Transport and Telecommunication, 2016, volume 17, no. 4, 289–297
Transport and Telecommunication Institute, Lomonosova 1, Riga, LV-1019, Latvia
DOI 10.1515/ttj-2016-0025

"MASS CENTRE" VECTORIZATION ALGORITHM FOR VEHICLE'S COUNTING PORTABLE VIDEO SYSTEM

Vladislav Gaidash[1], Alexander Grakovski[2]

[1,2] Transport and Telecommunication Institute
Lomonosova 1, Riga, LV 1019, Latvia
[1]Ph.: +371 28621472, Fax: +371 67100535, e-mail: v.gaidass@gmail.com
[2]Ph.: +371 67100654, Fax: +371 67100535, e-mail: avg@tsi.lv

Vehicle counting is one of the most basic challenges during the development and establishment of Intelligent Transport Systems (ITS). The main reason for vehicle counting is the necessity of monitoring and maintaining the transport infrastructure, preventing different kind of faults such as traffic jams. The main applied solution to this problem is video surveillance, which is presented by different kind of systems. Some of these systems use a network of static traffic cameras, expensive for establish and maintain, or mobile units, fast for redeployment, but fewer in diversity.

In this paper, one particular concept of a low-cost mobile vehicle counting system is investigated, which uses an object detection method based on calculating "mass centre" of detected features of possible object. A hypothesis of improvement of the basic algorithm was formulated and a modification was proposed. In order to prove the hypothesis, both basic and modified algorithms were tested and evaluated.

Keywords: video surveillance, object detection, mass centre, vectorization, Kanade-Lucas-Tomasi method

1. Introduction

Nowadays, Intelligent Transport Systems (ITS) are becoming more integrated in domestic and global transport infrastructure, helping to maintain and monitor the situation of transport infrastructure as whole, optimizing traffic flows, preventing faults on roads and highlighting possible improvements and adjustments.

In order to make ITS more effective, wide range of sensors should be deployed and scattered across transport infrastructure. These sensors pick up and aggregate raw traffic data, which can be acquired by different means: radars, lidars, piezoelectric and infrared sensors and traffic cameras, and transform the data into more convenient form for further analysis.

One of most common ways to acquire traffic data is to use video surveillance by traffic cameras only or by surveillance units, which consist of video cameras and additional devices. In order to deploy video surveillance system, a surveillance scene and object detection algorithm should be defined and implemented as well as means to distinguish detected objects in case of traffic flow of high density. As for today, object detection can be implemented (Fischer and Beyerer, 2012) by top-down approach (detection by known object model) or bottom-up approach (detection by designated features of object).

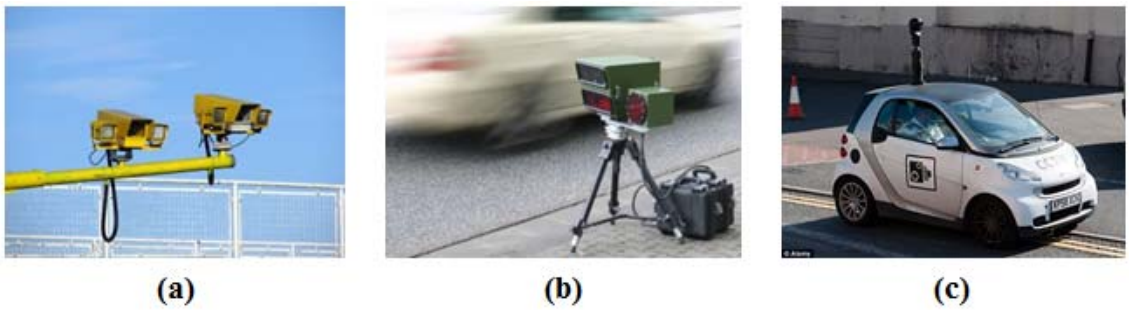

Figure 1. Examples of traffic video surveillance systems: (a) mounted on a pole (Vysionics, n.d.), (b) as a mobile unit (Vitronic, n.d.), (c) mounted on a surveillance vehicle (Gillman, 2014)

These systems can employ static surveillance units (Figure 1, a), which can be deployed in designated areas of dense traffic flow (mostly mounted on poles, bridges or highways) or mobile

surveillance units, small in size and easy to redeploy (Figure 1, b), or can even be mounted on surveillance vehicle (Figure 1, c). The main difference between systems is the complexity of architecture and cost of deployment and maintaining the surveillance units.

