Generation of Animated Stereo Panoramic Images for Image-Based Virtual Reality Systems

Fay Huang*, Chih-Kai Chang, and Zi-Xuan Lin

Department of Computer Science and Information Engineering,
National Ilan University, Yi-Lan, Taiwan, R.O.C.

Abstract. Panoramic image-based representation of real world environments has received much attention in virtual/augmented reality applications due to advantages such as shorter generation times, faster rendering speeds, higher photorealism, and less storage needed when compared to the 3D modeling approaches. In this paper, a novel approach is proposed for generating animated stereo panoramic images based on a single video sequence captured by a panning camera. The techniques involved include generating seamless stereo video textures, inpainting the unsatisfactory regions, and embedding video textures into stereo panoramic images. A player is also developed to provide user stereo visualization and real-time navigation of the image-based virtual environment that are featured with seamlessly-looping animated scenes. The quality of the generated animated stereo panoramic image is satisfactory for virtual reality applications and the computation time is acceptable for practical use.

Keywords: Image-based virtual reality, stereo panoramic imaging, video inpainting, video textures.

1 Introduction

Stereo imaging has been an active research topic due to the rapidly evolving and improving of stereoscopic display technology. Panoramic images have been widely used in various virtual reality (VR) applications, such as virtual touring and navigation, to achieve realistic rendering in real-time. The goal of the developed program is to combine those two features into an image-based virtual reality system. In this paper, the generated output images are referred to animated stereo panoramic images. There are several forms of panoramic images, such as spherical, cubic, or cylindrical panoramas. Among them, only cylindrical form can provide non-contradictionary relative depth perception of any given view within the captured scene. In other words, representing the virtual environment by a pair of cylindrical panoramic images is the only solution which allows stereoscopic visualization [7]. Therefore this paper aims to generate a pair of stereo cylindrical panoramic images featured with animated scenes from a video sequence acquired by an off-the-shelf digital camera. Figure 1 illustrates the concept of the proposed approach.

* Research funded by NSC Taiwan. (NSC 102-2221-E-197 -025 -)

R. Klette, M. Rivera, and S. Satoh (Eds.): PSIVT 2013, LNCS 8333, pp. 481–492, 2014.

Fig. 1. The concept of the proposed approach

A 360° cylindrical panoramic image can be generated by various techniques, such as mosaicing or stitching [2,15], or can also be acquired using specialized sensors such as catadioptric [11]. In particular, stitching is an efficient method if taking only camera panning motion into account. Therefore, panoramic image stitcher has become a standard built-in feature for many models of digital cameras. Many cameras' built-in image stitchers or commercially available panorama makers would automatically estimate all the parameters required for generating the panoramic image based on the provided set of sequential images. The underlying calculations include feature detection, matching, warping, and color blending. Their performances, in terms of the resulting panoramic image and the stitching speed, are in general satisfactory, but none of them are able to generate stereo panoramic images.

Generating a cylindrical panoramic image from a rotating video camera is an alternative approach [5,13]. Ideally the camera should be supported by a special designed rotating platform and only a narrow vertical image region from each capturing position is used to compose the final panoramic image. By this approach, it can naturally bypass camera parameter estimation and image warping processes. Image warping is a process highly depends on the accuracy of the estimated camera parameters. The resulting panoramic images would suffer from "double-image" effects if incorrect warping has been performed. Another major advantage of this rotating camera approach is its ability of generating the stereo panoramic images. However, it can only produce static stereo panoramic images.

Our program adopts the advantages from both methods. It accepts a video sequence captured by a rotating camera (on a tripod) as input, thus no special equipment is needed. The final stereo panoramic images are generated by combining side-by-side image columns from different video frames without image warping. Moreover, during the video capturing process, it is often unavoidable to have walking pedestrians or moving vehicles in the scene, which will then appear in the resulting panoramic images. The program provides a simple user interface, which allows users to specify the undesired object to be removed and then fills the hole with the available background scene by video inpainting technique. The concept of the inpainting algorithm used in the program is similar to [3,16].

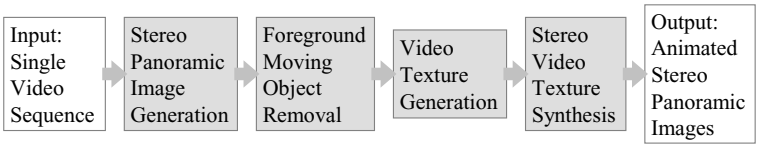


Fig. 2. The flowchart of the proposed framework for generating a pair of animated stereo panoramic images

In order to increase the realistic impression while navigating in a panoramic-image-based virtual reality platform, various hybrid image or video approaches have been proposed [4,8,9,12]. The goal is to preserve and illustrate the moving objects in the scene caused by nature or man-made forces, such as waving trees, fountains or streams, to enhance the realism of the virtual environment. Video data often contains spatial and temporal redundancy, especially for the above-mentioned object motions, information is highly repetitive or quasi-repetitive. The concept of video textures [14] is to create an impression that a video can be played continuously and infinitely based on a given sample video. Panoramic video textures [1] inherit the same concept and allow generating a wide field-of-view video texture. It employs graph-cut algorithms to generate the video textures, which is very time consuming.

