The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Traffic Light Recognition using Image Processing Compared to Learning Processes

Raoul de Charette, Fawzi Nashashibi, *Member IEEE*

***Abstract*—In this paper we introduce a real-time traffic light recognition system for intelligent vehicles. The method proposed is fully based on image processing. Detection step is achieved in grayscale with spot light detection, and recognition is done using our generic "adaptive templates". The whole process was kept modular which make our TLR capable of recognizing different traffic lights from various countries.**

To compare our image processing algorithm with standard object recognition methods we also developed several traffic light recognition systems based on learning processes such as cascade classifiers with AdaBoost.

Our system was validated in real conditions in our prototype vehicle and also using registered video sequence from various countries (France, China, and U.S.A.). We noticed high rate of correctly recognized traffic lights and few false alarms. Processing is performed in real-time on 640x480 images using a 2.9GHz single core desktop computer.

I. INTRODUCTION

SINCE traffic scenes are complex and include lots of information, keeping constant attention on the traffic signs is not an easy task for drivers. Therefore some traffic data (signs) can be missed for several causes such as the complexity of the road scene, the high number of visual information, or even the driver's stress or visual fatigue [1].

In order to assist this task, several driver assistant systems have been suggested in the past years using either database information (i.e. learned Geographic Information Systems) or on-vehicle sensors (i.e. laser, camera, etc.) to provide various environment information such as traffic signs, speed limits, traffic lights, crosswalks, … –, or any other information like pedestrian or obstacles. The specific functionality of traffic lights detection shall be very useful since traffic lights position and state (*go*, *stop* or *caution*) provide good knowledge of the traffic environment such as high probability of crossroads/crosswalk, dangerous area, etc. Furthermore detecting traffic lights with an on-vehicle camera could also be used to improve fusion of GPS and camera visual data in order to make visual projection of road information on the windshield.

Because all (or the upmost of) the previous works were only applied for *suspended traffic lights* recognition, we propose in this paper a generic modular real-time algorithm for traffic light recognition. That is to say, a method which will handle *supported* traffic light recognition as well as *suspended* traffic light. Furthermore in contrary to most of the existing works our algorithm not only works in rural or semi urban but also in full urban environment.

The paper outline is as follows. Section II is dedicated to the state of the art; we will describe some of the previous works published on traffic lights recognition. Our method based on image processing is presented in Section III while learning processes for traffic light recognition are detailed in Section IV. Section V describes outlines of learning processes methodology used for comparison with our algorithm. Finally, results of our method are presented and compared with learning processes in Section VI.

Fig. 1. 1st and 2nd pictures are two French *supported* traffic lights. 3rd picture is a Belgium *supported* traffic light. 4th picture is a U.S.A. *suspended* traffic light.

II. RELATED WORK

As shown in Fig 1, traffic lights are very different across the world. However we can still distinguish two main types of traffic lights: suspended traffic lights (last picture of Fig. 1), and supported traffic light (the first three pictures of Fig. 1).

Almost all the previous researches have been applied only on suspended traffic lights and use color sensor in all relevant cases. Indeed, recognition of *suspended* traffic lights is much easier since we could guess that the background is almost static and generally include a sky area. First attempts of traffic lights recognition were used either in non-real-time applications as presented in [2], or in real-time but with a fixed camera as in [3]-[4]. Thus, Tae-Hyun H. et al. proposed in 2006 an approach to detect traffic lights [5] which consists of a color thresholding on the upper part of the image completed by a Gaussian convolution on the result mask, in order to detect light emission of the traffic signals. In 2007, Kim Y.K. et al. [6] also showed that suspended traffic lights can be detected with an overall color-based

Manuscript received March 9, 2009.

R. C. is from the Robotics Centre, of MinesParisTech, 60 boulevard Saint-Michel F-75272, Paris cedex 06 FRANCE, (phone: (33) 1-40-51-94-54; fax: (33) 1-43-26-10-51; e-mail: raoul.de_charette@ensmp.fr)

F.N. is both from the Cent Robotics Centre, of MinesParisTech and from INRIA, Imara Team, BP 105, 78153 Le Chesnay Cedex, FRANCE. (phone: (33) 1-39-63-52-56, e-mail: fawzi.nashashibi@ensmp.fr)

thresholding and segmentation. But even if good results were achieved, this method is not suitable in our case, mostly because of the constraints due to supported traffic lights and urban environment.