The diversity of video processing methods and algorithms implemented in these systems varies by the complexity of utilized components or techniques, such as:

- image background subtraction (Cao *et al*., 2012);
- one or more virtual lines, specially positioned on static surveillance scene and that are being triggered by "crossing" over them (Zhang *et al*., 2007; Kadikis and Freivalds, 2013);
- motion detection and tracking by evaluation motion vectors (Aslani and Mahdavi-Nasab, 2013);
- machine learning methods which use sets of predefined training images (Wang and Zhang, 2014) or 3-D shapes (Buch, 2010).

The main objective of this research is the modification of the "mass centre" algorithm (Grakovski and Murza, 2010) in such way that it would minimize the impact of occlusions (moving object overlays between each other). This can be done by adding motion vectors to each point feature of the object (also known as local features or "blobs") (Tuytelaars and Mikolajczyk, 2007) and selecting only those point features that match predefined motion direction criteria, which can be applied not only to static video surveillance systems, but also to mobile systems (Figure 1, c).

Section 1 provides brief overview and characteristics of the research problem. Section 2 reviews the concept of mobile vehicle detection and counting system and the "mass centre" algorithm as well as the hypothesis of mitigating the effects of occlusions. Section 3 introduces the modification of the initial algorithm. The description and results of the algorithm comparison are compiled in Section 4. Finally, the overall results of this paper as well as proposals for the future work are being discussed in conclusions.

2. The concept of mobile vehicle counting system

The concept of mobile vehicle counting system, proposed by Grakovski and Murza (2010), is based on the usage of a simple laptop and a web camera, defined as one separate processing unit, which can be installed on a designated surveillance vehicle. The vehicle conducts video surveillance, while being parked by a roadside or on a parking place nearby, observing every lane of the road from side view (Figure 2). Together, this makes a low-cost mobile system, easy to deploy and relocate.

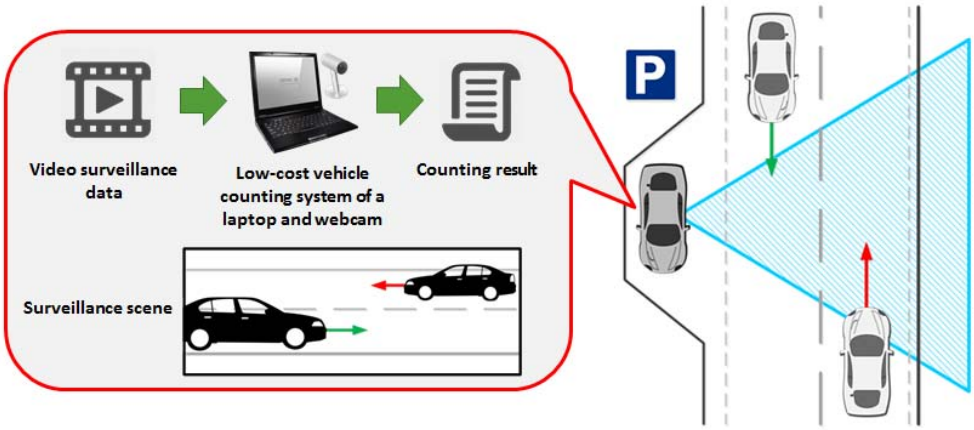

Figure 2. Concept of mobile vehicle counting system

2.1. The basic algorithm

The algorithm, proposed by Grakovski and Murza (2010) for their mobile vehicle counting system, is based on bottom-up approach.

As input data, the algorithm takes up a difference frame, which is the subtraction of two consecutive frames from input video stream. The difference frame would show all changes as well as movement of all objects between frames.

Then, a detector method is applied, which finds all point features on the difference frame. Point features (known as interest points, corner features or "blobs") are image patterns which differ from their neighbourhood by drastic change of an image property (e.g. colour intensity), which can be represented

by special points, edges or object corners. Point features have properties (Tuytelaars and Mikolajczyk, 2007) such as:

- repeatability (detection of the same feature on the same object under different viewing conditions);
- distinctiveness;
- locality;
- sufficient large quantity;
- accuracy (locality with respect to image location, scale and shape).

