Archives of
Transport System
Telematics

Volume 9

Issue 1

February 2016

Detection of vehicles moving in wrong direction

M. BAZAN[a], P. CISKOWSKI[a], K. HALAWA[a], T. JANICZEK[b], A. RUSIECKI[a], M. ŚMIGOWSKI[a]

[a] WROCŁAW UNIVERSITY OF TECHNOLOGY, Faculty of Electronics, Department of Computer Engineering, ul. Janiczewskiego 11/17, 50-372 Wrocław, Poland
[b] WROCŁAW UNIVERSITY OF TECHNOLOGY, Faculty of Electronics, Department of Control Systems and Mechatronics, ul. Janiczewskiego 11/17, 50-372 Wrocław, Poland
EMAIL: marek.bazan@pwr.edu.pl

ABSTRACT

In this paper we describe a method for detecting situations where a vehicle moves along a highway in the wrong direction. The first step of the algorithm is to build a pattern using Gaussian mixture model based on the optical flow calculated with the Lucas-Kanade method. The second stage concerns the detection of objects as a potential road hazard. The optical flow calculated on-line during the second stage is compared with the traffic pattern used in the first stage. Then the difference in movement direction is detected using predefined thresholds.

KEYWORDS: optical flow, wrong direction driving incidents detection, Lucas-Kanade method

1. Introduction

Detection of dangerous situations on roads is an extremely difficult task. Variety of risky events does not allow to develop a universal detection method. Situation complexity requires the use of many detection techniques at once. Moreover, the unique character of the place being observed does also influence the choice of the technique. Nevertheless, the development of technology help us dominate and affect the surrounding reality. With the aim of increasing the safety on roads, it is important to use modern tools and technology to classify the behavior and interaction between drivers. Very important aspect in maintaining security traffic is to prevent dangerous situations. For this purpose, a variety of methods is used, such as: big warning signs, microwave detection, termovision detection, etc. Until recently, the detection of dangerous incidents has also been conducted by human operators, who interpreted the situation by looking at camera images in traffic control centers. However the increasing number of cameras imposed the use of automated systems to notify about danger.

One of the currents trends in intelligent transportation systems (ITS) is the development of monitoring systems, in order to reduce the number of accidents and traffic jams, both in cities and on highways. Video systems provide huge amount of data about the situation on the roads. Our work aims at automating the process of detecting strange, unusual behavior on highways. By unusual behavior we mean sudden lane change or driving the wrong way. The latter activity is the cause of the most dangerous situations encountered on the roads. Therefore instant detection is crucial for preventing the death of many people. Data collected by The National Transportation Safety Board (NTBS) [1] indicate that the majority of drivers traveling in the wrong direction was under the influence of alcohol. The incidents happened mostly at night, between 12p.m. and 5a.m. Another large group of people going the wrong way were people above 70 years of age. Unfortunately, preventive measures for these two groups of drivers are often insufficient.

The solution presented in this paper is divided into two phases. During the first stage, we try to obtain the direction model for the specific camera view, using the Gaussian Mixture Model (GMM). The next stage is the detection – the optical flow is calculated for the situation on-line, and compared with the standard flow computed previously. Difference between directions is analyzed and if it reaches a certain threshold, the alarm is triggered.

2. Detection algorithm

The application was implemented in MATLAB 2014a environment. The program consists of several algorithms, described later. The training phase diagram is presented in picture 1.

At the beginning the video stream is read, using a special function from MATLAB libraries. In the next step the video frames are prepared. Functions create a table of pictures. After that picture processing may be performed, like size change in order to reduce the processing time. The preprocessed pictures are sent to the optical flow algorithm.

Optical flow describes movement between two frames of a video sequence. Each pixel in the image is assigned a velocity vector, that represents the direction and speed in which pixel is moving. Assigning speed to each pixel requires a lot of calculation and is time consuming. Such a structure is often called a dense optical flow. In this case the calculation is a demanding task. Imagine a movement of an object consisting of many white pixels. After moving to the next part of frame, the pixels will remain white. A change will occur only on the object edges. Therefore the method requires interpolation between points that are in the same area of brightness or color.

