Adaptive Traffic Control System

Prof. Ganesh Ubale[1], Atharva D. Kavade[2], Hardik S. Khade[3], Vivek N. Kendre[4], Sushant A. Katare[5], And Shripad P. Kanakdande[6]
Vishwakarma Institute of Technology, Pune, India

***Abstract*—The traffic congestion has become one of the major challenges, along with growing population and numbers of vehicles in cities. Traffic congestion causes not only delays and additional stress for car drivers but increases fuel consumption and air pollution. Although the issue is very commonly faced, in megacities, it is much more serious. The continuous growth underlines the demand for real-time traffic density measurement due to its importance for efficient signal control and traffic management. Traffic controllers need to play a very important role in regulating the flow of vehicles smoothly, and optimization of their operations is hence quite necessary in handling the growing demand. Our proposed system uses live camera feeds from the traffic junctions to measure vehicle density with image processing and AI techniques. It can also dynamically adjust the phases of a traffic light through an algorithm, which takes into consideration vehicle density, to reduce congestion and allow smoother flow on the roads and minimize levels of pollution**

I. INTRODUCTION

The traffic congestion in every city is caused by the growing number of vehicles, which, in turn, fashioned many road networks to have reduced capacity with subsequent reduced Levels of Service. An important reason for these traffic-related problems lies in the use of traffic control systems dependent on fixed timers of signals. They operate with the same sequence and duration for phases concerning traffic without adjustment to real traffic conditions. The new demand for road capacity urges the development of innovative traffic control, some of which is being developed under Intelligent Transport Systems.

The two case studies will be further discussed, those of Mumbai and Bangalore. A report that analyzed traffic in 416 cities within 57 countries reported that Bangalore has the worst flow of traffic in the world, while Mumbai ranks fourth. During peak hours, travel times in Bangalore are extended by 71%, and in Mumbai by 65%.
Currently, there are three big schemes of traffic control in operation:

Manual Control: The name itself suggests that this involves human control, wherein the police controlling the traffic regulate the flow of vehicles. This system requires a lot of manpower, which is hardly available since there are only a few traffic personnel. Hence, manual control cannot be implanted efficiently in every area of the city or town, and therefore, such an inefficient system requires the need for something more effective.

Static Control: In real life, road traffic does not change the fixed time of each phase in an intelligent traffic light system.

Electronic Sensor-Based Control: Advanced proximity sensors and loop detectors are employed to collect traffic data. However, here exists a trade-off between the quality and extent of coverage. The better the quality of collected data, the more technologically advanced and, thus, expensive the sensor technology is, so such means cannot always allow as much sensor deployment as highly restricted by budget. Besides this, these sensors have a very limited range; to assure adequate coverage of an entire network, many devices would be needed. Only in recent times, video monitoring and surveillance systems have widely been applied to traffic management for security purposes, ramp metering, and to provide real updates to the current travelers. These systems allow many other technologies that estimate traffic density and classify vehicles, enabling optimization of traffic signal timers and flow efficiency.

The system we propose will integrate an adaptive traffic light controller through Computer Vision that can respond dynamically to real-time conditions in traffic. It calculates the density of the vehicles using live feeds from traffic intersection CCTV cameras by counting the amount of vehicles waiting at a signal. Based on this, it sends the signal to adjust the green signal time. The type of vehicle is divided into car, bike, bus/truck, and riksha to get an approximate green signal time. It detects the traffic with the help

of the YOLO algorithm and times each signal direction depending on the number of vehicles it detects. This way, the technique optimizes this technique in such a way that the green light time for each stream of traffic is optimally utilized to pass through faster unlike the static system. Therefore, frustrating delays and congestions are eliminated, and because long-waiting times reduce, fuel consumption minimizes, as well as the emission of pollution.

II. LITERATURE REVIEW

Reference [2] describes a solution involving video processing. The continuous stream of C++ algorithm execution at the servers processes and analyzes the live video feeds. A performance comparison between hard-coded versus dynamically coded methods shows that the dynamic algorithm yields a 35% improvement in performance.

Reference [3]: To reduce congestion and hence waiting time, this method involves an Arduino-UNO system to begin the process. Images are captured somewhere in traffic using a camera and processed through MATLAB. Original images are converted into threshold images by removing their saturation and hues in order to calculate the traffic density. A USB connection is established between Arduino and MATLAB; then, simulation packages are installed. Arduino calculates the green light time for each lane given the traffic count and the traffic density. The system will have some problems with overlapped vehicles from correctly counting the number of passing vehicles. Moreover, billboards and trees in view are mistakenly detected as vehicles due to black-and-white sifting.

Reference [4] introduces a traffic light system, leveraging fuzzy logic control to adapt to the current condition of the flow. It uses two fuzzy controllers with three inputs and one output for primary and secondary roads. Simulations done in VISSIM-MATLAB show a better flow of traffic in low-density scenarios.

