Proceedings of the Fifth IIEEJ International Workshop
on Image Electronics and Visual Computing 2017
Da Nang, Vietnam, February 28 - March 3, 2017

EFFICIENT IMPLEMENTATION OF TOP-VIEW SURVEILLANCE SYSTEM USING MULTIPLE CAMERAS

Kohei Okahara† Tsukasa Fukasawa† Ichiro Furuki† Hironobu Abe†
† Information Technology R&D Center, Mitsubishi Electric Corporation
5-1-1 Ofuna, Kamakura, Kanagawa 247-8501, Japan

ABSTRACT

In this paper, we propose a surveillance system called 'Fairyview' which generates a wide area top-view image from multiple camera images. Image stitching technology is often used in applications that require a wide field of view such as surround view. However, surround view is not for surveillance: it doesn't consider the continuity of moving objects at the boundary. Creating a seamlessly stitched top-view image from multiple camera images requires different processing corresponding to moving objects and the background image; displaying a stitched image in interactive frame rates is also an important factor as a practical surveillance system. We describe how to create a continuous top-view image and how we have implemented the entire pipeline. In the case study presented in this paper, our GPU-based prototype system creates a continuous top-view image from two SXVGA camera input images at 15.2 fps which is 6.91x faster than that of CPU-based prototype.

1. INTRODUCTION

The number of surveillance cameras installed in the public places has been increasing with the increasing public interest in safety and security. However, there is no major change in the way of monitoring their images, which are jumbled and inconsistent on a display. In such a case, image stitching technology used in a surround view system [1][2] or a panoramic view system [3][4] is useful. Providing a wide area view in one image, these systems enable users to grasp the surrounding situation quickly. However, these systems are not for monitoring people: they don't consider the continuity of moving objects (such as pedestrians) at the boundary.

In this paper, we propose a novel surveillance system which generates a wide area top-view image from multiple surveillance camera images. We call the system 'Fairyview'. Moreover, we describe our implementation of a continuous top-view image stitching pipeline. In a top-view image generation system such as surround view [1][2], each camera is calibrated at the height of the ground. Therefore, images of objects above the ground are extended after viewpoint conversion. In case camera input frames are applied simple viewpoint conversion, stitching and blending, double image or missing of objects appears at the boundary. In order to make a continuous top-view image, additional processing on foreground moving objects is required. Nevertheless, display frame rate is also an important factor as a practical surveillance system. We developed a continuous top-view image stitching pipeline for interactive frame rates and implemented a prototype system of Fairyview using a graphics processing unit (GPU).

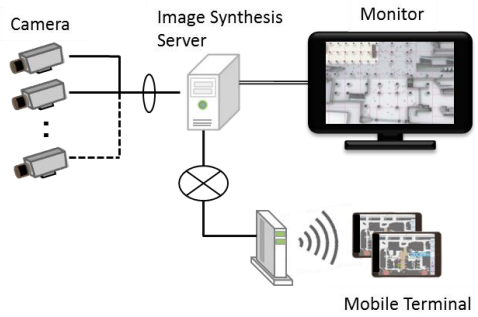
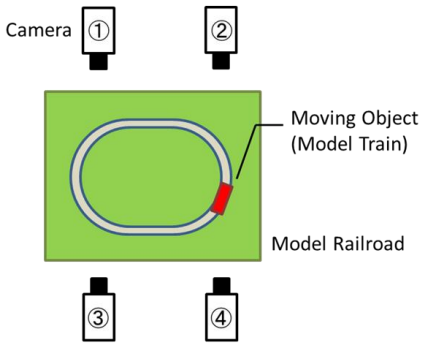


Fig. 1. The configuration of Fairyview system

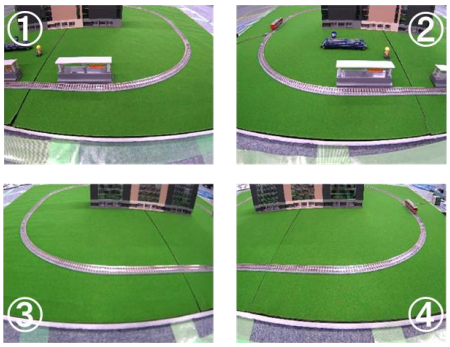
2. FAIRYVIEW SYSTEM

Figure 1 shows the concept of the proposed system, Fairyview. Two or more cameras are connected to an image synthesis server and each camera input image is converted to a seamlessly stitched top-view composite image. The created top-view image is displayed on a monitor or mobile terminals via network.