Problems with calculation of velocity vector for each pixel, motivated scientists to design an alternative sparse optical flow. Algorithms of this type are based on tracking groups of pixels. The best areas for monitoring are specific places, such as corners or edges of a moving object. In this work, the Lucas-Kanade method was used to calculate the optical flow. The algorithm is presented further on.

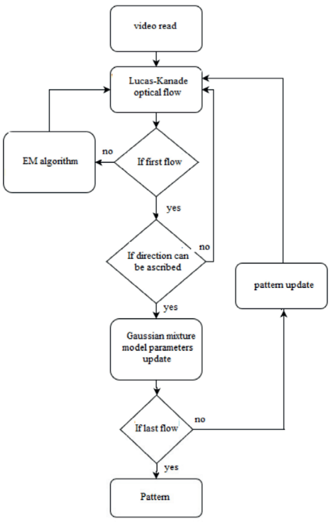

Fig. 1. Obtaining a pattern [own study]

2.1. Lucas-Kanade method

The Lucas-Kanade algorithm was presented in 1981 as an attempt to calculate the dense optical flow [2]. Nowadays the method is used to calculate the optical flow for groups of pixels. It is based on the information from small windows adjacent to the considered pixel block. The window is a field in which the algorithm seeks for pixels of the same parameters. The disadvantage of this approach is that if too frequent small windows are used, they cannot detect large displacements between images. Basic assumptions necessary for proper mathematical description of the algorithm are presented in [3-7]

- Constant pixel brightness. Pixel that belongs to a moving object does not change its brightness despite locations change.
- Small movements between consecutive video frames. Objects in the image move slowly over time. In practice, this means that movement should be confined to the window size.
- Spatial cohesion. The adjacent areas in the picture belong to the same surface and they move similarly.

These assumptions may be expressed in mathematical form. The first one is related to the constant pixel brightness over time, and it is shown in formulas (1) and (2).

$$f(x,t) \equiv I(x(t),t) = I(x(t+dt), t+dt) \quad (1)$$

$$\frac{\partial f(x)}{\partial t} = 0 \quad (2)$$

The second assumption means that the motion between two successive frames is small. To understand the consequences, let us consider a one-directional movement. Beginning with equation (3), which illustrates brightness stability for pixel over time, the chain rule must be used to solve the derivative:

$$\underbrace{\left.\frac{\partial I}{\partial x}\right|_t}_{I_x} \underbrace{\left(\frac{\partial x}{\partial t}\right)}_{v} + \underbrace{\left.\frac{\partial I}{\partial t}\right|_x}_{I_t} (t) = 0 \quad (3)$$

where I_x is the derivative over displacement of a pixel, I_t is the derivative over time which elapsed between two pictures, and v is the velocity we search. Finally a simple equation is obtained for the velocity in one-dimension:

$$v = -\frac{I_t}{I_x} \quad (4)$$

2.2. Gaussian mixture model

Let us recall the Gaussian distribution and how we define a mixture of Gaussian functions [8]. The Gaussian probability distribution function of a random variable X may be expressed as

$$p_X(x) = \mathcal{N}_X(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad (5)$$

where two parameters μ, and σ^2 are, the average of the measurement and variance (σ is standard deviation), respectively. The Gaussian mixture model function is a parametric function of the probability distribution, represented as a weighted sum of normal distributions

$$p(X|\lambda) = \sum_{i=1}^{M} w_i \mathcal{N}(x|\mu_i, \sigma_i) \quad (6)$$

where in our case x is one-dimensional data vector, w_i, i = 1,...,M are weights of component mixtures, $N(x|\mu_i,\sigma_i)$ are Gaussian probability distributions.