Reference [5]: The system is developed through the integration of ANN and a fuzzy controller. Cameras capture the images, which then undergo grayscale and normalization. Segmentation of the image is done through the segmented sliding window technique, which caries out the vehicle counting without considering the vehicle size. The ANN processes the data from segmented images, while the fuzzy controller adjusts the red and green light timer values based on the output given. The average error rate is 2% with a run time of 1.5 seconds.

Reference [6]: The method proposed here uses a support vector machine algorithm, combined with image processing. The frames extracted from live videos are changed into grayscale and, after processing through OpenCV, the SVM algorithm is applied. This way, the system can estimate the density of the traffic and detect traffic red light violations.

Reference [7] proposes an adaptive traffic light timer using image processing and traffic density. It is made of microcontroller-controlled timers, high-resolution image sensors, MATLAB, and UART-based data transmission. This proposed system does not provide any priority to emergency vehicles and also to any accidents at the intersection.

Reference [8]: A review of various traffic light management techniques is presented by this paper and points out a general framework followed by most of them: choice of input data, acquisition of traffic parameters, data processing, determination of density, and update of parameters.

Method 1: This approach is based on gathering vehicle data and locations using VANETs by means of GPS. The intelligent traffic lights update traffic statistics using this information and broadcast it to vehicles within that vicinity. In case of an emergency, the system will recommend diversion. Its deployment, however, costs much more.

Method 2: Special vehicle IDs are detected through a transmitter and receiver with infrared sensors. A radio frequency tag identifies the emergency vehicle, and violations of red light are identified. However, this is a limited method since infrared sensors require line-of-sight.

Method 3: This involves the application of fuzzy logic controllers to optimize the signals at an intersection. Cameras capture data on the flow of incoming and outgoing traffic, process information, and make decisions based on how congestion can be minimized.

Method 4: Another fuzzy logic-based approach considers the number of vehicles and their average

speed as the inputs. Sensors on the road collect data to change the timings accordingly.

Method 5: The photoelectric sensors, spaced apart on either end, record traffic data transmitted into a traffic cabinet for processing. The cabinet calculates the weight of each road and, therefore, adjusts the lights of traffic. However, maintenance is very expensive.

Method 6: It follows video imaging to track road traffic. The dynamic background subtraction, through morphological operation, increases the visibility in the traffic flow of vehicles. Each coming vehicle in the view area is marked by a new rectangle. For every car present in each rectangle, a counter is increased. This fitted method has several drawbacks, such as the inability to manage occlusions and overlap of shadows.

III. PROPOSED SYSTEM

A. Proposed System Overview

The proposed system would calculate the traffic density in real-time by analyzing images captured by CCTV cameras installed at traffic junctions through image processing and object detection. Data from captured images undergo, as illustrated in Figure 1, a vehicle detection algorithm making use of YOLO. Vehicles detected would fall into various categories of traffic, used by the system to calculate traffic density. This serves as an input for a signal switch algorithm that dynamically alters the green light time of each lane. The red light times are calculated as usual, but the green light times are kept within minimum and maximum limits to avoid lane starvation. A simulation model depicts the efficiency of this system compared to conventional static traffic signals.

B. Vehicle Detection Module

It is identified that real-time object detection using the later-end version of a Convolutional Neural Network forms the kernel of the vehicle recognition module, identified as YOLO. Training of a YOLO model using objects such as cars, bikes, heavy vehicles like buses and trucks, and rickshaws was performed for various categories of vehicles.

YOLO runs a single neural network over the entire image. YOLO divides the image into regions and predicts bounding boxes and probabilities for objects in each region. The bounding boxes are weighted with their respective probabilities. YOLO balances high accuracy and real-time performance well, as it needs only one network pass to make predictions. In order for every object to be detected only once, YOLO uses a technique called non-max suppression, returning the objects detected along with their bounding boxes.

The YOLO framework is based on Darknet—an open-source neural network framework in C and CUDA. Darknet allows computation performance both on CPU and GPU services, promising pace and simplicity. By implementing Darknet, YOLO achieved impressive accuracy levels of 72.9% and 91.2% top-5 accuracy on ImageNet.

Images were collected from the internet and, using LabelIMG, a graphical annotation tool, manually labeled for the training dataset of YOLO. The pre-trained weights of YOLO were downloaded, adapted to detect four categories of vehicles (cars, bikes, buses/trucks, and rickshaws), and fine-tuned with the model to reduce loss. The resulting weights were integrated into the system.

Before detection, the model pre-processed the images through the OpenCV library. The output data comes in JSON format: labels are keys, while confidence levels with coordinates make up the values. Using OpenCV, bounding boxes for the detected objects were drawn on the images. Examples of the results are given in Figure 1, showing original images together with processed images including bounding boxes and labels.