For this algorithm, the Harris detector (Harris and Stephens, 1988) was applied, which finds point features by sweeping rectangular or Gaussian window *W(u, v)* over image pixels and evaluating their change of colour intensity *I* by shifting the window in every direction (horizontal, vertical and diagonal):

$$M(x,y) = \sum_{u,v \in W} W(u,v) \begin{bmatrix} \left(\frac{\partial I}{\partial x}\right)^2 & \frac{\partial I}{\partial x} * \frac{\partial I}{\partial y} \\ \frac{\partial I}{\partial y} * \frac{\partial I}{\partial x} & \left(\frac{\partial I}{\partial y}\right)^2 \end{bmatrix}, \quad (1)$$

where *M(x, y)* is the autocorrelation matrix of the shift, $\frac{\partial I}{\partial x}$ and $\frac{\partial I}{\partial y}$ are the derivatives of pixel colour intensity *I* in the directions of *x* and *y* axis respectively.

A score *R* for each window is calculated to determine the presence of point feature (edge or corner):

$$R = \lambda_1 \lambda_2 - k(\lambda_1 + \lambda_2)^2, \quad (2)$$

where λ_1 and λ_2 are eigenvalues of matrix M, and *k* is the Harris constant (*k* = 0.04...0.06), proposed by Harris and Stephens (1988).

High λ_1 and λ_2 values determine the presence of corner feature. If one of eigenvalues is much greater than the other one, the presence of edge feature is determined.

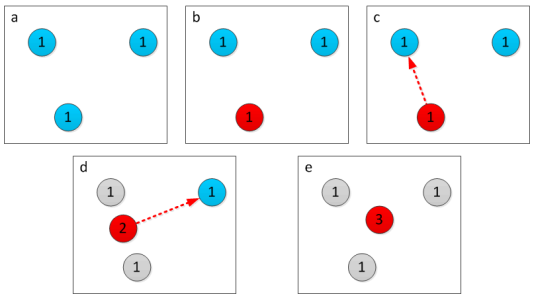

Figure 3. "Mass centre" calculation

After that, a "mass centre" is being calculated for the "cloud" of found point features. "Mass centre" is an estimate the centroid of point feature "cloud" and hence the possible moving object which can be used in localizing and tracking of detected objects (Grakovski and Murza, 2010). The "mass centre" is being calculated by iterative shifting between point features, increasing its "mass" by 1 each shift, while other features have assigned "mass", which equals 1 (Figure 3):

$$\begin{cases} offset = \dfrac{1}{mass_{centre} + mass_i}, \\ X_{centre} = X_{centre} \pm |X_{centre} - X_i| * offset, \\ Y_{centre} = Y_{centre} \pm |Y_{centre} - Y_i| * offset, \\ mass_{centre} = mass_{centre} + 1 \end{cases} \quad (3)$$

where $mass_{centre}$ is "mass" of the "mass centre", $mass_i$ is the "mass" of *i*-th point feature, X_{centre} and Y_{centre} are thecoordinates of "mass centre", $X_i$and Y_i are the coordinates of *i*-th point feature, and $offset$ is the value of "mass centre" shift towards *i*-th point feature.

By the end of calculation the "mass" would show how many point features are bind to the "mass centre", which would help filter out the small objects with small amount of point features.

During the object localizing and tracking, the "mass centre" together with bind features and calculated feature area are compared with expected vehicle dimensions (Figure 4).

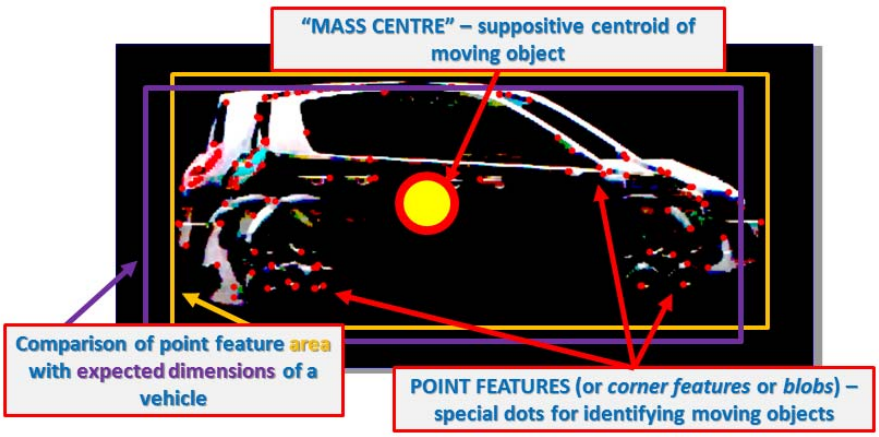

Figure 4. Point features, "mass centre" and comparison of feature area and expected dimensions

