Dynamic Control of Traffic Signals using Traffic Data from Google Maps and Road Cameras

Arunachalam Muthupalaniappan, B Shreehari Nair, Raakheshsubhash Arumuga Rajan, Raghesh Krishnan K

ABSTRACT--- Traffic congestion is a condition that occurs due to over usage of roads and results in increased waiting time, slower speeds and increased vehicle queuing. Every year billions of dollars are lost due to time wasted by waiting at traffic signals. In this paper, the problem of dynamic control of traffic signals using traffic data obtained from Google Maps and Road Cameras is explored. The traffic signal control algorithm described will adjust both the ordering of different lights in real time as well as the duration of each of them. The ordering (sequence) and length of each traffic signal light for a period of time will be affected by factors such as traffic volume, waiting time and vehicle queuing characteristics. The results from the above algorithm's simulations are compared to the average waiting times the vehicles experience in the case of static traffic signals which employ a fixed duration and sequence policy for each of the colors of the traffic signal.

I. INTRODUCTION

Smart transportation systems are adaptive road transportation systems that intend to improve vehicle throughput (num- ber of vehicles served in a given time period), and reduce average wait time of all the vehicles, in the area governed by a given system. The aim of this work was to design a real time traffic light control system, with a smart algorithm, based on queuing theory, and live vehicle data. This real time vehicle data is obtained from traffic cameras, present in each lane of a 4-way intersection, or traffic data from Google Maps. Computer vision techniques are applied on the live traffic video feeds, to detect and enumerate the number of vehicles present in each lane. Google Maps displays a dynamic colored overlay of live traffic and congestion data on top of roads, ranging from green to red, green being on the low end of the spectrum and red being on high end of the spectrum of traffic congestion. These colors, extracted from the maps, give an idea about the traffic density on the given road, at a given time. A method to extract this vehicle density data (colors on roads in Google Maps), and assign quantifiable values to the retrieved colors (actual number of vehicles present at that time), is discussed in the subsequent sections. Traffic data, obtained from the Google maps / road traffic cameras, is passed on to a scheduling algorithm that decides which lanes must next receive a green signal, and for what duration of time. After the chosen lanes are serviced, the updated traffic data is refreshed from the above sources and fed back into the algorithm.

The values obtained from the scheduling algorithm, processed in a central server, are communicated to a traffic signal controller, via the internet, in the case of traffic signals connected through wireless sensor networks (WSN), or through the local traffic control center, with the help of wired interconnects, as in the case of legacy traffic systems. Fig. 1 and Fig. 2 show the proposed system architectures and Fig. 3 represents the workflow. The average wait time for fixed time traffic signals is compared to the proposed system. The subsequent parts of this paper are organized as follows. Section II elaborates upon the traffic data extraction from Google Maps and from traffic cameras mounted atop traffic signals in each lane. Section III presents the proposed traffic light scheduling algorithm and the performance evaluation.

II. TRAFFIC DATA EXTRACTION

A. Traffic Data extraction from Google Maps

A central server, at the traffic control center (TCC), period- ically renders the map of a given area, containing traffic signals, to extract data from it and feed into the traffic scheduling algorithm to schedule the next signal timing. Google Map servers update the live traffic data, in real time. The colored lines, as shown in Fig. 4, represent the relative traffic densities. The following colors are indicators of the traffic congestion, on a given road: green, orange, red, dark red. These colors only give a relative view of the traffic at a given location, rather than quantifiable or observable values. Hence,

Revised Manuscript Received on July 10, 2019.

Arunachalam Muthupalaniappan, Department of Computer Science and Engineering Amrita School of Engineering, Coimbatore Amrita Vishwa Vidyapeetham, T.N, India (email: cb.en.u4cse15409@cb.students.amrita.edu)

B Shreehari Nair, Department of Computer Science and Engineering Amrita School of Engineering, Coimbatore Amrita Vishwa Vidyapeetham, T.N, India (email: cb.en.u4cse15446@cb.students.amrita.edu)

Raakheshsubhash Arumuga Rajan, Department of Computer Science and Engineering Amrita School of Engineering, Coimbatore Amrita Vishwa Vidyapeetham, T.N, India (email: cb.en.u4cse15437@cb.students.amrita.edu)

