Image Processing of Motion for Security Applications

Frantisek Duchon, Assoc. Prof.,
Peter Bučka, MSc.,
Martina Szabová, MA,
Martin Dekan, PhD.,
Peter Beňo, PhD.,
Michal Tolgyessy, PhD.
Slovak University of Technology, Slovakia

doi: 10.19044/esj.2017.v13n27p44 URL:http://dx.doi.org/10.19044/esj.2017.v13n27p44

Abstract

The aim of the article is a design, execution and examination of the computer vision systems, which processes digital video, reduces noise to a minimal level, and identifies a moving object together with estimation of its distance from the camera. For the image processing, library OpenCV was used. Two different methods were examined and implemented in control system. Some results are very similar in character and functionality with the use of security camera system, but the determining the distance of a given object is a new advanced ability of proposed system.

Keywords: Image processing, security applications

Introduction

Automatic image processing technologies are currently being applied in many industries. Typical applications include evaluation the quality of parts or workpieces, crop monitoring in agriculture, or detecting emerging queues on the road. Such systems are also used in security applications for objects or their parts supervision. Since labor costs are steadily increasing, the need for such automated systems capable of replacing tedious human activity is still growing. Properly designed technology can greatly help the operator not to omit important occurrences in the system. It may be on one hand motion recording, virtual border crossing, wall climb detection, vehicle detection or its license number, missed object detection (e.g. luggage at the airport), or on the other hand detecting a missing object in space. Such occurrences can be many in one system, and it is not in the human power to follow everything reliably. Therefore, it is necessary to meet with customer

requirements and find out which solutions will be implemented and how successful it is necessary to follow the desired result.

To choose the development environment when designing a system plays a very important role. There are several commercial solutions that have a certain amount of reliability, but other tools are more appropriate for prototyping innovative solutions. The first software is MATLAB from MathWorks Company (Mathworks 2017). MATLAB is a development environment with its own programming language that includes the Computer Vision System Toolbox (Mathworks 2017) for image processing. It is a powerful tool for implementing motion detection in a video or for detecting specific objects in the image (Hargas 2010) (Hargas 2016). However, its price is a bit higher; therefore the OpenCV library was used as the following solution. OpenCV is an open source library group designed primarily for the development of computer vision applications (OpenCV 2017). Several programming languages can be used for application development, e.g. C, C ++, Python, and other. At the same time, OpenCV provides a wide functionality and it is available for free.

Motion detection

Between the two following camera frames exists a time delay. From the user's point of view, this parameter is characterized as the image refresh rate (Cieszynski 2007). So it shows the number of frames recorded by the camera per second. In other words, this parameter indicates how often a change in the image can occur at a given time. The human eye can recognize a speed of about 25 frames per second. All speeds above this value are considered to be a smooth video for people. For most security applications, however, such recording speed is unnecessary. By reducing the refresh rate, it is possible to save the computing power and the storage space.

Motion in the image can be detected using multiple techniques (Varga 2016) (Sukop 2012). One of the most appropriate motion detection tools for security applications is the difference-matching technique in two following frames (Bradski 2008). The images are compared based on the difference of the corresponding pixels, in addition the difference being represented in the matrix. This matrix has the same size as the original images. Typically, the match of two pixels is represented by a 0 value. Any higher value represents the level of difference per pixel within the frames being compared (Fig. 1).

Despite the relatively simple and effective algorithm, such a comparison obviously has some shortcomings. Errors in such motion detection can be caused by a number of reasons. The most common error is the false motion detection (Cieszynski 2007). It occurs primarily within the exterior scanning and it is often caused by the movement of objects that do

not play a significant role in the security detection itself. These include, for example, trees, clouds, water movement, or animals. There is no universal algorithm to remove such false detection, and for each application it is necessary to eliminate these false signals separately and with specific conditions.

