REAL-TIME VIEW TRANSITION BETWEEN CAMERAS HAVING DIFFERENT PERSPECTIVES

[1]*Li-Wei Chang* (*張禮薇*), [1]*Kuan-Wen Chen* (*陳冠文*),
[2]*Wei-Ting Peng* (*彭維廷*), [1,2]*Yi-Ping Hung* (*洪一平*)

[1]Department of Computer Science & Information Engineering,
National Taiwan University, Taipei, Taiwan

[2]Graduate Institute of Networking and Multimedia,
National Taiwan University, Taipei, Taiwan

E-mail: hung@csie.ntu.edu.tw

ABSTRACT

This paper introduces a novel method for real-time view transition between two cameras having different perspectives. An important feature of our method is correspondences free, so that even when the viewpoints of two cameras are quite different, it still work well as long as monitoring areas of two cameras are adjacent in the real world. Using camera calibration data and a pre-constructed background 3D model, we can synthesize virtual views, and the texture for background and foreground objects are generated separately. The Experiment results have showed the reality and smoothness of synthesized view transition images and demonstrated the feasibility and real-time property of the proposed method.

1. INTRODUCTION

Multi-camera video system is popularly used, for example, it can be applied to the visual surveillance system, sports broadcast, and etc. Traditionally, multiple video streams are transmitted to the control center and shown on screens, and users can choose one to pay attention to. However, when switching the view from one camera to another, the view changes give users an uncomfortable flash, and further users can not understand the geometry and relationship between real cameras. Especially when users want to track a person from one view to another, the missing information will cause it difficult to comprehend the movement of the person. A directly perceived solution is to synthesize the view transition images between cameras.

This concept is first introduced in [6]. But there has been little work on the real-time generation of arbitrary virtual views from limited number of known views, especially when known views are captured by cameras far from each other. The basic concept of the view transition depends on view morphing [12], and there are many applications of the morphing techniques, such as virtual teleconference system [7]. These methods are usually applied on two views that are overlapped each other with large regions, and lots of correspondences are needed to find the fundamental matrix. A real-time novel view reconstruction has been addressed in [11]. This paper mainly deals with an augmented reality solution that deploys numerous cameras to monitor a large campus or an urban site, but a closer and more precise view is not allowed in its system. Another similar work is showed in [5], which copes with video synthesis problem, especially applied for the sporting videos. They use view interpolation for both the background and the foreground, but their system does not promise the real time synthesis and corresponding feature points are needed. These techniques can neither deal with both the automatically real-time view synthesis nor generate virtual images with almost different views captured by real cameras.

In this paper, we propose a new method to synthesize the viewing images of virtual cameras. When applying the immersive visualization for the surveillance system, automation and the real-time property are two important issues in our system. For the automation issue, minimum manual operations are needed, so in our method the camera calibration data and background 3D model should be known in advance and the transition images are generated automatically.

To achieve real-time frame rate, any time-consuming algorithms, such as finding feature correspondences between two real images, are not allowed. Further, there are many situations that the correspondences between two camera's views are difficult to find, for example when the viewpoints of two cameras are quite different. Therefore, we proposed a novel view transition method that is correspondence free and satisfy the automation and real-time properties.

In the next section, an overview of our system and workflow are presented. Then the algorithm about how to synthesize background and foreground transition views between real cameras is explained in the section 3. In section 4, the experiment environment and the results will be showed. Finally, we draw conclusions in the section 5.

2. OVERVIEW

In this section, we give an overview of the proposed system and explain its workflow. As shown in Figure 1, the system can be divided into three parts: 1) view interpolation for the background, and 2) view interpolation for the foreground, and 3) the composition of the background and the foreground images. The upper left block in Figure1 is the pre-processing part, and the running time of these steps is not taken into consider. In addition, the virtual viewing image is generated for the smooth transition between two real cameras, so the position of the virtual camera will be determined by interpolation between the locations of two real cameras.

In the first part, background 3D model and the camera calibration data are known; real texture images of the background 3D model are generated in advance. Once the viewpoint position is changed, its corresponding background image is rendered by background 3D model and camera calibration data.

