Traffic Lights State Estimation Using Image Processing Methods and Hidden Markov Model

Anna Böndicz and Attila Kiss

ELTE Eötvös Loránd University, Budapest, Hungary
Faculty of Informatics Department of Information System
e1u7up@inf.elte.hu

Abstract. With the advent of self-driving cars, computer vision and reliability are becoming increasingly important. Recognition and classification of traffic lights are essential during autonomous driving. To avoid accidents, it is a very important task for the car to recognize and accurately classify the headlight under any circumstances. Previous articles have dealt with the subject lot, and several forms of approach may be useful. In this article, we propose a method for recognizing the condition of traffic lights that have already been determined. This method classifies traffic lights based on the degree of intensity and their position, and we also give the accuracy of the estimate with a probability rate. Using this, the states encountered are estimated using the Hidden Markov Model [19] to find the order of the most probable states at the detected traffic light, thus filtering out image processing errors in a given traffic light sequence. The method was also tested on two test databases, refined to achieve the best possible ratio, resulting in a hit rate of more than 95%.

Keywords: traffic light, classification, estimation, image processing, Hidden Markov model, HMM, DTLD

1 Introduction

Autonomous vehicles (AV) can bring social and economic benefits in areas such as reducing road accidents or saving fuel. These situations reveal future changes to the current land transport system. The most of traffic accidents are caused by human negligence, often due to ignoring the traffic light. The introduction of autonomous cars could significantly reduce the number of accidents. Traffic lights play a big role in the movement of intelligent vehicles. Thus, recognizing the existence and condition of traffic lights plays an important role in the autonomous mobility of the urban environment. The car automatically detects and evaluates the condition of the traffic light which increases the safety and comfort of the driver and his surroundings. Even for non-autonomous vehicles, it would be useful to detect the condition of traffic lights, alerting inattentive drivers to change the state of light, thus making intersections safer.
Scenes are not easy to recognize as a traffic light because the traffic light is very

small compared to other objects and many objects are similar in colour and shape to the traffic light. Several other factors make it difficult to detect traffic lights, such as [20]:

- Changes in light conditions at different times of the day, seasons, and weather conditions also make it difficult to detect traffic lights;
- The colors of the traffic light may be confused with the colours of the other elements of the images;
- Traffic lights can be obscured by other objects such as trees or poles, making detection and classification much more difficult;
- Traffic lights are visible at different angles in the picture or maybe deformed;

Most of the methods described in the articles are based on image processing techniques for detecting traffic lights and their condition. However, they warn, that it is worthwhile to supplement this approach with another technique to ensure greater consistency in the validation of results. To remove the traffic lights and their condition from the image, image distortion and uncertainty should be checked in the best possible way. In this context, this study proposed image processing and estimation of the conditions encountered using the Hidden Markov Model (HMM) [15] to find the order of the most probable conditions at the detected traffic light. The results obtained after applying the proposed method made it possible to increase the stability in determining the current state of the traffic light.
The rest of the article is arranged as follows, Section 2 describes the related work. Section 3 describes the proposed method. Section 4 discusses the results and the constraints encountered. Finally, Section 5 presents our conclusions and future work.

2 Related works

Because computer vision is essential in autonomous driving, several articles have addressed the classification of traffic lights over the years, so researchers have developed many different methods to achieve the most accurate estimate possible. In theory, in 1999, in Article [4], general methods for detecting traffic lights and other urban driving information were described. After some processed, the images based on color similarity spaces [19], while others presented methods for recognizing high-intensity circular areas [14] for recognizing traffic lights. The method in [12] focuses on real-time detection of light using balanced filtering and detection based on Hough transformation. In [9], a threshold was determined based on the intensity information to determine if there were lights in the camera image. Also, in [3], the states were determined using color intensity, the method discussed in the article consists of three steps. Grouping, filtering, and status determination. With this method, an accuracy of 85-95% was achieved. In [11], the authors proposed a new approach to signal light recognition based on Euclidean distance transformation and local contour pattern that uses both color and shape information to detect traffic lights. In [10], a histogram filter is used