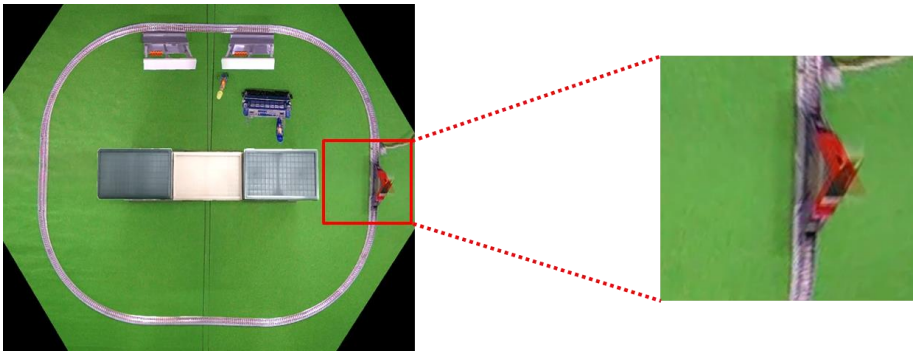
Figure 2 show an example of stitched top-view image from four surveillance camera input images and individual camera input images. In this example, two cameras are disposed in the same direction and the others are in the opposite direction. Presenting as a continuous top-view image, Fairyview system enables users to grasp the opposite side of the building at once which cannot be seen in individual camera image. As shown in Figure 2(c), objects above the ground are extended and it causes double image or missing of objects at the boundary. In Fairyview system, we solve this problem by dividing the image stitching process into foreground processing and background processing.



(a) Fairyview system setup



(b) Individual camera images



(c) Stitched top-view image and missing of model train at the boundary

Fig. 2. Fairyview system prototype

3. CONTINUOUS TOP-VIEW IMAGE STITCHING

Due to different camera installation direction, moving objects in individual camera image are extended after viewpoint conversion, and it causes double image or missing of objects at the camera boundary in the stitched top-view image. Figure 3 shows the proposed stitching pipeline to create a continuous top-view image. In this pipeline, moving objects are extracted from individual camera images, and the same object's images are refined into one when attached onto a background composite image.

3.1. Cam Reader module

Cam Reader module is responsible for capturing packets and decoding camera images. We use network surveillance cameras supporting 30fps MJPEG stream at a resolution of SXVGA (1280×960).

3.2. Background Subtractor module

In order to process foreground images such as moving objects and background image separately, Background Subtractor module extracts moving objects from decoded camera images. We use the Gaussian mixture model based background subtraction algorithm [5]. Finally, foreground images, foreground masks and background images of each camera are created in this module.

3.3. Background Inpaintor module

Subtracted background images have foreground defective regions. In Background Inpainter module, these regions are filled with several previous camera images and foreground masks preserved in memory.

3.4. Background Synthesizer module

Background Synthesizer module is responsible for warping and stitching each background image generated in upper modules. In this module, we use a reference table which has corresponding pixel data from camera input image to top-view composite image. The reference table is created in the initialization process using camera calibration data.

3.5. Foreground Selector module

In order to create a continuous composite image, the same foreground images captured in multiple cameras are refined into one image. An extension rate of moving object's image after viewpoint conversion depends on the distance between camera and moving object. For this reason, distances between each camera and moving object are calculated using camera parameters, and the image of the camera which has the shortest distance is selected in Foreground Selector module.

3.6. Foreground Warper and Stitcher module

A separately created foreground image has to be attached on background composite image created in Background Synthesizer module. In Foreground Warper module, the image of the moving object is warped based on camera parameters. Moreover, the warped image is adjusted to an appropriate size in case of a high extension rate warping. After warping and adjusting, the foreground image is attached onto the contact point of the moving object and the projection plane.

Figure 4 show the visual difference at the boundary between the result of simple stitching and the result of proposed pipeline. Simple stitching caused doubled image of model train at the boundary (Figure 4 (a)). On the other hand, proposed pipeline generated a clear model train image at the boundary (Figure 4 (b)). From this result, our proposed pipeline is found to be effective for the prevention of double image or missing at the boundary.

