DAYANANDA SAGAR UNIVERSITY

KUDLU GATE, BANGALORE – 560068

Bachelor of Technology
in
COMPUTER SCIENCE AND TECHNOLOGY

Major Project Phase-II Report(20CT4802)

On

"Smart Traffic Management System using IoT and ML"

By

Aditya A Navale - ENG20CT0003
Ansh Gupta – ENG20CT0005
Ruchi Singhal – ENG20CT0022

Under the supervision of

Dr. Santosh Kumar
Associate Professor,
Department of Computer Science and Technology

SCHOOL OF ENGINEERING

Department of Computer Science & Technology
Kudlu Gate, Bangalore – 560068
Karnataka, India

CERTIFICATE

This is to certify that the work titled **"Smart Traffic Management System using Iot and Machine Learning"** is carried out by **Aditya A Navale (ENG20CT0003), Ansh Gupta (ENG20CT0005), Ruchi Singhal (ENG20CT0022),** Bonafide students of Bachelor of Technology in Computer Science and Technology at the School of Engineering, Dayananda Sagar University, Bangalore in partial fulfillment for the award of degree in Bachelor of Technology in Computer Science and Technology, during the year **2023-2024**.

Dr. Santosh Kumar
Associate Professor, Dept. of CST,
School of Engineering,
Dayananda Sagar University.

Date:

Dr. M Shahina Parveen
Chairperson CST,
School of Engineering,
Dayananda Sagar University.

Date:

Department of Computer Science & Technology
Kudlu Gate, Bangalore – 560068
Karnataka, India

DECLARATION

We, Aditya A Navale (ENG20CT0003), Ansh Gupta (ENG20CT0005), Ruchi Singhal (ENG20CT0022), are student's of the seventh semester B.Tech in Computer Science and Technology, at School of Engineering, Dayananda Sagar University, hereby declare that the project phase-II project titled " **Smart Traffic Management System using IOT and Machine Learning**" has been carried out by us and submitted in partial fulfillment for the award of degree in Bachelor of Technology in Computer Science and Technology during the academic year 2023-2024.

Student Signature
Name1: Aditya A Navale
USN : ENG20CT0003
Name 2: Ansh Gupta
USN : ENG20CT0005
Name3: Ruchi Singhal
USN : ENG20CT0022

Place : Bangalore

Date :

ABSTRACT

As cities become more densely populated, traffic management becomes increasingly complex. A new smart traffic management system is tackling these challenges by utilizing IoT (Internet of Things) and machine learning technologies. By analysing real-time traffic data, this innovative system can optimize traffic flow, increase safety, and reduce congestion, providing a seamless transportation experience for all.

With features like real-time traffic monitoring, automatic incident detection, and dynamic traffic signal optimization, this system is revolutionizing the way traffic is managed in urban areas. Whether you're driving, walking, or taking public transportation, this system is designed to make your commute safer and more efficient. Using YOLO Technology.

Overall, the smart traffic management system represents a major step forward in urban transportation technology, and has the potential to significantly improve the quality of life for millions of people around the world.

Keywords: Smart traffic management system, IoT, machine learning algorithms, CNN, YOLO, data analysis, real-time monitoring, intelligent transportation systems, case studies, future research directions.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Technology has made many elements of our life more appealing, but it has taken the transportation sector a while to realise how easily it can become rat trapped. These developments can no longer be avoided. The effects of the rapid increase in the number of vehicles and the development of intelligent vehicles have led to an increase in complexity in the road environment. Smart cities are being used in parking lots, offices, hospitals, and other sectors of the economy[1].

Multiple efforts are being created to use intelligent transportation systems to make the current transportation systems "cleverer" [2]. Every day, millions of cars travel through cities and on roads. A number of social, cultural, and economic factors influence how traffic congestion develops. The level of traffic congestion has a significant effect on emergency delays, time lost, money spent, and accidents. The extension of roadways is one strategy the majority of states are using to address this problem. This strategy does, however, still provide some difficulties. Older roadways can be expensive to demolish. The majority of urban areas don't have the open space needed for such an endeavour. It is clear that the rate at which travellers purchase cars has outpaced the creation of new infrastructure, regardless of advancements in road infrastructure. Roads are able to accommodate more vehicles as a result of expansions as well, making use of the extra capacity [3]. This supports Downs' [4] "fundamental law of highway congestion," which holds that an increase in the number of available roads inevitably results in an increase in the volume of vehicles on the road [4]. The main issue is that users' quality of life is greatly impacted by the delays brought on by congestion. In fact, there is a higher chance of accidents and other negative effects, such as stress and noise, for drivers who are caught in traffic jams. This phenomenon directly affects the rise in vitality congestion and is commonly recognized as an indirect cause of health issues.Regardless of the situation, the preset time alternates between green, orange, and red colours. This results in longer wait times, more fuel usage, and air pollution, among other negative effects. Furthermore, there is a deficiency in drivers' enhanced awareness of the condition of the roadways. By enhancing traffic signals, it is possible to create more inexpensive, sustainable.
transportation systems that better satisfy the demands of road management. Controlling the timing of traffic lights to determine the quickest path between two locations in order to

minimize traffic and cut down on travel time is the main difficulty. The proposed traffic management system utilizes traffic light control to avoid congestion[1].