The aim of this paper is to achieve a comparable quality of animation effects within a practically acceptable time. The animated stereo panoramic images are a pair of cylindrical panoramic images embedded with multiple stereo video textures. A virtual reality navigation tool (i.e., a player) has been implemented, which is capable of playing such format of animated stereo panoramic images and supports stereo visualization.

2 Program Framework

The framework of the proposed approach is illustrated in Fig. 2. The developed program accepts a single MPG video file and converts it to a set of JPG images for the subsequent tasks. First, a pair of still stereo panoramic images in cylindrical form is generated by image stitching technique. At this point, users are already able to visualize the result through mouse interaction. This is the same as many other commercially available panoramic image viewers/players, except that two panoramic images (one for the left and the other for the right eye) instead of a single image are displayed at the same time. Within the image displaying window, users are able to specify image regions containing undesired moving objects. Then, those regions will be inpainted by copying background patches from other image frames. This procedure can be performed repeatedly until a satisfactory result is achieved. Next, for those background image regions containing dynamic objects, upon user's selection, individual (single-view) video texture will be generated. Solely based on the single input video sequence, it is impossible to construct a "geometrically-correct" stereo video textures. Hence,

the last step of the framework is to synthesize another left/right video texture for each of the previously generated video textures. The parallax is estimated by stereo analysis. Finally, those stereo video textures are embedded on the stereo panoramic images while visualizing them through the developed player.

3 Stereo Panoramic Image Generation

The task is to generate two cylindrical panoramic images containing parallax information from a finite sequence of video frames captured by a single rotating camera. The method for generating the stereo panoramic images used in this paper can be considered similar to approaches reported in [7,13]. Peleg and Ben-Ezra firstly proposed the idea of generating a stereo panorama using a single camera. However, the selection of image strips and the determination of image strip[1] width have not been taken into concern. In Huang et al.'s approach, a special designed rotating camera was used to capture a cylindrical panoramic image. The camera rotation speed was assumed constant during image adequation. There was no discussion on the situation when the camera is rotated manually, and as a result, constant rotation speed is on longer possible. For a manually controlled motion, it is possible to intentionally slow down the camera rotation speed or even pause the rotation a bit to capture dynamic features within the scene. Approaches to the above-mentioned issues will be addressed in this section.

3.1 Image Acquisition

The image acquisition approach can be considered as an approximation to the method presented in [7]. Instead of the special designed rotating rig and a line-camera, a video camera mounted on a standard rotatable tripod were used to simplify the image acquisition process. In order to generate a pair of stereo-viewable panoramic images, the camera should be placed away from the rotation axis during capturing, and two symmetric[2] portions of image columns are used to compose left and right images, respectively. The proof that this approach can deliver a pair of stereo-viewable panoramic images is given in [7]. The imaging geometry is depicted in Fig. 3 and an example of a pair of stereo panoramic images is illustrated in Fig. 4. Moreover, it is highly recommended to use leveler to minimize geometrical distortion in the resulting panoramic images. From our experiences, the scan time for a 360 degree rotation between 70 to 100 seconds is the optimal speed to balance the image quality and the processing time.

The video camera is rotated with respect to a fixed rotation axis as shown in Fig. 3. For a 360 degree rotation, the camera's trajectory form a circle. The radius and the center of the circle are denoted as R and $\mathbf{O}$, respectively. Camera's projection center is denoted as $\mathbf{C}_i$, where index i is used to indicate different

[1] In [13], image strip is referred to a set of adjacent image columns.
[2] Two image stripes are symmetric with respect to the normal of the camera trajectory.

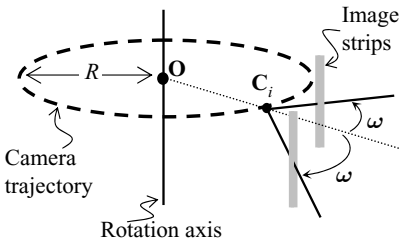


Fig. 3. The imaging geometry of a pair of stereo panoramic images

camera positions along the trajectory. This index will be used also corresponding to individual image frame. Later in the paper, an image frame captured at position $\mathbf{C}_i$ will be denoted as E_i. The two image strips are formed by a set of image columns (or vertical lines) with angular distance ω away from the optical axis of the camera (i.e., dashed line).