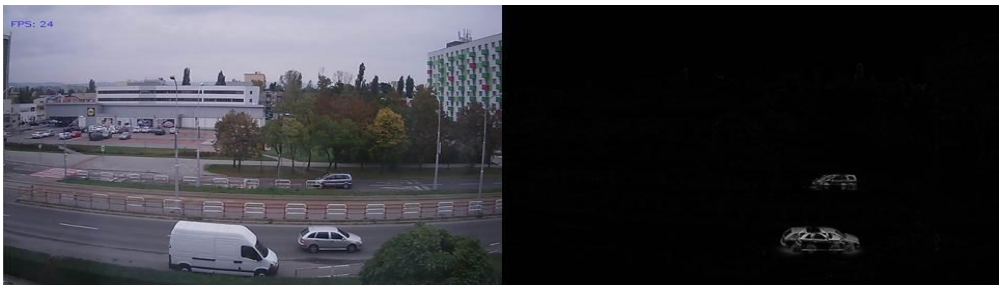

Fig. 1 Scene (on the left) and the result of comparing two following frames (on the right).

Another type of often-occurring error is the poor lighting of the scene, which causes an increased presence of noise in the image (Cieszynski 2007) (Fig. 2). There are two ways to reduce the noise level. The first is to increase the scene lighting intensity, but this is not always possible. Another way is to extend the shutter opening time, but this can be done only if the camera supports it. In this solution, however, it must be assumed that with the increasing opening time of the shutter, blurring of fast-moving objects (Cieszynski 2007) occurs. By default, however, cameras producers build in the different algorithms and filters in their products to help eliminate this noise (Table 1). Lighting conditions are also associated with fake motion detection caused by the different light reflections or changing the intensity of light sources in the scene. Such a situation is particularly typical when monitoring the area where the vehicles are moving under the conditions of insufficient natural lighting. The car with lights on the road does not have to go directly to the scene, but the light from its reflectors is evaluated as a moving object.

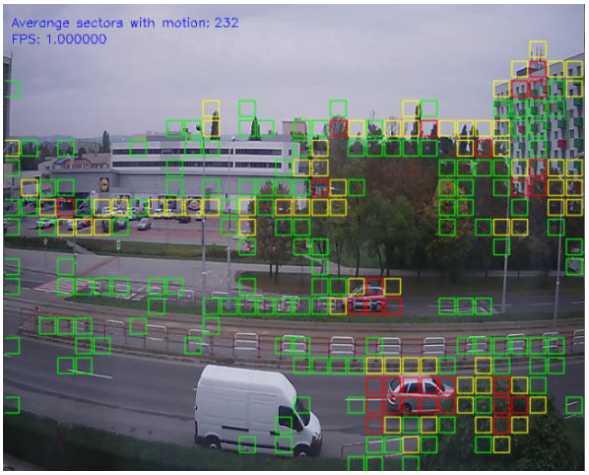

Fig. 2 Incorrectly set the level of noise filtering in the image causes the false motion detection in the image (colored squares show the detected motion and the color determines its intensity level - the red is the highest).

Tab. 1 Example of camera producers and used noise filters

Producer	Camera type	Used filters to remove noise
Sony	IPELA E series	XDNR – eXcellent Dynamic Noise Reduction
Honeywell	HCD1FX	AGC – Automatic Gain Control DNR – Dynamic Noise Reduction
HIK Vision	Dark Fighrer series	3D DNR
Samsung	SNB 6004	SSNRIII 2D+3D noise reduction

Experiments with the Motion Detection

For the initial experiments, a security camera from the Dahua manufacturer with designation IPC-K100AP, the Nikon Coolpix L120 camera, and the Lenovo P70-A mobile phone were selected. The detection algorithm consisted of the following steps (Fig. 3):

1. Opening a video stream.
2. Loading the two following frames from a video.
3. Color model change from RGB to black and white.
4. Frames defocusing (noise filtering).
5. Frames comparison and differences recording.
6. Application of erosion and dilation to frames.
7. Image thresholding and writing to the new matrix.
8. Specification the region for detecting non-zero values in the frame section.
9. Designation of regions with detected motion.
10. Overwriting the previous frame with the current one.
11. If the video is not finished, continue with the point 2.
12. End.