Raghesh Krishnan K, Department of Computer Science and Engineering Amrita School of Engineering, Coimbatore Amrita Vishwa Vidyapeetham, T.N, India (email: k raghesh@cb.amrita.edu)

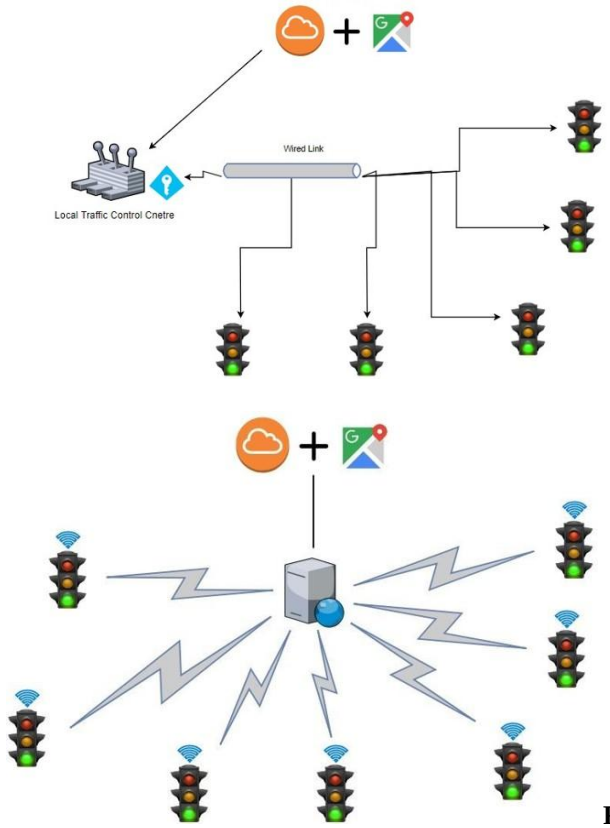


Fig. 2: WSN Based Traffic Systems - Proposed architecture

this calls for a means to determine the actual traffic density and create a link between the number of vehicles and the color density shown in the map. In order to achieve this, at different times of the day, for a given color in the map, the number of different types of vehicles in a 15 m x 100 m (1500 square meter) stretch of road is observed. The average values for each color of traffic density, as specified by Google maps, obtained by manual observation are tabulated in Tables (1 4). These values were obtained by manual observation of the traffic, over a 3-day period, at different times of the day, at Singanallur Junction, in the city of Coimbatore, India. Since the number of vehicles on a road dynamically change at different times of the day, the traffic congestion on a given road changes, and with it, the color of the road changes. The steps to deduce the traffic data are mentioned in Algorithm 1. [7]. The results of Algorithm 1 are shown in Fig. 5. Google API calls and image processing techniques are used to extract the required data. [14]

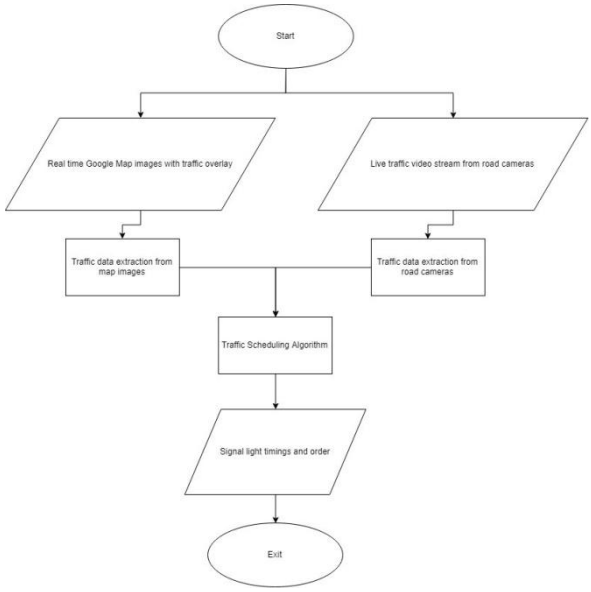


Fig. 3: Workflow

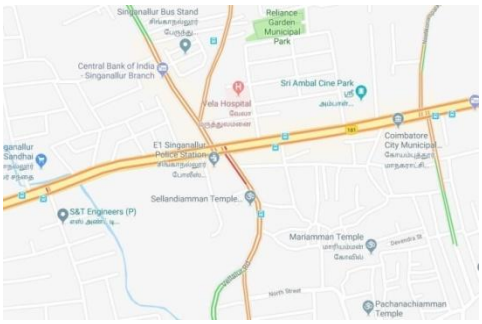


Fig. 4: Sample screenshot of intersection in Google Maps with traffic overlay

Table 1: Green Density Values			
Day	4 Wheelers	2 Wheelers	Avg. Speed
Day 1	3	10	60 km/hr
Day 2	5	4	58 km/hr
Day 3	5	6	52 km/hr

