumper wires

3.2 Software Requirements

- Arduino IDE
- Efp-idf
- Google Cloud vision

4. Software Requirements Specification

4.1 System Features

The cameras installed at the traffic junctions will capture videos and photos and send it to the system. Using AI algorithm count the number of vehicles and adjust the traffic signal. Using RFID tags emergency vehicle is detected.

4.1.1 Traffic Analysis

REQ-1: Real-time Video Feed Capture

The system will capture the video or image of the lane for which the signal is going to be green in next 5 seconds.

REQ-2: Sending Image to system

The image taken by the camera is sent to AI algorithm in order to calculate the number of vehicles.

4.1.2 Object Detection and Classification

REQ-1: Detecting vehicles

Algorithm is used to detect the number of vehicles in the image and classifying the vehicles such as car, bus, bike, truck etc.

REQ-2: Traffic Flow Analysis

Analyzing the traffic flow Based on the number and type of vehicle calculate the green delay of the signal for one lane in order to avoid the congestion.

4.1.3 Emergency Vehicle Detection

REQ-1: Detecting the Emergency vehicle

Using RFID tags the emergency vehicle like ambulance is detected.

REQ-2: Traffic Flow Analysis

As soon as ambulance is detected the signal for the particular lane is turned to green so that the ambulance reaches the hospital without facing any traffic.

5. System Design Description (SDD)

5.1 System Overview

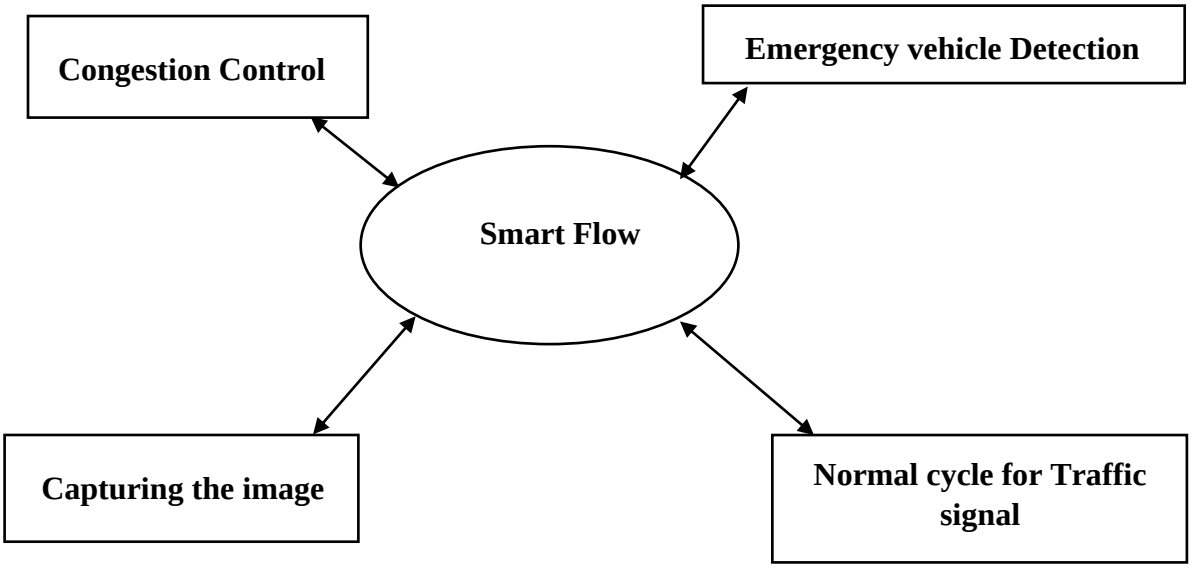

Figure 1: System Architecture

The Smartflow system consist of normal cycle for traffic signal, congestion control using camera, detecting the number of vehicles and classifying them using AI YOLO algorithm and emergency vehicle detection using RFID tag. For camera the code is loaded with the help of Ardino board once and then the images taken by the camera are viewed in the AI screen this image is sent to algorithm as input and algorithm gives the vehicle count and also the value of green delay. Then the signals are managed. The RFID tag is used to detects the emergency vehicle and inform the system so that green signal is turned for emergency vehicle.

5.2 Data Set Description

The number of vehicles detection and the classification vehicles is processed by YOLO algorithm. The data set consist of all different categories of vehicles is given as input which has lists of vehicles such as car, bus, truck, bike etc. The image taken by the camera is stored on the system.

