Dynamic Traffic Signal using Image Processing

Urjita Thakar, Nikita Tiwari, Vinti Jain, Harshali Satwani, Anand Multani, Nancy Agrawal, Sumangal Manke,

[1] Department of Computer Engineering, Shri Govindram Seksaria Institute of Technology and Science, 23, Sir M. Visweswaraiya Road (Park Road), Indore (MP) 452003, India

Abstract*: *In cities, traffic on road is increasing rapidly resulting in traffic jams and reduced traffic flow. Traffic lights play a major role in regulation and management of traffic flow at junctions. These lights are controlled by traffic light controllers (TLCs) that are programmed to assign timely directions to drivers in Red, Yellow and Green signals. Current TLC(s) are based on programmable micro-controllers, which have limitations as their signal timings function according to a program that needs to be modified manually and does not have flexibility to accommodate the functionality to modify signal timings based on variable traffic. In this paper, inspired by advanced technologies, we have studied and tested a method to improve traffic flow at squares run by traffic lights in real-time. To make TLC(s) more efficient, we are proposing a technique which controls green, yellow, red signal switching based on the real-time calculation of traffic density through capturing and processing images of all incoming lanes at any junction. This paper also focuses on the algorithm for scheduling these lights according to vehicle density on the road, thereby aiming at reducing the traffic congestion on squares which will help people reach their destinations safe and in the least time possible.

***Keywords*: Dynamic traffic lights, traffic density, image processing, OpenCV, background subtraction, traffic light automation, python pillow, PIL**

(Article history: Received: 6th February 2021 and accepted 17th May 2021)

I. INTRODUCTION

With time, the number of vehicles on road are increasing exponentially. With increasing vehicles, congestion is becoming a major problem in cities. There can be different causes of congestion in traffic, like insufficient capacity to handle vehicles on lanes, large waiting time at red lights, etc. Currently installed traffic lights do not consider variable traffic present at squares at different times of the day. They are merely pre-programmed embedded circuits. Mutual interference of traffic between different squares is also not considered while programming these TLC(s). This leads to traffic jams and congestion. Weekdays show heavier traffic than weekends and during peak hours (morning and noon), traffic density increases at every square.

A number of solutions have been proposed for the prevailing embedded TLC problem. Some of them use infrared(IR) sensors [1] to count the number of vehicles passing by a lane, photoelectric sensors [2] and continuous monitoring of vehicles, while others use vehicle count techniques [3] but these techniques have their limitations. With sensors, maintaining and interfacing is a problem and in the vehicle count technique, the size of the vehicle might affect the count of these vehicles.

Unlike other dynamic control signals that adjust the timing and phasing of lights according to limits that are set in controller programming, the proposed system uses an available data set of different squares and gives a novel approach to quantitatively tell the amount of traffic present at any lane and how can we adjust respective green-light times of those lanes. Here the processor processes these images and monitors vehicle density and controls TLC in real time to avoid congestion wherever possible. This paper proposes dynamic traffic signaling system based on traffic density using Image Processing approach.

The paper is organized in following sections: Section II gives us a slight background to the problem and proposed solution. Section III walks us through some related literature on various methods and approaches to building smart traffic light controller systems. Section IV presents our proposed method. The experimentation and results are given in Section V. Section VI concludes our study and gives some future prospects followed by acknowledgement and references.

II. BACKGROUND

This study, in a broad sense, comes under smart city planning, wherein it comes under smart traffic management. Traffic management has many nuances like smart street lights, roadway planning, automated challan system, etc. All these problems are faced by every city and need to be addressed to reduce accidents, avoid heavy congestion on the roads and provide better commute experience to residents.

Currently, most of the signals are designed based on surveys carried out for a fixed period of time. On the basis of the average traffic flow, green light time, clearance time and red-light time are calculated. These surveys give fixed red/green/clearance time and are

conducted by methods like manual counting or automatic counters using electronic, pressure sensitive and magnetic devices. The problem with these methods is that they provide a fixed time interval at each signal irrespective of traffic or time of the day, resulting in not so good traffic management. To overcome the above problem, studies to design and develop flexible/dynamic signals have bloomed. The traffic density at a particular signal is calculated and accordingly a clearance time is allocated. Various strategies are used to calculate density of the lane like: counting number of vehicles, length of traffic jam on each lane, photo-electric sensor, analyzing the pressure on each lane. But finding the density using above approaches is difficult and in-efficient.