Various algorithms were previously used to attempt to recognize traffics lights: from Hidden Markov Models [2] to color segmentation [5][6] but almost all previous attempts were done on *suspended* traffic lights in rural or semi urban environment. Yet, Lindner F. et al. present in their paper "Robust Recognition of Traffic Signals" [7] a comparison of different methods in order to detect and recognize German traffic lights in an urban environment. Various detectors are presented, such as a color detector, a shaped based detector with GPS, or even an AdaBoost learning method. But in contrast to our method the only detector which meets real-time constraints in their research is a combination of *"costly color sensor"* and differential GPS map.

In our system, in order to be more flexible and less dependent on the sensor quality and therefore, meet global cost requirements, we decided not to use color as main information in the detection step.

III. Method Based on Image Processing

A. System Overview

Our goal was to design a modular algorithm for Traffic Lights Recognition (TLR) that is able to detect *supported* and *suspended* traffic lights in a dynamic urban environment. Regarding the previous works, we decided to base our algorithm on the light property emission of traffics lights. Therefore, the layout of our TLR algorithm consists mainly of three steps as illustrated in Fig. 2.

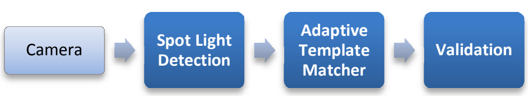

Fig. 2. Representation of TLR layout.

First Step First a robust Spot Light Detection (SLD) is executed on the whole grayscale source image in order to detect all visible spots lights.

Second Step After an intermediate association step which goal is to define relevant candidates region from the previously detected spots, an Adaptive Template Matcher (ATM) is executed. The matcher algorithm which contains geometric and algorithmic templates defined as explained in Section III.C evaluates matching confidence for every template candidates.

Third Step Finally, a simple validation step is used to filter the candidates marked by the ATM.

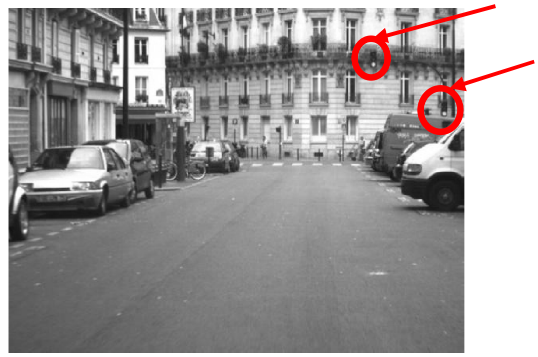

Fig. 3. Source image that was used as reference for further demonstrations. Traffic lights *facing the camera* are emphasized only to be more easily visible by the readers.

B. Spot Light Detection (SLD)

Despite the fact that traffic lights can be very different, all share the common property to emit light. Therefore a robust SLD appears to be the best base for TLR. The aim of this SLD step is to miss the fewest sought lights as possible. This task is not an easy task due to the complexity of an urban scene and especially when trying to detect small spots light in order to detect far traffic lights.

To detect spots light in the source image we use an algorithm we already described in [8]. This algorithm uses the top-hat morphological operator [9]-[10] to detect the bright areas and then applies shape filtering and region growing algorithm to keep only the consistent areas. Only grayscale information is used in the SLD step. Therefore, it is few sensitive to illumination variation and color distortion.

The SLD we use detect up to 90% of all the visible lights emitted from the traffic lights and is able to find any spot light as soon as it is 4 pixels wide or more. Fig. 4 illustrates the result of the SLD algorithm.

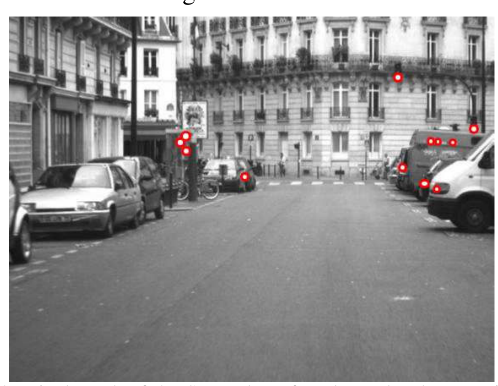

Fig. 4. Final result of the SLD where found spot has been projected on the source image. As we can see even if there are still false alarms, no spots light were missed. (5 spots light detected).

At this step false alarms are not yet a problem as the next steps of our TLR will be able to reject them later. Conversely, any missed lights by the SLD prevent detection of the associated traffic light.