In the second part of foreground interpolation, first, background modeling is constructed, and textures related to the foreground object will be obtained automatically from real cameras. Second, an alpha mask is estimated by foreground detection for each camera view. According to the alpha mask value, the foreground texture is pasted on the object billboard. The higher the alpha value of one pixel is, the more opaque this pixel is taken to be the foreground. After that, foreground image of each view is blended into a final one, and their weights are according to the virtual camera position.

Finally, the 3D position of the foreground object is estimated by the camera calibration data and the 2D image position of the foreground object, so that the foreground billboard can be placed at the 3D model background. Then, both the foreground billboard texture and background texture are projected to the virtual view image and rendered as the result image.

3. ALGORITHM FOR VIEW TRANSITION

An image at each time instant from a virtual camera viewpoint is generated, and it contains background model textures and foreground billboard textures. Textures of the background are obtained in the pre-

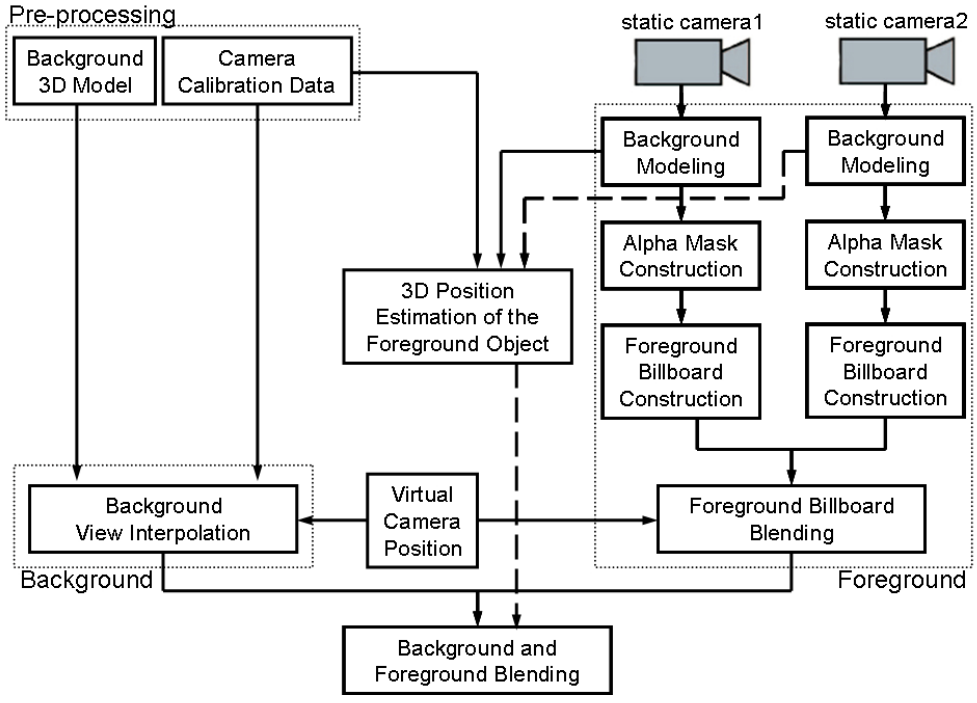

Figure 1: The overview of the proposed system

processing phase because they are considered as static regions; textures of the foreground, however, are generated at each frame since the shape or content of the foreground object alters as time changes.

3.1. View interpolation for the background

In order to construct the virtual view images of the background, we construct 3D model of the background, and the corresponding textures captured from real cameras are pasted on the model. Then, the position and the orientation of the virtual camera is interpolated from that of real cameras and used to re-project the 3D model texture on the image plane. Because we have two different cameras, there will be two virtual background images that are synthesized with real captured data of each camera. These images are blended with different weights according to the virtual camera position, and then the background view transition image is generated.