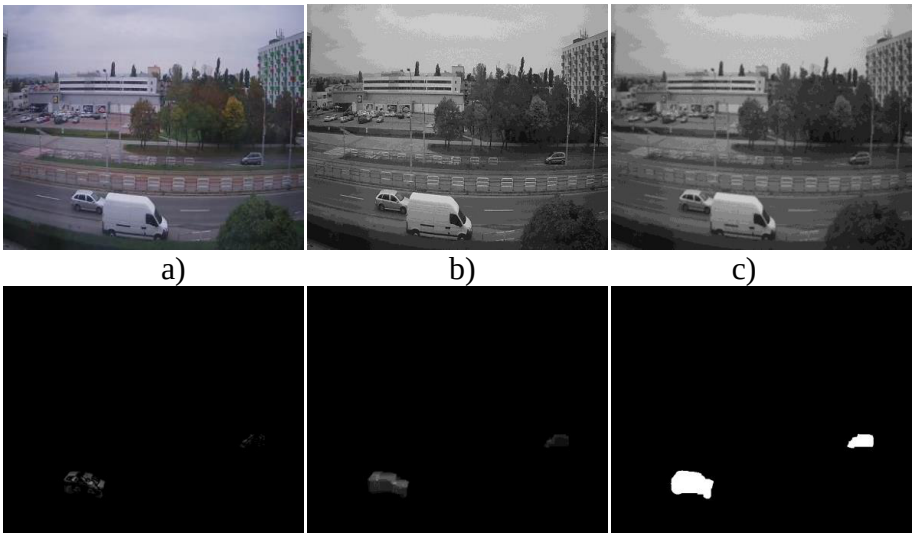

Fig. 3 Part of the algorithm applying transformations and filters: a) the original image, b) the change of the color model, c) the blur of the frame, d) frames difference, e) erosion and dilation, f) threshold value overload

Consequently, the algorithm divides the image into multiple regions where movement is detected separately (Fig. 4). For a 640x480 pixel video, the 10x10 pixel area was selected, creating 64 times 48 regions in the image. This is sufficient to determine the area of motion. Quantification of motion in the region is based on the number of pixels identified as "in motion". If this level is below 40%, the area is marked with green color, if between 40% and 80% is with yellow and above 80% red color. Such a view will be able to promptly warn the operator for movement in the image.

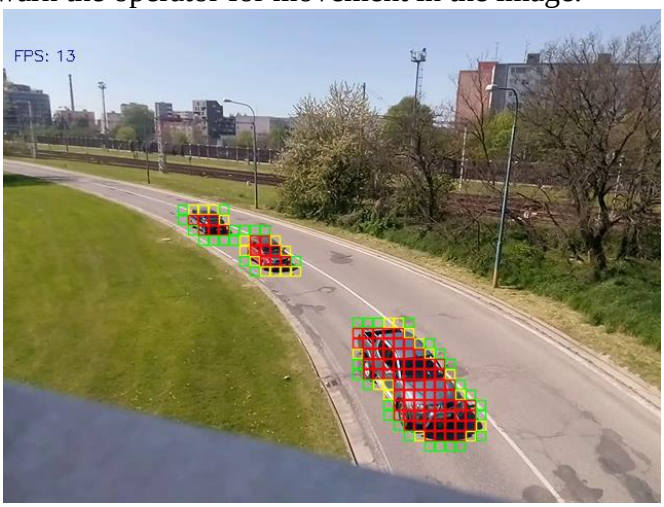

Fig. 4 Motion detection in the image by division into regions

Another way of identifying the motion is to use the contours object finder. Based on the contours of moving objects, a bounding box was plotted. Its dimensions and position are given by the boundaries of contours of moving objects (Fig. 5).

Fig. 5 Motion detection by searching the contours of floating objects

Both ways of identifying motion in the image can identify multiple moving objects. The first method is limited by the number of areas on which the image is divided. This method does not even distinguish the objects themselves; it only marks areas with motion in the image. The second method is able to identify any number of objects, while algorithms using higher intelligence need to be applied to distinguish two nearby moving objects.

Distance measurement

In order to determine the distance of an object from a single camera, it is necessary to know the parameters of the camera that the scene is being shot, the lens parameters, the location of the camera in the space and the angular displacement of the camera compared to the ground. One of the camera's important parameters is the angle of view. The angle of view is defined as the horizontal and vertical viewing angles that the camera is able to record (Cieszynski 2007) (Fig. 6). If its size is not fixed, i.e. is optional, so this camera feature is called optical zoom. In this case, the value indicates how many times it is possible to zoom in on the object at a minimum angle to the maximum. These parameters are not directly dependent on the camera, but they are depending on the used lens. It is important to calibrate the camera to determine the camera parameters correctly. There are several camera calibration options and they usually use modeling of known shapes

and dimensions from multiple angles. Consequently, using the triangulation methods, the camera's internal parameters are determined (Letanovska 2014).