C. Adaptive Template Matcher (ATM)

As mentioned above, one of our constraints was to have a fully generic TLR. Therefore, and in order to be able to adapt our algorithm to different types of traffic lights, we designed Adaptive Templates. Those templates are evaluated with our Adaptive Template Matcher (ATM).

Template Matching was used previously in different recognition process [11]-[2]. This technique is usually slow when applied on the whole image. However, we use the previously detected spots as hypothesis for template matching. Hence, we create candidates only where spots were detected. Those candidates will then be evaluated and either accepted or rejected according to their matching confidence value. The fact that we use the template matching only where spots were previously found, make it a lot faster (as detailed in the Results section).

An Adaptive Template can be defined as a combination of the 2D visual shape representations of the 3D elements which form the real object. In addition, templates also define algorithmic operators linked to one or more elements and which will be evaluated at run time. Both elements and operators can have different weights and matching confidence thresholds, or even be set to *non discriminatory*. Therefore, if any *non discriminatory* element or operator failed, it prevents the candidate from being rejected.

The matching process is the recursive evaluation of a template and its hierarchy, until all its elements and linked operators have been evaluated. The confidence value of an element is computed according to weight and confidence value of each child element or linked operator.

Geometry Definition ATM uses the common *top-down* approach to define template geometry. This simple approach, shown in Fig. 5, involves decomposing a real 3D object (traffic light in our case) into 2D visual shapes.

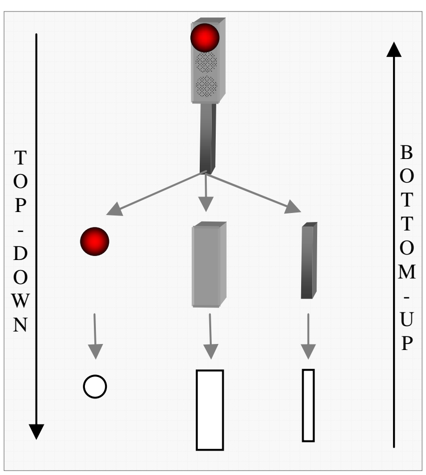

Fig. 5. Simple representation of both *top-down* and *bottom-up* approaches.

Adaptive templates contain elements which describe the visual appearance of the real object corresponding to the template. Those elements can be instance of various types - such as *Circle*, *Square*, *Rectangle*, *Spot*, *Container*, etc. -, and each type can have specific behavior. Fig. 6 illustrates relations between the elements types.

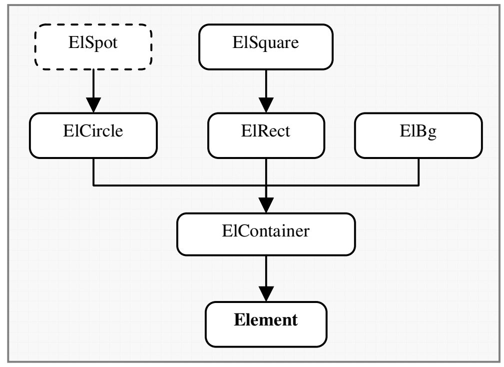

Fig. 6. Illustration of the element types classes relations. All element types derivate from the base class Element. *Container* type allows element to contain children elements. *Spot* is a special type which derivates from the *Circle* type and can be linked with a spot object previously returned by the Spot Light Detector.

Both element position and size are expressed in the parent referential taking into account that the parent element width is normalized. For instance, a child element which width, is one quarter of its parent width and which is also right aligned in its parent, will have a 0.25 width, and its x-axis position will be set to 0.75 (that is to say, 1.0-width). Conversely, height is not expressed explicitly but a width/height ratio has to be defined for each element.

According to those simple rules, we can easily describe fully scalable templates geometry. Furthermore, programmers can easily add new element types with special behavior, like we did for *Spot* type.

Operator Definition In addition to geometry definition, Adaptive Templates contain also operators which can be linked to one or more elements. Those operators are algorithmic processes which take elements as inputs and return output values depending on their inner behavior. Therefore, adding operators to an adaptive template leads to more complex matching and consequently more efficient matching.

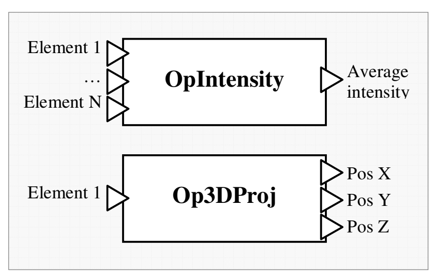

Fig. 7. Scheme of two different operators. Operator Intensity output the average intensity of the first input relatively to the others inputs. Operator 3DProj takes only one input, and outputs position of this element in the 3D world, according to its metric properties and according to camera calibration data.

