Modeling of cylindrical scene in CCTV systems with motorized camera

Adam Dąbrowski, Paweł Pawłowski, Jan Kurpisz, Mateusz Stankiewicz, Agnieszka Krzykowska
Division of Electronic Systems and Signal Processing, Chair of Control and Systems Engineering
Poznań University of Technology
Poznań, Poland
Adam.Dabrowski@put.poznan.pl

Abstract: **One of the main tasks of monitoring service operators is the video data inspection in order to search for interesting events. Assuming a CCTV system equipped with a motorized camera operating in a pan cycle (a quite typical situation), we propose a new visualization and event search method consisting: first, in generation of a cylindrical (panorama) scene model, and second, in creation of video sequences containing the selected panorama parts for a convenient automatic and/or manual video analysis.**

Keywords: CCTV; PTZ camera; cylindrical scene; panorama

I. INTRODUCTION

Monitoring systems are used in almost all cities and many premises. They strongly support critical urban services like police, security services, crisis management groups, etc. Unfortunately, due to many operational tasks, there are tight limits concerning management and analysis of the recorded video materials. It is nothing uncommon that many events, which are categorized in lower layers of hierarchy, are omitted. This is still emphasized by time limits related to finite video storage space. Typically used PTZ (pan-tilt-zoom) cameras, operating in cycles, offer large coverage of the monitored area, but drastically hamper the analysis of a fixed place.

We propose a new method that significantly facilitates searching for a given event in a selected area, if the video material was recorded with a rotating PTZ camera. The method consists: first, in generation of a cylindrical scene model (panorama model), and then, in creation of a video sequence (sequences) containing the selected panorama part (parts) for convenient automatic or manual video analysis (Fig. 1).

Let us consider a typical example: a CCTV console operator should find an event that happened in the known place but in an unknown time, e.g., during one day. A full fast-forwarding (FF) analysis of the video recorded with a PTZ camera (pan cycle mode of, e.g., 360^0 per minute) would be a very time consuming task. In out method, the operator merely selects the area, then the software calculates panoramas and finally produces a video sequence (Fig. 1). The operator can watch 2.4 seconds per hour of the 25 fps video sequence. The achieved speed is impossible to reach by the solutions like FF.

The presented idea, although is very helpful to reach fast (faster than the real-time) and accurate processing, requires cautiously selected and precisely optimized video analysis algorithms.

Figure 1. Video sequence frames created of consecutive panoramas

II. STATE OF THE ART

Panorama making and stitching algorithms are extensively studied and published. The known algorithms use mainly feature-based techniques [1, 2] and direct solutions [3].

However, generation of a cylindrical panoramic image with a rotating camera is not an exhaustively investigated task, discussed e.g. in [4, 5]. A more precise model is spherical or conical, but they are much more complicated during the video processing. With zoom (the last one of the PTZ functionalities) it is possible to achieve another dimension of the analysis – the depth [6].

Making of panoramas is burdened by many artifacts. Some solutions to deal with them are proposed in [7, 8].

Our solution with the panorama processing makes it possible to: first, fast search for the events, and second, efficiently compress data for a long period storage.

III. TOOLS AND ALGORITHMS

In order to be implemented in a CCTV monitoring system, the application for creating panorama images has to meet certain requirements. First, the algorithm should be robust enough to create panoramas in a changing environment with moving objects such as cars and pedestrians. Furthermore, the created images have to be similar enough to allow the analysis and comparison of subsequent panoramas. Finally, for practical reasons, the algorithm should function in the real-time to be efficient for monitoring systems. This section discusses tools

and algorithms, which we used to solve these problems and implemented in the prepared software. The algorithm workflow for a panorama creation, on which our program is based, is shown in Fig. 2.

