BULETINUL INSTITUTULUI POLITEHNIC DIN IAŞI
Publicat de
Universitatea Tehnică „Gheorghe Asachi” din Iaşi
Volumul 65 (69), Numărul 1, 2019
Secţia
ELECTROTEHNICĂ. ENERGETICĂ. ELECTRONICĂ

AUTOMOTIVE NIGHT VISION ALGORITHM FOR DETECTION OF THE LIGHT MOVING OBJECTS

BY

ALEXANDRU RUSU and GRIGORE MIHAI TIMIŞ[*]

"Gheorghe Asachi" Technical University of Iaşi,
Faculty of Automatic Control and Computer Engineering

Received: July 29, 2018
Accepted for publication: March 15, 2019

Abstract. In this paper, the authors present an automotive night vision algorithm that automatically detects the light moving objects present in car traffic. The system is part of a more complex project aimed at controlling a vehicle's headlamps based on the analysis of an image stream recorded by the video camera installed on the vehicle. The algorithm consists in image conversion, segmentation, grouping and filtering techniques, validation for the detected objects and a method for sending the resulted information to the headlights' controller. Running and simulating the application was done through a WiFi connection between the controller and the computer. System testing is done independently from the whole headlight system, but during real life scenarios, at night time.

Keywords: light moving objects; automotive algorithm; image conversion; moving object detection; headlight system beam; simulink; computer vision system toolbox.

1. Introduction

Road safety is the most important aspect to be considered in the field of car transports. Compared to other modes of transport such as airplane, train or ship, car transports present the highest risk of fatal accidents, according to road fatal accident statistics. Also, the rate of fatal accidents increases three to four

[*]Corresponding author: *e-mail*: mtimis@tuiasi.ro

times at night. The reason for this increase is the substantial diminishing of night visibility over daytime visibility. Thus, the possibility for the driver to remotely recognize obstacles, traffic signs or other hazard-generating elements is reduced due to the absence of colors, in-depth perception and daytime contrast, (Yen-Lin Chen *et al.*, 2011). Taking this into account, it is intended to build a vehicle headlight system that significantly improves visibility and, implicitly, increases driving safety at night, (Yen-Lin Chen *et al.*, 2011; Yen-Lin Chen *et al.*, 2009). Currently, series cars are equipped with a manual headlamp control system. In most cases, it has mainly two functions: long beam and short beam, (Xin-Li *et al.*, 2012; Antonio LópezJörg *et al.*, 2008).

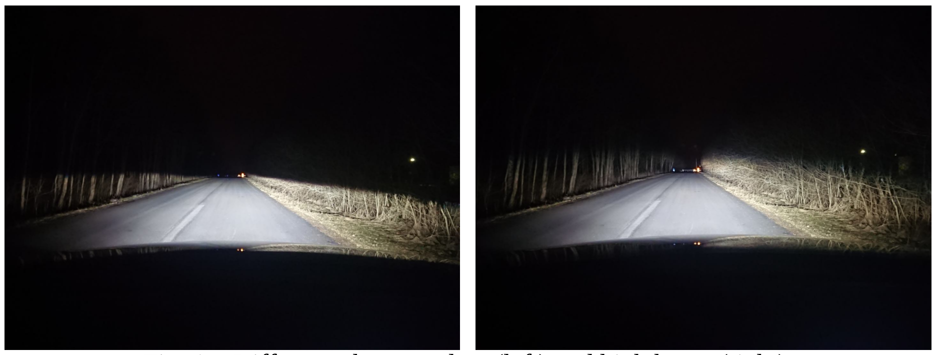
Fig. 1 – Difference between low (left) and high beam (right).

It can be observed in the Fig. 1 that the short beam covers a much smaller distance and is oriented predominantly on the road, while the long beam is dispersed on a much larger surface, both on the road and on the objects on the road or on its edge.

The main problems of this system come from the fact that the selection between the two illumination modes is done manually by the driver. The consequences of manual operation consist in the inaccuracy of choosing the switching moment. Specifically, there are many cases where drivers forget or do not switch in time between the short and long phase, blinding the drivers of the approaching cars.

An effective way to eliminate such a drawback is to automate the switching between the two phases and to adjust the luminous intensity in such a way that the driver of the approaching car is not blinded and the driver of the reference car does get his sight reduced (Yu-Chun Lin *et al.*, 2011; Ninomiya *et al.*, 1995). Situations that may arise in traffic and how to behave in such a system are the following, in case of a right-hand traffic (Chen Wei-Gang *et al.*, 2010):

a) car approaching from the opposite direction, in which case the left headlight of the car must reduce its intensity and change its orientation;

b) reference car approaching another vehicle going in the same direction - in which case the two headlamps must change their orientation in order to avoid the light beam hitting the car ahead, but keep their current intensity.

This specific headlight control can be achieved only by recognizing the previous scenarios in real life environment. This is why a vehicle detection method is needed on the road. More exactly, this consists in processing the video stream recorded by the video camera installed on the car so that the presence and, implicitly, the position of other moving cars is detected, (Ninomiya *et al.*, 1995; Weijie Liu *et al.*, 2003).