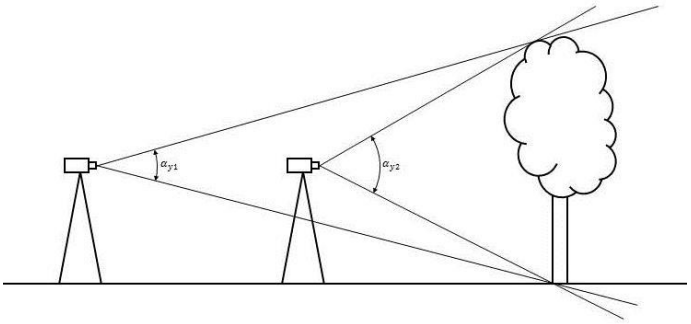

Fig. 6 Lens selection based on the distance from the observed object (αy - vertical viewing angle)

To determine the distance of an object from a single camera, it is necessary to know the location of the camera, its rotation to the ground, and the scanned scene parameters (Cheng 1997). If we consider the camera image to be a sector of a sphere in the area of the scene with a known vertical viewing angle (αy), then the angle between each line in the image (βy) is equal to a share of the viewing angle and the vertical resolution of the camera sensor (R_y) (Fig. 7):

$$\beta_y = \frac{\alpha_y}{R_y}$$

A similar relation can also be applied to the horizontal direction:

$$\beta_x = \frac{\alpha_x}{R_x}$$

Thus, each point in the image represents a vector and its angle is given by its position in the image matrix and it is also depending on the camera's rotation in the space. The size of this vector is given by the parameters of the scanned scene.

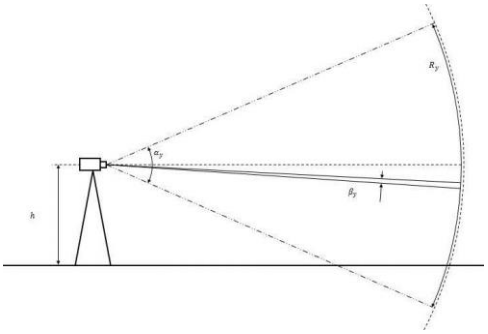

Fig. 7 Representation of the relationship between the vertical resolution and the vertical angle of the shot

For simplicity, it will be assumed that the object, whose distance it is needed to be found, is located in the horizontal axis at the center of the scanned scene and in any vertical direction. It is assumed that it is a horizontal flat scene without significant terrain inequalities (e.g. a room, a parking place) and the camera is placed on a tripod with a given height without being angled at any angle to the ground. We can consider the distance between the camera and the lowest point representing the object to be the object distance from the camera. In the case of selecting the reference point for distance measuring from another part of the object (e.g. the center or the top edge), it would be necessary to have the knowledge about its exact height. Assuming the object is on the ground, it is able to estimate the distance from the camera by applying the transformations:

$$l_c = h.tan\left(\gamma_y\right)$$

$$l_c = h.tan\left(90 - 0.5\alpha_y + n_y.\beta_y\right)$$

$$l_c = h.tan\left(90 - 0.5\alpha_y + n_y.\frac{\alpha_y}{R_y}\right)$$

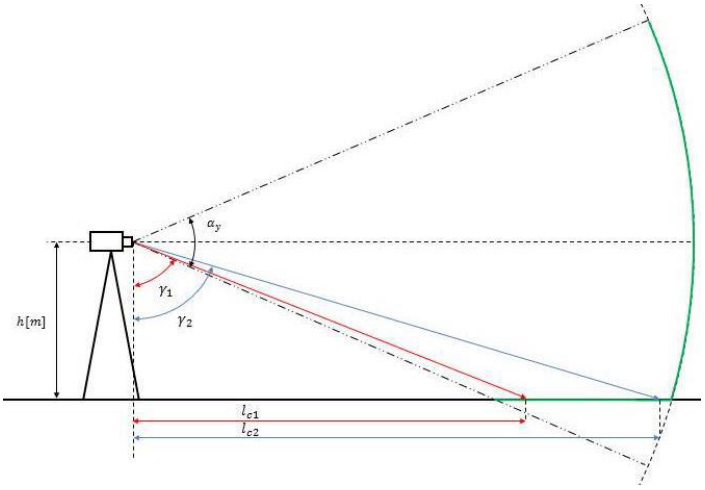

Fig. 8 Distance calculation of the object located in the vertical axis of the shot