To facilitate the process of obtaining a pattern, it is assumed that the direction distribution obtained from optical flow is Gaussian, and the Gaussian mixture function consists of two components. These assumptions are correct, as images form fixed cameras on highways show usually two lanes. In the case of one lane, the parameters of mixture Gaussian function take the same values, what has no effect on direction verification. The purpose of the Gaussian mixture function used in the presented work is to obtain parameters describing the mixture. Using them, we can verify whether the direction is reliable and we can update the model [8]. Update direction is performed according to the following procedure:

1. Algorithm checks if the value of velocity obtained from optical flow is significantly greater than the threshold 0,5. Values below the threshold may be considered as noise or errors.
2. Direction is calculated for each velocity vector, and if the value is between ±2.57 standard deviation parameters and mean are updated, according to the following formulas:

$$\mu_i^t = (1-\alpha_i^t)\mu_i^{t-1} + \alpha_i^t\theta^t \tag{7}$$

$$\sigma_i^t = (1-\alpha_i^t)\sigma_i^{t-1} + \alpha_i^t(\theta^t - \mu_i^t)^2 \tag{8}$$

where value α is calculated according to (9)

$$\alpha_i^t = \tau\mathcal{N}(\theta^t, \mu_i^{t-1}, \sigma_i^{t-1}) \tag{9}$$

τ is learning coefficient, its value was determined empirically. Best results were obtained for τ = 0.005.

3. The next step is to update the direction for the block in pattern. It involves arithmetic mean calculations

$$W^t(i,j) = \frac{W^{t-1}(i,j) + O^t(i,j)}{2} \tag{10}$$

If any direction did not occur in a pattern, we assume the value given by the optical flow

$$W^t(i,j) = O^t(i,j) \tag{11}$$

2.3. The expectation-maximization method

The EM algorithm is most commonly used as a method of solving problems with missing data. In this work it was used for obtaining information about direction distribution parameters, what is exactly a problem of missing data. Obtaining parameters for Gaussian mixture function helped with collecting the data. According to the definition given in [9], EM is a way of calculating the maximum likelihood estimators.

To recall the maximum likelihood estimator, let us consider a function:

$$f(\theta; x_1, \ldots, x_n) = \begin{cases} P_\theta(X_1 = x_1, \ldots, X_n = x_n), & \text{dla rozkładów dyskretnych} \\ f_\theta(x_1, \ldots, x_n), & \text{dla rozkładów ciągłych} \end{cases} \tag{12}$$

where the parameter θ is unknown. The likelihood function L: Θ → R is given by

$$L(\theta) = f(\theta; x_1, \ldots, x_n), \tag{13}$$

and is a function of parameter θ with fixed values $x_1, ..., x_n$. θ is the maximum likelihood estimator of parameter θ, if

EM method is an iterative method. The following two procedures are performed in each iteration: E expectation step and M maximization step. For this reason it is called EM algorithm.

Consider a statistical model, with measured values X = (x_1, . . . , x_n) of Gaussian distribution and latent data Z = (z1, . . . , zn), which determine from which component a given value is derived. Next we stick to assumption that composition of Gaussian mixture function consists of two components:

$$\begin{aligned} X_i|(Z_i = 1) &\sim \mathcal{N}(\mu_1, \sigma_1) \\ X_i|(Z_i = 2) &\sim \mathcal{N}(\mu_2, \sigma_2) \end{aligned} \tag{14}$$

where

$$P(Z_i = 1) = \tau_1 \quad \text{i} \quad P(Z_i = 2) = \tau_2 = 1 - \tau_1 \tag{15}$$

Our aim is to estimate the unknown parameters representing components in Gaussian function - the mean and the standard deviation

$$\theta = (\tau, \mu_1, \mu_2, \sigma_1, \sigma_2) \tag{16}$$

The maximum likelihood function for incomplete data:

$$L(\theta; x) = \prod_{i=1}^{n} \sum_{j=1}^{2} \tau_j f(x_i; \mu_j, \sigma_j) \tag{17}$$

and for the complete data:

$$L(\theta; x, z) = p(x, z|\theta) = \prod_{i=1}^{n} \sum_{j=1}^{2} \mathbb{I}(z_i = j) f(x_i; \mu_j, \sigma_j)\tau_j \tag{18}$$

where $\mathbb{I}$ is characteristic function, and function f is a probability density function.