to infer the image range of light and determine color. Diaz, Cerri and Sanchez [21], gave search colours and,visual characteristics similar to traffic lights, Then follows the trace phase, to help retain areas that are appropriate traffic lights and filter out those that are not.
Binangkit and Widyantoro [1] detected the color of traffic lights using a machine learning method. HSV color format was used to extract the characteristics of the images. The color model of traffic lights was constructed using a learning algorithm. The learned model was then classified based on whether or not the area of the pixels contained the color of traffic lights. Salary et al [17] presented an approach that perceives and recognizes traffic light-based image processing. This method included HSV color space segmentation and support vector machine training. Michael and Schlipsing [13] have developed a method for efficient video-based classification of said condition, with particular reference to the displayed pictogram and the additional ability to reject false perceptions. This gave a classification quality of 89.9%. Ji et al [7] proposed a method for integrating the visual selective attention (VSA) model and HOG functions for the detection and recognition of traffic lights. The HOG characteristics of the traffic lights and the SVM classifier were used to determine the exact regions of the traffic lights. The method of Sathiya et al. [18] involved displaying the timer using a threshold by colour segmentation, and this method recognized the LED digits on the timer display based on colour segmentation. With the proposed method, 96% an accuracy was achieved.
In [8], Jie et al. Integrated methods based on shape and color distribution. Choi, Ahn, and Kweon [2] propose the integration of two algorithms for the detection of intersections and traffic lights for real-time and low computational complexity. It is assumed that the traffic light and the intersection are often together. Image processing methods were used to detect the intersection and the traffic lights, but a probability template was also used to detect the traffic lights only to check the condition of the traffic lights. Gomez et al. [6] proposed a method that combines image processing with the estimation state routine developed by Hidden Markov Models (HMMs). This method helps determine the current state of the indicator light based on the states obtained by image processing and filters out any errors for states determined based only on image processing.

Article	**Number of States**	**Number of Estimates**	**Accuracy**
[14]	3 (red, yellow, green)	-	more than 80%
[12]	2 (red, green)	4000	98%
[11]	3 (red, yellow, green)	35 710	91%
[1]	3 (red, green, off)	800	96%
[17]	5 (red, red-yellow, yellow, green, off)	1802	96%
[13]	2 (red, green)	4537	98%
[2]	4 (red, green, yellow, off)	649	90%

Table 1. A few accuracy results achieved so far

3 Traffic light status determination

The method can be divided into 3 major parts:

A Pre-processing
B Classification
C Inaccuracy correction

A Pre-processing

Our method already assumes a processed image where we know the position of the traffic light and can identify the given light. The topological analysis determined the position of the traffic light and the x and y coordinates of the two corners of the rectangular shape, based on which we were able to highlight the traffic light from the image using OpenCV during the implementation. The result of which is shown in Figure 1. Then we only had to deal with the pictures of the traffic lights. In the next step, we pre-processed the images using image processing technologies. Figure 2 This means that all input images have been converted to the same format. To do this, we rotated the image so that the traffic light was as visible as possible on the rectangle and centred it, using a Sobel filter. Then, based on edge recognition, we made a rectangular mask. Finally, each image was resized to 32×32. It is important to make all the images the same size so that we can send them through the same classification steps. It is good to have square input dimensions because they can be rotated (and remain the same) and analysed in smaller square spots. For each signal traffic

Fig. 1. Traffic light highlighting from them image

light image, the expected output is also given, which can be coded for all traffic lights at the same time. To this, we added to each image a five element zero-array

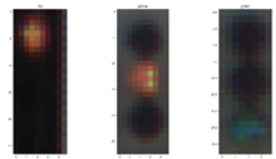

Fig. 2. Original images

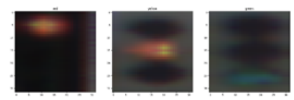

Fig. 3. Standardized images

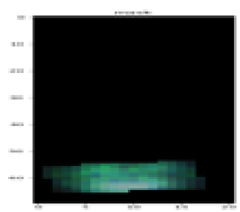

Fig. 4. Masked image

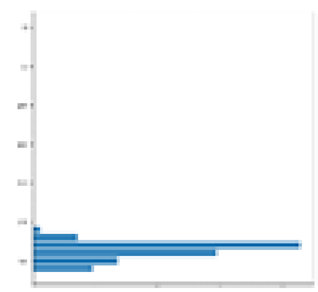

Fig. 5. Brightness feature

label, to representing the class of traffic lights, where each element represents a class. Since we have five classes $(off, red, yellow, red-yellow, green)$, we set the order to [red value, red-yellow value, yellow value, green value]. If the traffic light is later classified in a class, the element corresponding to that class is rewritten to 1. For example, if the image originally labeled $[0, 0, 0, 0, 0]$ is considered a yellow rating, let it be $[0, 0, 0, 0, 0]$.