5.3 Functional Design

5.3.1 Describe the functionalities of the system:

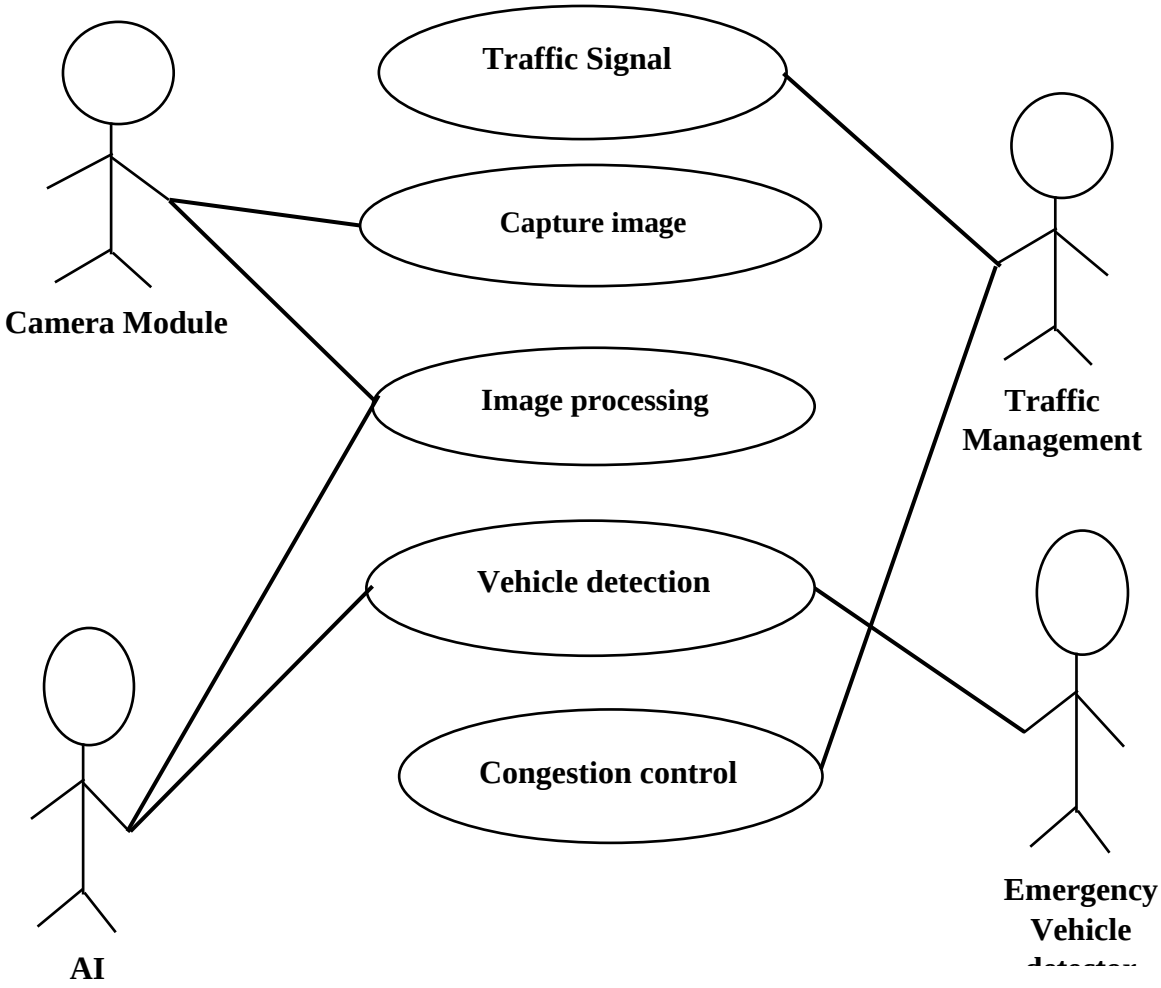

Figure 2: Use case Diagram

Use Cases

Traffic signal: The traffic junction where the traffic lights are controlled by the traffic management system.

Capture image: The cameras at junction takes images and send it to the system.

Image processing: The AI algorithm processes the image to detect the number of vehicles and classify them.

Vehicle detection: The AI algorithm uses data set and detect type and number of vehicles. The RFID is used to detect the emergency vehicle.

Congestion control: Traffic congestion , were there are more vehicles in one lane and less number of vehicles in another lane leads to traffic.

Actors

Camera Module: Captures the real time image or video to control the congestion based on the number of vehicles on a lane.

AI: Artificial intelligence algorithm YOLO used to detect the vehicles and classify them to calculate the green delay.

Traffic Management: The main system which controls the normal cycle of traffic and congestion control of the traffic, it uses the data given by the AI algorithm and RFID tag to control the traffic by adjusting green delay.