2.2. The hypothesis of modification

It was found that the basic "mass centre" algorithm with designated surveillance scene (Figure 2) is very sensitive to object obstructions or occlusions. In image processing and computer vision occlusion is an effect when far objects are occluded partially or completely by objects closer to the viewer or camera (Figure 5).

During occlusions, the probability of incorrect detection by the basic algorithm greatly increases, due to inability to distinguish point features of different objects (especially when the objects move towards each other).

Figure 5. Examples of occlusion with designated side-view surveillance scene

Calculations of additional "mass centres" (Grakovski and Murza, 2010) in order to prevent occlusion are quite burdening and would complicate the processing, so the hypothesis was formulated, that it is possible to decrease the probability of incorrect detection by adding evaluation of point features and objects motion direction.

3. The modification of the "mass centre" algorithm

One of most popular solutions to track object movement is using methods based on motion vectors also known as optical flow (Aslani and Mahdavi-Nasab, 2013). Optical flow represents vector field of all objects on the image, which is achieved by relative movement between the viewer (camera) and the scene. For convenience purposes, the modified algorithm was named as *"mass centre" vectorization* (MCV) algorithm.

To add vectorization of point features and "mass centres", pyramidal form (Bouguet, 2000) of Kanade-Lucas-Tomasi (KLT) algorithm (Tomasi and Kanade, 1991) was introduced into MCV algorithm.

KLT algorithm consists of two stages: point feature detection by Shi-Tomasi detector (Shi and Tomasi, 1994) and vector field calculation by pyramidal form (Bouguet, 2000) of Lucas-Kanade method (Lucas and Kanade, 1981). Instead of one difference frame, KLT algorithm requires two consecutive greyscale frames from input video stream.

Shi-Tomasi detector is an extension of the initial detector introduced by Tomasi and Kanade (Tomasi and Kanade, 1991), which is based on finding "good features to track" (Shi and Tomasi, 1994), which are more reliable to track for longer time spans. Their presence can be defined by next change to equation (2):

$$R = \min(\lambda_1, \lambda_2) > \lambda, \tag{4}$$

where parameter λ represents the threshold for eigenvalues of autocorrelation matrix M from equation (1).

Lucas-Kanade method (Lucas and Kanade, 1981) is one of the most popular methods to calculate optical flow of point features. It is based on the assumption that the displacement of image contents between two consecutive frames is small and constant within some window. Thus, it is possible to approximate the displacement by using first order derivatives of Taylor's series as differential equation:

$$\frac{\partial I}{\partial x}\frac{\Delta x}{\Delta t} + \frac{\partial I}{\partial y}\frac{\Delta y}{\Delta t} + \frac{\partial I}{\partial t} = 0, \tag{5}$$

and rewrite (5) as optical flow equation with unknown $\vec{v}$:

$$\nabla I^T * \vec{v} + I_t = 0, \tag{6}$$

where

$$\nabla I^T = [I_x \quad I_y],\ \vec{v}^T = [V_x \quad V_y],$$
$$I_x = \frac{\partial I}{\partial x},\ \ I_x = \frac{\partial I}{\partial y},\ \ I_t = \frac{\partial I}{\partial t}, \text{ and } V_x = \frac{\Delta x}{\Delta t},\ \ V_y = \frac{\Delta y}{\Delta t},$$

where ∇I is the spatial gradient, $\vec{v}$ is shift vector of optical flow, I_x, I_y, and I_t are derivatives of pixel colour intensity I in the x and y direction and time t respectively, V_x and V_y represent shift of the point feature in the x and y direction respectively between two frames.

The pyramidal form of Lucas-Kanade method (Bouguet, 2000) is the modification of the classical Lucas-Kanade method with increased accuracy and robustness, which consists of two procedures:

- Representation of initial images as L-layered pyramids in order to handle large pixel motions;
- Iterative optical flow calculation on each of L layers of the pyramid in order to minimize the error.