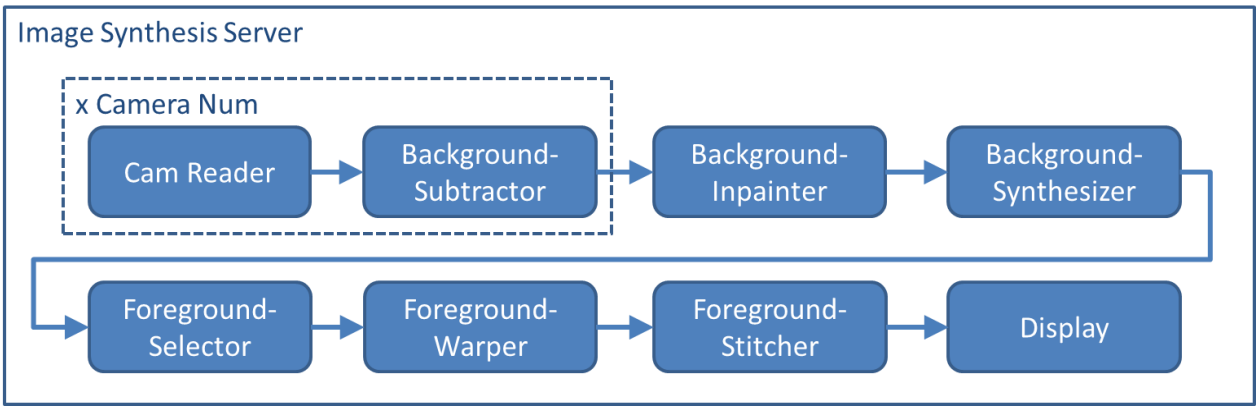
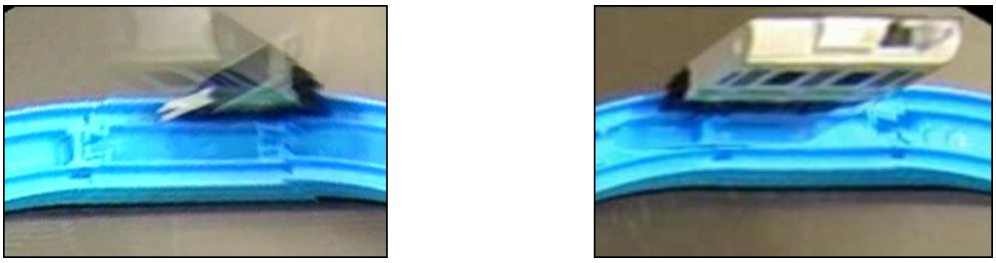
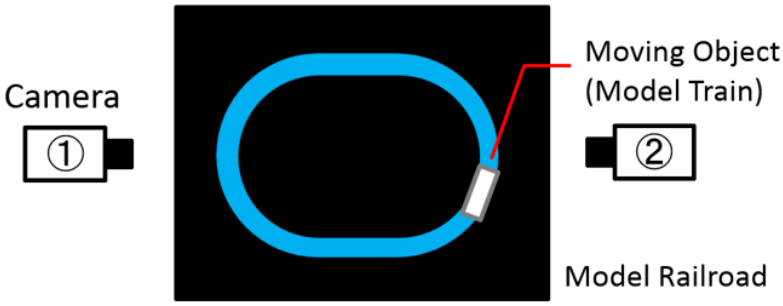


Fig. 3. Continuous top-view image stitching pipeline

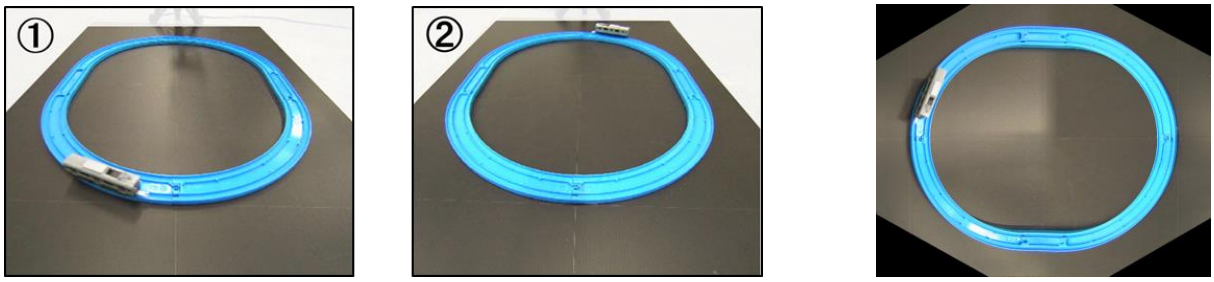


(a) Without foreground processing (b) With foreground processing

Fig. 4. Visual difference at the boundary between with and without foreground processing