B Classification

Images will be converted from RGB colour space to HSV colour space, as classification will be done based on the brightness characteristics of the images. Then comes the actual classification, where we classify traffic lights into one of five classes. This was done by dividing the image horizontally into 3 equal parts. These will represent the red, yellow, and green ranges, as with most traffic lights: the top region is red, the middle is yellow, and the bottom is green. After this, the brightness intensity was calculated for all three regions. One would think that in which region the intensity is the highest, we classify the traffic light in the corresponding class. However, we still have to deal with two states, the red-yellow, and the off-states, so we cannot solve the classification so easily. We calculate that the intensity of a given region is a fraction of the total intensity of the whole image, thus assigning a probability to each region. To calculate the probability of the red-yellow state, calculate the combined intensity of the red and yellow regions as a fraction of the total intensity of the total image and then multiply this value by the difference between the intensities of the red and yellow regions and the quotient of the combined intensities of the red and yellow regions. (i.e., this value will be 0 if the combined red and yellow intensity values are divided between the red and yellow regions so that all intensities are either in the yellow or red region and will be 1 if the red and yellow region has the same intensity value.) Thus, a good approximation can be obtained for the probability that the given traffic light is in the red-yellow state. The probability of the off-state was given similarly to that of the red-yellow state. We first calculated the difference of the region intensity values in pairs: red and yellow, yellow and green, red and green. We then calculated for these three values the proportion of the total intensity and took its complement. Finally, we multiplied these three values. This probability will be 0 if all intensities are in a region and will be 1 if the intensity value is the same in each region. Since the sum of these probabilities does not give 1, so that once we have calculated the probabilities for each state, we still need to add them together. To do this, we simply add the five probabilities and divide them by five. This will give us a relatively accurate probability for each condition. The traffic light is classified in the class with the highest probability. These probability rates also point to how reliable the result obtained is and will be useful in filtering out erroneous results in the future.

C Inaccuracy correction

To eliminate image processing errors, we recommend using HMM [15] to improve the results obtained. This method helps to maintain the viability and flexibility of image processing, for which HMM proves to be an effective mathematical tool in which we find the most probable sequence of states $Q = q_1, q_2, ..., q_t$ for a given observation sequence $\mathbf{O} = O_1, O_2, ..., O_t$. To do this, we determined the following quantity:

$$\delta_t(i) = \max_{q_1, q_2, \ldots, q_t} P[q_1 q_2 ... q_t = i, O_1 O_2 ... O_t | \lambda]$$

that is, $\delta_t(i)$ is the best value (highest probability) along a sequence at time t that takes into account the first observation t and ends in the S_i state. We have induction

$$\delta_{t+1}(j) = [\max_i \delta_t(i) a_{ij}] b_{ij}(O_{t+1}).$$

To actually retrieve the state sequence, we need to trace the argument that maximized $\delta_{t+1}(j)$ for each j and t. This is done through array $\Psi_t(j)$. The complete procedure for finding the best state sequence can now be defined as follows.
The first step was to create a Markov chain that describes the change in the state of the traffic light. For this, a 5-state Markov chain as shown in Figure 4 was used. Figure 6 shows the off state, which means that an undetected, i.e., either incorrectly detected traffic light that is not actually a traffic light or a non-functioning (non-lit) light, which means that certain limitations in image processing do not allow other classification into a possible condition. The next step is to define each element of the HMM:

- N, the number of model states. Each state is denoted as $S = S_1, S_2, ..., S_N$.

