Camera Sabotage Detection for Surveillance Systems

Damian Ellwart, Piotr Szczuko, and Andrzej Czyżewski

Gdańsk University of Technology, Multimedia Systems Department
Narutowicza 11/12, 80-233 Gdańsk, Poland
{ellwart,szczuko,andcz}@sound.eti.pg.gda.pl

Abstract. Camera dysfunction detection algorithms and their utilization in real-time video surveillance systems are described. The purpose of using the proposed analysis is explained. Regarding image tampering three algorithms for focus loss, scene obstruction and camera displacement detection are implemented and presented. Features of each module are described and certain scenarios for best performance are depicted. Implemented solutions are evaluated as independent events and final results are discussed. A detection efficiency improvement method is proposed.

Keywords: surveillance, tampering, defocus, displacement, obstruction.

1 Introduction

Video surveillance systems are commonly used as a mean of safety. Researchers develop complex methods allowing for so called smart or intelligent systems to be built. The recordings from monitoring systems often serve as an evidence in case of crimes. Therefore vital areas need to be observed continuously and the acquired videos should meet a good quality level requirements. There are many recommendations describing as how the cameras should be located and what additional conditions should be fulfilled to acquire the highest area coverage and yield best system performance. Hence, stationary cameras placed once should not be moved providing reliable image at all times. Most of the currently produced outdoor cameras are equipped with a weather and damage proof enclosure securing the camera itself, and guarantying proper work in various conditions. Despite these precautions the video provided by the camera can be damaged in different ways, for example by painting the camera lens or enclosure. In case of indoor monitoring, the cameras might be accessed quite simply making them an easy target susceptible of being tampered or damaged. To deal with this kind of threats, tamper detection algorithms are developed. There are some hardware implemented solutions already on the market but most of them require additional processing card and a dedicated software. Currently much work is dedicated to video analysis which enables detection of threat related situations. The tampering detection algorithms described in this paper are considered as a part of a larger smart surveillance system. Therefore, besides the algorithm detection accuracy, its low complexity is important making it useful for real-time processing.

2 Camera Dysfunction Detection

The main purpose of detecting camera dysfunctions is to assure that reliable video is being recorded. The problem of camera sabotage detection is mentioned in the literature briefly. Only a few methods for detecting camera tampering were proposed [1][2][3][4]. This task is difficult as the problem itself is hard to define. As it was mentioned in the introduction, this algorithm is to be integrated as a part of a larger smart surveillance system. Typically, the first operations in such systems include background subtraction and object detection. Hence, during tampering detection a background model is being used, utilizing Gaussian Mixture Model based method [5]. Such model becomes the template to compare the analyzed frame to during processing. Detailed description of detecting each tampering event is presented in the following sections. The detection process is performed by three separate modules called in a certain order. In the literature [3] it is shown that the detection of camera occlusion should be performed first, then camera focus loss, and finally the scene shift detection. Our experiments show that the focus analysis of already displaced camera leads to errors, therefore the sequence of these modules is changed. First the camera obstruction is detected, then displacement and finally focus loss.

2.1 Obstruction Detection

To assure that the observed area is visible at all times a module for detecting camera occlusions is used. It utilizes the information from a grey scaled analyzed frame and the corresponding background model. Misting over the camera view causes the entropy of the image to be reduced. This assumption stays true as long as the obstructing object is placed close to the camera or directly on the cameras lens, resulting in lower light exposure (Fig. 1). In such situations entropy of the image (1) reduces dramatically.

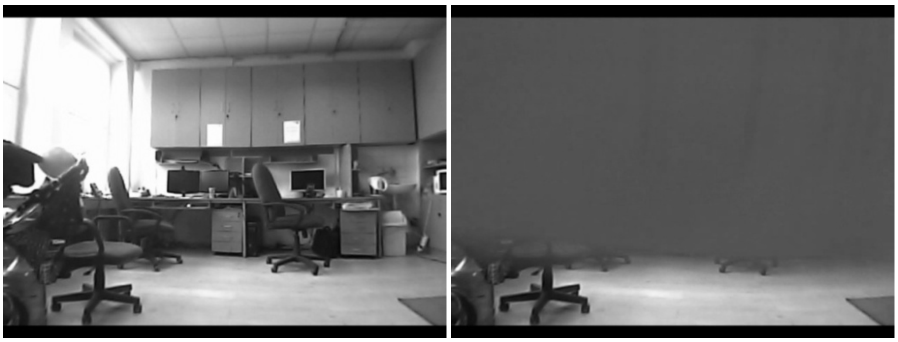

Fig. 1. Sample camera view under normal conditions (left) and after camera occlusion (right)

$$E = -\sum_{K} P(I_k) log_2[P(I_k)] \tag{1}$$

where: $P(I_k)$ is the probability of k-level pixel intensity.