Table 2: Orange Density Values			
Day	4 Wheelers	2 Wheelers	Avg. Speed
Day 1	10	9	35 km/hr
Day 2	12	15	30 km/hr
Day 3	7	19	32 km/hr

Table 3: Red Density Values			
Day	4 Wheelers	2 Wheelers	Avg. Speed
Day 1	10	34	20 km/hr
Day 2	26	20	22 km/hr
Day 3	22	17	22 km/hr

Table 4: Dark Red Density Values			
Day	4 Wheelers	2 Wheelers	Avg. Speed
Day 1	31	30	4 km/hr
Day 2	39	13	7 km/hr
Day 3	33	27	5 km/hr

Algorithm 1: Google Maps Traffic Data Extraction

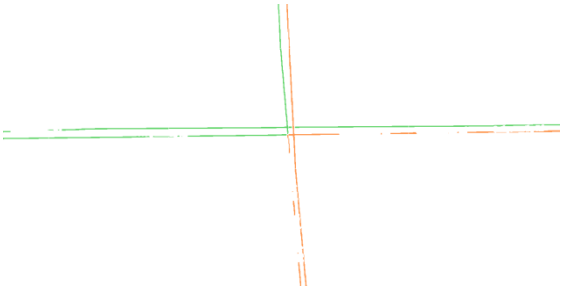
1. Render map image of road intersection of interest using google maps api.
2. Set required parameters for the given map: zoom level and remove buildings and labels.
3. From the obtained map use Canny edge detection and probabilistic Hough transform to extract the traffic lines on road and separate the lines from other unwanted map features.
4. Use selective colour segmentation to extract different colours of traffic lines present.
5. The traffic data for each lane is now obtained by comparing the above result with predetermined relative location of each lane using region-based segmentation.
6. The obtained colour results of each road are used to estimate the number of vehicles in the given lane by multiplying the traffic density for 1500 square metre strip size with the total size under consideration (per 1500 square meter). [5][9][12]

A. Traffic Data extraction from Road Cameras

For signal scheduling, traffic data can also be extracted from road cameras, in the case of major intersections, to provide a better estimate. Google Maps can be used in case of non- availability of road cameras. The algorithm to detect vehicles consists of three steps: background subtraction, blob detection and analysis, and blob tracking.

Background subtraction: The background of a video is the part of the video that undergoes little or no motion. A histogram-based statistical model of the image background, for every pixel, is created. This model collects the color co-occurrences of groups in the video. A Bayesian decision rule, for classification of foreground or background, is then used. The Bayesian rule considers the posterior probabilities of the given feature vectors and calculates the probability that a feature vector belongs to a particular class. The classification is then performed by comparing the probabilities, which help in detection of the foreground and background

(a)



(b)

Fig. 5: Results of algorithm 1

1) Blob detection and analysis: To detect blobs, the pixels in the current frame are grouped together by utilizing a contour detection algorithm [1]. The individual pixels are grouped into disconnected classes, and the algorithm then locates the contours that surround each class. Each class is considered as a candidate blob. Small candidates are removed to prevent false detection of vehicles. Several candidate blobs may belong to the same vehicle. To prevent re-counts of the same vehicle, K-Means clustering is used, to determine if all the blobs belong to the same vehicle. Each segmented blob is now bounded into a rectangle. [9] To determine if the vehicle is a two-wheeler or a four-wheeler, its height to width ratio is observed. If the value falls below 1, it is considered a four-wheeler, if it is above 1, it is considered a two-wheeler. For ease of calculations, a four-wheeler is considered equivalent to 2 instances of two-wheelers. [8][11]

Blob Tracking: To track the movement of blobs, within successive video frames, the blobs with their bounding boxes and centroids are extracted from each frame. The two blobs, whose centroids are closest to each other, and whose areas are almost equivalent to each other, denote the same blob (vehicle). Euclidean distance is used to measure the distance between their centroids. A moving blob is counted as a vehicle as soon as the centroid of a well-defined blob touches the passing line in one of its frames. A two-wheeler is considered as one vehicle and a four-wheeler as two vehicles. The cumulative vehicle count from each lane, at the traffic signal intersection, is fed into the scheduling algorithm. A screenshot of detected vehicles, in a sample road camera video, is shown in Fig. 6.