Every person uses automobiles in this era of population growth, meaning that the use of vehicles has expanded dramatically. This had a negative impact on the traffic control system. There are no cutting-edge methods or algorithms for identifying moving cars on the road in order to calculate the vehicle density. All of the algorithms that have been developed up to this point only identify cars in specific weather conditions. Most vehicle detection and tracking algorithms used in traffic management don't work well in all-weather situations. In this study, we **Use CCTV to monitor road conditions**. This can help identify areas of congestion and allow for more efficient management of traffic. By using CNN, YOLO (You Only Look Once), and SSD (Single Shot Detector). CNN Algorithm, we will be capturing portable pictures captured by CCTV, pre-process using CNN technique and identify the timing using particular data captured. Based on identification, managing traffic, adjustment of signaling and time reduction.

Thus, the number of cars is determined using the You Only Look Once (YOLO) machine learning model, which enables dynamic signal allocation based on the vehicle count. A neural network model called YOLO aids in object detection. A collection of region-based convolutional neural networks makes up the suggested model(CNN) which greatly accelerate the method, more potent and efficient. Apart from identifying objects with OpenCV, The YOLO model reduces noise and eliminates backgrounds from the input image[2].

CHAPTER 2

Literature Survey

Sr. No.	AUTHOR	NAME OF PAPER	PROPOSED THEORY	LIMITATIONS
1.	C. L. Wan and K. W. Dickinson	ROAD TRAFFIC MONITORING USING IMAGE PROCESSING-A SURVEY OF SYSTEMS, TECHNIQUES AND APPLICATIONS	The document discusses road traffic monitoring using image processing. It reviews existing image processing techniques and their limitations, as well as the development of the TRIP II system, which uses a neural network for vehicle detection. The document also mentions the potential applications of image analysis systems in traffic control and planning. The authors are currently researching pattern recognition techniques for traffic monitoring. The document concludes by mentioning the future possibilities in image processing for traffic monitoring, such as in-vehicle vision systems and automatically driven vehicles.	The limitations of existing image processing techniques for traffic monitoring include low-level interpretation of sequences of images, lack of intelligence, reliance on manual abstraction of information from video pictures, difficulty in handling video images with continuously varying light levels, inability to handle rapid changes in pixel intensity, susceptibility to shadows and reflections, inability to operate in real-time, and the need for additional built-in intelligence to handle more sophisticated applications. These limitations have led to the development of systems that only observe limited portions of the full image or "windows" and do not analyze complete images.
2.	Pushpalata Patil and Prof. Suvarana Nandyal	Vehicle Detection and Traffic Assessment Using Images	The document discusses the development of automatic vehicles detection and classification using image feature. The system aims to improve traffic surveillance by accurately identifying and tracking vehicles in real-time. It also includes traffic assessment based on object count and predefined knowledge about traffic surveillance.	The limitations to this paper are even though it discusses about the datasets in real-time, there is no specific information about the practical implementation or performance. The paper also does not provide information about the evaluation metrics used.

				The paper only focuses on vehicle detection, it does not delve in other aspects of traffic management.
3.	Ali Wided, Brek Assia and Bouakkez Fatima	Traffic Management system and Traffic Light Control in Smart City to Reduce Traffic Congestion	The document discusses the problem of traffic congestion in smart cities and proposes a traffic management system and traffic light control to reduce congestion. The traditional traffic light system is unable to handle the increasing traffic, so an intelligent traffic light system is introduced that can detect traffic levels and react accordingly.	The document does not explicitly mention the limitations of the proposed traffic management system and traffic light control. However, it does mention some limitations of previous research, such as scalability and fault tolerance issues with the SUMO simulator and the use of Java-based scheduling algorithms. It is possible that the proposed system may also have similar limitations.
4.	Du, W.; Dash, A.; Li, J.; Wei,	Safety in Traffic Management Systems: A Comprehensive Survey	Traffic management systems play a vital role in ensuring safe and efficient transportation on roads. However, the use of advanced technologies in these systems has introduced new safety challenges. This survey reviews the literature on safety in traffic management systems, discussing safety issues, current research, and proposed techniques.	The limitations of existing research in the field of traffic safety are that crash data is limited. There is diffulty is labelling accident data collected in real-world scenarios.
5.	A. Rasaizadi, A. Ardestani, S. E. Seyedabrishami	Traffic management via traffic parameters prediction by using machine learning algorithms	The study focuses on the short-term prediction of traffic parameters, specifically hourly traffic volume and hourly average traffic speed, on the Karaj-Chaloos road in Iran. The research uses machine learning algorithms, namely artificial neural network (ANN) and k-nearest neighbor (K-NN), to make these predictions. The results show that the K-NN model outperforms the ANN	The limitations of the documents is that it does to use C-NN algorithm for its neural network, which is difficult for us. The feature extraction are not relevant to the data collected.