The camera occlusion detection occurs by comparing the entropy of currently analyzed frame and the background model, to a defined sensitivity related factor (2):

$$Occlusion = \begin{cases} true, & \dfrac{E_{frame}}{E_{background}} < \propto \\ false, & \dfrac{E_{frame}}{E_{background}} > \propto \end{cases} \tag{2}$$

where: α is the detection sensitivity,
E_{frame} denotes the entropy of current image entropy,
$E_{background}$ is the entropy of background model built and updated according to the literature description [5].

This condition allows detecting only a specified set of situations. To extend the detected cases additional condition based on the comparison of both images histograms is used. If this difference in the meaning of Bhattacharyya distance continues for a set number of frames, camera obstruction alert is triggered despite high frame entropy. This situation can occur if an object is placed in close proximity to the camera.

2.2 Displacement Detection

Moving the camera from its original position can result in a loss of vital information. An example of a shifted camera is shown in the Fig. 2. For the purpose of camera translation detection a popular block matching method is used. Although it was developed for motion estimation in video compression field, it performs well for this application.

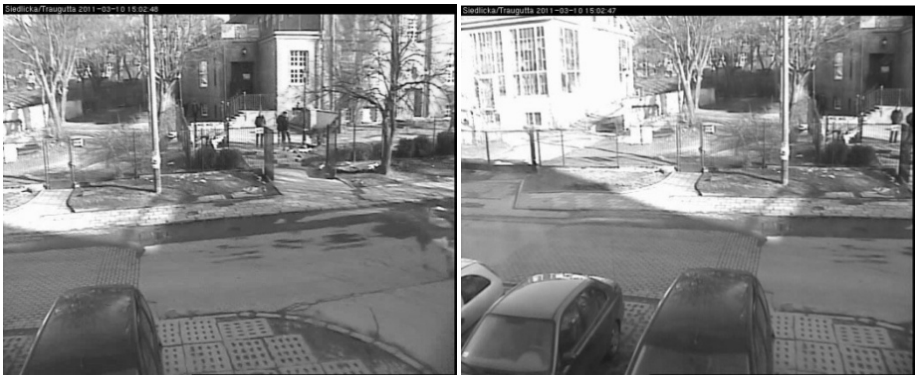
Siedlicka/Traugutta 2011-03-10 15:02:48
Siedlicka/Traugutta 2011-03-10 15:02:47

Fig. 2. Originally oriented camera view (left) and the view after camera shift (right)

There are several shift detection algorithms beside the block matching method. As it is required to operate in real-time system a compromise between the computational time and acquired accuracy has to be reached [6]. Therefore a slightly simplified Three Step Search (TTS) algorithm is implemented fulfilling this assumption. The procedure describing the algorithm is presented in Fig. 3.

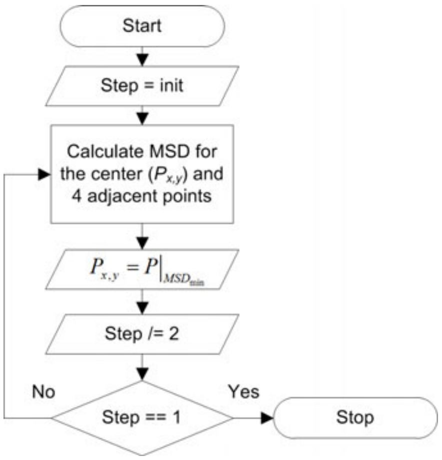
Start
Step = init
Calculate MSD for the center ($P_{x,y}$) and 4 adjacent points
$P_{x,y} = P|_{MSD_{min}}$
Step /= 2
No
Yes
Step == 1
Stop

Fig. 3. Three Step Search algorithm calculation procedure. As the result image shift estimation is acquired.

As the minimization criteria the Minimum Squared Difference (MSD) (3) is estimated in each block-matching step.

$$MSD = \frac{1}{XY}\sum_{i}^{X}\sum_{j}^{Y}\left(\mathrm{I}(i,j) - \mathrm{B}(i,j)\right)^2 \tag{3}$$

where: *I(i,j)* is the gray scaled analyzed frame,
B(i,j) depicts the background image.