1. The expectation step (E step). In this step a function Q is created:

$$\begin{aligned} Q(\theta|\theta^t) &= E[\log L(\theta; x, Z)] \\ &= E[\log \prod_{i=1}^{n} L(\theta; x_i, z_i)] \\ &= E\left[\sum_{i=1}^{n} \log L(\theta; x_i, z)i\right] \\ &= \sum_{i=1}^{n} E[\log L(\theta; x_i, z_i)] \\ &= \sum_{i=1}^{n}\sum_{j=1}^{2} T_{j,i}^t \left[\log \tau_j - \frac{1}{2}\log \sigma_j - \frac{1}{2}(x_i - \mu_j)^2\sigma_j^{-1} - \frac{d}{2}\log(2\pi)\right] \end{aligned} \tag{19}$$

where $T^t_{j,i}$ is equal to:

$$T_{j,i}^t = P(Z_i = j|X_i = x_i; \theta^t) = \frac{\tau_j^t f(x_i; \mu_j^t, \sigma_j^t)}{\tau_1^t f(x_i; \mu_1^t, \sigma_1^t) + \tau_2^t f(x_i; \mu_2^t, \sigma_2^t)} \tag{20}$$

2. The maximization step (M step). Parameters τ, (m_1,s_1) and (m_2,s_2) may be maximized independently. At the beginning we are looking for maximum τ whereas $\tau_1 + \tau_2 = 1$

$$\tau^{t+1} = \arg\max_{\tau} Q(\theta|\theta^t) = \arg\max_{\tau} \left\{ \left[\sum_{i=1}^{n} T_{1,i}^{t}\right] \log \tau_1 + \left[\sum_{i=1}^{t} T_{2,i}^{t}\right] \log \tau_2 \right\} \quad (21)$$

finally:

$$\tau_j^{t+1} = \frac{1}{n}\sum_{i=1}^{n} T_{j,i}^{t} \quad (22)$$

for other parameters (m_1,s_1)

$$(\mu_1^{t+1}, \sigma_1^{t+1}) = \arg\max_{\mu_1,\sigma_1} Q(\theta|\theta^t) = \arg\max_{\mu_1,\sigma_1} \sum_{i=1}^{n} T_{1,i}^{t}\{-\frac{1}{2}\log\sigma_1 - \frac{1}{2}(x_i-\mu_1)^2\sigma^{-1}\}$$

$$\mu_1^{t+1} = \frac{\sum_{i=1}^{n} T_{1,i}^{t} x_i}{\sum_{i=1}^{n} T_{1,i}^{t}} \quad (23)$$

$$\sigma_1^{t+1} = \frac{\sum_{i=1}^{n} T_{1,i}^{t}(x_i-\mu_1^{t+1})^2}{\sum_{i=1}^{n} T_{1,i}^{t}}$$

In subsequent iterations, we get a more accurate estimate parameters by repeating E and M steps, bringing us closer to the real distribution. Iteration steps for two-dimensional data are shown in figure 2, where areas marked darker color mean higher Gaussian mixture function.

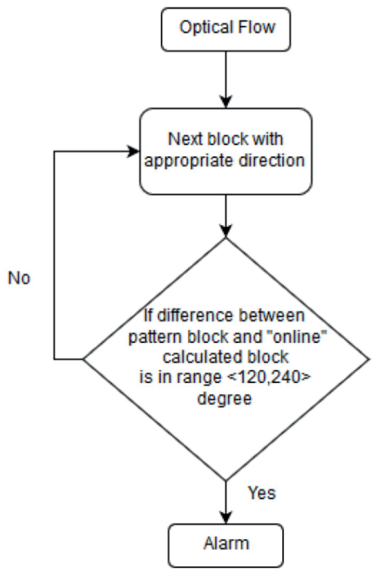

Fig. 2. Block diagram for the detection of dangerous behavior on the road [own study]