			model and the historical average benchmark in terms of prediction accuracy.	
6.	B. Gomathi and G. Ashwin	Intelligent Traffic Management System Using YOLO Machine Learning Model	The paper discusses the use of a YOLO machine learning model for an intelligent traffic management system. Conventional traffic control techniques are ineffective in managing growing traffic congestion, so there is a need for a solution that improves traffic control. The YOLO model is used to detect the number of vehicles on the road and dynamically allocate signal switching based on the vehicle count. The model analyzes input images from traffic surveillance CCTV cameras and removes backgrounds and noise to accurately detect vehicles	The limitations of this paperare the lack of empirical evidence or experimental results to support the effectiveness. The paper also lack of implementation of the proposed system.
7.	Jayesh Ambulkar, Pranay Bhoyar, Dr. N.D. Ghawghawe	Intelligent Traffic Signal Management	The document discusses the use of a YOLO machine learning model for an intelligent traffic management system. Conventional traffic control techniques are ineffective in managing growing traffic congestion, so there is a need for a solution that improves traffic control. The YOLO model is used to detect the number of vehicles on the road and dynamically allocate signal switching based on the vehicle count. The model analyzes input images from traffic surveillance CCTV cameras and removes	The paper uses outdated references and the informations are also incomplete. It lacks comparison between management systems.

			backgrounds and noise to accurately detect vehicles	

CHAPTER 3

Project Requirement Specification

3.1 Simulation Environment

3.1.1 MATLAB Environment

Use MATLAB for simulating the Smart Traffic Management System.

Implement a realistic urban road network for simulation.

3.1.2 Visualization

Create a user-friendly graphical interface to visualize real-time traffic data and simulation results.

3.2 Integration of IoT and Machine Learning

3.2.1 Data Integration

Establish a seamless integration of IoT data into the MATLAB simulation environment.

3.2.2 Algorithm Integration

Incorporate machine learning algorithms into the simulation for dynamic traffic control.

Chapter 4

Problem Definition

4.1 Problem Statement

The overarching goal of this project is to design and implement a Smart Traffic Management System capable of capturing real-time traffic dynamics through the deployment of Closed-Circuit Television (CCTV) cameras at key intersections. The system will leverage IoT to transmit the captured data to a central control unit, where sophisticated machine learning algorithms will be employed to dynamically adjust traffic signal timings. The primary focus is on achieving a responsive and adaptable traffic management system that can mitigate congestion, optimize signal timings, and enhance overall urban mobility. The current transportation systems have become "cleverer" through a variety of initiatives utilizing intelligent transportation systems. Millions of cars use the roads and cities every day. Additionally, there are several social, cultural, and economic factors that impact the development of traffic congestion. The amount of traffic congestion has a big impact on accidents, lost time, money, and emergency delays. One approach that most states are employing to deal with this issue is extending roads.

4.2 Relevance of the Problem

Traditional traffic light control systems operate on fixed timers or pre-programmed schedules. This lack of flexibility makes it difficult to adapt to changing traffic patterns and real-time conditions. As a result, congestion and delays can occur, especially during peak hours or in areas with unpredictable traffic flow.

Conventional traffic light control systems often rely on fixed cycles, where each direction of traffic is given a predetermined amount of time to proceed. This approach can lead to inefficient traffic flow, with long wait times at certain intersections and underutilised capacity at others. As cities expand and traffic volumes increase, this inefficiency becomes more pronounced.

Fig 4.1: existing system traffic management

CHAPTER 5

System Architecture

The suggested model creates a more intelligent, well-organised, and controlled traffic management system by analysing the presence of vehicles on the road and addressing issues with traffic flow. YOLO, a capable object detection machine learning model, can be used to accomplish this analyze vehicle count based on input from CCTV cameras installed in traffic signals. The traffic data input is passed into the YOLO machine learning model that counts the vehicles based on the object detection in input video stream [2].

YOLO use a totally different approach. In contrast to R-CNN systems, which need thousands of evaluations for a single image, it also generates predictions with a single network assessment. Because of this, it operates at a speed that is over 1000 times quicker than R-CNN and 100 times faster than Fast R-CNN. Benefits of Accuracy and speed (45 frames per second, better than real-time) are two advantages of YOLO over other models has a generalised object representation understanding [3].