When defining an adaptive template, valid range, weight, or confidence threshold can be defined for each operator and even for each operators output. Using those properties, when operators are evaluated, a confidence value is set for each operators output depending on the ability of the value output to fit in the valid range previously defined in the template. If the confidence value of an operator output is below its confidence threshold, the operator it belongs to will be set to invalid, except if the output was defined as non discriminatory. Likewise, if an operator is set to invalid or, if its confidence value does not reach the confidence threshold, the element it belongs to will be set to invalid except if the operator was defined as non discriminatory.

Template Candidate In order to evaluate a template matching hypothesis the ATM uses templates candidates. Those candidates inherit all their properties from their template. Using the known geometric relations between elements inside each template, candidate elements and operators are evaluated hierarchically. Depending on the confidence value each element and operator returned, a candidate confidence value is computed. Yet, as soon as any discriminatory operator/operators output/element is set to invalid during the evaluation process, the candidate is immediately rejected. Since some elements are not always visible, they can be set to non discriminatory. Therefore, it could increase the confidence value in case a *non discriminatory* element is visible.

In case of traffic light recognition, we defined three different templates; one for each state such as go, caution or stop. To represent a French traffic light with the top-down approach, we use three different elements: a *Spot* element for the lamp, a *Background* element for traffic light body, and a *Rectangle* for the pole. Since the pole can be hidden (because of vehicles, traffic signs, hanging traffic light, etc), this element is set to *non discriminatory*. Fig. 8 illustrates the geometry of the red traffic light template while Fig. 9 is the representation of the operators used for this template.

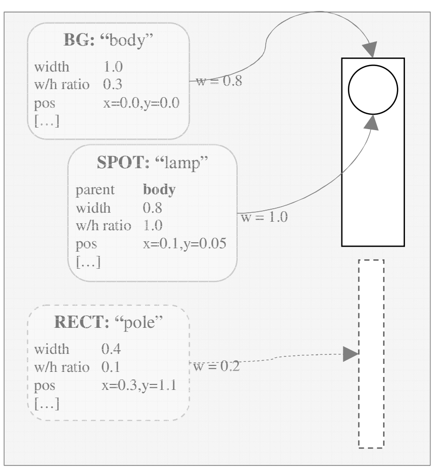

Fig. 8. Scheme representing adaptive template geometry of the "French red traffic light". Boxes on the left show some properties of the elements being pointed at. Label on the arrow is the weight of the element. As shown with dashed contours, *pole* is set to facultative.

Note that one of the improvements in progress will enable defining states inside templates; therefore, we will only have one template for the traffic light.

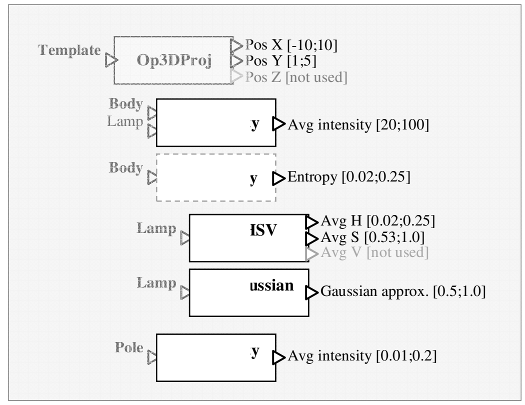

Fig. 9. Scheme representing operators used in adaptive template of "French red traffic light". Grey objects are set to *non discriminatory* such as operator *Entropy* or *PosZ* output of the 3DPosWorld operator.

The strength of our ATM is its fully generic nature. Since the whole class architecture was kept modular, programmers can easily add new element types or operators. Furthermore, defining new adaptive templates can be done very easily for non-programmer users also, and could even be done using XML format.

IV. Learning processes

A task such as object recognition can usually be achieved with two very different approaches: image processing or learning processes. In the section III and IV we proposed a TLR only based on image processing but the question arises whether learning processes could give better results for such a task. In order to evaluate the previously described TLR we also developed traffic light recognition based on the use of learning processes.

Using object-marked sequences acquired with an on-vehicle camera, we extracted traffic lights and non-traffic lights samples. Table I describes different datasets used for training and Fig. 10 show some of the samples used.