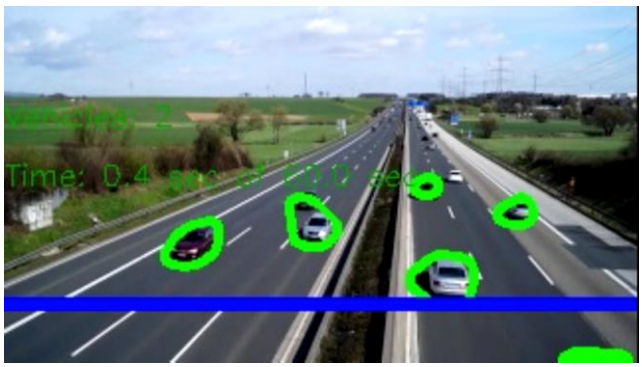


Fig. 6: Detected Vehicle Blobs in Sample Traffic Video Frame

III. TRAFFIC SCHEDULING ALGORITHM

The traffic scheduling algorithm must respond to a dynam- ically changing traffic environment by improving efficiency, and at the same time guaranteeing fairness for each lane of traffic. To model the scheduling problem, we will consider a four-way intersection (north, south, east, and west), as shown in Fig 7. each of which has two lanes, one for going forward

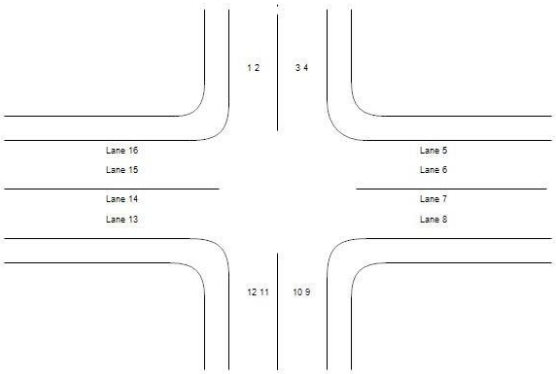


Fig. 7: A typical intersection

Vehicles in some lanes have the option for a free left turn. Each lane is controlled by a traffic light that displays 2 lights, red and green, which correspond to stop (Halt) and move (Go), respectively. Subject to traffic safety rules, there exists a maximum of twelve different possible cases of green lights as shown in Fig. 8. The next step is to determine which lane will obtain the green signal and the duration of time, for which the green signal must last for. [12][13] Algorithm 2 describes the steps required by the traffic scheduling algorithm. The following, notations will be used by the algorithm. [3][4][6]