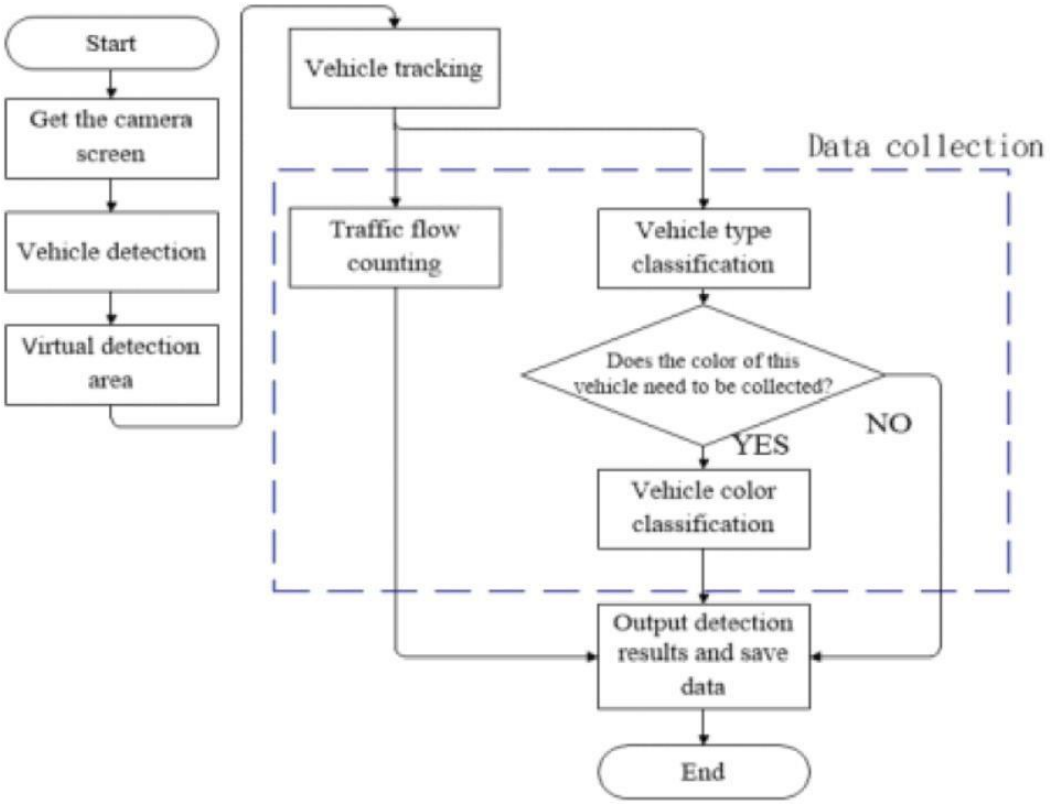


Fig 5.1: Flow diagram of Proposed system

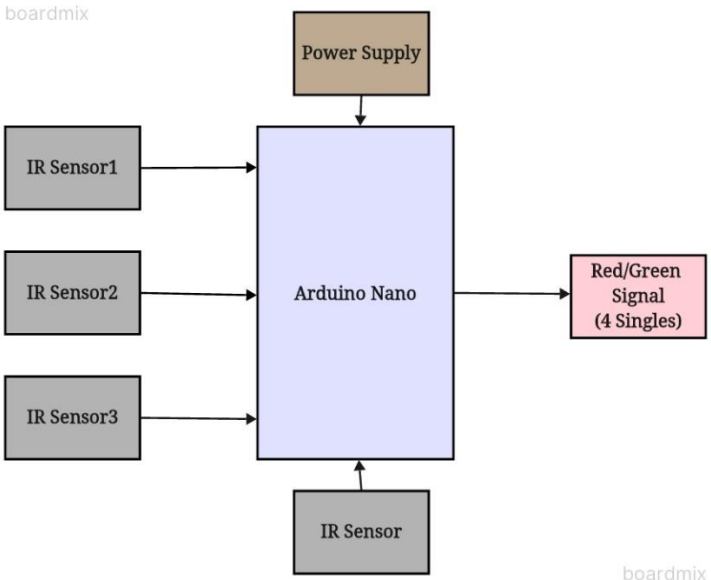


Fig 5.2: Flow diagram of IoT device

Components:

Power Supply: This provides electrical energy to all components.

IR Sensors (1, 2, 3): These infrared sensors detect the presence of vehicles at specific points on the road.

Arduino Nano: The microcontroller processes sensor data and makes decisions.

Red/Green Signal (4 Singles): Represents four sets of traffic lights (red and green).

Operation:

When a vehicle approaches an intersection:

IR Sensors detect it.

The Arduino Nano receives signals from the sensors.

Based on the sensor input, the Arduino decides whether to change the traffic light from red to green (or vice versa).

The Red/Green Signal is controlled accordingly.

Scenario:

Imagine a busy intersection:

If no vehicles are detected, the light remains green.

When a vehicle approaches, the IR sensor triggers the Arduino.

The Arduino switches the light to red for the intersecting road.

After a set time, it switches back to green for that road.

Benefits:

Efficient traffic flow: Lights adapt dynamically to vehicle presence.

Reduced waiting times: No unnecessary red lights.

Improved safety: Real-time adjustments prevent collisions.

Challenges:

Fine-tuning timing: Ensuring smooth transitions.

Sensor reliability: False positives/negatives.

Scalability: Handling multiple intersections.

In summary, this flowchart represents an intelligent traffic management system that optimizes light changes based on real-time vehicle data. It showcases the fusion of technology and urban planning for safer, more efficient roads. Sure, let me explain the process of how this traffic light management system would work step-by-step:

1. Initial State: When no vehicles are detected, the system would be in a default state, likely with one direction having a green light and the other directions having red lights.

2. Vehicle Detection: As vehicles approach the intersection from different directions, the corresponding IR Sensor1, IR Sensor2, or IR Sensor3 would detect their presence.