The currently-developed headlight control system is based on two LED headlamps, each of which is made of a 8×4 led array, a central unit that relies on the information received after the video is analyzed, a video recording and processing component designed to detect moving objects in traffic, and a system simulation and verification component. The subsystem that recognizes other cars in traffic is composed of a video camera connected to a development board that can communicate wirelessly with the center headlamp control unit, (Weijie Liu *et al.*, 2003; Moein Shakeri *et al.*, 2017).

2. Developing the Application

The module responsible of the light moving objects detection is composed of two modules: the video camera and the development platform. Still, for debug purposes, testing and simulations, a PC was used, connected to the development microcontroller.

This component can be run in two ways: directly on the vehicle, connected to the headlight control unit, and independently, controlled through a development environment from the PC. In this paper we will present the latest case where the simulations and tests were made both statically, with data sets on the laptop’s hard disk, and dynamically, on the vehicle, in real traffic conditions using a real-time video processing.

Thus, from a hardware point of view, a platform based on 3 components was used to develop and implement the application: Raspberry PI development board, Raspicam video camera and laptop to simulate and view the results obtained by running the application.

To reduce the complexity of the platform, a wired connection between the camera and the development board was used, and a wireless connection with SSH communication protocol, using wi-fi modules, between the development board and the PC.

From a software point of view, implementing the application involves the following steps:

• Set up Raspbian operating system on the development board to control video camera and interact with the development environment on laptop

• Set up and make a wi-fi connection between the PC and the development board

• Development of a frame by frame processing algorithm in the Matlab R2014a environment

• Export the algorithm in C code to be integrated with the application running on the headlamp control unit by using the Simulink development environment.

At the same time, the application can be divided into several parts, depending on the mode of operation that was chosen for the simulation. If a simulation that requires viewing of processed images is required, the application will need to use the existing graphics system on the operating system on which it is running. If the images viewing is not wanted, but only transmitting the LED control information is required, that information will be written to a file, taking into account that the simulation is performed independently of the headlamp control unit. The overview of the main elements of the application is as follows: Frame capture, Process the captured frame, Write information to file, Display the processed frames flow.

Using such a structure for the platform on which the application will run comes with a series of advantages and disadvantages.

The shortcomings of this structuring approach stem mainly from software limitations in the resource management done by the development environment and operating systems. Specifically, frame capture will differentiate between running time on the development board and how it runs in the development environment on the PC. The same kind of differences will be noticed in the frame processing part. Displaying the flow of frames implies increased complexity and a considerable decrease in efficiency as time. A display using openGL on the development board will be much more time costly than using the Matlab display mode on the Windows operating system. Because of this, the application's display function will only be used for development simulations. In addition, writing information to the file is optional, since the signals will be transmitted directly to the ECU once the object detection component has been integrated into the rest of the headlamp control system.

The advantages of such a structure are represented by the efficiency of the computational effort generated by the processing algorithm, effort redirected to the development board and not to the control unit. In addition, for such a structure it is very easy to check the efficiency. A real-time traffic test involves this interconnection of 3 components to be placed in the vehicle, each of which can be powered independently by the vehicle. At the same time, another advantage is the portability of the algorithm in C code through the development environment, Simulink being able to automatically generate C code on an operating system like the one used.

The hardware components used are:

- Raspberry PI development board B model
- Raspicam video camera
- TP-Link wn725n wi-fi dongle
- 5V power supply
- Ethernet cable
- SD Card 8Gb.

With regard to the software part of the application, the following development environments and additional components were used:

- Matlab R2014a with the Image Processing Toolbox for Matlab package;
- Simulink with the Computer Vision System Toolbox for Simulink;
- Raspberry PI Support Package for Matlab & Simulink;
- Matlab & Simulink camera driver;
- Raspbian Raspberry PI operating system;
- SmarTTY.

To implement the algorithm, Matlab was used, along with the Image Processing Toolbox. The motivation for this choice lies in the multitude of image processing functions contained in this toolbox and, last but not least, in the easy-to-use optimisation of the application by the specific and efficient method used by Matlab to operate with matrix elements underlying any image. The Simulink component of this development environment is used to construct a block-type model that masks most Matlab functions. The Computer Vision Toolbox includes blocks for image processing, being equivalent to the Matlab Image Processing toolbox. The advantage of using Simulink is that the model, created based on the Matlab algorithm, can run either on the operating system on which the environment is installed or directly on the development board using only its resources. In addition, the most important element is automatic C code generation, either in the form of an application or in the form of a code that can be integrated into the headlamp control application. The Raspberry PI support package for Matlab and Simulink contains a number of functions and blocks for each of the two components of the environment. They are primarily methods of easy interaction and control of Raspberry PI's hardware with features such as initialising or controlling the camera, making a SSH connection with the development board, or offering blocks responsible of video capture, calling on graphics libraries on development board for display, control of input-output pins, etc.

3. Application Structure

The application is structured in three major parts, as can be observed in Fig. 2.