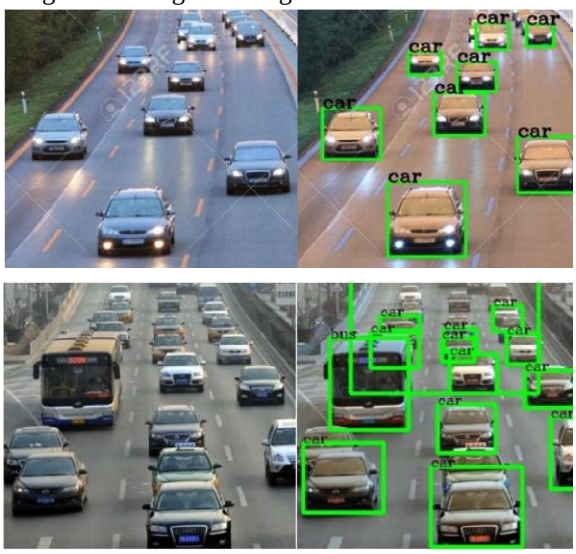

Fig 1. Vehicle Detection Result

C. Signal Switching Module

The signal switching algorithm dynamically updated the green light durations based on data provided concerning traffic density by the detection module. The updated red light timings are provided for the other signals to ensure smooth transitions. Signals cycle in a fixed order to preserve consistency with existing systems.

The algorithm considers several features:

- Processing Time: The time taken to analyze the density of the traffic and to calculate the duration of the green light, which decides when images are to be captured.
- Number of Lanes: Total number of lanes at the intersection or junction.
- Vehicle Counts: Number of vehicles in each category, including cars, bikes, and trucks.
- Density of Traffic: Deduced from the factors above.
- Start-Up Lag: Time consumed by vehicle start-up and additional time for vehicles further back in the queue.
- Speeds of Vehicles: Average speed for each category of vehicle when the green light begins.
- Timing Constraints: Minimum and maximum green light durations to avoid lane starvation.
- The algorithm starts with the default green light time for the first signal. Subsequent signals are dynamically timed based on traffic density and vehicle categories. Vehicle detection is handled by a separate thread, while signal timer management is handled by the main thread. Images are taken five seconds in advance of the signal turning green, providing enough time (10 seconds) for processing, density calculation, and timer adjustment.

The green light duration (GST) is defined by:

$$GST = \frac{\sum_{vehicleClass} (NoOfVehicles_{vehicleClass} * AverageTime_{vehicleClass})}{(NoOfLanes + 1)}$$

Where:

- GST : Green Signal Time
- No Of Vehicles Of Class: Number of vehicles in each class
- average Time Of Class: Average time a vehicle category needs to cross
- noOfLanes: Number of lanes

It performs a cyclic switching of the signals:

Red → Green → Yellow → Red

D. Simulation Module

A simulation done using Pygame illustrates the proposed system and some static counterparts for comparison. The simulation depicts a four-way intersection, with signals displaying timers for green, yellow, and red transitions, and showing vehicle counts for each signal. The vehicles' entry at the intersection is randomized for cars, bikes, buses, and rickshaws. Some of the vehicles in the rightmost lane turn while crossing, adding realism. Random behaviors are assigned to vehicles while generating them to simulate real-life scenarios. A timer marks the simulation time elapsed, and Figure 3 shows the simulation output.

Pygame is a set of Python modules designed for writing video games. It allows programmers to write fully featured games and multimedia programs in Python. It is highly portable and an extension of the SDL library, making it well-suited for this simulation. Its portability and LGPL license are additional advantages.

Fig 2. Simulation Module

IV. RESULTS

Use either SI (MKS) or CGS as primary units. (SI units are strongly encouraged.) English units may be used as secondary units (in parentheses). This applies to papers in data storage. For example, write ―15 Gb/cm^2 (100 Gb/in^2).‖ An exception is when English units are used as identifiers in trade, such as ―3½ in disk drive.‖ Avoid combining SI and CGS units, such as current in amperes and magnetic field in oersteds. This often leads to confusion because equations do not balance dimensionally. If you must use mixed units, clearly state the units for each quantity in an equation.

The SI unit for magnetic field strength H is A/m. However, if you wish to use units of T, either refer to magnetic flux density B or magnetic field strength symbolized as μ0H. Use the center dot to separate compound units, e.g., ―A·m^2.‖

V. CONCLUSION AND FUTURE WORK

The basic logic of the system lies in changing the time period of the green signal based on the traffic density. Heavier directions would be given more priority by extending the time duration of the green light. Consequently, unnecessary delays will be reduced, congestion will decrease, and drivers will not have to waste much time waiting, which wastes more fuel and accordingly increases pollution.