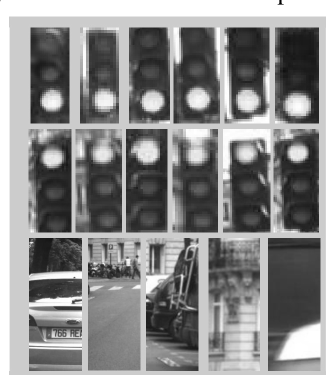

Fig. 10. Some of the positives and negatives samples used for training.

Each set of positive samples contain equal number of green traffic lights and red traffic lights. Because we didn't have enough *yellow* traffic lights this state won't be taken into account for the learning processes.

TABLE I
DATASETS USED FOR TRAINING

Set	Type	Number of positive samples	Number of negative samples	Samples size (pixels)
Set1_6x15	French traffic lights	2 112	11 180	6x15
Set2_13x33	French traffic lights	2 112	11 180	13x33

Using the data samples described we attempted several learning methods to detect traffic lights in grayscale image, such as: Multi Layer Perceptron, genetic, AdaBoost, etc. We also tested different features like: control points, connected control points [12], or haar features [13]. In this paper we will only detail the cascade classifier training with Adaboost and Haar features since this fast method gives good results for most of the objet-recognition tasks [13]. A future paper should be written to detail the exhaustive performances of each learning process.

To apply cascade classifier to the traffic light recognition problem we used the sets detailed in Table I. Before training, brightness of each sample was normalized. Each cascade classifier contains at least 9 boosting classifiers which lead to a big computation time. The cascade classifiers trained are described below in Table II and the evaluation of the training is detailed in Section V.

TABLE II
CASCADE CLASSIFIERS TRAINED

Boosting	Set	Number of boosting classifiers	Number of features used in the pipeline
Gentle AdaBoost 1	Set1_6x15	9	321
Gentle AdaBoost 2	Set2_13x33	11	330

V. RESULTS

A. Results

To evaluate the performances of either our TLR or the learning processes we used several urban sequences acquired with an on-vehicle camera. Then, we used a ground truth editor to mark the position and state of all traffic lights on each frame. Finally, we executed both methods on the sequences and compared the output of our algorithm with the ground truth. To measure the performances of our algorithm we used the standard precision and recall values computed as follow:

$$Precision = \frac{truePositives}{truePositives + falsePositives}$$

$$Recall = \frac{truePositives}{truePositives + falseNegatives}$$

Thus far, the tests were performed using a video stream database consisting of more than 20 minutes of "useful" urban scenes sequence. French sequences were acquired in Paris using our prototype vehicle. In addition to these sequences and in order to evaluate the ability of our method to recognize various traffic lights we also tested our algorithm on sequences acquired in others countries such as China, or USA. Table III describes the sequences used for the evaluation of our algorithms.

TABLE III
SEQUENCES USED FOR THE TESTS

Sequence	Country	Length	Camera	Number of frames with traffic lights
French Urban Sequence 1 On-vehicle	France	7'51"	Marlin F-046C	7 723
French Urban Sequence 2 On-vehicle	France	8'48"	Marlin F-046C	2 616
Chinese Urban Sequence 3 On-vehicle	China	12'07	-	

Fully detailed results of those tests are shown in table IV and some screenshots are displayed in Fig. 11 and Fig. 12. Processing was performed on a 2.9 GHz single core computer with 1GB RAM.

Fig. 11. Result of the whole Traffic Light Recognition process (2 traffic lights recognized). Sequence 1.

Fig. 12. Result of TLR (2 red traffic lights recognized). Sequence 2.

TABLE IV
RESULT OF THE FRAME PER FRAME MATCHING

Method	French Urban Sequence 1	French Urban Sequence 2	Chinese Urban Sequence 1
Our method based on image processing	P: **94%** R: **63%**	P: **75%** R: **44%**	P: **96%** R: **39%**
Gentle AdaBoost 1 Set 6x15	P: **51%** R: **66%**	P: **33%** R: **31%**	-
Gentle AdaBoost 2 Set 13x33	P: **24%** R: **87%**	P: **43%** R: **42%**	-

Letters "P" and "R" stand for "Precision" and "Recall".

Results detailed in Table IV show that our method based on image processing achieves better recognition results than learning processes. In addition to that our method is capable of recognizing traffic lights from other countries since it is very easy to add new traffic lights templates.