Using this approach currently processed image is compared with the scene background model. To reduce the image shift calculation time further the analyzed video frame is downscaled. If cameras within the system are not equipped with hardware or software stabilization, a small tolerable transpositions of the observed scene can occur. Therefore, before triggering camera displacement alert, a threshold for minimal detected shift is applied (4):

$$Displacement = \begin{cases} true, & P_{x,y} > th_{min} \\ false, & P_{x,y} < th_{min} \end{cases} \tag{4}$$

where: $P_{x,y}$ denotes the estimated analyzed frame displacement
th_{min} is the minimum shift threshold.

This approach allows detecting camera shifts assuming low and medium object movement in the scene. If the camera observes a small area, the visible objects can cover a greater part of the view resulting in errors. Therefore, as well as for two other modules, time averaging is applied reducing the false positive errors.

2.3 Defocus Detection

In surveillance systems focus loss may occur in various situations. The least possible one is caused by an electro-mechanic failure in the camera itself. More often it may be a result of the system operator error or by atmospheric conditions such as fog. Focused image contains relevant details allowing people and vehicle identification. A camera observing an entrance gate is presented in Fig. 4, where the vehicle license plate becomes unreadable if the camera is out of focus.

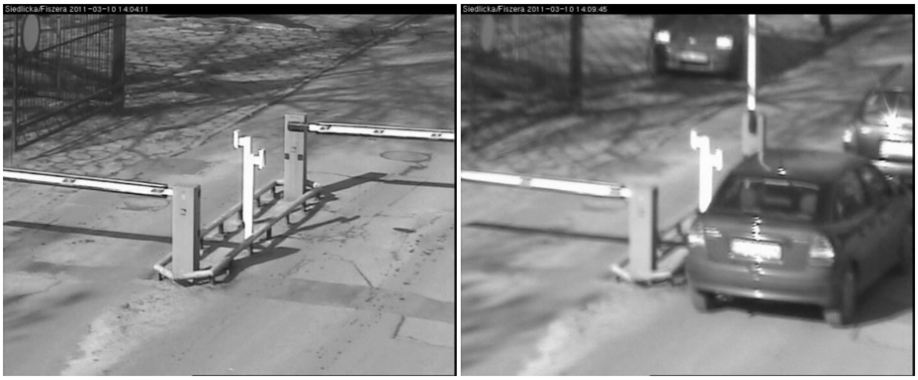
Siedlicka/Fiszera 2011-03-10 14:04:11
Siedlicka/Fiszera 2011-03-10 14:09:45

Fig. 4. Illustration of the focus loss problem. The defocused image (right) shows the lack of detailed information making it impossible to read the license plate.

The method utilized for focus loss detection in this work is based on a slightly modified solution proposed in the literature [3], as the original version of this algorithm was proven to be quite robust for variously changing light conditions. For the purpose of processing, input RGB image is being reduced to its gray scale representation. Edges present in the image are detected utilizing Sobel operator and the Canny algorithm. The result acquired in this way is accumulated using the weighted sum (5):

$$B_{edges} = \mu \cdot I_{edges} + (1 - \mu) \cdot B_{edges} \tag{5}$$

where: B_{edges} represents the accumulated edges image,
I_{edges} are the edges of the currently processed frame,
μ is the learning rate factor.

The acquired image contains a set of edges, but after thresholding only "strong edges" are present. This process is presented in Fig. 5. As the camera losses focus, gradients in the process image decrease simultaneously, especially for the strong edge parts of the image.

In the literature [3] a measure of focus based on a gradient energy along these edges is proposed. As the activity in the observed scene rises, a large amount of the edges can be covered resulting in a non-reliable energy estimation. It is noticed that even in case of highly defocused images a set of edges can be detected. Therefore, in this work the gradient energy of a common part of frame edges and strong edges is calculated (6):

$$G = \frac{\sum_i^X \sum_j^Y S(i,j) \cdot C(i,j)}{\sum_i^X \sum_j^Y I(i,j) \cdot C(i,j)} \tag{6}$$

where: *I(i,j)* is the input gray scaled image,
S(i,j) is the same image after applying Sobel operators,
C(i,j) denotes the common part of the strong edges and the edges in currently processed frame.