- Obtaining and transmitting a sequence of images for processing;
- Processing each captured frame;
- Transmitting and reporting processing results.

a) Obtaining and transmitting a sequence of images for processing:

At this stage, it is intended to connect to the video camera, initialize it, and configure the resolution and frame capture frequency. After that, the camera is scheduled to record frames for processing. Also, for static simulation, camera initialization is replaced by selecting a set of frames stored on the hard drive.

b) Processing each captured frame: This stage consists of 4 steps of processing a frame. The information from images is reduced based on the intensity in each color channel (RGB) and a single intensity value is obtained for each pixel.

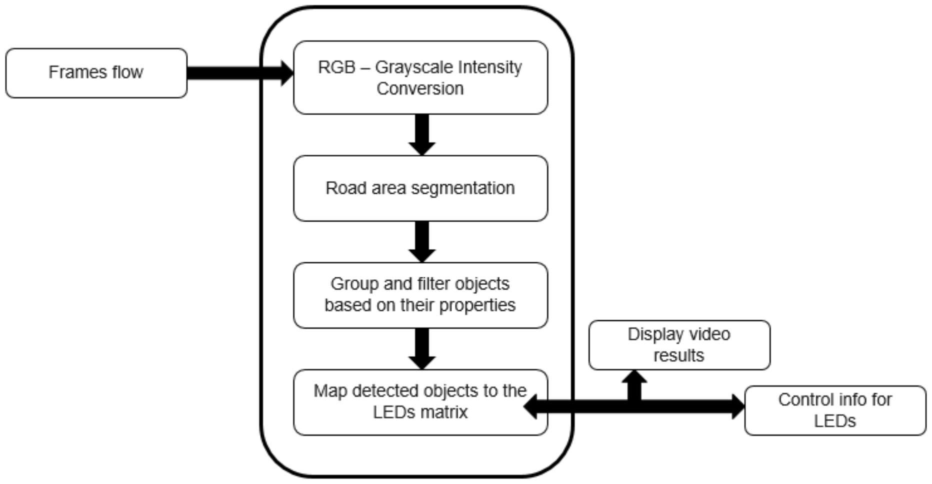
Frames flow
RGB – Grayscale Intensity Conversion
Road area segmentation
Group and filter objects based on their properties
Map detected objects to the LEDs matrix
Display video results
Control info for LEDs

Fig. 2 – Application workflow diagram.

Segmentation of the road area is necessary because, when traveling through the city, the number of bright objects is significantly higher than outside the city. Many of these objects come from the ambient lights from buildings on the roadside or from street lighting. This is why we are looking at a narrow range of the frame, where one can find the headlights - the area of the road.

Grouping and filtering the objects based on properties is required by the formation of other light objects in the area of the road. Headlamp lights can have reflections on the road or reflections on the body of other vehicles. Also, traffic signs or road markings may appear in the area of interest, which, due to the headlamps, may have a high luminous intensity. All these objects must be removed based on a group based on their properties.

The mapping of the detected objects is the part whereby the LED control signals can be generated. This is accomplished by overlapping a matrix of rectangles in the friction, each of which is mainly the area that each LED of the headlight will illuminate.

c) Sending and displaying the results of the processing: The video display of the results is optional, since it is not intended to implement a display inside the machine on which to stream the frame flow, the reasons being related to costs, safety and utility. However, when performing simulations, video display is required, despite the fact that it can affect the time complexity of the algorithm.

What is desired is to transmit to the headlamp control module signals as simple as possible, signals which, for verification purposes, are to be written to a file and log the information related to the frame number, the number of the object detected and the corresponding LED.

4. Implementing the Application

Each frame recorded by the camera is processed. The proposed algorithm for processing a frame is the following: Convert the image from color format (RGB) to a format based on tones of gray (grayscale intensity); Analyze the histogram of the grayscale image, Create a "mask" that emphasizes the area of interest for detection, starting from the brightest area of the background; Convert the image to a binary image, depending on a predetermined intensity threshold; Remove the light objects that are considered too small and those that do not meet a certain roundness threshold; Delimit a horizontal zone for determining headlamp pairing; Delimit a vertical zone for determining the headlamp mirroring on the road surface; Mark the light objects on the original image; Overlay a map over the area of interest with a matrix of rectangles representing the light area produced by each of the LEDs of each headlamp.

• RGB – Grayscale conversion: The conversion is done by removing the hue and saturation information from the image, but keeping the luminance information. The Matlab function that performs this conversion is rgb2gray and the results are shown in Fig. 3.

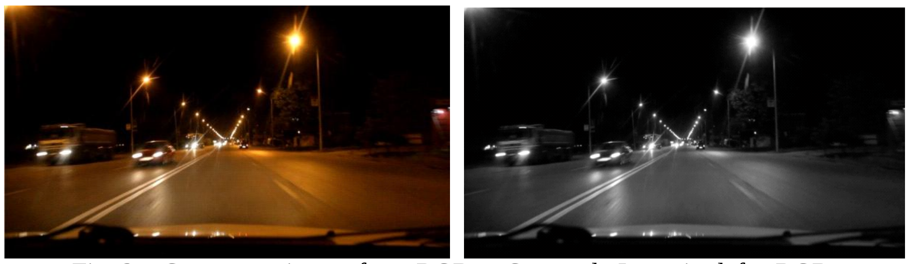
Fig. 3 – Convert an image from RGB to Grayscale Intensity left - RGB; right – Grayscale.