Emergency Vehicle Detector: The RFID tag is used to detect the emergency vehicle and inform system to turn on the green signal for emergency vehicle

5.3.2 Behavioral design:

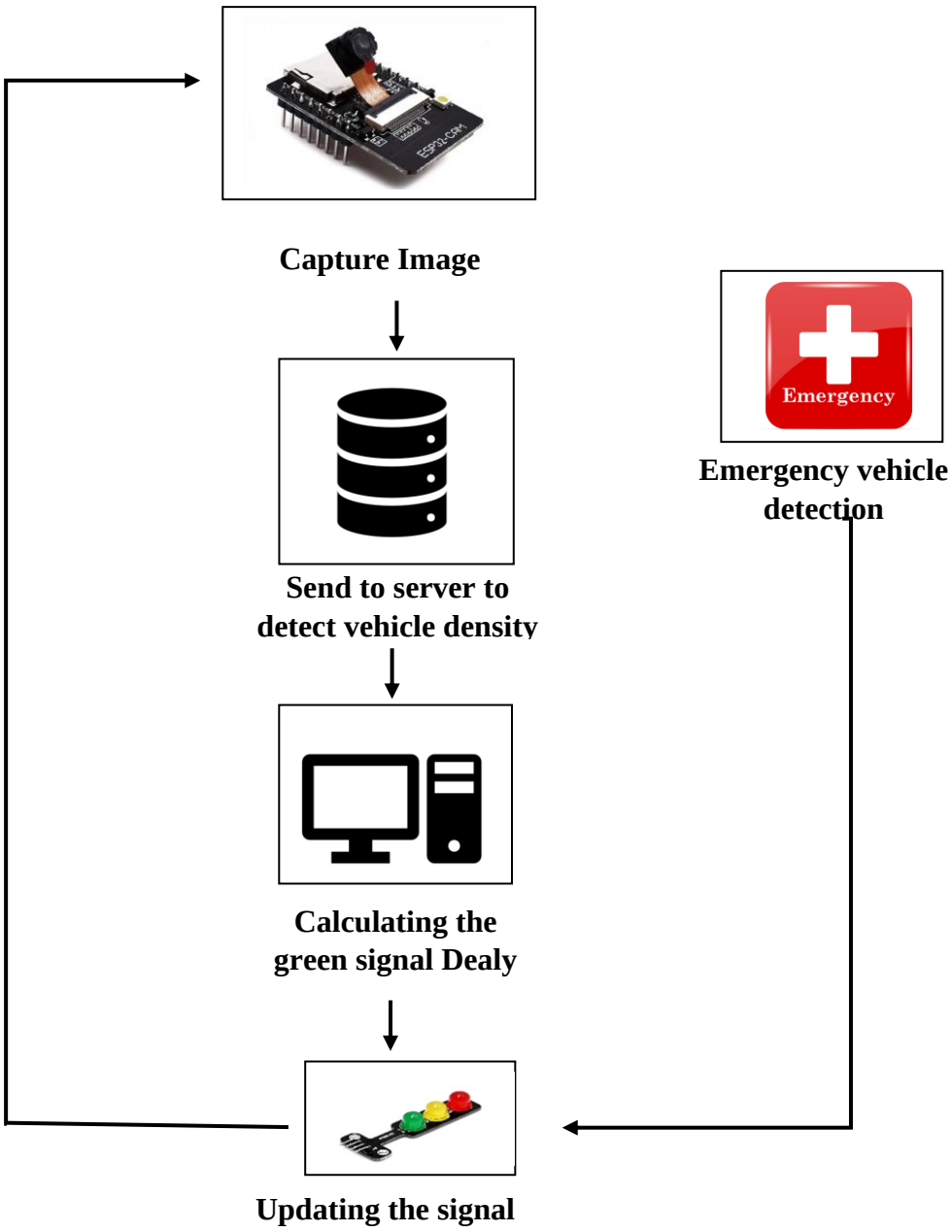

Figure 3: Behavioural Design

The camera captures the image and store it in the server, then YOLO algorithm processes the image and detect the congestion and calculate the green signal timing, according that calculation the traffic signal is controlled. If there is any emergency vehicle is arriving towards the traffic junction then RFID tag detects the vehicle and green signal is turned on for the emergency vehicle.