The word “estimation” is used intentionally, because the accuracy of the calculation is limited by the resolution of the camera. And the error in the distance estimation increases with the distance of the object. By expanding the movement of the object in any direction, it is necessary at first to determine the center of the object to which the distance will be determined:

$$l_r = \frac{l_c}{cos\gamma_x}$$

$$l_r = \frac{l_c}{cos\left(0.5\alpha_x - n_x \frac{\alpha_x}{R_x}\right)}$$

$$l_r = \frac{h.tan\left(90 - 0.5\alpha_y + n_y . \frac{\alpha_y}{R_y}\right)}{cos\left(0.5\alpha_x - n_x \frac{\alpha_x}{R_x}\right)}$$

Where *h* is the height of the lens from the terrain, *lr* is the real distance of the object from the camera and n_x is the number of columns from the left side of the image to the center of the detected object.

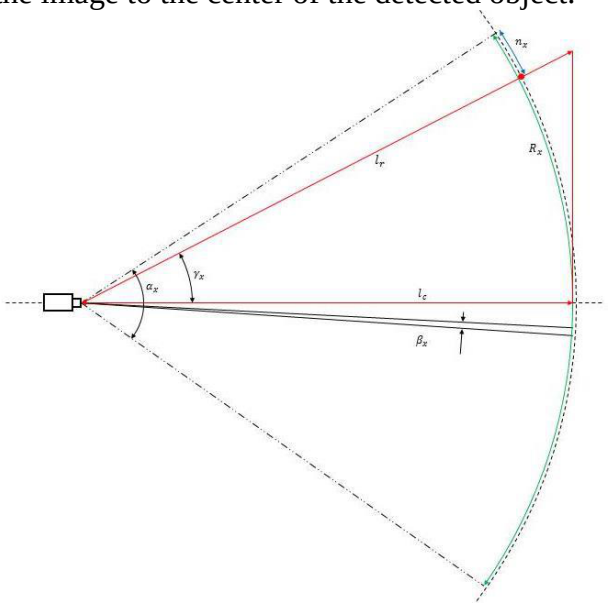

Fig. 9 Spatial relations for the distance calculation of the object located at any point in the scanned scene.

Of course, in such measurement a various errors can occur. The camera resolution causes together with the increasing distance of the observed object an inaccuracy in the distance estimation of the object. This is due to the fact that the pixel in the image is defined by a certain angle from the viewing space both horizontally and vertically. But the object's position in the image can only acquire the discreet values (Fig. 10).

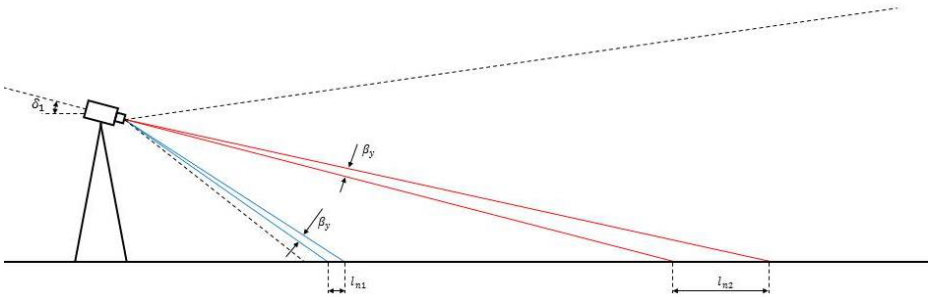

Fig. 10 Effect of camera resolution on distance estimation (δ - Vertical camera storage, l_{n1}, l_{n2} - Pixel distance)

To measure this problem, measurements for the IPC-K100AP camera with a 480 pixel vertical resolution, a vertical angle of 50 °and a camera height above the terrain of 4 m (Tab. 2) and 8 m (Tab. 3) were performed. It is clear from the results that the measurement accuracy not only affects the location of the camera above the ground, but also the camera resolution and the width of the camera shot.

Tab. 2 Measuring the difference in distance between adjacent pixels depending on the camera's slope angle for the camera above the terrain of 4 m.