3.2 Determination of Image Strips

Due to that the rotation speed of the video camera varies during recording, each frame should contribute different numbers of image columns to compose the final panorama. Before determine the image strip in each frame, a quick image preprocess is performed to identify image frames captured at the same location $\mathbf{C}_i$. There are two purposes for this process: First, those frames should be ignored while determining the optimal width of the image strips. Second, those frames are needed for video texture generation. The pseudocode of the algorithm is presented in Algorithm 1.

In particular, function *Sampling*() in the algorithm is to resample all image frames. The original image resolution has been set to 480(width) × 720 (height) pixels, and the resampled images have resolution of 90× 120 pixels. This would reduce the preprocessing time. After function *CannyEdge*(), images were also binarized. Therefore, function *Thresholding*() is able to consider the resulting D_j as an one-dimensional curve, and the values represent the similarity information of two adjacent frames calculated by function *Difference*(). While the similarity

Fig. 4. An example of a pair of stereo panoramic images

Algorithm 1 : Removing paused image frames

Input: Image frames E_j for $j = 1, 2, \ldots, m$.

Output: Image frames E_i for $i = 1, 2, \ldots, n$.

for $j = 1$ to m **do**
  $S_j = Sampling(E_j)$, $S_j = CannyEdge(S_j)$, $S_j = Dilation(S_j)$.
end
for $j = 1$ to m-1 **do**
  $D_j = Difference(S_{j+1} - S_j)$.
end
$I = Thresholding(D_j$ for all $j)$ *% I is an 1D array recording indexes*
% associated to the paused image frames
$E_i = Updateting(I, E_j$ for all $j)$ *% paused image frames have been eliminated*
% from resulting E_i, where $n \leq m$.

value is above a given threshold (i.e., 90%), it is to assume that those images were captured at the same location and thus are referred to the "paused image frames". In the remaining contents, n will be used to denote the number of image frames after the removal of the paused image frames.

Next, it is necessary to determine which image columns are to be used to generate the panoramic images. This is equivalent to computing the value of ω (in Fig. 3). A larger value of ω would provide greater depth resolution for stereo-viewing. However, in some cases, especially for the indoor scenes where some objects are relatively close to the camera, a large ω value might cause difficult or impossible for your eyes to fuse the stereo images. The determination of the optimal value of ω requires the knowledge of scene range of interest, the value of R, and the maximum angular image disparity. The scene range of interest defines a near and a far distances of the interested scene, which is to say that users would like to make sure stereo fusion is possible for all objects lie within this range. Let D_1 and D_2 denote these two distances, where $D_1 < D_2$. The maximum angular image disparity, denoted as θ, may vary depending on the applied stereoscopic visualization method. The relation among the optimal value of ω and all other variables defined above is given by Equation (1).

$$\frac{\theta}{2} = \sin^{-1}\left(\frac{R\sin\omega}{D_1}\right) - \sin^{-1}\left(\frac{R\sin\omega}{D_2}\right) \tag{1}$$

To calculate ω given R, D_1, D_2, and θ is not straightforward. Therefore, an approximation formula was derived for this purpose as shown in Equation (2).

$$\omega = \sin^{-1}\left(\frac{0.0078\theta}{R(\frac{1}{D_1} - \frac{1}{D_2})}\right) \tag{2}$$

In the equation, all the distance measurements are given in meters, and angles are given in degrees. It gives a good approximation while the values of R is below 0.5 meters, which is practically true due to the limitation of the extension slider.

The last step in this subsection is to obtain a suitable image strip width. Starting from the left panoramic image, let l_i denote the number of image columns contributed from the ith frame to generate the panoramic image. Further, let

w denote the expected average number of image columns contributed from all frames. Ideally, $w = \frac{1}{n}\sum_{i=1}^{n} l_i$, where n is the total number of frames. If the camera's intrinsic parameters are known, then it is possible to estimate w by the following equation:

$$w = \frac{2f\pi}{\tau n},$$

where f and τ denote the focal length of the camera and the pixel size (assuming squared pixels), respectively. Again, this formula gives a good approximation while the values of R is below 0.5 meters. The value of w is to be rounded to the neatest integer for later calculations. The image strips used for generating the left panoramic image are calculated based on the estimated w.