A. *Need*

Indore has only 378 traffic personnel against 3.2 million people [4] and with increasing number of vehicles, handling such densely populated city's traffic is extremely difficult. With the Traffic Police doing its best to maintain good traffic flow in the city, even a small optimization in the traffic management system can create better streamlining of traffic flow. Therefore, an adaptable traffic light system to manage dynamic traffic is required.

B. *Software and Technologies*

Videos being taken as input, processing and extracting density from these is done through **Image Processing.**

Image processing is a method to perform some operations on an image, in order to get an enhanced image or to extract some useful information from it. As we know images are a matrix of pixels and videos are multiple image frames, image processing is done using the values of each pixel that represents the color that the pixel carries. It is a type of signal processing in which input is an image and output may be image or characteristics/features associated with that image.

Image processing basically includes the following three steps:

1. Importing the image via image acquisition tools
2. Analyzing and manipulating the image
3. Output in which result can be altered image or report that is based on image analysis

Some inbuilt Python libraries and modules for image processing that are used in proposed method are:

- cv2 - OpenCV
- pillow
- Python Imaging Library (PIL)

C. *Details about Image Processing Technologies specific to the implementation*

The following three technologies are studied and applied in the implementation of this proposed method, namely:

- *Image Averaging:* Image averaging is done to get an image, out of a video clip (multiple frames), which is an average of these frames. It calculates the temporal mean (average of values over a time period) of respective pixels in all the frames. For a video with same number of frames captured per second, temporal average is calculated as sum of the value over the time period divided by total number of frames in that period. The relevant application of this method for our study is to extract background or a reference frame for comparison with current frame to calculate density.
- *Background Subtraction:* This is one of the most important pre-processing steps in many image processing applications. For example, in situations where we have to count number of visitors entering or leaving a room, or a camera at a traffic signal for extracting information about the incoming and outgoing lanes, for achieving this, we need to separate the vehicle or person from the background. That is subtracting the foreground from the background.
- *Thresholding:* It is a method to separate fore ground from the background of an image. Here, the image is segmented so that the objects get isolated from the background, this is achieved by converting the greyscale images into binary images. This method is the most effective way to differentiate between objects and their background in images with high level of contrast.

All these methods are implemented through OpenCV*.

III. RELATED WORK

In this section, the contributions made by various researchers in the relevant field are discussed.

Calculating density using vehicle count can be one approach to find out the solution to this problem. K. Vidhya, A. Bazila Banu [3] calculated density by using openCV tool as software for image processing by just displaying the various conversion of images in the screen and finally the number of vehicles were calculated by surrounding the box on the vehicle in the given image. Similarly, S.Lokesh, T.Prahlad Reddy [5] implemented an artificial density traffic control system using image processing and Raspberry pi. The hardware used is webcam, Raspberry pi and the software used are Occidentalis and Matlab. The camera is interfaced with the Raspberry pi and thresholding method is used to analyze the image sequences from a camera to find the density of vehicles. Subsequently, the number of vehicles at the intersection is evaluated and traffic is efficiently managed.

But these methods come with their drawbacks. Since dimensions of vehicles vary, calculating density on the basis of vehicle count may not give accurate results as a truck or a trolly might hide a bike or a car behind it.

Another approach of calculating density can be using infrared sensors. Er. Faruk Bin Poyen, Amit Kumar Bhakta, B.Durga Manohar, Imran Ali,

*An open source library for computer vision and image processing

ArghyaSantra, AwanishPratap Rao [1] proposed a system in which infrared sensors are used with OP-Amp LM324 for the comparator operation. The total number of IR sensors required at a junction are 4 and LED's 12. They, therefore, are connected to any of the two ports of microcontroller.

IR transmitter and receiver pairs are used which work as proximity sensor. The output voltage changes according to distance from an object and is fed to the comparator with a reference set. The reference set is set by a variable resistance according to required range of sensing. These sensors will detect the presence of the vehicles and sends the information to the microcontroller where it will decide how long a flank will be open or when to change over the signal lights.