l = {1,2,3,4,5,6,7,8}
c = {1,2,3,4,5,6,7,8,9,12}

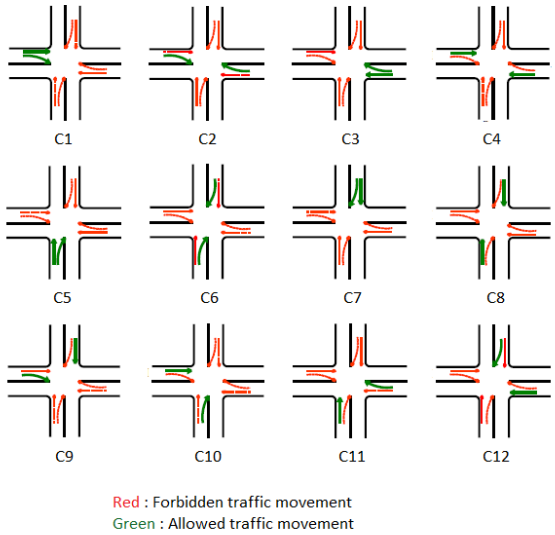


Fig. 8: 12 possible cases of traffic flow

- *departRate* = rate at which vehicles are served during green light. (vehicles/min - considered a constant for ease of calculations)
- *curTime:* current signal time.
- *serveTime:* Time to serve a particular lane. This is the time for which the signal will be in green colour for a particular lane. (green length time).
- *glCase:* The case that currently has green light, *glCase* $\in c$ (One of the cases as mentioned in figure 5.)
- *glLane*1: Lane one of current *glCase*.
- *glLane*2: Lane two of current *glCase*.
- $w[x, t]$: no. of vehicles present in lane x at time t where $x \in l$.
- $uns[x,t]$: cumulative number of unserved vehicles present in lane x at time t where $x \in l$.
- *hunger*[x]: hunger level of lane x where $x \in l$.
- *greenReqLvl*[y] : the green light requirement level of each of the twelve cases of traffic flow (Fig 5) where y c. it is calculated as *greenReqLvl*[y] = $w[x1, t]$ + $w[x2, t]$+0.25 $uns[x1, t]$+0.25 $uns[x2, t]$ where $x1, x2$ are the two lanes corresponding to traffic flow case. Eg : for case 7 in Fig.5, it is lane 2 and 10.

IV. PERFORMANCE EVALUATION & RESULTS

To evaluate the algorithms performance, simulations have been performed in matlab and in python (based on usual traffic flow characteristics). [2]

The first metric of measurement is the number of vehicles served, in a given amount of time, by the proposed system, when compared to the default static system, commonly present in all traffic signals. Fig. 9 shows a comparison of

Algorithm 2: Traffic Scheduling Algorithm

1. Set *curTime* = startup time of signal.
2. Set *hunger*[x] = 0 for all the lanes.
3. Obtain values of traffic level for each lane using maps/cameras. Store them in w [x, *curTime*].
4. Check if for of any one of the lanes *hunger*[x] > 3 for x from 1 to 12. If present, choose any one x. Let it be k. Calculate *greenReqLvl* for all cases with k as one of the lanes. If not present calculate *greenReqLvl* for all the 12 cases.
5. *glCase* = case corresponding to max *greenReqLvl*[y] for the above calculated cases.
6. *serveTime* = *max*(w[*glLane*1, *curTime*], w[*glLane*2, *curTime*]) /*departRate*.
7. Set *hunger*[*glLane*1] = 0 and *hunger*[*glLane*1] = 0.
8. For all lanes i l other than *glLane*1 and *glLane*2, set *uns*[i, *curTime*] = w[i, *curTime*] and increment *hunger*[i] by 1.
9. Set *curTime* = *curTime* + *serveTime*.
10. Set *uns*[*glLane*1] = 0 and *uns*[*glLane*2] = 0.
11. Go back to step 3.

The traffic throughput of the two different systems, over a time period of 600 seconds (10 minutes) averaged over three

simulations. As time progresses, it can be inferred that the time saved, in certain signal cycles, is used for serving other vehicles efficiently due, to the dynamic nature of signal timing, and hence, the cumulative number of vehicles served is higher.

The second metric of measurement is the average waiting time of all vehicles combined. As described in the algorithm, since we consider hunger level as a factor, the average waiting time performs well in cases of high traffic congestion as shown in Fig. 10. In cases of lower congestion level, the average waiting time of the proposed algorithm is like that of static signals. However, a noticeable difference is seen in case of higher congestion levels characterized by longer waiting times as seen in Fig 7. [10]