3. Signal to Arduino: The IR sensor(s) that detected the vehicle(s) would send a signal to the Arduino Nano microcontroller.

4. Arduino Processing: The Arduino Nano would be programmed with an algorithm that takes input from the IR sensors and determines the appropriate action based on factors like:

- Which sensor(s) detected vehicles
- The current state of the traffic lights
- Predefined rules for traffic light sequencing and timing

5. Light Sequence Adjustment: Based on the programmed algorithm, the Arduino Nano

would send signals to the Red/Green Signal (4 Singles) lights to change their state accordingly. For example:

- If vehicles are detected on the road with the green light, the algorithm may extend the green light duration.
- If vehicles are detected on a road with a red light, the algorithm may trigger a sequence to change that light to green after a safe interval.

6. Intersection Monitoring: The additional IR Sensor at the intersection itself could be used to detect vehicles that have entered the intersection, allowing the Arduino to make further adjustments if needed, such as extending the green light or preventing conflicting green lights.

7. Cycle Repetition: The process of vehicle detection, signal processing, and light sequence adjustment would continue in a loop, with the Arduino constantly monitoring the IR sensors and updating the traffic lights based on the programmed algorithm.

8. Power Supply: Throughout this process, the Power Supply would provide the necessary electrical power to all components, including the Arduino Nano, IR sensors, and traffic lights.

The key aspects of this system are the use of IR sensors to detect vehicle presence, the Arduino Nano as the central processing unit running a predefined traffic light control algorithm, and the ability to dynamically adjust the traffic light sequence based on real-time vehicle detection data.

It's important to note that this is a simplified example, and in practice, more advanced algorithms, additional sensors (e.g., for pedestrian detection, emergency vehicle detection), and fail-safe mechanisms would likely be implemented for a robust and safe traffic light management system.

The above-described YOLO object detection method can be used to identify a vehicle and its location information from a single picture. However, in actual traffic applications, a continuous image frame is provided as the input. The vehicles detected in different image frames are independent of each other. Therefore, the same vehicle is counted multiple times, and the

collected vehicle information would be wrong. To solve this problem, the ID of the detected vehicle must be configured to prevent double counting. In the proposed system, an object counting method is added to correlate and match the vehicles detected in different image frames and to determine whether a detected vehicle is newly added[5].

For the proposed approach to accomplish the goal of vehicle counting, only basic distance computations are needed. Additionally, the system can lessen the effects of incorrect detection by including more checkpoints. The accuracy and overall performance of the suggested system are examined using pictures taken from various viewpoints and locations.

The results show that our system can count objects with a high degree of precision in an environment with sufficient daylight.

Fig 5.3: Real Time traffic Monitoring

Objectives:

1. Collect the data (video or images) from traffic signals.
2. Preprocess the video or image to identify the vehicles.
3. Analyze the traffic based on vehicle count and time stamp.
4. Automate the traffic signal flow based on analysis.

CHAPTER 6

Implementation

Methodology:

1. Data collection using high end cameras.
2. Using machine learning algorithms preprocess the data.
3. Data analysis using machine learning.
4. Automate the traffic using IoT components.

6.1 Traffic Scenarios

Define and simulate various traffic scenarios, including peak hours, special events, and unexpected incidents.

Evaluate the system's response to different traffic conditions.

6.2 Performance Metrics

Implement metrics for evaluating the system's performance, such as traffic flow improvement percentage, average travel time, and congestion reduction.

6.3 Real-time Monitoring

Verify real-time monitoring capabilities through the GUI, ensuring accurate representation of simulation data and machine learning predictions.

6.4 Steps to Run on Matlab

Open MATLAB:

Launch MATLAB on your computer.

Navigate to the Code Directory:

Use the MATLAB command window or the MATLAB interface to navigate to the directory where you extracted or cloned the GitHub repository.

Review Documentation:

Check if the repository includes any documentation or README file. This file often provides instructions on how to use the code.

Set MATLAB Path:

Add the code directory to MATLAB's path to ensure that MATLAB can access the functions and scripts.

addpath("C:\Users\AdityaNavale\Dropbox\MyPC(DESKTOP-R5IBUNU)\Downloads\OpenTrafficLab-master");

OpenTrafficLab: Simulating simple traffic scenarios with provided human driving models, or user defined models

This example shows how to use the OpenTrafficLab environment in MATLAB® to simulate simple traffic scenarios. By the the end of this document you will have seen:

How to create a drivingScenario object and link it with OpenTrafficLab in order to generate traffic simulation.

How to assign junction controllers to intersections and their turns and movements.

How to create vehicles and choose a driving logic for them, as well generate their entry times by specifying injection rates at entry nodes.

How to simulate and visualize the created scenario

How to create your own vehicle and junction controllers to simulate the performance of your own algorithms.

We will use several helper functions contained in the Testing folder of the repository, so we first add it to the path:

```
% Add that folder plus all subfolders to the path.
addpath(genpath(pwd));
```

OpenTrafficLab builds upon the structure of the Driving Scenario Designer App, and its drivingScenario object by adding three main classes used for simulaiton.