The pyramid is being built in a recursive fashion (Bouguet, 2000), starting from "zeroth" layer (which is also the initial image) $I^0 = I$ until the top level of pyramid I^{Lm} is reached.

The next layer of the pyramid $I^L(x, y)$ is being computed from the previous layer $I^{L-1}(x, y)$ with the usage of low-pass filter coefficients for image anti-aliasing (Bouguet, 2000):

$$\begin{aligned} I^L(x,y) = \frac{1}{4} I^{L-1}(2x, 2y) \\ + \frac{1}{8}\big(I^{L-1}(2x-1, 2y) + I^{L-1}(2x+1, 2y) + I^{L-1}(2x, 2y-1) \\ + I^{L-1}(2x, 2y+1)\big) \\ + \frac{1}{16}\big(I^{L-1}(2x-1, 2y-1) + I^{L-1}(2x+1, 2y+1) \\ + I^{L-1}(2x-1, 2y-1) + I^{L-1}(2x+1, 2y+1)\big) \end{aligned} \tag{7}$$

Bouguet (2000) defines equation (7) only for x and y values that satisfy:

$$\begin{aligned} 0 \le 2x \le n_x^{L-1} - 1, \\ 0 \le 2y \le n_y^{L-1} - 1, \end{aligned} \tag{8}$$

where n_x^{L-1} and n_y^{L-1} are width and height of image I^{L-1}.

Hence, during the construction of the pyramid, each next layer is two times smaller than the previous one.

The iterative optical flow calculation starts from the top layer of the pyramid L_m (or the smallest image). At the start of computation (Bouguet, 2000), the initial guess for pixel displacement is $\vec{g}^{L_m} = [0, 0]$. Then, the residual optical flow $\vec{d}^{L_m} = \vec{v}$ is being calculated as defined in equation (6). In order to minimize the error between the images, the calculation is being performed in Newton-Raphson fashion (Bouguet, 2000): in *k* iterations until the error would be smaller than the predefined threshold or the maximum count of iterations is reached.

After that, both guess and residual optical flow are being passed to next level *L-1* of the pyramid as the new guess (Bouguet, 2000):

$$\vec{g}^{L-1} = 2(\vec{g}^{L} + \vec{d}^{L}) . \quad (9)$$

The final result is available after the computation on the "zero[th]" layer L_0:

$$\vec{d} = \vec{g}^{0} + \vec{d}^{0}. \quad (10)$$

The illustration of the pyramidal form of Lucas-Kanade method is shown in Figure 6.

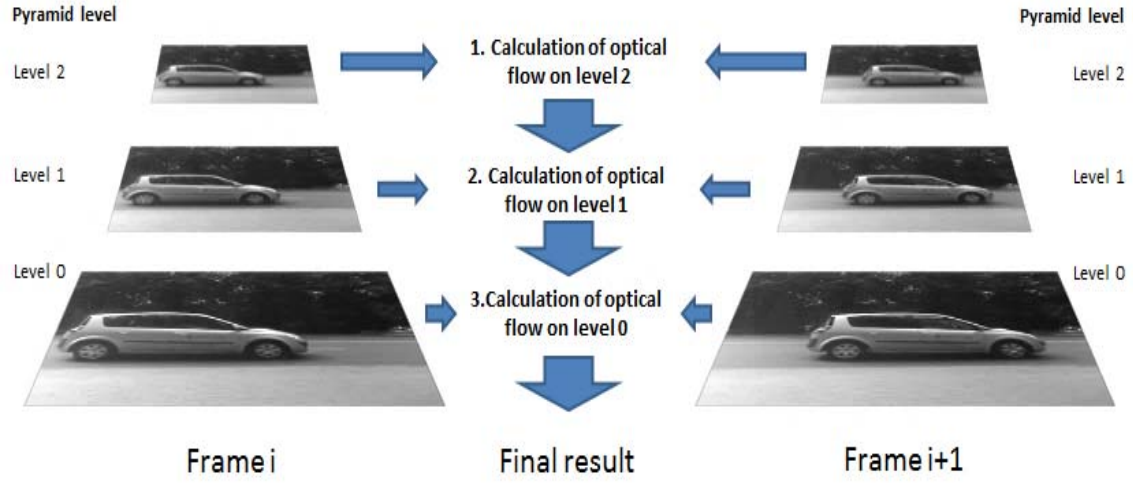

Figure 6. The pyramidal form of Lucas-Kanade method

The result frame would show point features and its displacement between frames as vectors (Figure 7, a), which could be filtered (Figure 7, b) by their length and direction (by evaluating vector angle).