• Analyzing the grayscale image histogram: The need for this analysis comes from the fact that obtaining the background used for creating the "mask" is done by selecting the brightest area on the road. The absence of street lighting implies a much smaller area of light (more precisely the one achieved from the headlights of the car on which the camcorder is installed). For the algorithm to be effective both in the presence of ambient lights, as well as in their absence, dynamic background selection is required depending on the situation. This aspect can be obtained by calculating the histogram and equalizing it. Equalizing the histogram has the effect of improving the contrast of the image. The results of this method are highlighted in Figs. 4 and 5.

By analyzing the two histograms one can notice a decrease in the number of pixels with different intensity values and also a significant increase in the number of pixels with a high luminous intensity to the detriment of the low light pixels. The usefulness of these changes will be noticed in the growth of the enlightened area, even in the absence of ambient lighting.

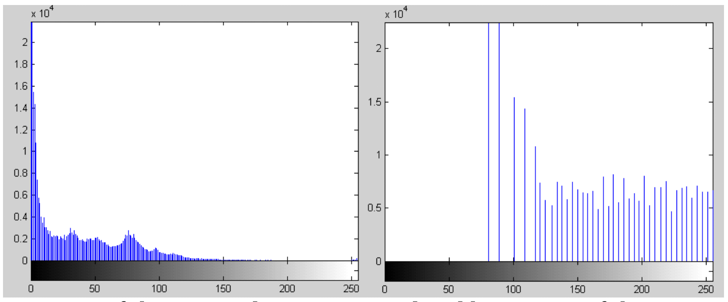

Fig. 4 – Histogram of the grayscale image, Equalized histogram of the grayscale image.

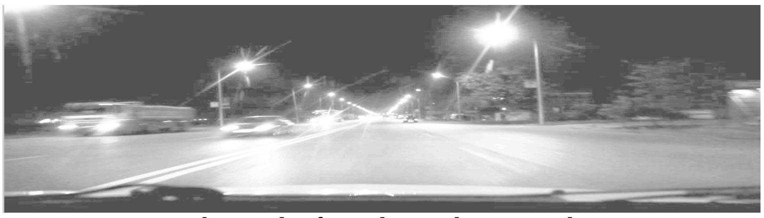
Fig. 5 – The result of equalizing the grayscale image.

To demonstrate effectiveness in selecting the area of interest by analyzing the histogram, the comparison between the variant where the method is absent and the variant in which it is present is displayed. The case considered is one in which ambient lighting is absent, and the images will highlight the way the luminance-based area of interest is selected. This comparison is displayed in Fig. 6.

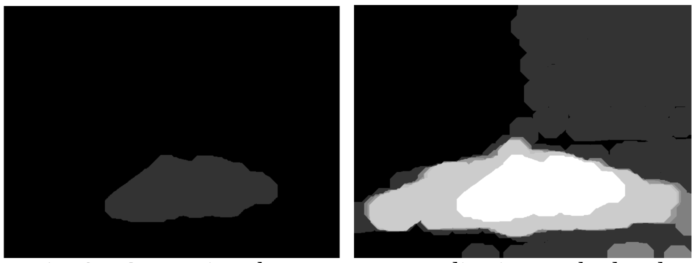
Fig. 6 – Comparison between non-equalization method and histogram equalization method.

• Create the mask to select the area of interest in the image: The area of interest is represented by the background obtained by applying the morphological opening operation with a structural element in the form of a disc with a surface of 30 pixels. In particular, the background is first subject to erosion, then to dilation, leaving out areas too small to contain the structural element. The result of applying these operations to a frame containing ambient lighting is shown in Fig. 7.

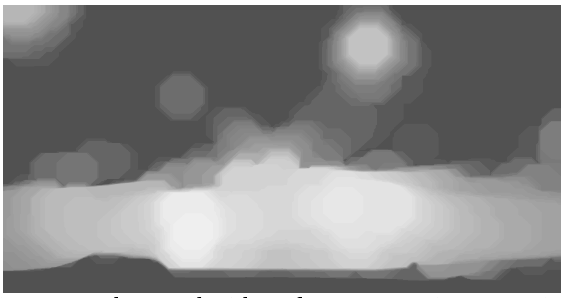

Fig. 7 – Background with undergone opening operation.

From this point on the actual mask selection is made, by selecting pixels with a value greater than a certain threshold for intensity and dilate the area thus obtained. Therefore, a shape close to that of the road is obtained, but at the same time the bright objects of very large dimensions remain inside the image, namely those represented by the street lighting pillars, as can be seen in Fig. 8.

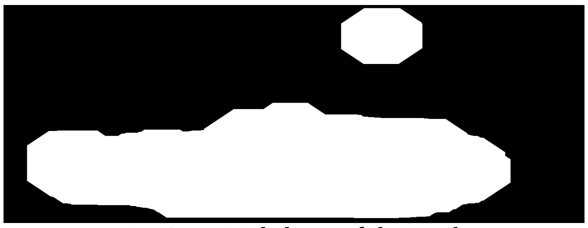

Fig. 8 – Initial shape of the mask.