A Node class that is used to represent the road network as a directed graph where each node correspond to a lane within a road segment or turn/maneuver at an intersection. The edges of the graph indicate how cars can go from one node to the next, which in turn encodes the allowed traffic flow directions and turns.

A DrivingStrategy class that serves as a vehicle controller. To implement a desired control logic, users can create a child class to DrivingStrategy and override the main methods where the driving logic is instantiated. The nominal logic implements standard car following models, namely the Gipps model or the Intelligent Driver Model (IDM) suitably modified to respong to upcoming traffic controllers (traffic lights etc...)

A TrafficController (JunctionStrategy) class that allows users to model traffic signalization and control like traffic lights, stop signs or any user defined logic.

The TrafficController is an Actor in the simulation that actts on a group of Nodes, say all turns/maneuvers at a junction, and controls wether each one is open or closed at any given time. A closed Node will a act as a stationary vehicle for incoming traffic, which will make the vehicles stop at the exit of the previous lane. Users can inherit from TrafficController to implement their own control logic. We provide one such inherited class: TrafficLight, which implements the logic of a traffic light controller.

Simulations can be run by creating the road network and its associated nodes, populating the scenario with actors and their control strategies (i.e DrivingStrategy and TrafficController), and then calling the advance function like in other drivingScenario simulations.

Creating drivingScenario object and connecting it to OpenTrafficLab

Drawing the roads

The simulator builds heavily on the Driving Scenario Designer and its different classes. The first step in running a simulation is to create a drivingScenario object containing the road segments of the network. Each turn or maneuver at a junction must be defined as its own road segment. Two different functions are provided for generating the drivingScenario object, one for a T-junction, and one for a 4-way intersection. In this example, we will use the T-Junction scenario. The createTJunctionScenario creates programatically 3 two lane roads that feed an intersection, and it creates 6 road segments at the intersection, one for each possible turn cars can make.

```
scenario = createTJunctionScenario();
```

We can set the simulation's final time and time step, in seconds, here:

```
scenario.SampleTime = 0.1;
scenario.StopTime = 200;
```

We can plot the scenario using plot.

```
plot(scenario)
```

Note

Road segments for use in OpenTrafficLab need to be created carefully. In the Driving Scenario Designer App, roads that overlap ceate common road tiles for both roads and lane markings are not shown; however, no new road segments that represent the possible turns in the intersection are created. For the purposes of traffic simulation using OpenTrafficLab, you will usually need to create road segments for every possible turn at an intersection.

Create the OpenTrafficLab graph representation of the network

The simulator represents the network in a graph of Node objects (See Node class documentation). The Node object is associated with a single lane of a single road segment. Upon creation, the Node object generates a mapping between global position and station coordinate < Title of your Project>

(i.e. the distance travelled along the length of the lane). We can create the Node objects and specify their connectivity for the above graph using the corresponding helper function, which takes as input the scenario we just created. See help Node, and the comments on the helper function for details on how to assign each lane to a new Node.

```
network = createTJunctionNetwork(scenario);
```

There are three lanes feeding the intersection, three lanes leaving it, and 6 possible maneuvers at the intersection. There are therefore 12 Nodes in the network we just created.

```
length(network)
```

```
ans = 12
```

Each of these Nodes has different properties used by the simulator and its vehicles and traffic controllers. ConnectsTo, ConnectsFrom, and SharesRoadWith are links to other Node objects that define the connectivity of the network. The Scenario, RoadSegment and Lane property are set upon initialization and connect the Node object to the underlying drivingScenario object. The Length and Mapping properties define the mapping between the station distance along the length of the lane and the global position; the Mapping property also contains information about the direction/curvature of the lane at each station distance. The Vehicles property stores a handle to vehicles currently travelling down the Node, and the TrafficController property points to the agent controlling the traffic at node, which is done by either allowing or not allowing vehicles to enter.

```
network(1)
```

```
ans =
Node with properties:

          ConnectsTo: [1×2 Node]
        ConnectsFrom: [0×0 Node]
      SharesRoadWith: [1×1 Node]
            Scenario: [1×1 drivingScenario]
         RoadSegment: [1×1 driving.scenario.RoadSegment]
                Lane: -1
              Length: 50
             Mapping: [501×6 double]
            Vehicles: [0×0 driving.scenario.Vehicle]
          PlotHandle: []
   TrafficController: []
```

We can use the plotPath method of the Node class to show the nominal trajectory a vehicle would follow within each node. For example, Nodes 7 through 12 correspond to all the maneuvers at the intersection, let's visualize those trajectories.

```
plot(scenario)
plotPath(network(7:12));
```

You can use the ConnectsTo property to navigate the networ of Nodes. For example, let's plot the paths of the first entry Node, and the two Nodes it connects to at the intersection.

```
plot(scenario)
plotPath([network(1) network(1).ConnectsTo]);
```

Creating a junction controller

Now that we have created a road network and its OpenTrafficLab graph representation, we can create a controller for the intersection using the TrafficController class. Specifically, we will create a TrafficLight object which inherits from the TrafficController class.
When created, all TrafficController objects need a single input:

A vector of Node objects which the traffic controller operates on. Traffic is controlled by setting Nodes to be either open or closed to incoming traffic. Wehn a vehicle's upcoming Node is closed, the vehicle should stop.