Figure 7. (a) Result of Lucas-Kanade method, (b) Filtered optical flow

In order to group and distinguish "clouds" of optical flow, "mass centres" are being calculated for every "cloud" (Figure 8, a).

An adjustment was made to the calculation (3): if *i*-th point feature's Euclidean distance from the current "mass centre" is too long, it would not be bound to it, but to another "mass centre", forming a new

cluster (Figure 8, b). This adjustment imitates k-means clustering algorithm, where, in order to minimize within-cluster variance, Euclidean distance between the points and cluster centre is evaluated.

(a) (b)

Figure 8. Distinction of optical flow clusters of: (a) opposite direction, (b) same direction

In order to prevent detection errors during occlusions, an adjustment to object localization and tracking was made. MCV algorithm localizes and identifies new objects only when they are entering video camera's coverage area (from the right side of the frame, if object moves "to the left" and vice versa), where probability of occlusion is much lower than in the middle of coverage area (middle of the frame). During object's movement through the coverage area, MCV algorithm tracks the object each frame until it leaves the coverage area.

4. Implementation of MCV algorithm and comparison

Finally, both "mass centre" algorithm (Grakovski and Murza, 2010) and MCV method (paragraph 2.3) were evaluated by two experimental videos. Both videos were made accordingly to designated surveillance scene. The first video clip depicts a two-lane road with two opposite traffic flows (Figure 9, a), and the second video clip is a four-lane road with two pairs of opposite traffic flows (Figure 9, b).

(a) (b)

Figure 9. Frame from: (a) video clip 1, (b) video clip 2

True positive rate (TP) and false positive rate (FP) were chosen as comparison metrics (11), which are quite popular in evaluating object detection. TP indicates the probability of correct object detection and FP represents the probability of false alarm:

$$TP = \frac{detected_{true}}{N_{vehicle}},$$
$$FP = \frac{detected_{false}}{detected_{true} + detected_{false}}, \quad (11)$$

where $detected_{true}$ is the amount of correct detected objects, $detected_{false}$ is the amount of false alarm, and $N_{vehicle}$ represents actual number of vehicles that moved through coverage area of the video camera.

The results of algorithm comparison are compiled in tables 1 and 2.

Table 1. Algorithm comparison results, video clip 1

	Detected vehicles					
	Going "right"		Going "left"		Total count	
	TP	FP	TP	FP	TP	FP
"Mass centre" algorithm	0.737	0.067	0.66	0.076	0.729	0.01
MCV algorithm	1	0.05	1	0	1	0.026

Table 2. Algorithm comparison results, video clip 2

	Detected vehicles					
	Going "right"		Going "left"		Total count	
	TP	FP	TP	FP	TP	FP
"Mass centre" algorithm	0.88	0.21	0.736	0.22	0.805	0.325
MCV algorithm	0.88	0.117	0.84	0.058	0.86	0.088

5. Conclusions

To sum up, experimental results showed that "mass centre" vectorization algorithm increases the probability of correct object detection and decreases the probability of false alarm. Also, the true positive rate and false positive rate values for "right" traffic flow differs from the "left" traffic flow, which can be explained by the positioning of the camera: one of traffic flows will always be closer to viewer or camera and would show more point features for further analysis.

The results prove the hypothesis, that the evaluation of optical flow increases the effectiveness of object detection algorithm.

However, even "mass centre" vectorization algorithm's effectiveness will decrease in case of multiple occlusions over multiple traffic lanes. The reason is the designated surveillance scene which has its own limitations.

The possible solutions to multiple occlusion problem can be additional filtering of optical flow, more sophisticated clustering procedure, vehicle classification as well as adjustments to surveillance scene, which will be investigated in future work.

Acknowledgements

This research was granted by Latvian state research program project "The next generation of information and communication technologies (Next IT)" (2014-2017).