Conversely, to recognize other traffic lights with learning processes we would need to extract new samples from sequences to train the classifiers with these new samples. Since we didn't have enough sequences in foreign countries the learning processes couldn't be evaluated on these sequences.

It is important to notice that Table IV details the result of frame per frame matching. In case of temporal matching we can easily achieve better performance. For instance, our proposed method reaches 98% and 97% of precision and recall when evaluated with temporal matching.

B. Computation Time

For the tests, the TLR we suggested in this paper was executed in *real-time* (up to 26FPS) whereas the methods based on learning processes couldn't be executed in *real-time* due to the computation time needed. These performances confirm the results published in [7] in 2004. In their paper, Lindner et al. reached only 2Hz using a boosting cascade classifier for the detection.

Table V detail the computation time of our TLR based on image processing.

TABLE V
COMPUTATION TIME DETAILS

Main steps	Sub steps: Name	Sub steps: Duration (ms)	Duration (ms)
Spot Light Detection	Bright area analysis	22.2 ms	**34.2 ms**
	Blob extraction	1.6 ms	
	Blob filtering	10.4 ms	
Adaptive Template Matching	Candidates creation according to SLD result	0.1 ms	**2.1 ms**
	Template matching	2.0 ms	
Validation	Candidate selection	1.1 ms	**1.1 ms**
		TOTAL	**37.4 ms**

Whole Traffic Light recognizer computation time performed on a 2.9GHz single core desktop computer on video sequences.

In our method the reader will notice that the Spot Light Detection is the most time-consuming step. This is due to the use of morphological operators (bright area analysis) which are known for being really long to compute. Conversely, the Adaptive Template Matching is almost insignificant because candidates for the ATM are only created where spots were previously detected by the SLD.

C. Conclusion

We proposed in this paper, an image processing method to recognize *real time* Traffic Light in urban and rural environment. The proposed method is fully modular and capable of recognizing traffic lights from various countries.

Even though our TLR could still be improved (by adding for instance a tracking algorithm) we compared our method with standard object-recognition learning processes and proved that it reached up to 95% of precision which is better than the results achieved with cascade classifiers.

REFERENCES

[1] Kimura F.; Takahashi T., Mekada, Y., Ide I., Murase H., Miyahara T., Tamatsu Y.: "Measurement of Visibility Conditions toward Smart Driver Assistance for Traffic Signals" *Intelligent Vehicles Symposium, 2007 IEEE*.

[2] Zhuowen T., Ron L.: "Automatic recognition of civil infrastructure objects in mobile object mapping imagery using a markov random field model" *ISPRS, 2000 Amsterdam.*

[3] Chung Y-C., Wang J-M., Chen S-W.: "A Vision-Based Traffic Light System at Intersections", 2002.

[4] Lai A-H-S., Yung N-H-C.: "A Video-Based System Methodology for Detecting Red Light Runners", *MVA 98, Workshop on Machine Vision Applications*.

[5] Tae-Hyun H., In-Hak J., Seong-lk C.: "Detection of Traffic Lights for Vision-Based Car Navigation System", *2006 PSIVT*.

[6] Kim Y.K., Kim K.W., Xiali Y.: "Real Time Traffic Light Recognition System for Color Vision Deficiencies" *International Conference on Mechatronics and Automation, 2007 IEEE.*

[7] Lindner F., Kressel U., and Kaelberer S.: "Robust Recognition of Traffic Signals" *Intelligent Vehicles Symposium, 2004 IEEE.*

[8] De Charette R., Nashashibi F.: "Real Time Visual Traffic Lights Recognition Based on Spot Light Detection and Adaptive Traffic Lights Templates" *Intelligent Vehicles Symposium, 2009 IEEE.*

[9] Meyer F.: "Contrast Features Extraction" Proceedings, *2nd European Symposium on Quantitative Analysis of Microstructures in Material Sciences, Biology and Medicine, Caen (pages 374-380), 1977.*

[10] Serra J.: "Image analysis and mathematical morphology", Ac. Press, Vol. I, 1982.

[11] Jizeng W., Hongmei Y.: "Face Detection Based on Template Matching and 2DPCA Algorithm", *2008 Congress on Image and Signal Processing*, 1982

[12] Viola P., Jones M.: "Robust Real-time Object Detection", *International Journal of Computer Vision*, 2001

[13] Moutarde F., Stanciulescu B., Breheret A.: "Real-time visual detection of vehicles and pedestrians with new efficient adaBoost features", *Workshop of International Conference on Intelligent RObots Systems*, 2008