The background 3D model can be built by finding the geometry of multiple views [3]. In the procedure of geometry construction, camera calibration is an important issue to establish the relationship between the 3D world coordinates system and their corresponding 2D image coordinates system. One of the most popular methods for camera calibration is Zhang's calibration methodology [14]. The advantage of Zhang's method is its accuracy and easiness to make a planar pattern. After the intrinsic parameters of the cameras are estimated, extrinsic parameters can be calculated by the nonlinear refinement within the Levenberg-Marquardt Algorithm [8] [9] given several correspondences of 3D points and 2D points.

3.2. View interpolation for the foreground

Like view interpolation for the background, there are also two virtual foreground images that are synthesized with real captured data of each camera, and these images are given different weights, depending on the distance between the virtual camera and one of the real camera, to blend together. Then, the camera calibration data and the 2D position of the foreground object are used to calculate the 3D position of the billboard under the assumption that the foreground person stands on the floor, so that the unknown position can be calculated. Finally, the foreground billboard can be combined with the background virtual image.

3.2.1. Body detection on the image

The general method to detect the 2D position of the foreground object is to use background models. Three background modeling methods [1] are introduced. The first method is a basic background subtraction algorithm. This is the simplest algorithm by computing the difference *(D(x), where x is considered as a pixel)* between the current frame and a pre-defined background image for each pixel. Then, a pixel is labeled as foreground if *(D(x))* is larger than a threshold *(T)*. There may be lots of fractions in the difference image, so that connected component analysis or morphological operations can be applied to eliminate isolated pixels.

W4 is another method to model the background [2]. Every background scene can be modeled by three values: maximum intensity *(Max)*, minimum intensity *(Min)*, and the difference threshold *(T)*. In this method, *Max* and *Min* are trained over several frames, and a pixel can be labeled as foreground if the difference between its intensity and either *Max* or *Min* is larger than *T*. Both morphological operations and connected component analysis are used to remove noise after thresholding.

Wren proposed the Single Gaussian model algorithm [13] to model the background. In his method, each pixel can be represented by YUV color model; the mean and covariance will be recursively updated. After these parameters are calculated, the log likelihood *(l(x))* of the difference between the current image and the background image is computed and thresholded to classify the pixel as either background or foreground.

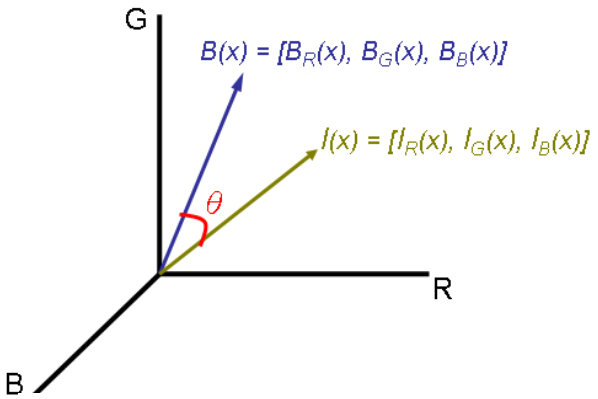

Figure 2: The color model for shadow detection.

After the background model is determined, shadow removal is another important issue for the foreground object detection because the unnecessary texture, if the shadow is considered as a foreground object, will let the seam between the foreground and the background unreal. In our system, a color model in RGB color space is used for shadow detection [4] [10]. Let the expected background color be $B(x)=[B_R(x),B_G(x),B_B(x)]$, and the captured color of the current image *be* $I(x)=[I_R(x),I_G(x),I_B(x)]$, so the distortion of the chrominance can be represented by θ, and that of the brightness is $|B(x)|/|I(x)|\cos\theta$ (see Figure 2). Because there are two main properties of shadow: one is that shadow is considered to be darker than the original background model, and the other is that the chrominance of shadow should not be distorted too much from the original background color, two

thresholds should be set up and one is labeled as shadow if the following constraints are satisfied.

$$\begin{cases} \theta < T_1 \\ 1 < \frac{|B(x)|}{|I(x)|cos\theta} < T_2 \\ |I(x)| - |B(x)| < 0 \end{cases}$$

For the real-time concern and the almost static scene in our experiment, only the background subtraction is first used in our system. Then the shadow removal is followed (see Figure 3, the left one is the result of directly background subtraction, and the right one is the result of removing the shadow in the left one.), and finally some thresholding, dilation and erosion operations are taken.