(a) System setup with two cameras



(b) Individual camera images (c) Stitched top-view image

Fig. 5. Fairyview system for performance mesurement

3.7. Pipeline Performance

We measured our pipeline performance above to check it can be executable in interactive frame rates. Our test machine of Fairyview system use an Intel Core i7-3770 (4-core) and NVIDIA GeForce GTX 750 Ti GPU. In this paper, our measurement environment is composed of two SXVGA cameras and one moving model train. We implemented the first prototype as CPU-based except Background Synthesizer module. Figure 5 show the measurement environment of Fairyview system. Two cameras are disposed in the opposite direction (Figure 5 (a)) and a model train runs on the railroad line with constant speed. Figure 5 (b) show the individual camera images and Figure 5 (c) shows the stitched top-view image from two camera images.

Table 1 shows the pipeline performance of our first prototype. This result is an average processing time and frame rate for a hundred frames of SXVGA camera input. In general, it is said that more than five frame rate is needed for video surveillance. From this result, we found that we had to improve our pipeline performance drastically for a practical surveillance system.

Table. 1. Average execution time (ms) of continuous top-view image stitching pipeline

	TIME	FPS
CPU-based (Intel Core i7-3770)	459.1	2.2

4. ACCELERATION FOR INTERACTIVE FRAME RATES

In this section, we describe how we have improved the pipeline performance. Figure 6 shows the new proposed pipeline to create a continuous top-view image in interactive frame rates. We applied GPU module to each component and parallelize independent processes such as foreground processing and background processing.

4.1. Using GPU modules

GPU-based co-processing is compatible with image processing, and GPU-based image processing is usually faster than that of CPU-based. Actually, GPU-based processing is used in real-time panorama system [4] [6]. In order to check GPU-based processing is effective to improve our pipeline performance, we measured the difference in performance of principal image processing function between with and without GPU module.

Table 2 shows the measurement results of execution time. This result is an average processing time for a hundred SXVGA camera input frames. From this result, we found that applying GPU module improves the performance of each component drastically. We applied GPU modules to entire pipeline except Cam Reader module and Display module shown in Figure 3, and measured the pipeline performance. The measurement environment is the same as we described in section 3. Table 3 shows the entire pipeline performance after applying GPU modules. As a result of applying GPU modules, the pipeline performance is improved by about six compared with CPU-based pipeline.

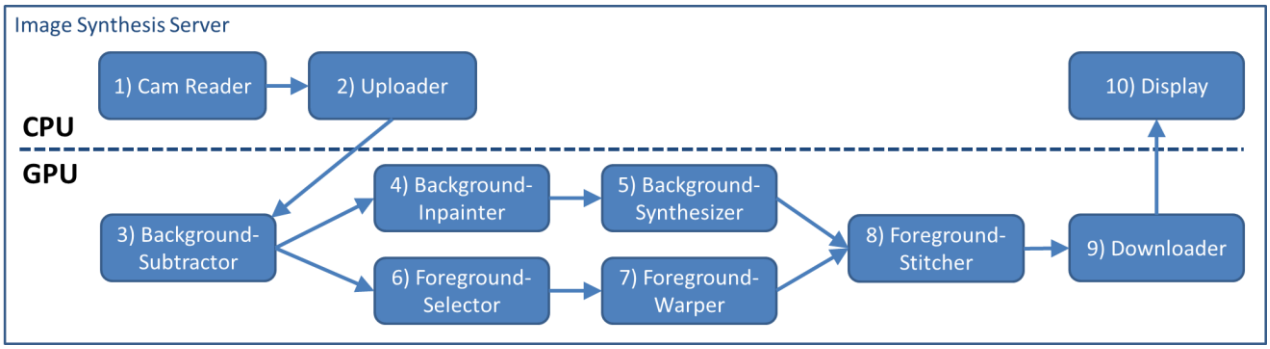


Fig. 6. GPU-based pipeline for interactive frame rates

Table. 2. Average execution time (ms) of principal image processing function with and without GPU module