To bring the mask to the desired shape, the area of the white objects in the image is calculated and only the one with the largest area, which obviously represents the area of the road, is preserved. Completing the mask creation consists in cutting the Y-axis of an unnecessary portion, such as the bottom, which contains the hood of the machine. Finally, the image with the mask applied will be shown in Fig. 9.

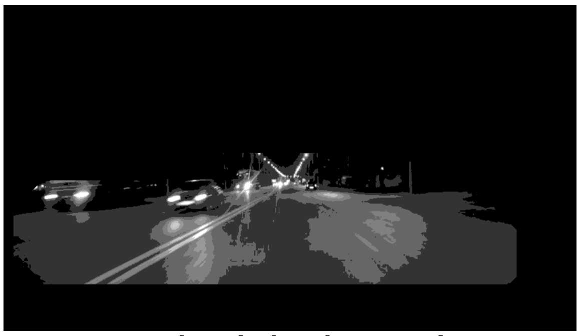

Fig. 9 – Mask applied on the grayscale image.

• Convert to binary image: In order to be able to separate the headlamps, the smallest objects must be determined. Conversion replaces all pixels in the base image that have a luminance greater than a set threshold with a value of 1 (white), and the other pixels are replaced with a value of 0 (black). Determination of the luminance threshold is obtained from the normalized value of the pixel intensities, plus a value for the threshold that represents between 85 and 100% of the maximum intensity. The outcome can be seen in Fig. 10.

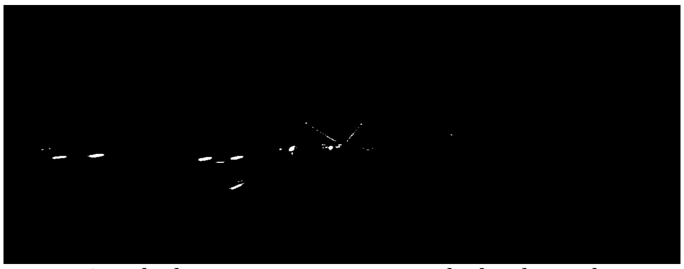
Fig. 10 – The binary image containing the brightest objects.

• Removing too small light objects and those that do not meet a certain roundness threshold: Removing too small objects is done by determining the area of each object, and those whose size does not exceed 7 pixels have their value changed from 1 to 0, making them black. The effect is to remove objects such as ambient light, light reflections on road markings, license plates, etc. outside a predefined threshold. The drawback of this method is the loss of the objects that can represent the tail lights of the cars moving ahead of the reference vehicle, in the same direction. Due to the conversion to greyscale intensity, the round red tail lights will be turned into less bright objects, meaning that the area of the light blob will make it prone to exclusion. Still, given the more and more complex shapes of the rear lights that are currently tolerated by the law, a significantly different approach is needed to identify those kind of light objects.

• Delimitation of a horizontal zone for the determination of the headlamp pair: This part of the algorithm will be based on three important aspects of image processing: the area of objects, their centre (centroid), and the bounding box.

The first step at this stage is to determine the pixel vector on the Y axis for each object, starting from the pixel whose Y coordinate is the smallest to the maximum height of the object. The next step is to check the intersection between pixel vector elements on the Y axis of each light object. In this way, n-tuples of light objects in the same "horizontal" strip are determined, limiting the possibilities of pairing certain objects. However, the problems that may arise at this time are the pairing of light objects belonging to different cars. An example is shown in Fig. 11.

The legend: Yellow line: the horizontal area of the object 1, Blue line: Horizontal area of object 2. It can be seen that on the basis of the horizontal strips generated by the pixel vector on the Y axis, the object 1 can be placed in the same pair with the object 3 and the object 2 can be put in pair with each of the objects 1,3,4, what would be wrong. To eliminate these errors, the distance between the light objects and their area is taken into account. Thus, the larger the area of the objects, the closer they are, which means that the distance between their centers can be greater. If the objects are in the distant plane, then the distance between their centers must be smaller. Still, this method imposes a few limitations, since the closer two cars are to each other (*i.e.* the case of overtaking) the higher the risk that an incorrect pair of headlights is not established. Therefore, a few blobs might be incorrectly unmarked. This is though not dangerous, since if at least a vehicle is considered as detected in the area where those two cars reside, then the headlights will be adapted to low beam for that area, not glaring any of those drivers.

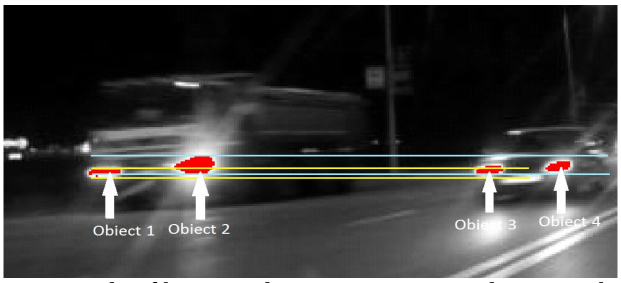

Fig. 11 – Example of horizontal "stripes" corresponding to each object.