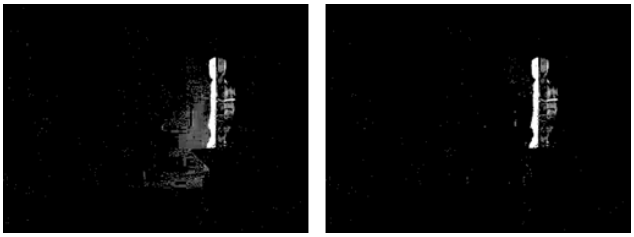

Figure 3: The result of shadow removal.

3.2.2. Alpha mask for the foreground texture

After deciding where the foreground object is on the 2D image, a probability can be used to model whether the foreground texture should be transparent or not. In our work, an alpha mask is used to represent how opaque (or transparent) the foreground image should be pasted on the foreground billboard. For each pixel considered as the foreground, the difference of the intensity between the current frame and the background image is taken as the initial alpha value of the foreground object. In other words, the higher the alpha value is, the more opaque the foreground image is blended into the billboard. However, the alpha value will be too small and foreground will be rendered too transparent if the difference value is directly used. Therefore, we double (or triple) the initial alpha value and check that the product result should not exceed 255, the maximum of the intensity. Also, some smooth operations will be taken so that the blending edge between the background and the foreground will not be too obvious. Finally, the foreground texture is re-generated by the alpha mask and warping to the plane perpendicular to the optic axis of the virtual camera.

3.2.3. 3D position estimation of the foreground object

Where the foreground billboard is established is another issue in our system; if the 3D position of the foreground body is not estimated correctly, the virtual image will not be convincing. In this section, what we mean "world coordinate system" is in the 3D model space that is mentioned before and pre-constructed in advance. In our system, there are three axis (x-axis, y-axis, and z-axis) in the world coordinate system, and let y-axis donate the direction toward up. We also define **P**=[P_x, P_y, P_z] as a 3D point, and p=[p_x, p_y] as a 2D point in an image. According to the pinhole camera model, the projective geometry of 2D and 3D points is given by $sp' = ARP' + T$, where p'=[p_x,p_y,1]^T, P'=[P_x,P_y,P_z]^T, R and T are extrinsic camera parameters, specifying rotation and translation matrix related from world coordinate system to camera coordinate system. Matrix $A = \begin{bmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix}$ stands for the intrinsic camera parameters, where α and β specifies the scale factor in image x-axis and y-axis respectively, γ defines the skewness of the two axes on the image, and u_0, v_0 are the principal point of the 2D image. In a general case, there are three unknown variables (p_x, p_y, p_z) if the 3D position of the foreground objects needs estimating, and two cameras, with large viewing regions overlapped, should be set up in order to get precise 3D position of the foreground objects. However, two cameras with large overlapped views will not be established in a real surveillance environment. Therefore, we assumed that all the foreground objects stand exactly on the floor, with position p_y=0, which means only two unknown variables should be calculated, and it works even with just one camera. This assumption is reasonable, because when people moving from one camera's FOV to another's they always stand on the floor. Even when climbing stairs, we also know the p_y, because we have the 3D model of the environment in advance.

That is to say: if the projection matrix, including intrinsic and extrinsic parameters, of one camera is known, a ray from the center of the camera to the 3D position of a person's foot can be determined. To go a step further, a 3D location of the foreground object is the intersected position of the ray and the floor in the world coordinate system. In our system, the position of a person's foot can be detected by traversing the image, which is captured by a real camera, from the bottom, counting the number of foreground pixels until 5% of total foreground pixels are scanned; in other words, the position where the scanning procedure stops is the 2D position of a person's foot.

When the 3D position of foreground objects is estimated, a billboard can be established with texture mapping if the width of the foreground object is pre-estimated. Notice that if a person's foot is occluded and its 2D position is inaccurate, the corresponding 3D position of a person's is not precise, either. However,

the inaccurate part is not obvious because the texture of that part will be transparent.