Function [Module name used in]	CPU	GPU
Distortion Correction [3) Background Subtractor]	12.6	3.6
Background Subtraction (Mixture of Gaussian) [3) Background Subtractor]	39.3	3.0
Homography Transformation [7) Foreground Warper]	5.2	1.0

4.2. Parallelizing Independent Process

In the first prototype shown in Figure 3, we implemented the entire pipeline sequentially. However, processing to background images and foreground images or processing to each camera image can be executed simultaneously. Moreover, the number of cameras connected to the image synthesis server will increase in the future. In order to prepare for such an increase of calculation cost, we tried to parallelize the entire pipeline as shown in Figure 6. In parallelization, the processing to each camera image will be synchronized at the end of each module, and the processing to foreground and background will be synchronized before Foreground Stitcher module.

Table. 3. Average execution time (ms) of entire pipeline with GPU modules

	TIME	FPS
With GPU Module (NVIDIA GeForce GTX750 Ti)	77.1	12.9

Table 4 shows the entire pipeline performance after applying GPU modules and parallelization. As a result of parallelization the pipeline performance is improved by 11ms compared with the GPU modules applied sequential pipeline.

The display frame rate was improved from 2.2 fps to 15.2 fps by applying GPU modules and parallelizing independent process. Therefore, Fairyview system can be used as a practical surveillance system so far as stitching two camera images.

Table. 4. Average execution time (ms) of entire pipeline with GPU modules and parallelization

	TIME	FPS
With GPU Module and Parallelization (NVIDIA GeForce GTX 750 Ti GPU)	66.0	15.2

5. CONCLUSIONS

In this paper, we proposed a surveillance system called ‘Fairyview’ and described how we have implemented the entire pipeline to create continuous top-view image from multiple camera images. Moreover, we also described the pipeline performance in case of two cameras and how we have improved it. The proposed pipeline with GPU modules and parallelization creates the continuous top-view image from two camera input frames at 15.2fps which is 6.91x faster than that of CPU-based prototype.

Monitoring area covered with two cameras is not so large, therefore ongoing work is scaling Fairyview system up to a higher number of cameras and moving objects. After constructing the entire system, we will also measure the pipeline performance and verify the number of cameras available in one image synthesis server

REFERENCES

[1] B. Zhang, V. Appia, I. Pekkucuksen, A. Z. Batur, P. Shastry, S. Liu, S. Sivasankaran, K. Chitnis, and Y. Liu, “A surround view camera solution for embedded systems”, in Proc. the 10th IEEE Embedded Vision Workshop (Held in conjunction with IEEE CVPR 2014), Columbus, Ohio, June 2014.

[2] Yu, M. and Ma, G., "360° Surround View System with Parking Guidance," in Proc. SAE Int. J. Commer. Veh. 7(1):19-24, 2014.

[3] W.-S. Liao, T.-J. Hsieh, and Y.-L. Chang, “GPU parallel computing of spherical panorama video stitching,” in Proc. IEEE 18th International Conf. on Parallel and Distributed Systems, 2012.

[4] Y. Xu, Q. Zhou, L. Gong, M. Zhu, X. Ding and R. K. F. Teng, "High-Speed Simultaneous Image Distortion Correction Transformations for a Multicamera Cylindrical Panorama Real-time Video System Using FPGA," in IEEE Transactions on Circuits and Systems for Video Technology, vol. 24, no. 6, pp. 1061-1069, June 2014.

[5] P. KaewTraKulPong and R. Bowden, “An Improved Adaptive Background Mixture Model for Real-Time Tracking with Shadow Detection,” in Proc. European Workshop Advanced Video Based Surveillance Systems, 2001.

[6] M. Tennøe, E. Helgedagsrud, M. Næss, H. K. Alstad,H. K. Stensland, V. R. Gaddam, D. Johansen,C. Griwodz, and P. Halvorsen, “Efficient implementation and processing of a real-timepanorama video pipeline.” in Proc. IEEE ISM, Dec.2013.