Similarly, Salama A.S., Saleh B.K. and Eassa M.M. [2] provided a design of an integrated intelligent system using Photoelectric Sensors for management and controlling traffic lights. Two types of sensors are used here. The sensors' readings are received and saved in a local database (DB) at the cabinet which can be real time replicated to a central DB located at the central traffic management department. This central DB can be used for the overall control of the city traffics and these data can be the base of data mining activities later to decide the proper values of the sensors' relative weights and roads' directions priorities. This is done by applying an algorithm based on the relative weight of each road. The system will then open the signal of that lane which is densely crowded and give it comparatively more time than less dense lanes. The real-time decision-making capability of the system stands out very notably. Moreover, the system can also be programmed for emergency situations such as passing of ministries, presidents, fire brigades and ambulance vehicles that require virtually zero congestion through an active RFID based technology. As a result, the system will assure smooth flow of traffic for such emergency cases or for the main vital streets and paths that require the fluent traffic all the time, without affecting the smooth flow of traffic generally at normal streets according to the traffic density and the time of the day. Also, the proposed system can be adjusted to run automatically without any human intervention and can be adjusted to allow human intervention at certain circumstances as well.

Saranya J, Jayashwanth J.S, Kiran J, Harish S, Linga Kumar T [6] also proposed a system that includes Arduino Mega platform which is interfaced with IR sensor and LEDs (Red & Green only). Infrared sensors with Arduino mega2560 microcontroller are used to detect the density of the traffic. The IR sensor connected with the Arduino senses the number of vehicles residing in the lane. The output of IR sensor is given to Arduino board for further actuation of either red or green light on the lane based on programming logic embedded inside the Arduino platform. The logic inside their controller is that, if any particular lane has green light recurring for two times, then for the third time the other lane is given priority thereby allowing the lane which has a greater number of vehicles to pass through. This usage scenario exhibits the efficacy of their developed system. It focuses on reducing the unwanted delays which is caused at road intersections. The unnecessary traffic congestion which occur due to a greater number of vehicles floating on the road lanes during prime hours (i.e.) morning and late evening hours are reduced by means of counting the number of vehicles at each lane and giving priority to the lane which has a greater number of vehicles. It also proposed the use of sound sensor to detect the ambulance and other emergency vehicles.

These methods proposed by [1], [6] and [2] have various limitations. Firstly, low range IR sensors may not be an answer for long range signalling system. We may resort to ultrasound or radar techniques for big scale set-ups. Next is the influence of stray signals that may alter the reading of sensor receptors and lead to conveying false information to the microcontroller. Also, periodic checking of the accuracy and precision is a must for efficacious operation of this model prototype.

Our proposed method requires background subtraction which is a pre-processing step in various image processing applications. Following is some research work carried out on various background subtraction techniques.

Mr. Deepjoy Das, Dr. Sarat Saharia evaluated three background subtraction techniques. The techniques [7] ranges from very basic algorithm to state-of-the-art published techniques categorized based on memory requirements, speed and accuracy. Frame Difference Technique (1) aims to maximize speed and limits the memory requirements which produce a low accurate output. It produces an erroneous result due to slow moving objects, lighting condition and many other challenges if introduced with. Whereas Real Time Background Subtraction and Shadow Detection Technique Theory (3) and Adaptive Background Mixture Model for Real Time Tracking Technique (2) aims to achieve the highest possible accuracy under possible circumstances. The result of (2) and (3) methods can be used for real time background subtraction method, but the result of mixture of Gaussian method (3) is computationally intensive. The most appropriate method is (3) due to average computation cost and average result.

Mr. Mark Smids, Mr. Rein van den Boomgaard implemented and compared two different background subtraction techniques [8] for traffic monitoring. The deterministic subtraction technique uses the current pixel values in a video frame to update the background model. To make sure that moving foreground objects are not registered in the background model, only the pixels classified as background are updated. Classifying is done by considering the distance between pixel values in the background model and the new incoming video frame. The second implemented technique is a statistical

approach which models each pixel in the background model by a Gaussian mixture model. Updating the model is done by a number of update equations that update the weight, mean and variance of each Gaussian component in the model. Classifying a new pixel is done by extracting those Gaussian components in the model that have high weights and low variances, which are the indicators of being a background component. Both algorithms are evaluated using an implemented video summarization technique. From this evaluation it can be concluded that the statistical approach has a better performance in all cases. No matter the weather conditions, the statistical approach outperforms the deterministic approach.

Looking at the above approaches and considering their advantages and limitations, we came up with a solution that is explained meticulously in the next section.