After the previous three steps are finished, the billboard is placed in the background 3D model; all objects, including the background and the foreground, will be re-projected on the image plane of the virtual camera.

4. EXPERIMENT RESULTS

In our experiment, there are two cameras established in an indoor environment. As shown in the Figure 4, one camera is set up inside a room and shoots toward one side of the door (we mention it as camera2, which captured image is also shown in Figure 5(e).); another camera is settled outside the room and shoots toward the other side of the door (we mention it as camera1, which captured image is also shown in Figure 5(a).). Notice that in our environment, views of the two real cameras differ significantly, and nothing, except the door, can be captured in both these two real images.

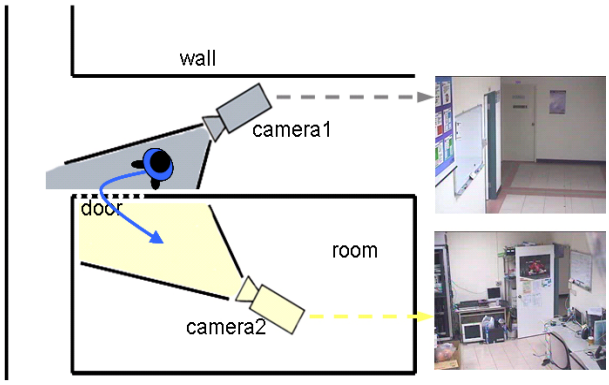

Figure 4: The environment of our experiment.

In addition, a person will walk from the outside to the inside, and similarly the positions of virtual camera will be interpolated from camera1 to cmaera2 as time changes. What we synthesized are views that captured at the position from the outside, moving through the wall, to the inside. In our implementation, several walls are used to represent the complicated background 3D model, and those walls that can not be seen by real cameras will be drawn gray. In Figure 5, the leftmost and rightmost images are captured directly by real cameras. As time goes by, images shown on the monitoring screen are those from left 5(a) to right 5(e). For more detail, images that captured from real cameras and synthesized for the virtual camera will be listed in Figure 7. The left column displays frames captured by camera2, and the right column shows frames captured by camera1. The results are showed at the middle column. In our virtual images, the synthesized one in the 4th row seems a little fragmental and some fractions of background textures still are rendered. However, if frame rate of the synthesized sequence is 20 frames per second or more, human eyes will not perceive these fractions and the virtual effect seems realistic. Operating on 320x240 pixels images, our results can achieve 22 frames per second on a 3.4GHz Intel Pentium IV with 2G RAM.

Only one case is dealt with in our experiment, and there are some other common ways to set up cameras. For example, cameras can be set up at two sides of a street corner, seeing toward the turn position from different viewpoints (See Figure 6(a)). Or cameras can be established at two end parts of a straight sidewalk, viewing toward the ground of the other camera (See Figure 6(b)). We will deal with these cases in the future.

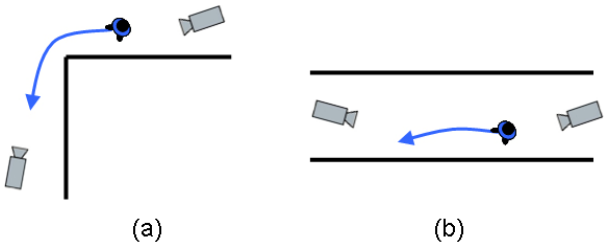

Figure 6: Other cases to be investigated.

Figure 5: The synthesized results of virtual views. (a) is captured by a real camera that is established outside the room while (e) is shot by the other real camera set inside the door, and (b), (c), (d) are images that synthesized in our system. There are seven frames between each two images; say, if (a) is the 1st image (b) is the 8th image, (c) is 15th image, and (d) is the 22th image.