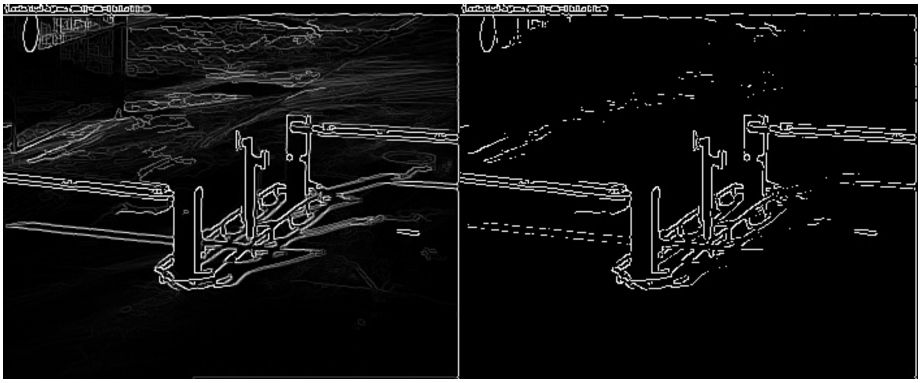
Fig. 5. Examples depicting averaged image edges and the result of applying a threshold

The normalized gradient measure is calculated for the processed frame and the background image. By comparing these values with a set sensitivity, the level of the camera focus loss can be detected (7):

$$Defocus = \begin{cases} true, & \frac{G_{frame}}{G_{background}} < \beta \\ false, & \frac{G_{frame}}{G_{background}} > \beta \end{cases} \tag{7}$$

where: G_{frame} and $G_{background}$ are the normalized gradient energy for the analyzed frame and the background image,
β denotes the detection sensitivity.

Before any detection occurs, strong edges need to be learnt. Therefore the detection process in this module is started after at least 1/μ frames (μ is the learning rate factor). Similarly to the previous detection algorithms, the results are averaged over a set number of frames before an alert is triggered.

3 Experiments

Two parameters for each tampering event were evaluated during experiments - the algorithm accuracy for a set of prepared recordings and the average calculation time. Algorithms were run using a regular PC with 2GB of RAM memory and 2.4GHz dual core processor. Approximately 6 hours of recordings were processed. The test recordings included scenes at different camera viewpoints shown in Fig. 6. Various weather and variable lightning conditions were considered as well. The original recordings resolution (704x576) was reduced for the purpose of processing to 352x288. During the recordings preparation camera focus as well as the white balance were set manually. It is important to keep these settings at a set value so they do not vary resulting in unwanted image changes.

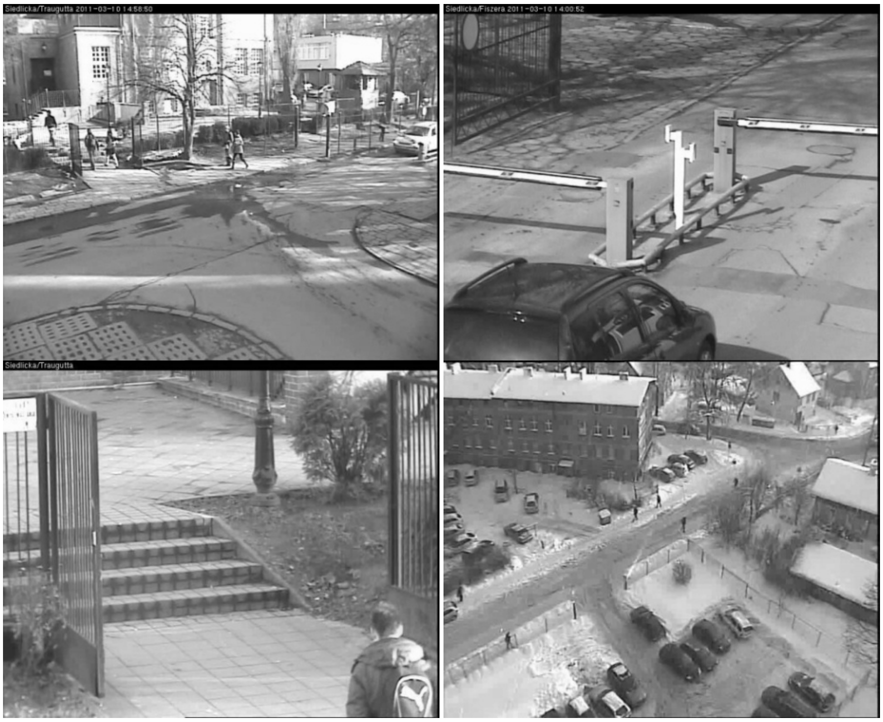
Siedlicka/Traugutta 2011-03-10 14:58:50
Siedlicka/Fiszera 2011-03-10 14:00:52
Siedlicka/Traugutta