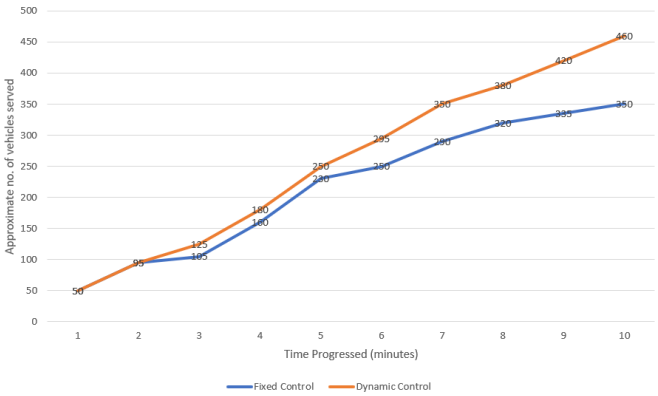


Fig. 9: Comparison of Vehicle Throughput

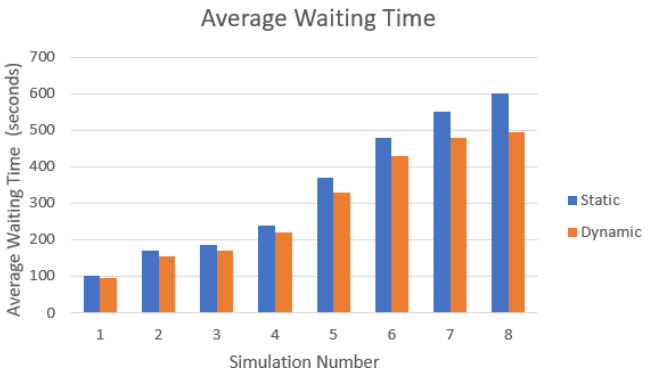


Fig. 10: Comparison of Average Waiting Times

V. CONCLUSION

The proposed system can schedule signals based on Google Maps and traffic camera data. The advantage of the proposed system is that it is cost effective and does not require the use of specialized RFID cards (radio frequency identification devices) / pressure / motion sensors.

The low cost of implementation will make this dynamic signal system suitable for developing countries. A city-wide implementation of the system would require a very reliable database which must be able to store traffic data from every intersection in the city. The current scheduling system takes into account an isolated intersection only. Implementing an algorithm which considers the influences of neighboring intersections and their traffic, would provide scope for future improvement.

REFERENCES

1. P. Arbelaez, M. Maire, C. Fowlkes and J. Malik, "Con- tour Detection and Hierarchical Image Segmentation", in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 33, no. 5, pp. 898-916, May 2011.
2. D. Heidemann, "Queue length and delay distributions at traffic signals" Transportation Research B vol. 28 no. 5 pp. 377-389 1994.
3. A.G. Sims K.W. Dobinson, "The sydney coordinated adaptive traffic (scat) system philosophy and benefits", IEEE Trans. Veh. Technol. vol. 29 pp. 130-137 1980.
4. R. Robertson R.D. Bretherton, "Optimizing networks of traffic signals in real time: the scoot method", IEEE Trans. Veh. Technol. vol. 40 no. 3 pp. 11-15 1991.
5. Wada T., Huang F., Lin S. (eds) Advances in Image and Video Technology, "Vehicle Detection from Aerial Images Using Local Shape Information". PSIVT 2009. Lecture Notes in Computer Science, vol 5414. Springer, Berlin, Heidelberg.
6. E. Pappis C.P. Mamdani "A fuzzy logic controller for a traffic junction" IEEE Transactions on Systems Man and Cybernetics Choi JY., Yang YK. (2009).
7. Thiago Rateke, Karla A. Justen, Vito F. Chiarella, Antonio C. Sobieranski, Eros Comunello, and Aldo Von Wangenheim "Passive Vision Region-Based Road Detection: A Literature Review", ACM Comput. Surv. 52, 2, Article 31 (March 2019).
8. Dr. Padmavathi S., Naveen, C. R., and V. Kumari, "A. Vision based Vehicle Counting for Traffic Congestion Analysis during night time", Indian Journal of Science and Technology, vol. 9, 2016.
9. Neethu John, B. Anusha, and Krishnan Kutty, "A reliable method for detecting road regions from a single image based on color distribution and vanishing point location", Procedia Computer Science 58 (2015), 29.
10. Z. Lu P. Hao, "Bi-system fuzzy control for single intersections", Information and Control 2001.
11. W. Wei Y. Zhang, "Fl-fn based traffic signal control" Proc. FUZZ-IEEE vol. 1 pp. 296-300 2002.
12. H. Kong, H. C. Akakin and S. E. Sarma, "A General- ized Laplacian of Gaussian Filter for Blob Detection and Its Applications", in IEEE Transactions on Cybernetics, vol. 43, no. 6, pp. 1719-1733, Dec. 2013.
13. I. Day "Scoot-split cycle and offset optimization tech- nique", Proceedings of TRB committee AHB25 adaptive traffic control 1998.
14. K. Pandit, D. Ghosal, H. M. Zhang and C. Chuah, "Adaptive Traffic Signal Control with Vehicular Ad hoc Networks", in IEEE Transactions on Vehicular Technology, vol. 62, no. 4, pp. 1459-1471, May 2013.
15. Jinyou Zhang and H. -. Nagel, "Texture-based segmen- tation of road images," Proceedings of the Intelligent Vehicles '94 Symposium, Paris, France, 1994, pp. 260-265.