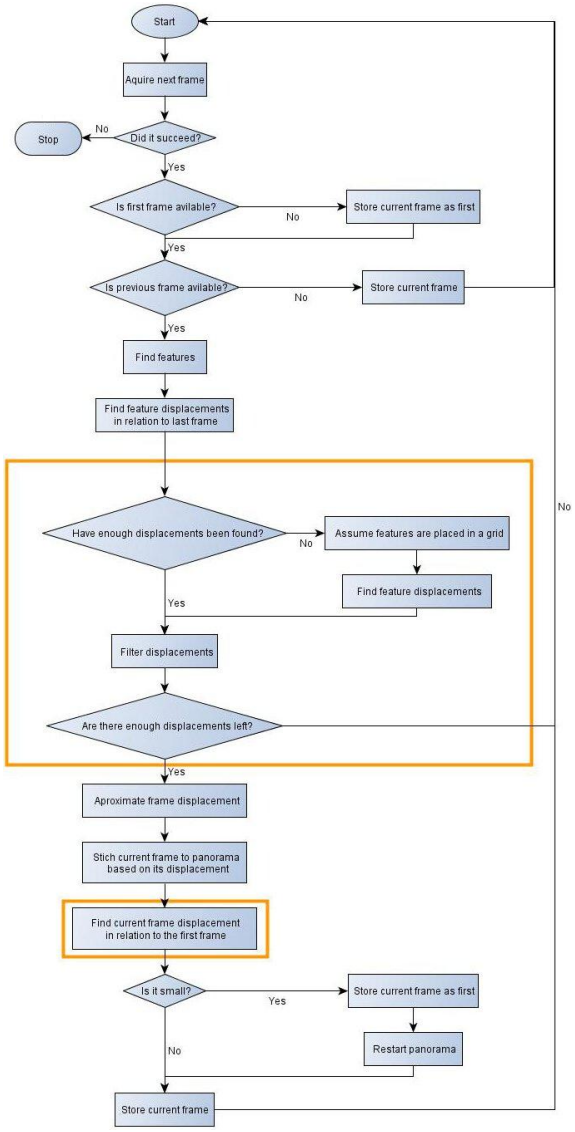

Figure 2. General scheme of algorithm

A. *Homography*

Homography, or a projective transformation, is a mathematical operation for image transformation, using perspective projections onto a different plane [9]. While using this method the size or angles are not preserved but incidence and cross-ratio stay the same. Thanks to this property, the lines that were straight in the original image, stay straight while projected into the panorama. Thus, it is possible to very precisely fit the consecutive frames. However, while using this method, the scaling error accumulates. Hence, it is not an ideal, but, nevertheless, a very efficient technique for creating panoramas.

B. *Background estimation and movement detection*

This subsection describes methods that are used in the developed software to solve two basic problems: background estimation and movement detection. Their effectiveness is crucial for proper functioning of the whole algorithm. This is because if a feature point (defined below) is chosen incorrectly (e.g., on a moving object) the fitting point will also be incorrect.

1) *Background subtraction*

One of the basic methods for background modeling is subtraction of current frame from the reference image. For places where the difference is bigger than the surroundings we can assume that there is a moving object in that position. It is possible to use this technique efficiently in case where the background is known earlier and does not change. The main disadvantage of this method is that objects, which do not differ much from the background, would not be properly detected.

2) *Background modeling*

In situations where the background is not known or changes in time, we use adaptive background modeling. First we take a frame as the background reference and then, in next steps, we mix the following frames with the last reference image. The main advantage is that the background model adapts to the changing conditions. However, objects, that remain stationary for a long time, will be classified as the background as well.

3) *Subsequent frames subtraction*

Another method for movement detection is subtracting next frames from each other. This technique does not require to create background model and is in some degree resistant from changing lightning conditions. The flaw is that only moving objects are detected. What is more, only a fragment of a moving object can be detected.

4) *Optical flow*

Optical flow method detects movement of pixels in successive frames. It works very well for estimating of movement between frames, although it is hard to detect moving objects.

C. *Image stitching using feature points*

Feature point is an image point that is characteristic in the sense that it can easily be distinguished from other points in its neighborhood. The most distinctive parts of an image are edges and they are usually used as good features to track. In the software application we implemented two methods of finding feature points. Both are described below.

1) *Harris & Stephens approach*

The Harris & Stephens method of finding feature points is based on two steps [8]. The first one consists in the analysis of the matrix of the second-order partial derivatives of an image. In the next step the so-called Harris matrix is created, which is the autocorrelation matrix of small parts (windows) of the image. The method classifies a point as an edge, if the eigenvalues of the Harris matrix are relatively bigger than the local surrounding.