As a result of the learning process, we get a pattern. The main advantage of the above described algorithm is that it considers different directions that may occur in block. The number of frames used to calculate the pattern depends on the number of cars in observed route section. It is important to include the whole road, in order to improve the efficiency of detection. The fragment of video footage must be carefully chosen. After the above steps, we may proceed to next stage, in which the on-line picture that came from camera is analyzed. For each new frame the optical flow is calculated. Each block has its own direction. An object is considered as moving in opposite direction when the difference between the direction of the block, and the direction of the corresponding block in the pattern is larger than a certain threshold, which in this work was set to < 120, 240 >. As a result, if the difference in block is greater than 120 then the alarm is activated. It is possible that camera movements or noise may trigger an alarm, The following block diagram shows procedure which should be performed in order to detect dangerous driving.

3. Synthetic data tests

In order to perform tests on synthetic data, an animation with moving cars has been created. The Gaussian mixture model allowed to verify the correctness of moving cars direction. The Gaussian mixture function chart, with marked directions assigned to particular areas is shown in figure 3.

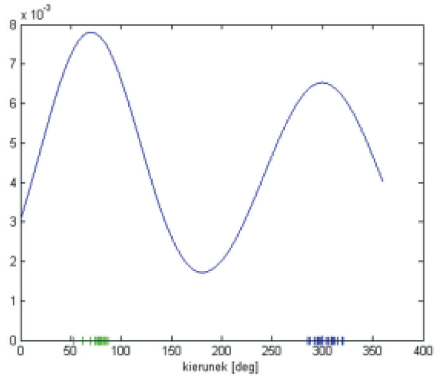

Fig. 3. Gaussian mixture function with calssified directions [own study]

We begin with checking the number of frames needed to obtain satisfactory data in the pattern. We cannot see a significant difference between number of directions, with the growing number of frames. We may notice a slight change between upper images – the pattern consisting of 32 frames is better. As mentioned before, we should choose a model with the number of frames equal to the number of small elements. Let us consider another use of the increased number of blocks and thus more accurate information about motion, remembering however that this requires larger computing power. It is possible that when the blocks are too large, some small objects are not included in the optical flow calculation. We may reduce the size of frames from 720x576 to 320x240.

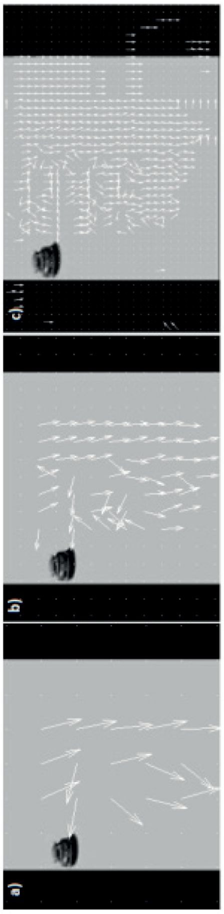

Fig. 4. Calculated patterns for a) 192, b) 768, c) 4800 blocks [own study]

There are 192 blocks in picture a) and the information is poor and worthless. For the pattern b) with 768 blocks, the number of directions is still insufficient, but the right lane is mapped. Image c) with 4800 block represents the richest information, and the right lane is mapped completely. Problems arise with the left lane, where the algorithm is not able to classify the vehicles moving from top to bottom due to the size of moving objects. In this case, we may reduce the size of search window, what will improve the ability to correctly classify shipments to the sequence of frames. Figure 5 shows a pattern with a square window consisting of 25 pixels. We can see noise on the left side, what shows that the reduction of window size does not improve the quality.

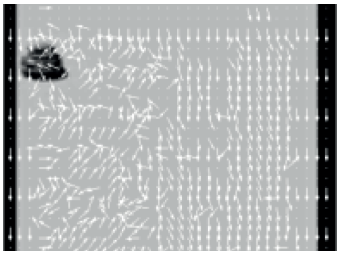

Fig. 5. Pattern with five pixel block [own study]

Alarm is triggered when three successive optical flows reveal improper direction. The car going in the wrong way was correctly classified as a potential danger object. We have also performed tests with cars moving horizontally on the screen, in this case the algorithm didn't react. The detection time of the algorithm may be easily changed by adjusting the detection threshold. In order to increase the confidence of malicious behavior detection, we should implement validation procedure.