6. Implementation

Module 1: Image Processing

To implement smart traffic management using an ESP32 camera module, start by connecting the ESP32 with a compatible camera like the ESP32-CAM. Capture images or stream video of traffic scenes using the camera. Utilize OpenCV for image processing tasks such as object detection, motion detection, and lane detection. Analyze the processed data to extract useful information like vehicle count, congestion level, and vehicle types. Based on this analysis, make decisions such as adjusting traffic signals or displaying real-time traffic information. Ensure network connectivity for data transmission and integration with servers or cloud platforms. Optimize power consumption to ensure reliable operation, especially in remote locations or battery-powered setups. Test the system thoroughly in various traffic conditions and optimize algorithms for better performance. Once tested, deploy the system in real-world scenarios and monitor its performance continuously while performing regular maintenance to ensure proper functioning.

Module 2: Vehicle Detection

The YOLO (You Only Look Once) algorithm will start collecting a dataset of labeled images containing vehicles, annotating each image with bounding boxes around the vehicles, and labeling them according to their classes (e.g., car, truck, motorcycle). With the annotated dataset, train a YOLO object detection model capable of recognizing various types of vehicles. Evaluate the trained model's accuracy and performance on a separate validation dataset. After detecting vehicles, classify them into categories such as car, truck, or motorcycle using YOLO's class probabilities along with bounding box coordinates. Implement a mechanism to count the number of vehicles detected, tracking the count over time to estimate traffic flow and congestion. Use this vehicle count data to estimate green delay at traffic signals by analyzing the rate at which vehicles pass through intersections after the signal turns green.

Module 3: Emergency Vehicle Detection

Emergency vehicle detection using RFID involves equipping emergency vehicles with RFID tags and installing RFID readers along roadways. When an emergency vehicle with an RFID tag approaches a reader, the reader detects the tag's unique ID, triggering actions such as changing traffic signals or notifying authorities. Integration with existing traffic systems, testing, and maintenance are crucial for reliability and effectiveness.

Setup.py

```
from Cython.Build import cythonize
import numpy
import os
import imp

VERSION = imp.load_source('version', os.path.join('.', 'darkflow', 'version.py'))
VERSION = VERSION.__version__

if os.name =='nt' :
    ext_modules=[
        Extension("darkflow.cython_utils.nms",
            sources=["darkflow/cython_utils/nms.pyx"],
            #libraries=["m"] # Unix-like specific
            include_dirs=[numpy.get_include()]
        ),
        Extension("darkflow.cython_utils.cy_yolo2_findboxes",
            sources=["darkflow/cython_utils/cy_yolo2_findboxes.pyx"],
            #libraries=["m"] # Unix-like specific
            include_dirs=[numpy.get_include()]
        ),
        Extension("darkflow.cython_utils.cy_yolo_findboxes",
            sources=["darkflow/cython_utils/cy_yolo_findboxes.pyx"],
            #libraries=["m"] # Unix-like specific
            include_dirs=[numpy.get_include()]
        )
    ]

elif os.name =='posix' :
    ext_modules=[
        Extension("darkflow.cython_utils.nms",
            sources=["darkflow/cython_utils/nms.pyx"],
            libraries=["m"], # Unix-like specific
            include_dirs=[numpy.get_include()]
        ),
        Extension("darkflow.cython_utils.cy_yolo2_findboxes",
            sources=["darkflow/cython_utils/cy_yolo2_findboxes.pyx"],
            libraries=["m"], # Unix-like specific
            include_dirs=[numpy.get_include()]
        ),
        Extension("darkflow.cython_utils.cy_yolo_findboxes",
            sources=["darkflow/cython_utils/cy_yolo_findboxes.pyx"],
            libraries=["m"], # Unix-like specific
            include_dirs=[numpy.get_include()]
        )
    ]

else :
    ext_modules=[
```

```
        Extension("darkflow.cython_utils.nms",
            sources=["darkflow/cython_utils/nms.pyx"],
            libraries=["m"] # Unix-like specific
        ),
        Extension("darkflow.cython_utils.cy_yolo2_findboxes",
            sources=["darkflow/cython_utils/cy_yolo2_findboxes.pyx"],
            libraries=["m"] # Unix-like specific
        ),
        Extension("darkflow.cython_utils.cy_yolo_findboxes",
            sources=["darkflow/cython_utils/cy_yolo_findboxes.pyx"],
            libraries=["m"] # Unix-like specific
        )
    ]

setup(
    version=VERSION,
    name='darkflow',
    description='Darkflow',
    license='GPLv3',
    url='https://github.com/thtrieu/darkflow',
    packages = find_packages(),
    scripts = ['flow'],
    ext_modules = cythonize(ext_modules)
)
```