The TrafficLight object in particular also needs two other inputs:

The first is a list of indeces that indicate to which clique (or phase group) each Node belongs to; this input is a vector which assigns each node to its clique.

The second input is a vector indicating the timing, in seconds, of the switches from one phase to the next. Each entry in this vector indicates when the light switches from servicing one clique to the next, in the order in which they are defined by the previous input.

```
% Rebuild scenario and network to clear plots
scenario = createTJunctionScenario;
network = createTJunctionNetwork(scenario);
scenario.SampleTime = 0.1;
scenario.StopTime = 100;
```

```
%Identify which Nodes in the network belong to the junction that will be controlled
nodesInJunction = network(7:end);
% Assign each Node to a its corresponding clique (i.e. signal phase)
cliques = [1,1,2,2,3,3];
% Create the fixed time cycle of the light
cycle = [0,10,35,45];
% Create the traffic light actor
trafficLight = trafficControl.TrafficLight(nodesInJunction,'Cliques',cliques,'Cycle',cycle);
```

Creating vehicles, their drivers and their entry times

The main actors in the simulation are the vehicles. In OpenTrafficLab, as in Driver Scenario Designer, vehicles are Vehicle objects, and they are controlled by a MotionStrategy object. In a Driving Scenario Designer scene, the motion strategy is normally a predetermined space-time trajectory, created using the trajectory function. In OpenTrafficLab, the motion strategy is a DrivingStrategy object (see DrivingStrategy documentation), which allows closed loop simulation.

To create vehicles then, we must create two objects and link them together:

First, we create a Vehicle object, using the function vehicle. The key input for this construction is the vehicle's entry time.

Second, we create a DrivingStrategy object (or a child class of it). The main and only required input for the DrivingStrategy is the vehicle's path through the network. This is given by a list Nodes in the order which the vehicle will traverse the network, including its first Node.

To facilitate the modelling of the arrival time, the simulator includes a function that generates arrival instances according to a Poisson arrival proocess. See drivingBehavior.generatePoissonEntryTimes.
For this example we will use the createVehiclesForTJunction helper function to create the vehicles and their drivers. This helper function takes as input:

The drivingScenario object

The list of Nodes in the network

The injection or arrival rate in vehicles per hour at each entry Node of the network (i.e .Nodes 1, 2, and 3 of the array network). This is given as a 3-by-1 vector of arrival rates.

The turn ratio for vehicles at the intersection. This is a 2-by-1 vector (or 3-by-1 if using the four-way-intersection scenario) indicating what ratio of vehicles turn into the first or second Node that their entry Node connects to. That is, for vehicles entering in network(1), a turn ratio of [40, 60] will create paths for vehicles so that 40% of vehicles procede into network(1).ConnectsTo(1), and 60% proceed to network(1).ConnectsTo(2)

Optional: A function handle to the DrivingStrategy constructor to be used. The default uses the standard DrivingStratgy construct, which would create the nominal DrivingStrategy object. See DrivingStrategy for details on how the nominal driving strategy works. To create user defined drivers, create a function handle using the constructor of a user defined class which inherits from DrivingStrategy.

```
InjectionRate = [700, 700, 700]; % veh/hour
TurnRatio = [30, 60];
fnc = @(varargin) DrivingStrategy(varargin{:},'CarFollowingModel','Gipps');
cars = createVehiclesForTJunction(scenario,network,InjectionRate, TurnRatio,fnc);
```