- M, the number of each observation symbol per state.
- Probability distribution of the state transition $A = a_{ij}$.
- Probability distribution of the observation symbol in state j, $B = b_j(k)$
- The initial state distribution $\pi = \pi_i$

In this article, we considered the following elements of HMM:
$states = Off, Red, Yellow, Red - yellow, Green$
$\pi_i = 0.2, 0.2, 0.2, 0.2, 0.2$ When determining each parameter, the probabilities of

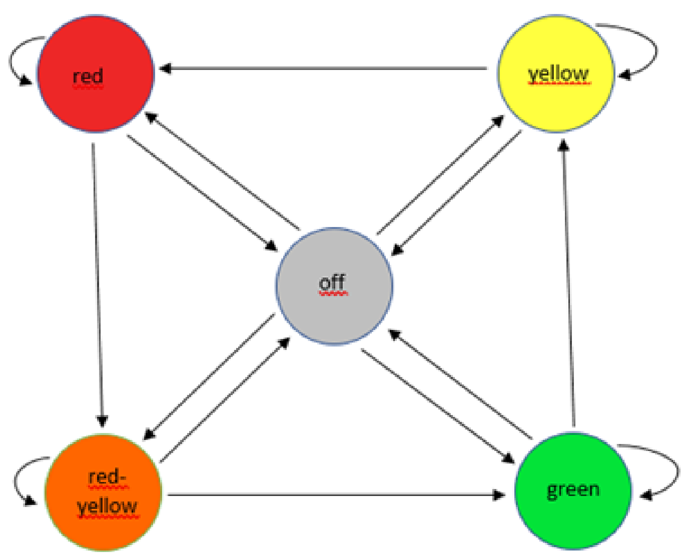

Fig. 6. Markov model with 5 sates

	Off	Red	Yellow	Red-Yellow	Green
Off	0.2	0.2	0.2	0.2	0.2
Red	0.3	0.35	0	0.35	0
Yellow	0.3	0.35	0.35	0	0
Red-Yellow	0.3	0	0	0.35	0.35
Green	0.3	0	0.35	0	0.35

Table 2. Hidden state transition matrix

the initial states were taken to be the same in all states. Transition and observation of the same condition were more likely determined. Additional relevant information is to consider that the probability of observing the yellow state is not representative of the system. It can be considered as the second basic problem of HMM to obtain a more coherent state, starting from the states provided by image processing, using the technique of the Viterbi algorithm [15]. The whole procedure can be defined as follows:

a Initialization:

$$\delta_1(i) = \pi_i b_i(O_1), \quad 1 \leq i \leq N$$

Where O_1, it is the first element of the given observation sequence.

$$\Psi_1(i) = 0$$

Where $\Psi_1(i), itisanarray.$

b Recursion:

$$\delta_t(j) = \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}] b_j(O_t), \quad 2 \leq t \leq T, \quad 1 \leq j \leq N$$

$$\Psi_t(j) = \arg\max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}], \quad 2 \leq t \leq T, \quad 1 \leq j \leq N$$

c Termination:

$$P* = \max_{1 \leq i \leq N} [\delta_T(i)]$$

$$q*_T = \arg\max_{1 \leq i \leq N} [\delta_T(i)]$$

d Path (state sequence) backtracking:

$$q*_T = \Psi_{t+1}(q*_{t+i}), \quad t = T-1, T-2, ..., 1.$$

The third basic problem of HMM is to define a method for adjusting the model parameters (A, B, π) to maximize the probability of a given observation sequence in the model. For this, we used the Baum-Welch algorithm, an iterative procedure. To describe the re-estimation (iterative update and correction) of the HMM parameters, we first determine the values of $\xi(i, j)$, the probability that we are in state S_i at time t and the state S_j at time t + 1, given the model and the observation in the case of order, i.e

$$\xi_t(i, j) = P(q_t = S_j, q_{t+1} = S_j | O, \lambda)$$

$$\xi_t(i, j) = \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{P(O|\lambda} = \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}$$

where, α and β can be given using the Forward-Backward procedure [6]:

$$\alpha_t(i) = P(O_1 O_2 ... O_t, q_t = S_i | \lambda)$$

i.e., the probability of a partial observation sequence, $O_1, O_2 ... O_t$, (to time t) and state S, t, taking into account model A.

a Initialization:

$$\alpha_1(i) = \pi_i b_i(O_1), \quad 1 \leq i \leq N.$$

Where $\Psi_1(i), itisanarray.$

b Induction:

$$\alpha_{t+1} = \Big[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\Big] b_j(O_{t+1}) \quad 1 \leq t \leq T-1, \quad 1 \leq j \leq N.$$

c Termination:

$$P(O|\lambda) = \sum_{j=1}^{N} \alpha_T(i)$$

$$\alpha_T(i) = P(O_1 O_2 ... O_T, q_T = S_i|\lambda)$$

Similarly, we can determine β backwards:

$$\beta_t(i) = P(O_{t+1} O_{t+2} ... O_T, q_t = S_i|\lambda)$$

i.e that is the probability of the partial observation sequence from $t+1$ to the end, the given S_i state at time t, and the model h.