Vehicle_detection.py

```
import cv2
from darkflow.net.build import  TFNet
import matplotlib.pyplot as plt
import os

options={
   'model':'./cfg/yolo.cfg',        #specifying the path of model
   'load':'./bin/yolov2.weights',   #weights
   'threshold':0.3                  #minimum confidence factor to create a box, greater
than 0.3 good
}

tfnet=TFNet(options)
inputPath = os.getcwd() + "/test_images/"
outputPath = os.getcwd() + "/output_images/"

def detectVehicles(filename):
   global tfnet, inputPath, outputPath
   img=cv2.imread(inputPath+filename,cv2.IMREAD_COLOR)
   # img=cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
   result=tfnet.return_predict(img)
   # print(result)
```

```
  for vehicle in result:
    label=vehicle['label']   #extracting label
    if(label=="car" or label=="bus" or label=="bike" or label=="truck" or
label=="rickshaw"):    # drawing box and writing label
      top_left=(vehicle['topleft']['x'],vehicle['topleft']['y'])
      bottom_right=(vehicle['bottomright']['x'],vehicle['bottomright']['y'])
      img=cv2.rectangle(img,top_left,bottom_right,(0,255,0),3)    #green box of
width 5
      img=cv2.putText(img,label,top_left,cv2.FONT_HERSHEY_COMPLEX,0.5
,(0,0,0),1)   #image, label, position, font, font scale, colour: black, line width
  outputFilename = outputPath + "output_" +filename
  cv2.imwrite(outputFilename,img)
  print('Output image stored at:', outputFilename)
  # plt.imshow(img)
  # plt.show()
  # return result

for filename in os.listdir(inputPath):
  if(filename.endswith(".png") or filename.endswith(".jpg") or
filename.endswith(".jpeg")):
    detectVehicles(filename)
print("Done!")
```

Simulation.py

```
def generateVehicles():
  while(True):
    vehicle_type = random.randint(0,4)
    if(vehicle_type==4):
      lane_number = 0
    else:
      lane_number = random.randint(0,1) + 1
    will_turn = 0
    if(lane_number==2):
      temp = random.randint(0,4)
      if(temp<=2):
        will_turn = 1
      elif(temp>2):
        will_turn = 0
    temp = random.randint(0,999)
    direction_number = 0
    a = [400,800,900,1000]
    if(temp<a[0]):
      direction_number = 0
    elif(temp<a[1]):
      direction_number = 1
    elif(temp<a[2]):
      direction_number = 2
    elif(temp<a[3]):
```

```
            direction_number = 3
        Vehicle(lane_number, vehicleTypes[vehicle_type], direction_number,
directionNumbers[direction_number], will_turn)
        time.sleep(0.75)

def simulationTime():
    global timeElapsed, simTime
    while(True):
        timeElapsed += 1
        time.sleep(1)
        if(timeElapsed==simTime):
            totalVehicles = 0
            print('Lane-wise Vehicle Counts')
            for i in range(noOfSignals):
                print('Lane',i+1,':',vehicles[directionNumbers[i]]['crossed'])
                totalVehicles += vehicles[directionNumbers[i]]['crossed']
            print('Total vehicles passed: ',totalVehicles)
            print('Total time passed: ',timeElapsed)
            print('No. of vehicles passed per unit time:
',(float(totalVehicles)/float(timeElapsed)))
            os._exit(1)
class Main:
    thread4 = threading.Thread(name="simulationTime",target=simulationTime,
args=())
    thread4.daemon = True
    thread4.start()

    thread2 = threading.Thread(name="initialization",target=initialize, args=())    #
initialization
    thread2.daemon = True
    thread2.start()
   # Colours
    black = (0, 0, 0)
    white = (255, 255, 255)

    # Screensize
    screenWidth = 1400
    screenHeight = 800
    screenSize = (screenWidth, screenHeight)

    # Setting background image i.e. image of intersection
    background =
pygame.image.load(r'C:/Users/ASUS/Desktop/Miniproject/Code/YOLO/darkflow
/images/mod_int.png')
    screen = pygame.display.set_mode(screenSize)
    pygame.display.set_caption("SIMULATION")

    # Loading signal images and font
```

```
    redSignal =
pygame.image.load(r'C:/Users/ASUS/Desktop/Miniproject/Code/YOLO/darkflow
/images/signals/red.png')
    yellowSignal =
pygame.image.load(r'C:/Users/ASUS/Desktop/Miniproject/Code/YOLO/darkflow
/images/signals/yellow.png')
    greenSignal =
pygame.image.load(r'C:/Users/ASUS/Desktop/Miniproject/Code/YOLO/darkflow
/images/signals/green.png')
    font = pygame.font.Font(None, 30)

    thread3 = threading.Thread(name="generateVehicles",target=generateVehicles,
args=())    # Generating vehicles
    thread3.daemon = True
    thread3.start()

    while True:
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                sys.exit()

        screen.blit(background,(0,0))
        for i in range(0,noOfSignals):  # display signal and set timer according to
current status: green, yello, or red
            if(i==currentGreen):
                if(currentYellow==1):
                    if(signals[i].yellow==0):
                        signals[i].signalText = "STOP"
                    else:
                        signals[i].signalText = signals[i].yellow
                    screen.blit(yellowSignal, signalCoods[i])
                else:
                    if(signals[i].green==0):
                        signals[i].signalText = "SLOW"
                    else:
                        signals[i].signalText = signals[i].green
                    screen.blit(greenSignal, signalCoods[i])
            else:
                if(signals[i].red<=10):
                    if(signals[i].red==0):
                        signals[i].signalText = "GO"
                    else:
                        signals[i].signalText = signals[i].red
                else:
                    signals[i].signalText = "---"
                screen.blit(redSignal, signalCoods[i])
        signalTexts = ["","","",""]

        # display signal timer and vehicle count
        for i in range(0,noOfSignals):
```

```
            signalTexts[i] = font.render(str(signals[i].signalText), True, white, black)
            screen.blit(signalTexts[i],signalTimerCoods[i])
            displayText = vehicles[directionNumbers[i]]['crossed']
            vehicleCountTexts[i] = font.render(str(displayText), True, black, white)
            screen.blit(vehicleCountTexts[i],vehicleCountCoods[i])

        timeElapsedText = font.render(("Time Elapsed: "+str(timeElapsed)), True,
black, white)
        screen.blit(timeElapsedText,(1100,50))

        # display the vehicles
        for vehicle in simulation:
            screen.blit(vehicle.currentImage, [vehicle.x, vehicle.y])
            # vehicle.render(screen)
            vehicle.move()
        pygame.display.update()

Main()
```