5. CONCLUSION

We have proposed a novel method for smooth transition between views of two real cameras by synthesizing the images of the virtual cameras. Our method does not depend on any feature points and correspondences. Also, our method can deal with reference images from cameras with almost different views. The foreground and the background images are generated separately. The virtual background image is produced by using pre-constructed 3D background model with texture mapping, where the textures are warped and re-projected as the view of the virtual viewpoint. For the foreground object, the background modeling method determines the 2D position on the image, and an alpha mask is constructed for the foreground texture. Also, 3D position of the foreground object can be automatically estimated by the 2D position of the real image and the camera calibration data, and then a billboard with foreground texture can be set up according to the 3D position. Experimental results have demonstrated the reality and smoothness of view transition. The program using our method can achieve 22 frames per second on a 3.4GHz Intel Pentium IV with 2G RAM.

In the experiment, the timing for view transition is determined manually and only two cameras are considered in our environment. In the future, our system will automatically determine when for view transition and which camera to switch to. Also, more general cases for camera positions should be dealt with. For instance, two cameras can be set up at two sides of a street corner, or at two end parts of a straight sidewalk.

ACKNOWLEDGEMENTS

This work has been supported in part by the Ministry of Economic Affairs, Taiwan, under Grant 95-EC-17-A-02-S1-032.

REFERENCES

[1] A. Katkere, S. Moezzi, D.Y. Kuramura, P. Kelly, R. Jain, "Towards video-based immersive environments," *Multimedia Systems,* pp. 69-85, May 1997.

[2] R. Hall, J. Nascimento, P. Ribeiro, E. Andrade, P. Moreno, S. Pesnel, T. List, R. Emonet, R.B. Fisher, J. Santos Victor, J.L. Crowley, "Comparison of target detection algorithms using adaptive background models," *Proc. 2nd Joint IEEE Int. Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance (VS-PETS),* pp. 113-120, Beijing, October 2005.

[3] I. Haritaoglu, D. Harwood, and L. S. Davis, "*W4*: real-time surveillance of people and their activities", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 22, No.8, pp.809–830, August 2000.

[4] R. Hartley, A. Zisserman, *Multiple view geometry*, Cambridge University Press, 1 edition, 2000.

[5] T. Horprasert, D. Harwood, and L.S. Davis, "A Statistical Approach for Real-time Robust Background Subtraction and Shadow Detection", *Proc. IEEE ICCV'99 FRAME-RATE Workshop*, Greece, September 1999.

[6] N. Inamoto and H. Saito, "Free viewpoint video synthesis and presentation from multiple sporting videos," *IEEE International Conference on Multimedia & Expo, The Netherlands, July* 2005.

[7] B. Lei and E. Hendriks, "Real-time multi-step view reconstruction for a virtual teleconference system," *Proc. EURASIP J. Appl. Signal Process*, Vol. 2002, No. 10, pp. 1067-1088, Oct. 2002.

[8] K Levenberg, "A method for the solution of certain problems in least squares," *Quarterly Applied Math*, Vol. 2, pp. 164-168, 1944.

[9] D. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters," *Journal of the Society for Industrial and Applied Mathematics*, Vol. 11, No. 2, pp.431-441, 1963.

[10] A. Prati, I. Mikic, M. Trivedi, and R. Cucchiara, "Detecting moving shadows: Algorithms and evaluation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 25, No. 7, pp. 918-923, July 2003.

[11] H.S. Sawhney, A. Arpa, R. Kumar, S. Samarasekera, M. Aggarwal, S. Hsu, D. Nister, K. Hanna, "Video flashlights: real time rendering of multiple videos for immersive model visualization," *Proceedings of the 13th Eurographics workshop on Rendering*, pp. 157-168, Pisa, Italy, June 2002.

[12] S. Seitz, C. Dyer, "View morphing," *Proc. of ACM SIGGRAPH*, pp.21-30, 1996.

[13] C. Wren, A. Azarbayejani, T. Darrell, A. Pentland, "Pfinder: Real-time tracking of the human body," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 19, No. 7, pp.780-785, July 1997.

[14]Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 22, pp. 1330-1334, November 2000.