References

1. Aslani, S. and Mahdavi-Nasab, H. (2013) Optical Flow Based Moving Object Detection and Tracking for Traffic Surveillance. In: *World Academy of Science, Engineering and Technology. International Journal of Electrical, Computer, Energetic, Electronic and Communication Engineering*, 7(9), 1252-1256, DOI: scholar.waset.org/1999.5/17157.
2. Bouguet, J.-Y. (2000) *Pyramidal Implementation of the Lucas Kanade Feature Tracker. Description of the algorithm.* Intel Corporation, Microprocessor Research Labs.
3. Cao, Y., Lei, Z., Huang, X., Zhang, Z. and Zhong, T. (2012) A Vehicle Detection Algorithm Based on Compressive Sensing and Background Subtraction. In: *AASRI Procedia*, 1, 480-485, DOI: 10.1016/j.aasri.2012.06.075.
4. Fischer, Y. and Beyerer, J. (2012) A top-down-view on intelligent surveillance systems. In: *Proceedings of the Seventh International Conference on Systems (ICONS)*, February 29 - March 5, 2012, Saint Gilles, Reunion Island, pp. 43–48.
5. Grakovski, A. and Murza, A. (2010) Development of Segmentation Method Based on "Mass Centre" Approach for Video Surveillance Data of Transport Vehicles Flow. In: *Transport and Telecommunication*, 11(2), 18-29.
6. Harris, C. and Stephens, M. (1988). A combined corner and edge detector. In: *Proceedings of the 4th Alvey Vision Conference*, August 31 – September 2, 1988, University of Manchester, Manchester, United Kingdom, pp. 147–151.

7. Kadikis, R. and Freivalds, K. (2013) Vehicle classification in video using virtual detection lines. In: *Proceedings of SPIE. Sixth International Conference on Machine Vision (ICMV 2013),* Volume 9067, DOI: 10.1117/12.2051028.
8. Lucas, B. and Kanade, T. (1981) An Iterative Image Registration Technique with an Application to Stereo Vision. In: *Proceedings of the 7th international joint conference on Artificial intelligence (IJCAI'81)*, August 24-28, 1981, University of British Columbia, Vancouver, B.C., Canada, Volume 2, pp. 674-679.
9. Shi, J. and Tomasi, C. (1994) Good Features to Track-In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94)*, June 1994, Seattle, USA, pp. 593-600.
10. Tomasi, C. and Kanade, T. (1991) *Detection and Tracking of Point Features.* Carnegie Mellon University Technical Report CMU-CS-91-132, April 1991.
11. Tuytelaars, T. and Mikolajczyk, K. (2007) Local Invariant Feature Detectors: A Survey. In: *Foundations and Trends in Computer Graphics and Vision*, 3(3), 177–280, DOI: 10.1561/0600000017.
12. Wang, H. and Zhang, H. (2014) A Hybrid Method of Vehicle Detection based on Computer Vision for Intelligent Transportation System. In: *International Journal of Ubiquitous Engineering*, 9(6), 105-118, DOI: dx.doi.org/10.14257/ijmue.2014.9.6.11.
13. Zhang, G., Avery, R. and Wang, Y. (2007) Video-Based Vehicle Detection and Classification System for Real-Time Traffic Data Collection Using Uncalibrated Video Cameras. In: *Transportation Research Record: Journal of the Transportation Research Board*, 1993, 138-147, DOI: http://dx.doi.org/10.3141/1993-19.

Web sources:

14. Buch, N.E. (2010) *Classification of vehicles for urban traffic scenes*. PhD thesis. Kingston University, viewed 2 November 2016, http://www.bmva.org/thesis-archive/2010/2010-buch.pdf.
15. Gillman, O. (2014) *Councils will be ordered to come clean about how much they make from parking fines.* October 4, 2014. Daily Mail Online, viewed 2 November 2016, http://www.dailymail.co.uk/news/article-2780616/Councils-ordered-come-clean-make-parking-fines.html#ixzz4C0sODcKY.
16. Vitronic, n.d. *Police Scan Remote Camera. Product Update*, viewed 2 November 2016, https://www.vitronic.com/traffic-technology/applications/traffic-enforcement/speed-enforcement/poliscan-speed-mobile.html.
17. Vysionics, n.d. SPECS, viewed 2 November 2016, http://www.vysionics.com/product/specs.