a Initialization:

$$\beta_T(i) = 1, \quad 1 \leq i \leq N.$$

Where $\Psi_1(i), itisanarray.$

b Induction:

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j), \quad t = T-1, T-2, ..., 1, \quad 1 \leq j \leq N.$$

We can then define,

$$\gamma_t(i) = P(q_t = S_i|O, \lambda),$$

the probability that we are in the Si state at time t concerning the order of observation and the model. We can then combine this with k:

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j).$$

which can be interpreted as the expected (temporal) number of visits to the Si state, or equivalent, the expected number of transitions from the SI state (excluding the time gap t = T from the summation). Similarly, the summation of $\xi_t(i,j)$ over t (from $t = 1$ to $t = T-1$) can be interpreted as the expected number of transitions from the S_i state to the S state. That is

$$\sum_{t=1}^{T-1} \gamma_t(i) = expected\ number\ of\ transitions\ from\ S_i$$

$$\sum_{t=1}^{T-1} \xi_t(i) = expected\ number\ of\ transitions\ from\ S_i to S_j$$

Using the above formulas, we can give a method for re-evaluating the parameters of HMM:

$$\bar{\pi}_i = excepted\ frequency\ (number\ of\ times)\ in\ state\ S_i\ at\ time\ (t=1) = \gamma_1(i)$$

$$\bar{a}_{ij} = \frac{excepted\ number\ of\ transitions\ from\ state\ S_i\ to\ state\ S_j}{excepted\ number\ of\ transitions\ from\ state\ S_i} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

An important aspect of the reassessment procedure is that the stochastic limitations of the HMM parameters,

$$\sum_{i=1}^{N} \bar{\pi}_j = 1$$

$$\sum_{j=1}^{N} \bar{a}_{i,j} = 1, \quad q \le i \le N$$

are met for each iteration. A detailed description of the Forward-Backward and Baum Welch algorithm can be found in [6]. The result state of the last step of the system is more consistent because the result state takes into account the results of image processing of several previous images. Consequently, the system is less vulnerable to inaccuracies in image processing.

4 Experiments and Results

To detect the robustness of different traffic lights and to demonstrate robustness under multiple lighting conditions, we used the DriveU Traffic Light Data Set (DTLD) [5]. The DTLD contains more than 230,000 annotated traffic lights in 2-megapixel camera images. The data set was recorded in 11 cities in Germany at a frequency of 15 Hz. Images have a resolution of 16 or 8 bits.

The notes include the coordinates of the bounding box (top left corner, width,

	Off	**Red**	**Yellow**	**Red-Yellow**	**Green**
Off	234	2	7	56	23
Red	12	15704	141	131	3
Yellow	4	23	3600	121	105
Red-Yellow	4	336	124	2248	34
Green	17	0	87	41	8386

Table 3. Confusion matrix

and height) as well as the properties of the traffic lights such as viewpoint orientation, relevance, orientation, lighting units, status, pictogram. From which we used relevance and status properties.
31534 traffic lights were used for testing, where the traffic light relevance was at least 55%. Thus, we managed to achieve an 85% hit rate without the help of HMM. Where there was no case where you would have classified a red traffic light as green. HMM improved the method a lot, so we finally managed to achieve a 96% success rate.

5 Conclusions and Future works

This paper presented a robust system based on image processing and a hidden Markov model (HMM) through image processing techniques as well as the Viterbi, Forward-Backward, and Baum-Welch algorithms that increase detection accuracy in rare sequences. In the future, the accuracy of the system could be improved by additional image processing technologies such as color recognition, or the HMM could also be improved by criteria tied to the sequence, such as how many elements can be used to observe a given state, and so on. Furthermore, with a new learning machine approach, such as the Gaussian process [16].

6 Acknowledgments

The project has been supported by the European Union, co-financed by the European Social Fund (EFOP-3.6.3-VEKOP-16-2017-00002).