• Delimitation of a vertical zone for the determination of headlamp mirrors on the road: This stage is similar to the previous one, distinguishing itself through the axis on which the pairing area is made. In this situation, it is being sought to find objects "one above the other". More precisely, determine the coordinate vector on the X axis of each object left in the image after the filtrations made in the previous step. The next step is to verify the existence of an intersection between the X coordinate of the center (centroid) of each object and the coordinate vector previously determined for all the light objects present in the image. Thus, n-tuples of objects in the same vertical "strip" can be determined, as shown in Fig. 12.

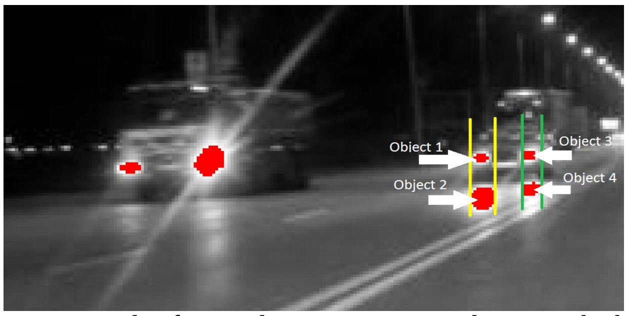

Fig. 12 – Example of vertical "strips" corresponding to each object.

The legend: Yellow line: the horizontal area of the object 2, Blue line: the horizontal area of the object 4. It can be seen that objects 1 and 2 are in the same vertical strip, similar to objects 3 and 4. Objects 2 and 4 respectively represent mirrors of objects 1 and 3. For this reason, they will be removed based on the Y coordinate of their center. Consider the fact that the illuminated object of the headlamp reflection is lower than the light itself. There may also be cases when the headlamps of another car can overlap in the vertical areas caused by the headlights of a car. In this case, it should be avoided to consider the front lights of the car as mirrors of the rear headlamps. This takes into account the area of light objects and the distance between their centers. The reflections caused by the headlights of a nearby car will be larger and farther away from the center of the headlamp, while the reflections caused by the headlights of a car in the far plan will be smaller and closer to the center of the headlamp. This step imposes the same risks as the previous one, raising the possibility of not detecting the correct number of vehicles, but ensuring the safety functionality of reducing the beam for the correct area of the road.

• Mark the light objects on the original image: In order to perform a video verification that can be used to determine the precision and accuracy of the algorithm, it is necessary to indicate on the grayscale initial image the objects detected by the processing method. Marking consists in setting the intensity of the pixel values in the image being processed. Thus, the pixels will have the maximum possible value of the intensity on the red channel (R) - 255, and the minimum value for the other two channels, green (G) and blue (B) - 0. After that, the processed image is superimposed on the original image, the result being as shown in the Fig. 13.

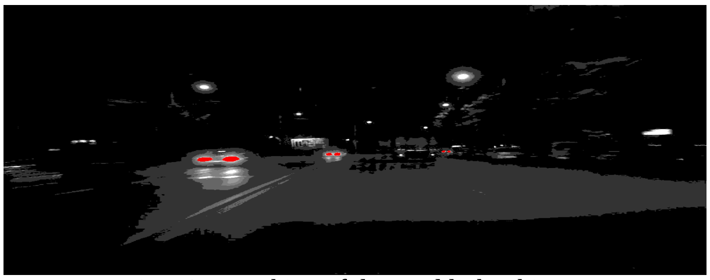
Fig. 13 – Marking of detected light objects.

• Map the area of interest with a matrix of rectangles representing the light area produced by each of the LEDs of each beacon: the LED control signal must contain information about the frame number, the number of the object detected in the image, and the LED that is assigned to it. To assign a led to one or more light-objects, an array of rectangles whose surface is the area illuminated by a led is superimposed over the processed image. Thus, the detected light objects will be located on the surface of a rectangle and, implicitly, will influence the intensity of the LED corresponding to that rectangle.

5. Testing the Application

There are two situations where the application can be tested:

• in a dynamic case in the car, acquiring real-time frames in traffic;

• in a static, off-vehicle scenario, using pre-recorded frames saved on the hard drive on which the application will run.

In both situations, it is necessary to connect the development environment from a PC to the development board in order to run the application and initiate the camera. The connection between the PC and the development board is wireless, the two components communicating through development environment.

• Dynamic simulation: The dynamical simulation imposes the use of a prototype setup inside the car. The video camera was placed behind the inside rearview mirror, facing the center of the lane on which the car is moving. The development board was powered by a 5 V battery connected to the car's 12 V outlet. With the development board powered on, the application can be run from the development environment. This will make the SSH connection between the PC and the card, start the camera and set its parameters, and then start the purchase. The duration of the acquisition will be controlled, following a finite processing time given by the number of snapshots the camera has been set up to take. During the acquisition, the result of the processing will be displayed inside the development environment.

• Static simulation: Taking into account that no capture of images is needed, since these are already acquired, the main role of the simulation is to visualize the efficiency of the algorithm as easily as possible. All frames stored and processed will be retrieved, the result being a video stream on which the detected light objects are marked. The only necessary element for a correct verification of the algorithm's operation is the set of data the acquisition of which must be done under conditions similar to those presented in the dynamic simulation method.