7. Testing

7.1 Description of Testing

Unit Testing

Each functionality of the SmartFlow is tested first the camera connectivity is connected to check video or image processing and sending to the system. Next the algorithm is tested using the test images to detect the number of vehicles and classify them. The normal working of the traffic signals is tested. The RFID tags are tested to detect the emergency vehicle.

Integration

The RFID tag and traffic signals both are combined and tested to check the signals turning green immediately for emergency vehicle to pass through junction and camera module and algorithm are combined and tested to solve the traffic congestion.

System testing

End to end functionality of the system is tested by integrating camera module and RFID with algorithm so that traffic congestion problem is solved based on the number of vehicles present on the lane and also the emergency vehicles like ambulance is given high priority at traffic junction so that it reaches the hospital as soon as possible.

7.2 Test Cases

Test case #	Test case Name	Test case Description	Inputs	Expected Output	Actual Output	Status
1	Normal traffic flow	To check whether the normal traffic cycle works or not.	Sending equal number of vehicles at each lane.	The duration of the green signal at each lane is same.	The duration of the green signal at each lane is same.	Pass
2	Camera connectivity	The connectivity of camera is tested to capture images or videos.	Inputting the program to capture the real time image or video.	Capture image or video.	Error.	Fail.
3	Executing the algorithm	Executing algorithm to detect the vehicles and classify them.	Image captured by camera.	Number of vehicles and value of delay of green signal.	Number of vehicles and value of delay of green signal.	Pass
4	Congestion control	Capturing the image using camera and sending to algorithm to detect vehicles.	Connect camera and send image to system.	Value of delay of green signal.	Value of delay of green signal.	Pass

Test case #	Test case Name	Test case Description	Inputs	Expected Output	Actual Output	Status
5	Emergency Vehicle detection	RFID tag is tested to detect the emergency vehicle.	Taping the RFID key on the board.	Turn on green signal for emergency vehicle immediately.	Turn on green signal for emergency vehicle immediately.	Pass
6	Camera connectivity	The connectivity of camera is tested to capture images or videos.	Inputting the program to capture the real time image or video.	Capture image or video.	Error.	Fail.
8	SmartFlow testing	Congestion control and emergency vehicle detection is tested.	More number of vehicles at one lane and emergency vehicle on other lane.	Turn on green signal for emergency vehicle immediately.	Turn on green signal for emergency vehicle immediately.	Pass