References

[1] Jeffrey L Binangkit and Dwi H Widyantoro. “Increasing accuracy of traffic light color detection and recognition using machine learning”. In: *2016 10th International Conference on Telecommunication Systems Services and Applications (TSSA)*. IEEE. 2016, pp. 1–5.

[2] Jongwon Choi, Byung Tae Ahn, and In So Kweon. “Crosswalk and traffic light detection via integral framework”. In: *The 19th Korea-Japan Joint Workshop on Frontiers of Computer Vision*. IEEE. 2013, pp. 309–312.

[3] Tin Trung Duong et al. “Near real-time ego-lane detection in highway and urban streets”. In: *2016 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia)*. IEEE. 2016, pp. 1–4.

[4] Uwe Franke et al. “Autonomous driving goes downtown”. In: *IEEE Intelligent Systems and Their Applications* 13.6 (1998), pp. 40–48.

[5] Andreas Fregin et al. “The DriveU traffic light dataset: Introduction and comparison with existing datasets”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 3376–3383.

[6] Andres E Gomez et al. “Traffic lights detection and state estimation using hidden markov models”. In: *2014 IEEE Intelligent Vehicles Symposium Proceedings*. IEEE. 2014, pp. 750–755.

[7] Yang Ji et al. "Integrating visual selective attention model with HOG features for traffic light detection and recognition". In: *2015 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2015, pp. 280–285.

[8] Ying Jie et al. "A new traffic light detection and recognition algorithm for electronic travel aid". In: *2013 Fourth International Conference on Intelligent Control and Information Processing (ICICIP)*. IEEE. 2013, pp. 644–648.

[9] Sung-Kwan Joo et al. "Traffic light detection using rotated principal component analysis for video-based car navigation system". In: *IEICE TRANSACTIONS on Information and Systems* 91.12 (2008), pp. 2884–2887.

[10] Jesse Levinson et al. "Traffic light mapping, localization, and state detection for autonomous vehicles". In: *2011 IEEE International Conference on Robotics and Automation*. IEEE. 2011, pp. 5784–5791.

[11] Yue Li, Guobao Zhang, and Yongming Huang. "Adaptive wavelet packet filter-bank based acoustic feature for speech emotion recognition". In: *Proceedings of 2013 Chinese Intelligent Automation Conference*. Springer. 2013, pp. 359–366.

[12] Frank Lindner, Ulrich Kressel, and Stephan Kaelberer. "Robust recognition of traffic signals". In: *IEEE Intelligent Vehicles Symposium, 2004*. IEEE. 2004, pp. 49–53.

[13] Matthias Michael and Marc Schlipsing. "Extending traffic light recognition: Efficient classification of phase and pictogram". In: *2015 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2015, pp. 1–8.

[14] Jin-Hyung Park and Chang-sung Jeong. "Real-time signal light detection". In: *2008 Second International Conference on Future Generation Communication and Networking Symposia*. Vol. 3. IEEE. 2008, pp. 139–142.

[15] Lawrence R Rabiner. "A tutorial on hidden Markov models and selected applications in speech recognition". In: *Proceedings of the IEEE* 77.2 (1989), pp. 257–286.

[16] CE Rasmussen. "Gaussian processes in machine learning Advanced lectures in machine learning". In: *Lect. Notes Comput. Sci* 3176 (2014), p. 6371.

[17] Mehdi Salarian, Andrea Manavella, and Rashid Ansari. "A vision based system for traffic lights recognition". In: *2015 SAI Intelligent Systems Conference (IntelliSys)*. IEEE. 2015, pp. 747–753.

[18] S Sathiya, M Balasubramanian, and D Vishnu Priya. "Real time recognition of traffic light and their signal count-down timings". In: *International Conference on Information Communication and Embedded Systems (ICICES2014)*. IEEE. 2014, pp. 1–6.

[19] Tadayoshi Shioyama et al. "Measurement of the length of pedestrian crossings and detection of traffic lights from image data". In: *Measurement Science and Technology* 13.9 (2002), p. 1450.

[20] Gangyi Wang et al. "Hole-based traffic sign detection method for traffic signs with red rim". In: *The Visual Computer* 30.5 (2014), pp. 539–551.

[21] Yu-Jun Zheng et al. "Localized biogeography-based optimization". In: *Soft Computing* 18.11 (2014), pp. 2323–2334.