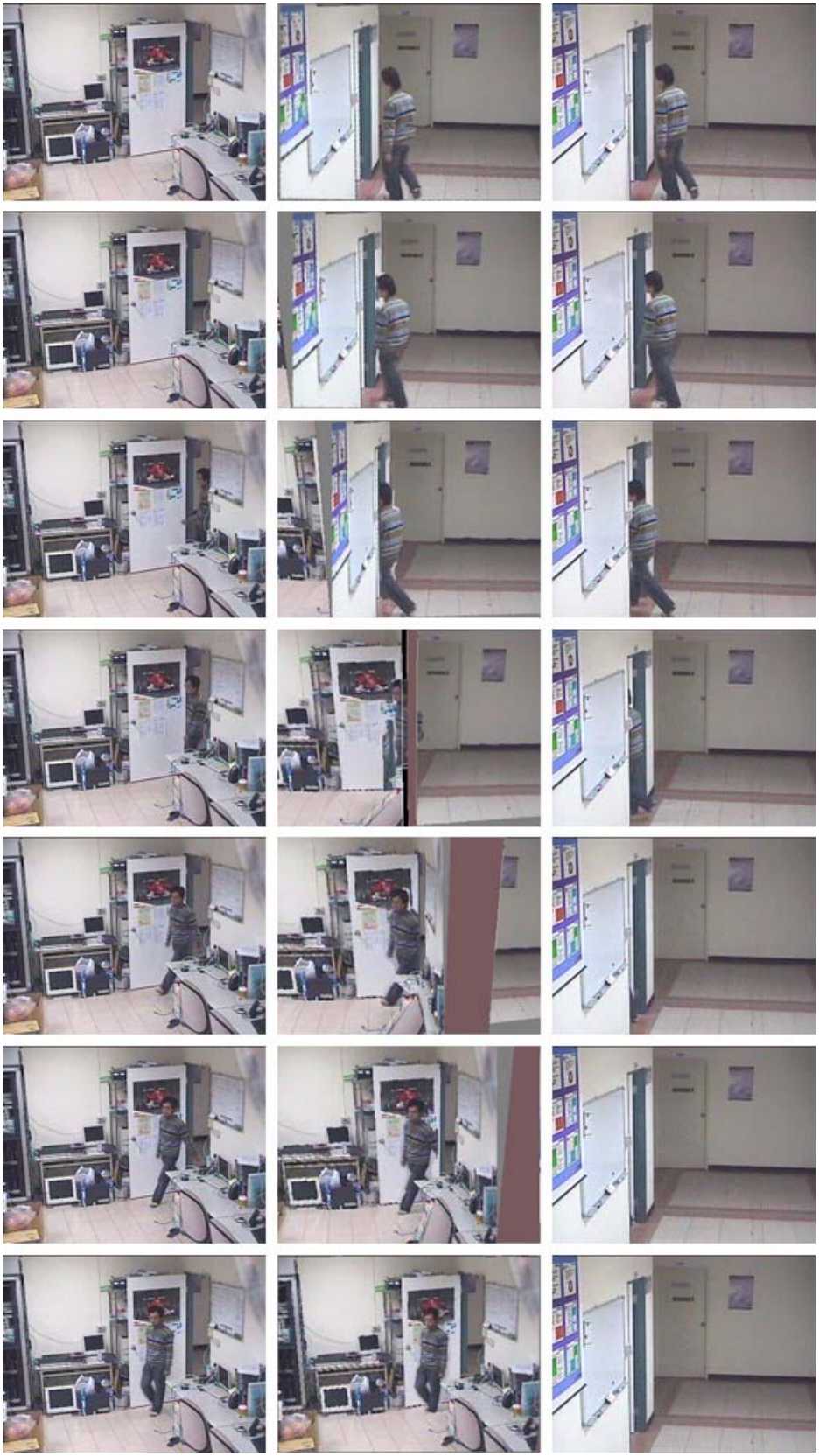

Figure 7: Comparison between real and synthesized views. The leftmost column shows the captured images from a real camera inside the room, the rightmost column shows images shot from a real camera outside the room, and the middle column displays our synthesized results. There are five frames interval between each row. In our experiment, the reality can be shown in the middle sequence although positions of the foreground object may not set the same as real locations.