Fig. 6. Sample frames of the test recordings presenting various camera viewpoints

All three modules were tested using same settings for all the prepared recordings. As to the focus loss and occlusion detection the sensitivity parameter was set to 0.9. Regarding displacement detection, the minimum shift was set to 4 pixels. As it was briefly mentioned in the module description, the detections results are averaged. The time window corresponding to 4 seconds of recording is utilized. Hence, the detection alert is expected to be slightly delayed. The evaluation results are presented in Tab. 1.

Table 1. The results of modules efficiency evaluation

Tampering event	**True detections**	**False alarms**	**Calc. time [ms]**
Occlusion	12/16	2	0,55 ± 0,03
Displacement	20/26	5	11,20 ± 0,58
Defocus	20/22	12	6,33 ± 0,25

The most time-consuming module is the camera displacement detector, although a simplified version of TTS is utilized. Still the overall algorithms calculation time analysis show that the presented approach is possible of being applied along with other complex video processing methods utilized in smart surveillance systems.

As to the detection accuracy, a lot of false alarms are present. Displacement detector as well as the focus loss detector seem to fail often for closed-up views. The shift detection errors manifest especially when a great part of the view is occupied by a moving object. This situation often occurs at the vehicle entrance gate. This problem can be dealt with by extending the average time window length. Moreover, the algorithm can fail if the shifted camera view has no significant common parts with the original scene. In such a case, depending on the view, defocus or occlusion detection may occur. For a precise camera displacement detection in this situation, further research needs to be done to solve the problem. Omitting close-up views in the testing set results in a decrease in false alarms, what makes the result more acceptable. Focus loss detection algorithm proposed in the literature turns out to be more quite dependent to the light changes variations. Hence, a high count of false alarms occurs during the mentioned algorithm testing.

4 Conclusions

A set of methods for camera tampering events detection was presented in the paper. Three processing modules were introduced for camera focus loss, displacement and view occlusion detection. As the conducted experiments show the algorithms are suitable of being used in real time processing with a large margin. Therefore, their application as a part of a complex smart surveillance system is possible. Unfortunately the system accuracy needs to be improved further before being applied in real monitoring systems, especially regarding still too high false positive error rate. Various approaches for close ups and distance views should be introduced in the future for better performance.

Acknowledgements. Research funded within the project No. POIG.02.03.03-00-008/08, entitled “MAYDAY EURO 2012 – The supercomputer platform of context-dependent analysis of multimedia data streams for identifying specified objects or safety threads”. The project is subsidized by the European regional development fund and by the Polish State budget.

References

1. Ribnick, E., Atev, S., Masoud, O., Papanikolopoulos, N., Voyles, R.: Real-Time Detection of Camera Tampering. In: IEEE International Conference on Video and Signal Based Surveillance, Australia (November 2006) ISBN: 0-7695-2688-8
2. Gil-Jiménez, P., López-Sastre, R., Siegmann, P., Acevedo-Rodríguez, J., Maldonado-Bascón, S.: Automatic Control of Video Surveillance Camera Sabotage. In: 2nd International work-conference on Nature Insired Problem-Solving Methods in Knowledge Engineering: Interplay Between Natural and Artificial Computation, Spain (June 2007) ISBN: 978-3-540-73054-5

3. Harasse, S., Bonnaud, L., Caplier, A., Desvignes, M.: Automated camera dysfunctions detection. In: 6th IEEE Southwest Symposium on Image Analysis and Interpretation, USA (March 2004) ISBN: 0-7803-8387-7
4. Sağlam, A., Temizel, A.: Real-time Adaptive Camera Tamper Detection for Video Surveillance. In: 6th IEEE International Conference on Advanced Video and Signal Based Surveillance, Italy (September 2009) ISBN: 978-0-7695-3718-4
5. Czyżewski, A., Dalka, P.: Moving object detection and tracking for the purpose of multimodal surveillance system in urban areas. In: Proc. 1st Int. Symp. on Intell. Interactive Multim. Syst. and Services, Piraeus, Greece (2008)
6. Turage, D., Alkanhal, M.: Search Algorithms for Block-Matching in Motion Estimation, Mid-Term project (1998), `http://www.ece.cmu.edu/~ee899/project/deepak_mid.htm`