The main tests that can be performed to verify the effectiveness of the application are those that determine accuracy and precision, and those that calculate time efficiency. For the calculation of accuracy and precision, two situations may be considered, with street lighting present or absent.

For each data set, all light objects that are categorized as true positive, true negative, false positive, false negative are determined.

The classification of the lights objects is listed below:

• True Positive = Objects that are headlights in reality and have been marked;

• Positive False = Objects that are not headlights in reality but have been marked;

• True Negative = Objects that are not actually headlights and have not been marked;

• Negative False = Objects that are headlights in reality and have not been marked.

In order to determine the accuracy of the algorithm, the true positive and true negative elements have been summed up and divided by the total number of light objects detected in a frame. In what regards the precision of the algorithm, the number of true positive objects is divided by the sum of true and false positive objects.

The results of the simulations for video sequences composed of a number of frames ranging from 150 to 300 frames, are as follows:

– Absence of street lighting:

• falsely unmarked objects (false negative): 2% of the total number of objects;

• accuracy: 89%;

• precision: 69%.

– Presence of street lighting:

• falsely unmarked objects (false negative): 4% of the total number of objects;

• accuracy: 82%;

• precision: 42%.

These results are independent of the simulation method used, since they rely on the flow of frames that are fed to the algorithm.

The processing time and processing capabilities are heavily dependent on the HW platform used. The results are purely for statistics, since the prototype setup is using several components that add major delays (SSH connection, Matlab environment running on Windows PC, Frame acquisition tool running on Raspbian etc.) that will be removed when running on dedicated hardware.

6. Conclusions. Improving Directions and Further Developments

Safety in traffic during night time is substantially diminished when compared to daytime due to the reduced visibility and lighting. The most widespread headlight system in today’s cars relies on switching manually between the high and low beams. Usually, the human factor introduces a delay when switching between the two beams, which can result in disturbances for the other traffic participants by blinding them temporarily and, therefore, increasing the risk of accidents. Currently, the emphasis is set on creating an automatic control system for the headlights, eliminating the need for the driver to manually adjust the beam levels. The light moving objects detection system is composed of a video camera placed inside the vehicle, connected to microcontroller running the frame-by-frame processing algorithm under a Linux based operating system. This device should be considered as a safety one. Our proposal provides a solution to a practical challenge of adapting the headlights to incoming traffic.

Compared to the public-shared documented approaches, the major advantage of the hereby presented solution is the high accuracy achieved at a reduced cost and moderate complexity. One of the main differences when compared to most of the other solutions is the use of a basic camera for video acquisition. It can be acknowledged that dedicated hardware was used for the frame capture, where a modified light sensor was used to enhance the detection of red and white objects (Antonio LópezJörg *et al*., 2008), or where a black and white camera was used to reduce the volume of information recorded (P.F. Alcantarilla *et al*., 2011). Similarities can be observed throughout the available documentation in what regards the image format used. Most of the approaches choose an intensity based conversion, although conversions to HSV (O'Malley *et al*., 2008) or RGC (Hyun-Koo KIM *et al*., 2011) can also be found, approaches which might impose drawbacks due to the high volume of information. A high volume of information that comes in for processing can be seen again in (Antonio LópezJörg *et al*., 2008), where the area containing the street lighting poles is scanned, only to switch to low beam when driving under ambient lighting. Therefore the current presented solutions capture the frames with minimal interaction, performing all the processing via a software that returns a high accuracy.

Moreover, based on available references, we demonstrate that the paper's research subject is an actual one.

Further development: the presented algorithm was developed to detect the lighting moving objects – cars, which are coming from the opposite side, straight ahead, in line. A few of the further improvement areas concern the special positioning cases of incoming vehicles, cases like cars approaching from uphill/downhill or cars entering the visual range after a road turn. Also, difficult weather conditions are to be taken into consideration. Rain and fog can trigger massive glare and widespread light objects. Correctly detecting incoming headlights under these conditions will depend on the light sensor mounted inside the windshield, which will most likely force the switch to low beam independent of incoming traffic, according to best practices when driving.

REFERENCES

Alcantarilla P.F., Bergasa L. M., Jiménez P., Parra I., Llorca D.F., Sotelo M.A., Mayoral S.S., *Automatic LightBeam Controller for Driver Assistance*, 2011.

Antonio López, Jörg Hilgenstock, Andreas Busse, Ramón Baldrich, Felipe Lumbreras, Joan Serrat, *Nighttime Vehicle Detection for Intelligent Headlight Control*, International Conference on Advanced Concepts for Intelligent Vision Systems ACIVS 2008: Advanced Concepts for Intelligent Vision Systems, 113-124.

Chen Wei-Gang, Xu Bin, *Detecting Moving Shadows in Video Sequences Using Region Level Evaluation for Vision-Based Vehicle Detection*, Fifth International Conference on Frontier of Computer Science and Technology, 2010, 142-146, DOI: 10.1109/FCST.2010.89.