8. Results and Discussion

Vehicle Detection

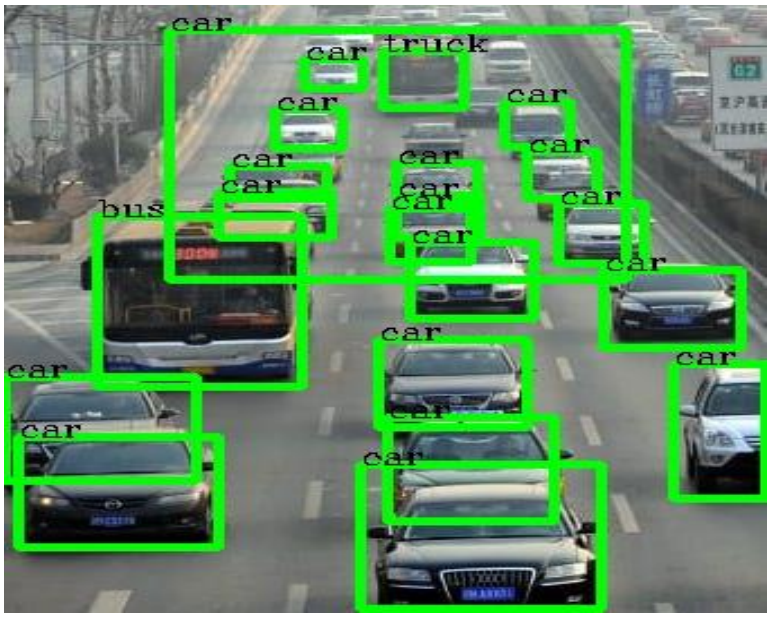

Figure 4: Vehicle Detection

As shown in the above figure, the YOLO algorithm classifies them into categories such as car, truck, or bike using YOLO's class probabilities along with bounding box coordinates. Then counts the number of vehicles and estimate the green delay.

Congestion control

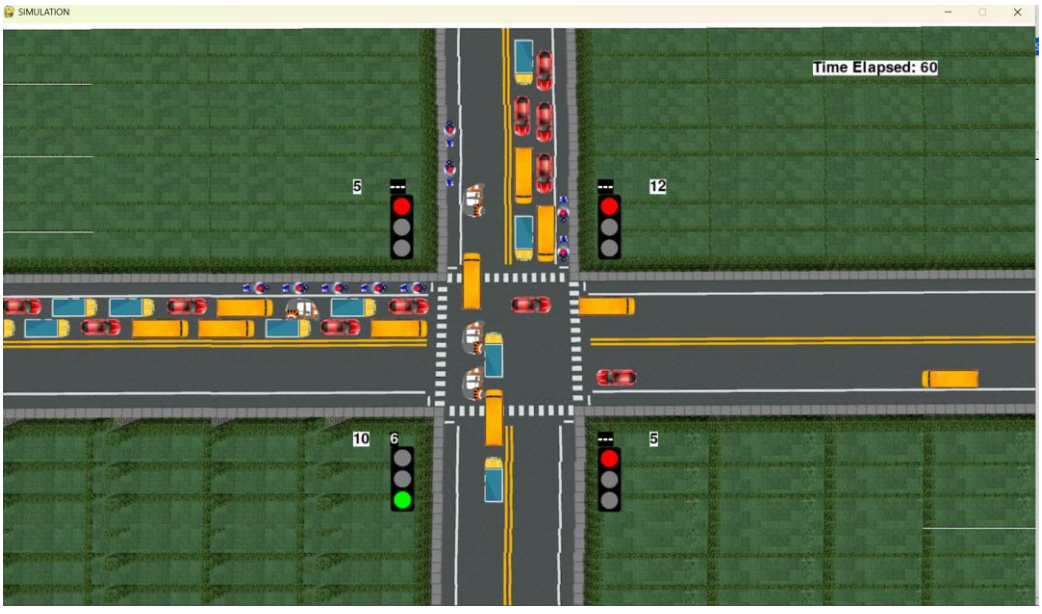

Figure 5: Congestion control

As shown in the above figure, the green delay is updated as it was estimated by the YOLO algorithm in order to control congestion.

9. Conclusion

The proposed system SmartFlow will control the traffic congestion based on the density that is by calculating the number of vehicles in the lane and adjust green delay according to the number of vehicles and the emergency vehicle is detected using RFID tag so that it does not experience any traffic or delay in reaching hospital.

This project provides an effective solution for rapid growth of traffic flow particularly in big cities which is increasing day by day and traditional systems have some limitations as they fail to manage current traffic effectively. The smart traffic management system is proposed to control road traffic situations more efficiently and effectively. It changes the signal timing intelligently according to traffic density on the particular roadside and regulates traffic flow by capturing the image. This project also makes sure that emergency vehicle like ambulance will not suffer from any interruption due to traffic signal so that ambulance can reach hospital as soon as possible.