2) Shi-Tomasi approach

This is a modification of the Harris & Stephens method. According to this algorithm, the edge is determined with eigenvalues that are above a given threshold .

D. Tracking feature points

The next step after finding and choosing good feature points is tracking them between consecutive frames in a sequence. In order to do so, we implemented three algorithms.

1) Lucas-Kanade algorithm

The purpose of the Lucas-Kanade method is to evaluate the optical flow by the feature points tracking. The following assumptions are made:

- same color of the tracked point between frames.
- relatively small movement of the tracked point.
- surrounding of the tracked point does not change between frames in sequence, thus the neighboring pixels move along with the tracked point.

The method tracks the displacement of certain pixels in order to track the features.

2) Pyramid Lucas-Kanade method

The basic Lucas-Kanade algorithm looks for the pixel displacement in a certain limited area. In order to find a longer shift the pyramid Lucas-Kanade method is used. Basically it is the same method repeated for one image in a different scale. Due to that it is significantly slower than the previous method.

3) SIFT algorithm

SIFT (scale invariant feature transform) is a complex method of comparing two images. It is designed to detect the corresponding points even if they change their size, orientation, lightness, or position. It is one of the best methods for comparing images with utilization of the feature points. Due to the method complexity, its usage in the real-time vision system must be carefully controlled.

IV. Implementation, Experiments and Artifacts

A. Implemetation

Implementation of the chosen algorithms was evaluated on Windows platform in Visual Studio C# environment with the OpenCV library. The input file for the application was a video sequence from the PTZ camera. The application creates panorama images and tracks the defined area (treaded as frames) in the sequence of panoramas.

The application consists of modules responsible for particular tasks. The main program parts are:

- Finding feature points,
- Finding feature points deviations,
- Creation of panoramas,
- Searching for a particular picture in the panorama images.

Second application was created to build a new video sequence based on a series of panorama pictures. The user selects a fragment of the first panorama image (with the 4:3 aspect ratio), which is treated as a frame. Then a new video sequence is created among all panoramas with the same feature points procedure, which is used in creation of panoramas. By this means possible displacements of the frames between panoramas are compensated. This video sequence is stored as a movie file.

B. Experiments

15 video sequences in 5 different locations were recorded for this experiment. Recordings have 320×212 resolution and were coded with XviD (MPEG4) codec with 900 kbps and 24 frames per second. Sample panorama is presented in Fig. 3.

The correct result occurs when from a sequence with n rotations, n full panoramas are made. A situation, in which more panoramas have been created than the number of the rotations were in the video sequence, is an example of incorrect detection of the beginnings of some panoramas. An opposite situation, i.e., when less panoramas have been created that the number of rotations, means that multi-panoramas were made due to lack of detection of the panorama beginning. These situations did not happen in our experiments.

Anther example of the incorrect result is an incomplete panorama and panorama that starts in a different location than the ones previously created.

For test recordings the application had 89 % efficiency. The computer used for tests was equipped with Intel Core 2 Duo E6750 (2.66GHz) processor, DDR2 2GB 800MHz) memory and 32-bit Windows 7 OS.

If we assume that rotating camera is the center of the coordinate system it can be said that whole scene rotates with a defined rotational speed. Though different stationary object in the scene have different linear speed (dependent from its distance to the camera), rotational speed of physical points in the scene corresponding to the feature points in the picture are the same (with an assumption that no moving objects occur). Calculation of displacement of feature points between frames is not equal (especially near edges) because of camera distortion. If feature points are located evenly in the frame, errors of their displacement will reduce during averaging. The distortion can be removed before calculation of these shifts, but this operation does not have a positive impact on finding the feature points in a picture.

C. Artifacts

Making of panoramas using the described method is burdened by some artifacts. We are presenting them below.

1) Contraction of moving objects

Objects that move in an opposite direction to the camera rotation are shown in cramped shape in the panorama image. An example of this phenomenon is presented in Fig. 4 (left). If the speed of rotation of the camera and the distance (radius) between the object and the camera are given, an approximate speed of the moving object can be computed from (1).