4. Real data tests

The algorithm has been tested on the video frames of the traffic on highways filmed by static camera. The images covered various day times, such as: the night, rainy day, clear day, in order to determine whether the algorithm works properly regardless of the weather conditions. It was very difficult to obtain video from static camera, showing cars diving in wrong direction. No such videos have been found. Therefore we added a specially prepared picture with a car and created a short animation of a moving car.

As in previous section, the tests were focused on finding how many frames are sufficient for a good pattern. Figure 6 shows the calculated pattern on a highway. a) for 576 blocks, while b) 2304 blocks. In both cases, patterns were taken from the same video sequence.

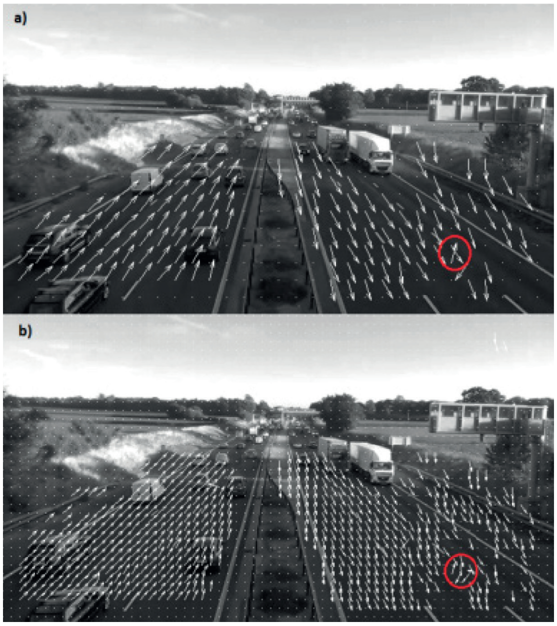

Fig. 6. Pattern on highway calculated in daylight a) 576 b) 2304 blocks [own study]

We can see that the patterns in picture b) provide more information. The places where directions are incorrect, inconsistent with our expectations, are marked with red circles. The cause of these errors is the wrong selection of a movie clip. In this case, the algorithm should use a different part of the video. There is also a possibility for manual intervention, to change the information about block directions.

Figure 7 shows a situation on a highway. The cars on the right lane are moving slowly, so their movement is not detected. Reduction of window size to 5px did not bring any result as the implemented algorithm captures moving objects that exceed a certain speed limit. It is possible to modify the amount of skipped frames, equivalent to the artificial change in objects velocity. To a large extent, this is a reason for the loss of information about objects traveling at higher speed and the noise.

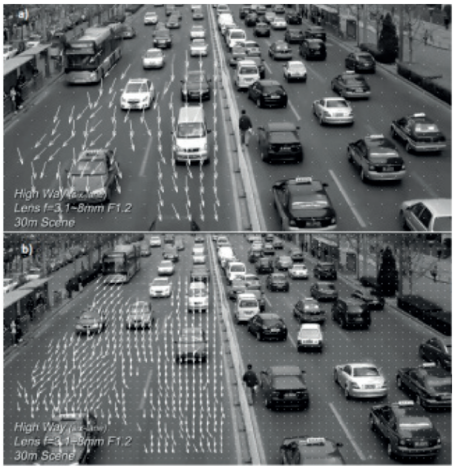

Fig. 7. Patteen calculated with high traffic density [own study]

5. Encountered problems

The aperture problem was the most significant during the implementation of the Lucas-Kanade algorithm. It emerges when our search window is too small. In that case only the edge of a moving object, not its corner , appear in the relevant window. Let us note that the edges are not enough to clearly and accurately determine object's movement. Consider the following example in figure 16. In the second row, gray square moves diagonally to the bottom right corner. This sequence shows real movement of a considered object. At the top sequence we see a movement from window perspective. It shows that gray square moves horizontally to the right, which in fact is not true.