10. Scope for Further Enhancement

For the proposed system controls the traffic congestion using camera and detect the emergency. For this system we can implement the accident detection and inform the nearby ambulance, also we can implement the Li-fi navigation so that tit will be easy for the ambulance driver to take the nearest rout to hospital with low traffic experience. Following the traffic rules id also important. We can implement the traffic rule violation detecting. The cameras will take the images of vehicles who violate the rules and by recognizing the plate number we can send message to the driver.

11. Bibliography

1. Dr. Vikram Bali, Ms. Sonali Mathur, Dr. Vishnu Sharma, Dev Gaur researched on Smart Traffic Management System using IoT Enabled Technology, 2020 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN). https://ieeexplore.ieee.org/document/9362753
2. Anil Kumar Biswal, Debabrata Singh, Binod Kumar Pattanayak, Debabrata Samanta and Ming-Hourb Yang worked on IoT-Based Smart Alert System for Drowsy Driver Detection (Received 29 December 2020; Revised 18 January 2021; Accepted 10 February 2021; Published 10 March 2021). https://www.hindawi.com/journals/wcmc/2021/6627217/
3. Shubham Kumar Chandravanshi1 , Hirva Bhagat2 , Manan Darji3 , Himani Trivedi worked on Automated Generation of Challan on Violation of Traffic Rules using Machine Learning paper released by International Journal of Science and Research (IJSR) in 2019. https://www.ijsr.net/archive/v10i3/SR21222190144.pdf
4. R Shreyas, Pradeep Kumar B V, Adithya H B, Padmaja B, Sunil M P worked on Dynamic Traffic Rule Violation Monitoring System Using Automatic Number Plate Recognition with SMS Feedback paper released on 2017 2nd International Conference on Telecommunication and Networks (TEL-NET 2017). https://ieeexplore.ieee.org/document/8343528
5. Roopa Ravish, Shanta Rangaswamy, Kausthub Char worked on Intelligent Traffic Violation Detection paper released on 2021 2nd Global Conference for Advancement in Technology (GCAT) Bangalore, India. Oct 1-3, 2021. https://ieeexplore.ieee.org/document/9587520
6. Niharika Mishra, Riya Mandal, Monika Rai, Harjeet Kaur worked on Navigation System using Light Fidelity Proceedings of the 2nd International Conference on Trends in Electronics and Informatics (ICOEI 2018) IEEE Conference. https://ieeexplore.ieee.org/document/8553760
7. Pedro Maximino, Rui S. Cruz, Miguel L. Pardal worked on Smart Healthcare Monitoring System For Healthy Driving in Public Transportation published paper in 2023 18th Iberian Conference on

Information system and Technologies (CSISTI) 20-23 June 2023. https://ieeexplore.ieee.org/document/10211847

8. Prof. Deepali Ahir, Saurabh Bharade, Pradnya Botre, Sayali Nagane, Mihir Shah worked on Intelligent Traffic Control System for Smart Ambulance paper released in International Research Journal of Engineering and Technology (IRJET) Volume: 05 Issue: 06 June-2018. https://www.irjet.net/archives/V5/i6/IRJET-V5I675.pdf
9. Meghana V, Prof. Anisha B S, Dr Ramakanth Kumar P worked on IOT based Smart Traffic Signal Violation Monitoring System using Edge Computing paper released on 2021 2nd Global Conference for Advancement in Technology (GCAT) Bangalore, India. Oct 1-3, 2021. https://ieeexplore.ieee.org/document/9587585
10. Nazia Parveen, Ashif Ali, Aleem Ali worked on IOT Based Automatic Vehicle Accident Alert System paper released in 2020 IEEE 5th International Conference on Computing Communication and Automation (ICCCA) Galgotias University, Greater Noida, UP, India. Oct 30-31, 2020. https://ieeexplore.ieee.org/document/9250904
11. X.S. Asha Shiny, D.Ravikumar, A. Chinnasamy, S. Hemavathi worked on Cloud Computing based Smart Traffic Management System with Priority Switching for Health Care Services paper released by Proceedings of the Second International Conference on Applied Artificial Intelligence and Computing (ICAAIC 2023) IEEE Xplore. https://ieeexplore.ieee.org/document/10140942
12. Abhishek Patni, Bhavini Mishra, Harsh Aditya, Yogesh Kumar, Rohit Sharma worked on Highway Navigation Using Light Fidelity Technology paper released by IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 2 Issue 4, April 2015. https://ijiset.com/vol2/v2s4/IJISET_V2_I4_209.pdf
13. Sarfraz Fayaz Khan researched on Health Care Monitoring System in Internet of Things (loT) by Using RFID paper released in 2017 the 6th International Conference on Industrial Technology and Management. https://ieeexplore.ieee.org/document/7917920