IV. Proposed Method

The primary objective of this project is to generate an effective way to control traffic signals in real time. To achieve this, we performed image processing on live video clips available online from City of Brentwood, Tennessee, USA [9]. These clips are taken from 4 lanes: Crockett Road @ Aberdeen Drive, Church Street @ Wilson Pike, Franklin Road @ Town Center and Moores Lane @ Galleria Blvd[†].

A. Algorithm for Density Calculation

To calculate density, initially averaging of all the lanes of all junctions at three different times (morning, dusk and night) is performed and frame of reference is extracted from them. These images are stored as reference images (referred as Average Main). The detailed algorithm to calculate density is as follows:

1. Start
2. Capture a 7 second video from piCam at three different times of the day.
3. Calculate the daily average through temporal image averaging for all the lanes at a given time and extract the frame of reference which is actual background without any objects in the foreground.
4. Compare it with the main average frame to make sure that they both match to a particular extent. If they do, jump to step 6.
5. Jump to step 2 if you find a frame of reference not comparable to the Average Main. This step is carried out at most thrice.
6. Capture the current frame of remaining 3 lanes from piCams when the yellow light starts and send it to raspberry Pi unit.
7. Extract the frame of interest (FOI) from the current frame by cropping it.
8. Convert both the daily average and the current frame to grayscale.
9. Subtract the daily average from the cropped frame to obtain the binary mask.
10. Apply a threshold function to get a binary black and white image which is a matrix of pixels with values ranging from 0 to 255. 0 depicting complete black and 255 to be white.
11. Count the number of non-zero pixels to determine the number of pixels occupied by vehicles.
12. In the resulting image, we get the density when we divide the number of white pixels by total number of pixels, the density's value lies between 0 to 1, where 0 is no vehicle on the lane and 1 is the lane completely filled with vehicles.
13. Send the obtained density to the synchronization code.
14. Stop.

Due to limited hardware specifications of raspberry pi, 7s video clip was found to be sufficient to provide perfect background. Background extraction was experimented with video clips of different time periods. Raspberry pi was taking more than required time to process video clips of 50 s, 30 s and 12 s. Also the result of lower time period like 3 s and 5 s was not accurate. Therefore, 7 s video clip was considered appropriate in which it captures 210 images as its frame rate is 30 FPs (Frames Per Second).

B. Algorithm for Time Calculation

After density calculation, the next lane which will go green and the time given to that lane is calculated. The algorithm is as follows:

1. Start
2. Obtain densities of all the lanes at the junction.
3. A count variable initialized to zero is maintained for every lane which increases every time it is set to Red.
4. A queue is maintained for proper functioning of signals and to avoid starvation. When the value of count variable reaches 4, that lane is inserted in the queue. The priority is always given to the lane at the front of queue.
5. In case the queue is empty, the time for which traffic light at the lane with maximum density should go green is a linear function of density present at that lane:

$$greenLightTime = (39 * density)^{\ddagger} + yellowLightTime$$

Where *yellowLightTime* is total distance to be travelled to cross the square divided by the average speed with which vehicles cross the junction. With the survey that we did for junctions in Indore, the average distance of a junction is assumed to be 20 m and the average speed

[†]These lanes were selected for testing as they provide comparatively better angle and field of view than other lanes and open source traffic signal data.

[‡]this factor of 39 is multiplied as density lies between 0 and 1, the maximum green light time can be 39 + 6 = 45, so that maximum threshold is 45 × 4 = 180.

with which vehicles cross that junction when the light goes green is 3.3ms^{-1}.

6. The green light time given to a lane is max(calculated time, 15) s AND min(calculated time, 45) s, i.e. minimum threshold for green light to a lane is 15 s and maximum waiting time threshold is 180 s.
7. Count variable is again set to zero for the lane which went green.
8. Stop.