The simulation results confirm that the system improves traffic flow by 23% compared to the current methods of traffic management, which is quite significant regarding the number of vehicles passing through intersections. The further optimization of this system using real-life CCTV data refines its performance even further.

A number of advantages exist in the proposed system compared to conventional intelligent traffic control systems using pressure mats and infrared sensors. The deployment costs are hence low since some of the existing CCTV cameras at junctions can be utilized. Maintenance costs too will be less because no components such as pressure mats, which may wear out after continuous use, need to be used. This is thus well-suited for implementation along with the existing CCTV infrastructure in big cities in order to manage the flow of traffic more successfully.

Other features that could be integrated into future expansions of this project include:

- Traffic Rule Violation Detection: Vehicles running red lights or changing lanes without indication could be detected by the definition of a violation line and capturing license plates through image or video processing.
- Accident or Breakdown Detection: It would immediately detect immobile vehicles in improper locations to attend to accidents or breakdowns, discarding parked vehicles. This would enable quick responses that can save lives and reduce property damage, as well as reduce traffic congestion.
- Synchronization of Signals Across Junctions: Coordinating signals along a route would allow vehicles to travel with fewer stops, improving traffic flow and overall efficiency.
- Adaptation to Emergency Vehicles: The system can be enhanced further to recognize the passage of emergency vehicles, like ambulances, and adjust the signals accordingly to ensure that such vehicles pass through an intersection efficiently for quick clearance.

REFERENCES

(Periodical style)

[1] TomTom.com, 'Tom Tom World Traffic Index', 2019. [Online]. Available: https://www.tomtom.com/en_gb/traffic-index/ranking/

[2] Khushi, "Smart Control of Traffic Light System using Image Processing," 2017 International Conference on Current Trends in Computer, Electrical, Electronics and Communication (CTCEEC), Mysore, 2017, pp. 99-103, doi: 10.1109/CTCEEC.2017.8454966.

[3] A. Vogel, I. Oremović, R. Šimić and E. Ivanjko, "Improving Traffic Light Control by Means of Fuzzy Logic," 2018 International Symposium ELMAR, Zadar, 2018, pp. 51-56, doi: 10.23919/ELMAR.2018.8534692.

[4] A. A. Zaid, Y. Suhweil and M. A. Yaman, "Smart controlling for traffic light time," 2017 IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT), Aqaba, 2017, pp. 1-5, doi: 10.1109/AEECT.2017.8257768.

[5] Renjith Soman "Traffic Light Control and Violation Detection Using Image Processing”.” IOSR Journal of Engineering (IOSRJEN), vol. 08, no. 4, 2018, pp. 23-27

[6] A. Kanungo, A. Sharma and C. Singla, "Smart traffic lights switching and traffic density calculation using video processing," 2014 Recent Advances in Engineering and Computational Sciences (RAECS), Chandigarh, 2014, pp. 1-6, doi: 10.1109/RAECS.2014.6799542.

[7] Siddharth Srivastava, Subhadeep Chakraborty, Raj Kamal, Rahil, Minocha, “Adaptive traffic light timer controller” , IIT KANPUR,

NERD MAGAZINE

[8] Ms. Saili Shinde, Prof. Sheetal Jagtap, Vishwakarma Institute Of Technology, Intelligent traffic management system:a Review, IJIRST 2016

[9] Open Data Science, 'Overview of the YOLO Object Detection Algorithm', 2018. [Online]. Available: https://medium.com/@ODSC/ overview-of-the-yolo-object-detection-algorithm-7b52a745d3e0

[10] J. Hui, 'Real-time Object Detection with YOLO, YOLOv2 and now YOLOv3', 2018. [Online]. Available: https://medium.com/ @jonathan _hui/ real-time-object-detection-with-yolo-yolov2-28b1b93e2088

[11] J. Redmon, 'Darknet: Open Source Neural Networks in C', 2016. [Online]. Available: https://pjreddie.com/darknet/

[12] Tzutalin, 'LabelImg Annotation Tool', 2015. [Online]. Available: https://github.com/tzutalin/labelImg

[13] Li, Z., Wang, B., and Zhang, J. "Comparative analysis of drivers' startup time of the first two vehicles at signalized intersections", 2016 J. Adv. Transp., 50: 228– 239. doi: 10.1002/atr.1318

[14] Arkatkar, Shriniwas & Mitra, Sudeshna & Mathew, Tom. "India" in Global Practices on Road Traffic Signal Control, ch.12, pp.217-242

[15] 'Pygame Library', 2019. [Online]. Available: https://www.pygame.org/wiki/about

[16] 'Traffic Signal Synchronization'. [Online]. Available: https://www.cityofirvine.org/signal-operations-maintenance/trafficsignal-synchronization