Angle γ [°]	Object distance for		Distance difference [m]
	Lower pixel edge [m]	Upper pixel edge [m]	
45	4.00	4.01	0.01
55	5.71	5.73	0.02
65	8.58	8.62	0.04
75	14.93	15.04	0.11
85	45.72	46.70	0.98
88	114.55	120.83	6.29

Tab. 3 Measuring the difference in distance between adjacent pixels depending on the camera's slope angle for the camera above the terrain of 8 m.

Angle γ [°]	Object distance for		Distance difference [m]
	Lower pixel edge [m]	Upper pixel edge [m]	
45	8.00	8.03	0.03
55	11.43	11.47	0.04
65	17.16	17.24	0.08
75	29.86	30.07	0.22
85	91.44	93.39	1.95
88	229.09	241.67	12.58

The second factor that greatly affects the distance measurement accuracy using camera is the irregular terrain in the scanned scene. This error can be suppressed by adjusting the relation for the particular scene distance calculation. However, such a modification does not achieve a universal solution. The most common irregularities, that can be relatively easily

included in the calculations, are the slope of the terrain. This can be done by transforming the slope of the terrain into a vertical observation angle. These angles are added together and using transformation will be the surface flat. And the camera is sloped by the corresponding angle compared to the surface (Fig. 11).

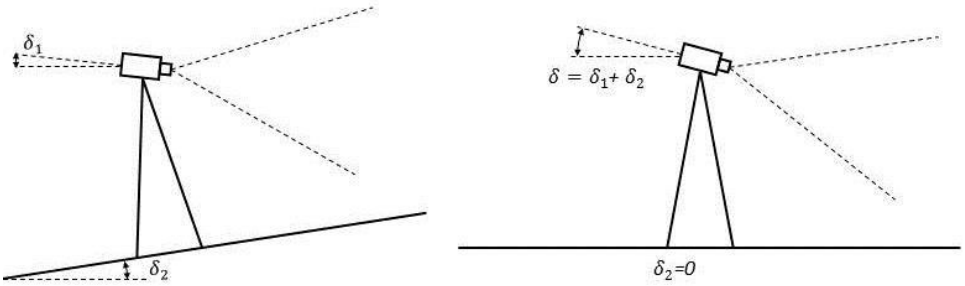

Fig. 11 Compensation of terrain irregularity

The last factor significantly affecting the resulting measurement of the object's distance is the light conditions. Light sources do not directly affect the measurement if there are no extreme lighting conditions (e.g. complete darkness or illuminated scene). The measurement is affected if the light source is placed behind the object under observation, then the observed subject cast the shadow in front of itself. In the case of a moving object, this shadow is also detected as a moving object, and therefore the distance of the object is determined by the distance of the shadow's beginning, not the object itself (Fig. 12). Otherwise, a situation may occur when the light source is evaluated as a motion in the image. Suppression of such errors is not easy and can be universally suppressed only by the knowledge of the observed object, e.g. vehicle color.

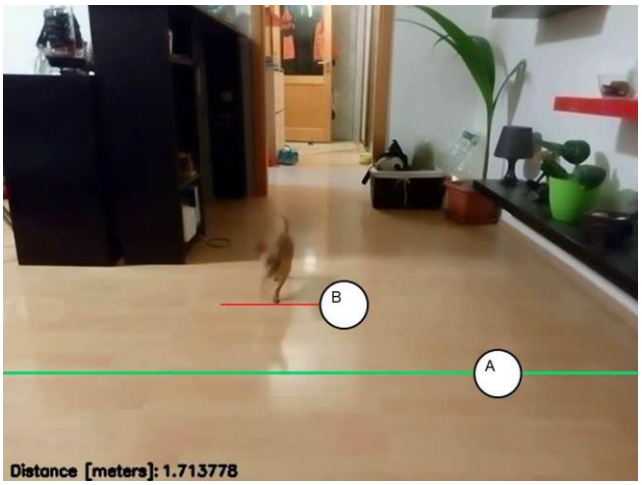

Fig. 12 Inaccuracy of measuring the distance of a moving object caused by the reflection of light, a - Incorrectly estimated distance, b - Reference distance

Results and measurement

Prior to performing the measurements themselves, appropriate scenes were selected, e.g. an object or parking guarding. To calculate the distance, parameters such as the height of the camera above the scene were correctly identified. To verify the identification, the distances of some objects from the camera were manually recorded (Fig. 13). The relations mentioned in the chapters above were applied using the OpenCV library.