Figure 3. Panorama image

$$\omega_o = (w_o/w_{or} - 1) * \omega_c, \quad (1)$$

$$\upsilon_o = \omega_o * r. \quad (2)$$

where w_o stands for the width (in pixels) of the object in a panorama image, w_{or} is the width of the object in the video sequence, ω_o is a rotational speed of the object, ω_c is a rotational speed of camera, υ_o is a linear velocity of the object and r is a distance form the object to the camera. Equations (1) and (2) can be used if the distance between the object and the camera does not change significantly during the video registration and its linear velocity is constant. The object will not be preserved in the panorama if it moves with higher velocity then the camera rotational speed and in the opposite direction.

2) Extension of moving objects

Another phenomenon is extension of moving objects. It occurs when the object moves in the same direction as the camera but with lower velocity. An example of extension of a moving object is show in Fig. 4 (right). To define approximate velocity of the object expression (3) can be used.

$$\omega_o = \omega_c * (1 - w_{or}/w_o) \quad (3)$$

Figure 4. Contraction (left) and extension (right) of moving objects

Accuracy of these computations will be lower because the radius between the camera and the object will change during the recording (excluding a situation of the object moving in a circle with the center in the camera position).

3) Inversion of moving objects

If the object is moving in the same direction as the camera but with higher velocity it will be reversed in the panorama image as illustrated in Fig. 5. Based on the inverted image of the object its approximate velocity can be computed from (4):

$$\omega_o = \omega_c * (1 + w_{or}/w_o). \quad (4)$$

An interesting fact is that the object is not directly inversed, but its fragments are preserved in the reversed order. An algorithm that was used to find the shifts between frames does not take movements of objects into account. As a result the displacement of feature points in a moving object is also computed. However, even if the moving objects are correctly determined, the shift between subsequent frames may be not.

Figure 5. Inversion of moving objects

V. Summary and Future Work

The presented method of video processing using generation of panoramas and creation of a new time-zoom video sequence shows usefulness and new possibilities of CCTV data analysis. Having present experience [10], we plan to get further improvement of the video processing speed using GPGPU solutions (general-purpose processing by graphics processing units)

Beside the video processing, the presented solution can also be used for a high and unconventional video compression. This subject will be undertaken in a future work. A compression of consecutive panoramas using the known intra and inter frame processing (like in H.264) can bring very promising results.

References

[1] M. Brown, D.G. Lowe, Recognizing Panoramas, Proc. of the 9th International Conference on Computer Vision, pp. 1218-1225, 2003.

[2] M. Brown, D. G. Lowe, Automatic Panoramic Image Stitching using Invariant Features. IJCV(74), No. 1, pp. 59-73, Aug. 2007,

[3] M. Irani, P. Anandan, About Direct Methods. In Vision Algorithms: Theory and Practice, pp. 267-277, 1999

[4] F. Huang, R. Klette and K. Scheibe. Panoramic Imaging: Sensor-Line, Cameras and Laser Range-Finders. Wiley, West Sussex, England, 2008

[5] K.M. Yang; F. Huang; S.H. Lin, Generation of animated panorama from single video sequence, 3rd International Congress on Image and Signal Processing (CISP), vol. 1, pp. 477 – 481, 2010

[6] S.K. Wei; F. Huang; R. Klette, The design of a stereo panorama camera for scenes of dynamic range, Proc. of 16th Int. Conf. on Pattern Recognition, Vol. 3, pp. 635 – 638, 2002

[7] Y. Wan; Z. Miao, Automatic panorama image mosaic and ghost eliminating, Int. Conf. on Multimedia & Expo, pp. 945 – 948, 2008

[8] Z. Xiao-chun, H. Ming-yi, Z. Xin-bo, F. Yan, A robust mosaic panorama technique for video, 2nd Int. Conf. on Computer Engineering and Technology (ICCET), Vol. 2, pp. V2-641 – V2-644, 2010

[9] M. Lourakis; A C/C++ Library for Robust, Non-linear Homography Estimation; Institute of Computer Science, Foundation for Research and Technology - Hellas, Heraklion, Crete, Greece, 2009

[10] F. Misiorek, M. Stankiewicz, P. Pawłowski, A. Dąbrowski, Efficient use of graphics cards in implementation of parallel image processing algorithms, Int. Conf. Mixed Design of Integrated Circuits and Systems MIXDES, p. 152, 2011

The paper is prepared within the INDECT and DS projects.