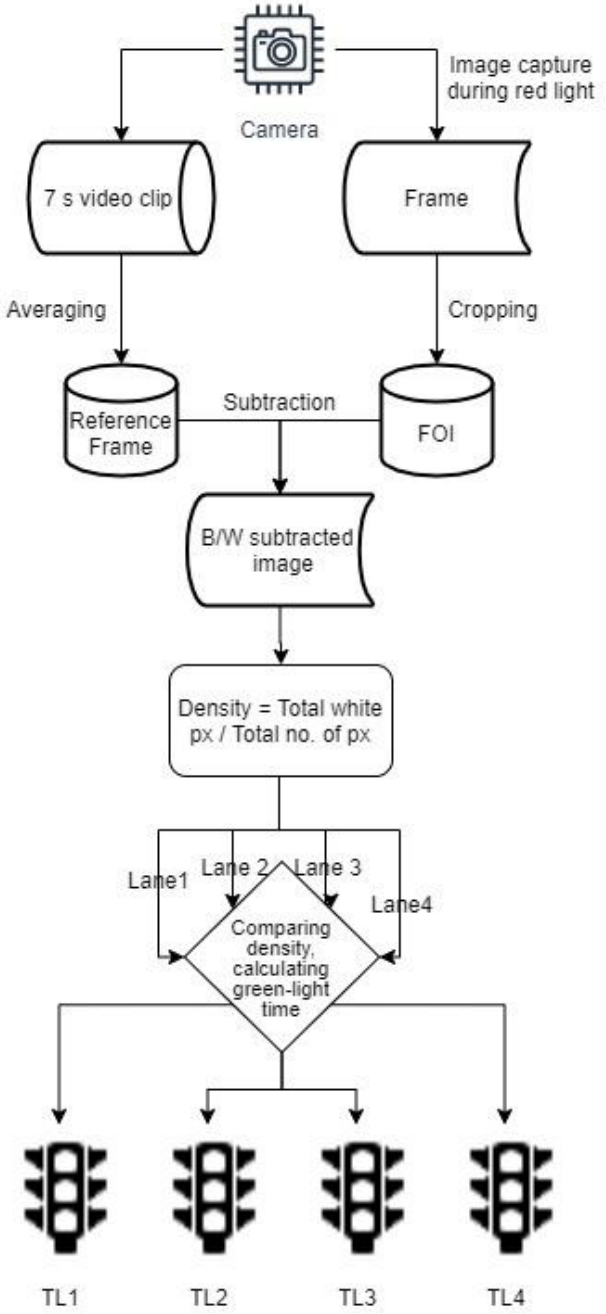

Fig. 1. Algorithm flow chart

Fig. 1 shows a simple algorithm flow. It shows the processing of the images right from the moment they are captured till the TLC sends one traffic light (TL) green-light time value.

V. Experiment and Results

In Fig.2 the temporal averages of respective 7 s video clips of these lanes are taken and the resulting image is shown. These images give us the actual background of the road and a reference frame for further processing of images.

Fig.3 shows frames captured when vehicles have stopped due to red light. This frame is still not the FOI. To get the actual frame of interest, we crop these images, this is done in order to reduce any discrepancy caused by the objects that are of no importance for the lanes' traffic density. The FOI is shown in Fig.4.

The step-wise results are shown in the following figures:

- *After averaging the 7 s video clip:*

Fig. 2. Image Averaging of Lane 1 and Lane 2

- *Testing frames (images taken during red light):*

Fig. 3. Current view of lanes 1 and 2

- *Extracting the current Field Of Interest:*

Fig. 4. Choosing background's FOI

- *Cropping the current FOI:*

Fig. 5. Current FOI

- *Subtracting the cropped reference frame from FOI:*

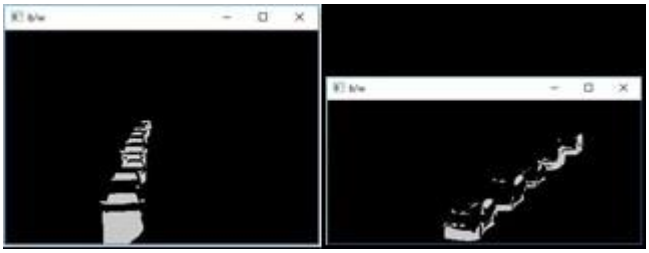

Fig. 6. Subtracted images of lane 1 and lane 2

This FOI is calculated automatically throughout the day for all the signals. Since It is possible that the mounted cameras might shift due to external factors such as wind, birds, human interference or mechanical faults in the mount, which might result in inaccurate results if FOI is not found. As seen in the figure, as long as the FOI is present in the captured image, the system extracts it after the
end of every time-zone so that the position of the camera can be monitored throughout the day.

In case that the camera shifts to the point where the FOI moves out of the frame, an error log is registered with the relevant information. The system stops capturing images and extracting densities and instead a static sequential traffic light system is initiated. The error log is reported and until the camera position is fixed, this sequential system is run instead. Even if there is only a shift in the position, the shift is logged in the file with the relevant information.