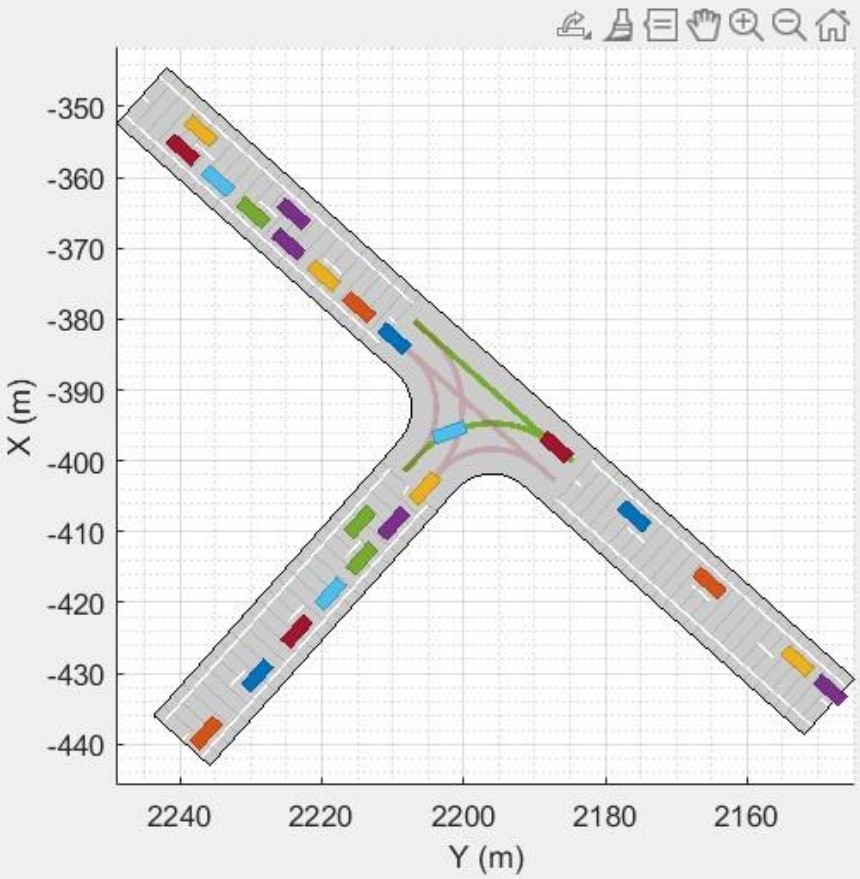
Simulating traffic

Traffic is now ready to be simulated. To run a simulation we call the advance function on the drivingScenario object. We can use the plotOpenPaths of the TrafficLight class to show the state of the traffic light at the intersection

```
% Make figure a pop-up, for animation purposes
figure('visible','on')
ax = gca;


% Plot the Scenario
plot(scenario,'Parent',ax)
```

```
%Advance the Scenario
 while advance(scenario)
    plotOpenPaths(trafficLight)
   drawnow limitrate
 end
```




```
Final output after stimulation
```

Fig 6.1: Output of traffic simulation

CHAPTER 7

Conclusion

The primary objective of intelligent traffic management systems is to address the traffic issue that nearly all urban and suburban regions face. Therefore, the goal of the suggested model is to lessen traffic caused by cars. The proposed model decides smart switching timing for the signal on all sides of the road. Hence, people do not want to wait for an extended interval of time on the road and provide a smooth flow of traffic on the road. Since the system learns from time to time, the system can be updated in future to become fully autonomous with training and learning.

REFERENCES

[1] Ali Wided, "Traffic Management system and Traffic Light Control in Smart City to Reduce Traffic Congestion". July 2023

https://www.researchgate.net/journal/International-Journal-of-Automation-and-Smart-Technology-2223-9766?_tp=eyJjb250ZXh0Ijp7ImZpcnN0UGFnZSI6InB1YmxpY2F0aW9uIiwicGFnZSI6InB1YmxpY2F0aW9uIn19

[2] **Gomathi Babu** , "Intelligent Traffic Management System Using YOLO Machine Learning Model". July 2022.

(PDF) Intelligent Traffic Management System Using YOLO Machine Learning Model (researchgate.net)

[3] **Jayesh Ambulkar**, **N. D. Ghawghawe**, "Intelligent Traffic Signal Management". July 2023

(PDF) Intelligent Traffic Signal Management (researchgate.net)

[4] Paul Viola, Michael Jones, “Rapid Object Detection using a Boosted Cascade of Simple Features” ACCEPTED CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION 2001.

[5] Cheng-Jian Lin; Jyun-Yu Jhang, "Intelligent Traffic-Monitoring System Based on YOLO and Convolutional Fuzzy Neural Networks". Published : 2021

Intelligent Traffic-Monitoring System Based on YOLO and Convolutional Fuzzy Neural Networks | IEEE Journals & Magazine | IEEE Xplore

[6] Mallikarjun Anandhalli , Vishwanth P. Baligar , Pavana Baligar , Pooja Deepsir , Mithali Iti, "Vehicle detection and tracking for traffic management"

IAES International Journal of Artificial Intelligence (IJ-AI) Vol. 10, No. 1, March 2021

[7] A. Rasaizadi, A. Ardestani, S. E. Seyedabrishami, "Traffic management via traffic parameters prediction by using machine learning algorithms". Published: 09 September 2020

International Journal of Human Capital in Urban Management (IJHCUM)

[8] Mohammed A. A. Al-qaness, Aaqif Afzaal Abbasi, , Hong Fan, Rehab Ali Ibrahim, Saeed H. Alsamhi & Ammar Hawbani

"An improved YOLO-based road traffic monitoring system". Published: 06 January 2021

An improved YOLO-based road traffic monitoring system | SpringerLink

[9] B. Gomathi & G. Ashwin

"Intelligent Traffic Management System Using YOLO Machine Learning Model". Published : 02 August 2022

Intelligent Traffic Management System Using YOLO Machine Learning Model | SpringerLink

[10]Huansheng Song, Haoxiang Liang, Huaiyu Li, Zhe Dai & Xu Yun

"Vision-based vehicle detection and counting system using deep learning in highway scenes". Published: 30 December 2019

Vision-based vehicle detection and counting system using deep learning in highway scenes | European Transport Research Review | Full Text (springeropen.com)

[11] Prof. Saraswati Nagtilak1 , Parth Jadhav2 , Shreyas Chaudhar3 , Pratik Bhosale4 ,Om Godase, "Smart Traffic Management using Deep Learning". Volume 3, Issue 2, March 2023

International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)