Two solutions may be suggested for the presented problem. First is to find the characteristic points of the two-dimensional object, such as corners. We may also use the assumption posed in section 2.1. If adjacent areas of image are moving the same way, then we can determine motion of the central block using the information from the adjacent blocks. The set of equations (7) are created and then the velocity vector is calculated by the least squares method.

Another problem, already mentioned, was to obtain the real video clip with a dangerous situation (a car driving the wrong way). Several such recordings have been found, but they were captured by cameras installed in a car, or a patrol helicopter. To work the problem around, we applied an artificial animation to our video sequences.

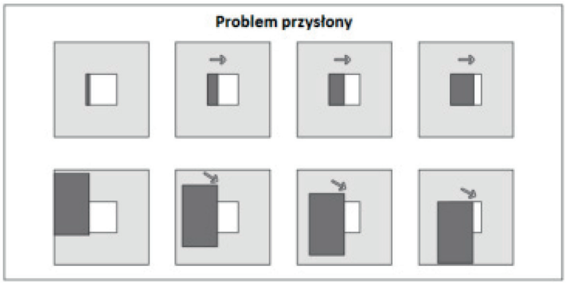

Fig. 8. Aperture problem [2]

6. Conclusion

This paper presents a technique of detecting vehicles moving in the wrong way using the calculation method called Lucas-Kanade optical flow. Dozens of experiments were conducted on six recordings, reflecting the different weather conditions. The suggested algorithm worked well and detected dangerous situations. Unfortunately, in some cases the alarm is activated despite the absence of danger on the road. A very important aspect of the algorithm is that it works regardless of weather conditions. Implementation of validation method is needed in order to improve the algorithm, that will check whether a moving object is a car.

Validation may consist of two stages. The first stage is temporary validation if wrong way alarm occurs in three consecutive frames, and if that happens the trajectory of a car will be created. Second order Kalman filter can be used to track and predict the position of the vehicle on subsequent frames. Frames with moving object detected are stored in memory for the second phase of validation. The main objective of this part of validation is to find a small set of Haar features. Object detection is performed by moving the search window along the whole image obtained from previous validation. It will check whether a region can be classified as a car.

As mentioned at the beginning, the algorithm suggested in this work may be used by institutions responsible for maintaining order on roads. It does not require much effort, money and is simple to use. When an alarm appears on the screen, appropriate person should check whether or not it is a mistake, and then take remedial actions.

Acknowledgments

This work was funded from the grant no S50242 at Wrocław University of Technology.

Bibliography

[1] SAETERN T. L.: Wrong-Way Driving Prevention Methods, Sacramento: Caltrans, 2015.
[2] BRADSKI G., KAEHLER A.: Learning OpenCV, p. 322-337, Sebastopol: O'Reilly, 2008.
[3] BARRON J.L., FLEET D.J., BEAUCHEMIN S.S.: Performance of optical flow techniques, International Journal of Computer Vision, Volume 12, p. 43-77, 1994).

[4] LUCAS B., KANADE T.,: An iterative image registeration technique with an application to stereo vision, DARPA Image Understanding Workshop, p. 121-130, 1981).
[5] FORESTI G.: Obejct detection and tracking in time-varying and badly illuminated outdoor environments, SPIE Journal on Optical Engineering, Volume 37, p. 2550-2564, 1998.
[6] PAŁCZYNSKI K.: Segmentacja na podstawie analizy pola ruchu sekwencji obrazówcyfrowych, Praca Dyplomowa, Kraków: Akademia Górniczo-Hutnicza im. Stanisława Staszica, 2002.
[7] FLEET D.J., WEISS T.: Optical flow estimation, Mathematical models for Computer Vision: The Handbook, s. 239-257, 2005.
[8] MONTEIRO G., et al.: Wrongway drivers detection based on optical flow, Image Processing, San Antonio, Volume V, s. 141-144, 2007.
[9] HUPTAS R.: Zastosowanie algorytmu EM do estymacji parametrów rozkładu na podstawie danych pogrupowanych, Zeszyty Naukowe Akademii Ekonomicznej w Krakowie 2007 .