One thing to note here is that the angles of both cameras are different and thus give us different sectional view of the vehicles. This can cause a slight difference in calculation of densities as they cannot be compared. So while installing cameras for the implementation, they should be placed at an appropriate height and angle as mentioned in Fig. 7.

- h: height at which camera will be placed (known)
- x: range of camera (known)
- d: distance of stop line from camera post (known)
- A+O: angle of inclination of the camera (to be calculated)

$$tan(A+O) = (d + (x/2)) / h$$
$$(A+O) = tan^{-1}((d + (x/2)) / h)$$

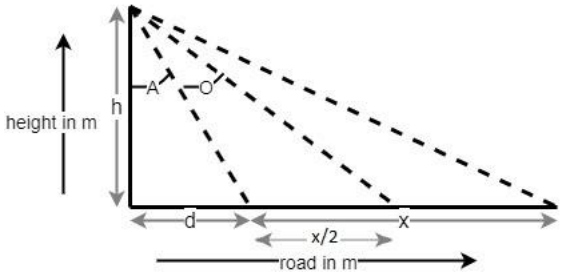

Fig. 7. Proposed Camera PositionThe maximum value of x can be 77 m approximately as maximum depth of field of piCam v3 is 77 m which is calculated using a fact i.e. depth of field is inversely proportional to the square of its focal length and in this case the focal length of piCam is 3.60 mm +/- 0.01.

When cameras at all four lanes (assuming a square with 4 incoming lanes) are installed at the same angle and height, calculated by the method mentioned above, they provide the same field of view. Thus giving us the densities that are comparable, through which we can decide which lane should go green and for how much time. For example, in Fig.6 the densities of the lanes are found to be 0.03531688437348815 and 0.029985981903912322 respectively, and hence, lane 1 goes green for 15 s after which lane 2 goes green for 15 s.

This *greenLightTime* is calculated by the formula mentioned above. Another aspect of the calculation of this time is the area of the FOI for a particular side, which is directly proportional to the number of pixels present in the subtracted images and is one of the parameters to control the resultant density calculated. The FOI in these images were calculated to accommodate all the vehicles in the depth of field for the camera units. Within a specified range, all FOI areas must remain similar to one another so that the calculated densities from all the sides (i.e. from all the cropped FOI images) can be normalized so that the system gives accurate results.

Table I shows value of density and time for every lane till 7 iterations. It also shows which lane is selected and turned green for some particular time. The count for every lane is incremented in each iteration when it is asked to wait and reset to zero once turned green. In the first iteration, density of all four lanes is calculated which is followed by calculation of density of remaining 3 lanes. After fourth iteration, the count of second lane becomes 4 which results in red count exception and hence in fifth iteration second lane is given priority to go green. Similarly, in next iteration lane 2 is given priority due to red count exception.

One limitation while getting the image average from the 7 s video is: if a vehicle is parked or is stationary for that span of time, it becomes the part of the background. But, unless it is a comparatively big vehicle, it will account for less than 5% inaccuracy. However, this limitation can be overcome by calculating image average at specific intervals to avoid any inaccuracy while calculating the average. For this, special high processing microprocessor will be required that can handle multiple image processing tasks parallelly.

VI. Conclusion and Future Work

In this paper, the proposed method can reduce traffic congestion at junctions due to unessential and lengthy waiting time at red-lights. An adaptive system is used which, unlike the hard-coded lights, produces less green-light time for less traffic lanes and more green-light time for crowded lanes. This will help people reach their destinations efficiently. This method, when coordinated for multiple junction will lead to drivers encountering string of green lights thus optimizing travel-time, reducing delays, road accidents and fuel consumption. The proposed method overcomes the limitations of previously implemented methods of density calculation using various

methods like vehicle count, infrared sensor, photoelectric sensor and increases the accuracy significantly.