Fig. 13 The reference scene for the visual system calibration. The distance of the object is determined in the lower left corner and the green line represents the lower edge of the moving object. The reference band that was used for the system calibration is applied in the scene.

This calibrated system was used to measure the objects distance in the real conditions. These were underground garages in a multifunctional building. Due to the good illumination of these spaces a phenomenon of variable routing of the moving objects shadows occurs (Fig. 14).

Fig. 14 Changing the direction of the moving object's shadow (the direction of the shadow is indicated by a red arrow).

For this reason, it was not possible to use the motion detection method by comparison of the frames for a sufficient accurate measurement, and finally, a color-based HSV model (the method of specific color monitoring) was used (Figure 15).

Fig. 15 Distance measurement using the color detection (left - image with plotted measured values, right - display of the searched color in the image).

Another issue that occurred during the measurement was the incorrect data provided by the camera manufacturer. For this experiment, the DS-2CD2132F-I camera was used and the manufacturer gave an image width of 98.5 °. Of this, a 4:3 ratio and a resolution of 2048x1536 pixels results the vertical viewing angle of 73.875 °. However, from the measured distances it is a vertical viewing angle of 65.28 ° (Fig. 16). This difference also caused an inaccuracy in the calculations of distance to the object. After setting the right parameters, experiments were made to measure the distance of the moving object. The results (Fig. 17 and Fig. 18) confirm the correctness of the set algorithms. The uncertainties in the measurement are due to the above-mentioned factors. However, these results are sufficiently accurate for object-guard applications.

Fig. 16 Experiment for the accurate determination of the camera's vertical viewing angle.

Fig. 17 Distance measurement to the object (the reference distance of 10 m), measured distance of 11.07 m.

Fig. 17 Distance measurement to the object (the reference distance of 5 m) measured distance of 4.73 m.

Conclusion

The aim of this article was to design:

- a simple real-time algorithms for motion detection in the image,
- motion detection at multiple locations,
- algorithms for estimating the distance of a moving object.

The results and experiments show some interesting findings:

1. These simple algorithms can be used as a support tool for operators. An example may be a situation requiring the estimation of the moving

object's distance or the determination of the moving object's speed. Such algorithms can be used, for example, in the fight against terrorism, where non-standard moving objects could be detected, objects with too high / low speed or often changing direction of movement.

2. Parameters that affect the quality of motion detection have been determined. These can be suppressed by improving the camera and computing technology. Despite some shortcomings, motion detection is sufficiently sensitive and accurate. The level of proposed algorithms is comparable to the commercially available security applications.

Acknowledgements

This work was supported by grants Req-00347-0001, VEGA 1/0065/16 and ITMS 26240220033.

References:

1. Bradski, G., Kaehler, A. (2008). *Learning OpenCV*. O'Reilly Media.
2. Cieszynski, J. (2007). *Closed Circuit Television*. Oxford: Elsevier Ltd.
3. Cheng, G., Zelinsky, A. (1997). *Real-Time Visual Behaviours for Navigating a Mobile Robot*, Canberra.
4. Hargas, L., Koniar, D., Bobek, V., Stofan, S., Hrianka, M. (2010). *Sophisticated measurement of non-electrical parameters using image analysis*. Robotics in education : proceedings of the 1st international conference : RiE 2010 : Bratislava, Slovakia, September 16-17, 2010.
5. Hargas, L., Koniar, D., Loncova, Z., Hrianka, M., Simonova, A., Joskova, M., (2016). *Artefacts detection for video sequences analysis*. ELEKTRO 2016: 11th international conference : May 16-18, 2016 Slovak Republic.
6. Letanovská, L. (2014). *Calibration of intrinsic parameters of camera* (in Slovak), Bratislava.
7. Mathoworks (2017). *Design and simulate computer vision and video processing systems*. Retrieved from:
8. https://www.mathworks.com/products/computer-vision.html
9. OpenCV (2017). Open Source Computer Vision Library. Retrieved from:
10. http://www.opencv.org/
11. Sukop, M., Hajdku, M., Varga, J., Vagas, M. (2012). *Image processing and object founding in the robot soccer application*. International Scientific Herald, Vol. 3, no. 2 (2012), p. 203-206.
12. Varga, J., Sukop, M. (2016). *Simple algorithm for patterns recognition*. Applied Mechanics and Materials : Automation and Robotics in Production Engineering, Vol. 844 (2016), p. 75-78.