Gabriel Zahi, Shigang Yue, *Reducing Motion Blurring Associated with Temporal Summation in Low Light Scenes for Image Quality Enhancement*, International Conference on Multisensor Fusion and Information Integration for Intelligent Systems (MFI), 2014, 1-5, DOI: 10.1109/MFI.2014.6997725.

Hyun-Koo KIM, Kuk Sagong, Ju H. Park, Fabien Moutarde, Ho-Youl Jung, *A Color Based Real-time Vehicle Detection for Adaptive Headlamps Control*, 2011.

Jinchang Ren, Astheimer P., Feng D.D., *Real-Time Moving Object Detection under Complex Background*, 3rd International Symposium on Image and Signal Processing and Analysis, 2003. ISPA 2003, Volume **2**, 662-667, DOI: 10.1109/ISPA.2003.1296359.

Moein Shakeri, Hong Zhang, *Moving Object Detection in Time-Lapse or Motion Trigger Image Sequences Using Low-Rank and Invariant Sparse Decomposition*, IEEE International Conference on Computer Vision (ICCV), 2017, 5133-5141, DOI: 10.1109/ICCV.2017.548, ISSN: 2380-7504.

Ninomiya Y., Matsuda S., Ohta M., Harata Y., Suzuki T., *A Real-Time Vision for Intelligent Vehicles*, Intelligent Vehicles '95 Symposium., 315-320, DOI: 10.1109/IVS.1995.528300.

O'Malley R., Glavin M., Jones E., *Vehicle Detection at Night Based on Tail-Light Detection*, 1st International Symposium on Vehicular Computing Systems, Trinity College Dublin, 2008.

Weijie Liu, Kensuke Maruya, *Detection and Recognition of Traffic Signs in Adverse Conditions*, IEEE Intelligent Vehicles Symposium, 2009, 335-340, DOI: 10.1109/IVS.2009.5164300.

Xin Li, Zenggang Zhou, Xiaoyuan Li, Yan Wan, *Vehicle Segmentation and Speed Detection Based on Binocular Stereo Vision*, Eighth International Conference on Computational Intelligence and Security, 2012, 369-373, DOI: 10.1109/CIS.2012.89.

Yen-Lin Chen, Bing-Fei Wu, Chung-Jui Fan, *Real-Time Vision-Based Multiple Vehicle Detection and Tracking for Nighttime Traffic Surveillance*, IEEE International Conference on Systems, Man and Cybernetics, 2009, 3352 - 3358, DOI: 10.1109/ICSMC.2009.5346191.

Yen-Lin Chen, Bing-Fei Wu, Hao-Yu Huang, Chung-Jui Fan, *A Real-Time Vision System for Nighttime Vehicle Detection and Traffic Surveillance*, IEEE Transactions on Industrial Electronics, **58**, *5*, 2030-2044 (2011).

Yu-Chun Lin, Yi-Ming Chan, Luo-Chieh Chuang, Li-Chen Fu, Shih-Shinh Huang, Pei-Yung Hsiao, Min-Fang Luo, *Near-Infrared Based Nighttime Pedestrian Detection by Combining Multiple Features*, 14th International IEEE Conference on Intelligent Transportation Systems (ITSC), 2011, 1549-1554, DOI: 10.1109/ITSC.2011.6083015.

ALGORITM PENTRU DETECTAREA OBIECTELOR LUMINOASE AFLATE ÎN MIŞCARE PE TIMP DE NOAPTE

(Rezumat)

Siguranța în trafic pe timp de noapte este afectată de reducerea substanțială a vizibilității în comparație cu cea din timpul zilei. Sistemele de iluminat existente în prezent la maşinile produse în serie au la bază un mecanism manual de schimbare a intensității luminoase a farurilor între faza lungă şi faza scurtă. De cele mai multe ori inexactitatea umană ce survine în comutarea între cele două faze are ca efect deranjarea participanților la trafic, apărând situațiile în care şoferii sunt orbiți de luminile altor vehicule, crescând astfel riscurile de accidente. În prezent se urmăreşte eliminarea necesității ca şoferul să intervină în controlul farurilor prin automatizarea acestora.

In lucrarea de față se prezintă metoda de realizare a prototipului unui sistem de detecție a obiectelor luminoase aflate în mişcare în trafic pe timp de noapte. Sistemul este prevăzut ca un subansamblu al unui proiect complex ce are ca scop controlul farurilor cu led-uri ale unui autovehicul pe baza analizei unui flux de imagini provenit de la camera video instalată pe autovehicul.

Sistemul de detecție a obiectelor luminoase este compus dintr-o cameră video amplasată în interiorul maşinii, conectată la o placă de dezvoltare cu un sistem de operare de tip Linux pe care rulează algoritmul de prelucrare a fluxului video, cadru cu cadru. Algoritmul constă în tehnici de conversie, segmentare, grupare şi filtrare a imaginilor, validare a obiectelor detectate şi o variantă de transmitere a unui semnal cu informații de control ale led-urilor farurilor. Simularea şi rularea aplicației se realizează printr-o conexiune Wi-Fi cu PC-ul. Testarea sistemului a fost realizată independent de ansamblul ce controlează farurile, optându-se pentru o verificare în timp real pe timp de noapte, în trafic.