TABLE I. TEST CASES

Iterations	Lane 1		Lane 2		Lane 3		Lane 4		Selected Lane		Count for every lane			
	Density	*Time*	*Density*	*Time*	*Density*	*Time*	*Density*	*Time*	*Lane No.*	*Time*	*1*	*2*	*3*	*4*
1	0.042	15	0.614	31	0.376	21	0.542	28	2	31	1	0	1	1
2	0.737	35	-	-	0.719	35	0.340	20	1	35	0	1	2	2
3	-	-	0.498	26	0.723	35	0.638	31	3	35	1	2	0	3
4	0.991	45	0.145	15	-	-	0.268	17	1	45	0	3	1	4
5	-	-	0.713	34	0.972	44	0.490	26	4	26	1	4	2	0
6	0.031	15	0.391	22	0.679	33	-	-	2	22	2	0	3	1
7	0.933	43	-	-	0.530	27	0.587	29	1	43	0	1	4	2

In future, this algorithm can be implemented at a larger scale incorporating factors like traffic density at neighbouring squares in the city, and connecting this complete graph which coordinates and synchronizes in such a way that any person leaving for a place reaches his destination in the minimum possible time and waiting time reduces 5-6 times. This project can also generate data that can be useful to determine traffic trends and help in city planning.

ACKNOWLEDGMENT

This research was supported by Department of Computer Engineering, Shri G. S. Institute of Technology & Science, Indore, India.

REFERENCES

[1] A. K. B. Er. Faruk Bin Poyen, "Density based traffic control." IJEAMS. [Online]. Available: https://ijaems.com/detail/density-based-traffic-control/

[2] E. M. Salama A. S., Saleh B. K. , "Intelligent cross road traffic management system(icrtms)." IEEE [Online] Available :https : / / www. semanticscholar. org / paper / Intelligent-cross-road-traffic-management-system-Salama-Saleh / c6fe217f3fdcdf835cb5995a33d5f5ec20e5a977

[3] A. B. K.Vidhya, "Density based traffic signal system." IJIRSET.

[4] T. N. Network. 68,927 challans: Traffic rules take back seat here. [Online]. Available: http://timesofindia.indiatimes. com/articleshow/47704283.cms?utm_source=contentofinterest&utm_ medium=text&utm_campaign=cppst

[5] T. R. S.Lokesh, "An adaptive traffic control system using raspberry pi." IJESRT. [Online]. Available: https://www.researchgate.net/publication/282612189_An_Adaptive_Traffic_ Control_System_Using_Raspberry_PI

[6] K. J. Saranya J, Jayashwanth J.S, "Automated density-based traffic light control system using arduino platform." IJEAT. [Online]. Available: https://www.ijeat.org/wp-content/uploads/papers/v9i1/A2956109119. pdf

[7] D. S. S. Mr. Deepjoy Das, "Implementation and performance evaluation of background subtraction algorithms." ResearchGate. [Online]. Available:

[8] https://www.researchgate.net/publication/262189479_Implementation_And_Performance_Evaluation_Of_Background_Subtraction_ Algorithms

[9] M. R. v. d. B. Mr. Mark Smids, "Background subtraction for urban traffic monitoring using webcams." Semantic Scholar. [Online]. Available: https://pdfs.semanticscholar.org/d511/3ff70fe3c328f8d161974c70911633e67ee4.pdf

[10] U. Gov. City of Brentwood, Tennessee: Live traffic cameras. [Online]. Available: http://traffic.brentwoodtn.gov/

AUTHOR PROFILE

Urjita Thakar is presently Professor in Department of Computer Engineering at Shri G. S. Institute of Technology & Science, Indore. Her research interests include the field of information security, Microservices, Machin Learning and IoT. She is guiding doctoral research in these fields and has also guided about 100 projects at UG and PG level. She has more than 80 research publications in international and national journals and conferences.

Nikita Tiwari is a faculty of Computer engineering with teaching experience of about five years. Her areas of interest are Computer Networks and Information Security.

Vinti Jain, Masters student at San Jose State University, California. A Software Engineer with experience in developing large scale web applications. Data Science, Enterprise lopment enthusiast.

Harshali Satwani completed her graduation from Shri G.S. Institute of Technology & Science, Indore and is currently working at Optum, United Health Group. A Software Engineer with experience in developing large scale web applications. She has profound interest in Data Science, Machine Learning and Enterprise Software development.

Anand Multani is a graduate level student currently working at Persistent Systems ltd. in Nagpur. He has profound interest in Robotic Automations and Machine Learning Applications.

Nancy Agrawal is a graduate level student at Indian Institute of Science, Bangalore pursuing her post-graduation in Computer Science and Engineering. She has profound interest in Machine Learning and Artificial Intelligence Applications.

Sumangal Manke is a graduate level student graduated from Shri G.S. Institute of Technology & Science, Indore. He has profound interest in Data Science and Machine Learning Applications.