Let W denote the width of an image frame in pixels. Image strips bounded by columns c_s through c_e in each frame are gathered by the program to perform image stitching calculations. The formulas for obtaining c_s and c_e are as follows:

$$c_s = max\{1, round\left(\frac{W-w}{2} - \frac{f\tan\omega}{\tau}\right) - 3\},$$
$$c_e = round\left(\frac{W+w}{2} - \frac{f\tan\omega}{\tau}\right) + 3.$$

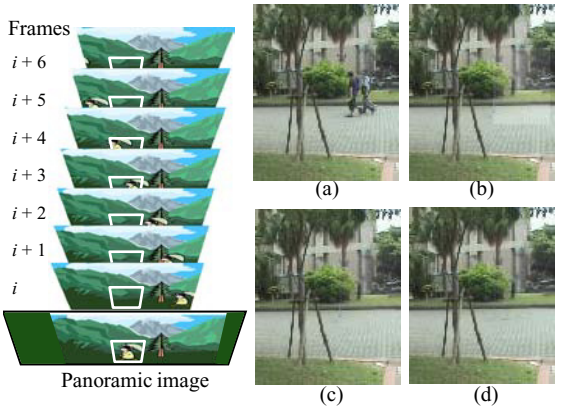
If users have no information of camera intrinsic parameters, then by default, program would select left image strip from column $\lfloor\frac{W}{20}\rfloor$ to column $\lfloor\frac{W}{5}\rfloor$. For the right image strips, symmetric image regions are selected with respect to the central image column.

The value of l_i for each $i \in \{1, 2, \ldots, n\}$ is determined by the result of image stitching, which will be discussed in the next subsection. Finally, let r_i denote the number of image columns contributed from the ith frame to generate the right panoramic image. An important key in generating a pair of stereo panoramic images is to make sure the resulting left and right images have the same number of columns (i.e., equal image width). Although l_i is not necessarily equals to r_i for each i as they are stitched separately, the following constraint has been proposed in this paper to ensure $\sum_{i=1}^{N} l_i = \sum_{i=1}^{N} r_i$. We assume that $l_i = r_j$, where

$$j = i + \frac{n}{\pi}\left[\omega - \sin^{-1}\left(\frac{2R\sin\omega}{D_1 + D_2}\right)\right].$$

3.3 Image Stitching

There have been many algorithms developed for video registration for different purposes. Here, the task is to generate a pair of stereo panoramic images. Base on the proposed camera setup, registration problem is relatively simple due to the constrained camera motion. The selected image strips are used for accelerating the video registration speed. All successive image strips from two adjacent frames are to be registered to form a panoramic image. This process is also referred to as image stitching.



Algorithm 2 : Inpainting

Input: Image regions *Rj* for *j* = *i*, *i*+1, …, *i*+*k*.
Output: Image patch *B* *% the background patch*

for *j* = *i* to *i*+*k*-1 **do**
Dj = *Difference*(*Rj*+1 – *Rj*) *% image subtraction*
End
b = *Min*(*Dj* for all *j*) *% b is a frame index*
B = *Brightness*(*P*, *Rb*)
B = *Blending*(*P*, *B*) *% P is the panoramic image*

Fig. 5. Left: The concept of foreground moving objects removal. Middle: An example of inpainting result. (a) is the original resulting panoramic image, (b) is an intermediate result after the region has been replaced, (c) is the result after brightness correction, and (d) is the result after blending. Right: Inpainting algorithm.

For each pair of successive image strips, a similarity measure is defined as the average pixel intensity difference within the overlapping region. Image strips are stitched based on the obtained minimum similarity value. For instance, if the left strips from the the ith and the $(i+1)$th frames have been determined having o_i columns overlapping. Then we have $l_i = w - o_i$. The resulting left panoramic image is composed by stitching together l_i image columns from the ith frame, for all $i \in \{1, 2, \ldots, n\}$. An example of image stitching result is illustrated in Fig. 4.

3.4 Moving Object Removal

An image refinement feature has been implemented, that has the ability to remove the undesired foreground moving objects. This makes the developed program more advance than other existing panoramic image makers or stitchers, such as Panorama Maker, PixMaker, Panorama Factory, AutoPano, and PTGui. The program provides an interface which allows users to specify regions to be refined, and the specified regions will then be replaced with background scene by exemplar-based image inpainting technique.

Figure 5(left) shows the concept of our inpainting approach, where the undesired object in the panoramic image (at the bottom) has been enclosed by a white rectangle. The available source image frames from i to $i + 6$ are displayed above the panoramic image. The actual number of available source image frames depends on the camera rotation speed and the detected paused frames. In each source image frames, the corresponding image regions (also enclosed by white

rectangles) are then determined based on the previously obtained l_i and r_i. The program automatically replaces the specified region in the panoramic image with the determined region among the source image frames. The inpainting algorithm is given in Algorithm 2.

It is naturally that the brightness of the panoramic image and the replacement batch R_b (obtained by the algorithm) are different due to that they were captured at different time instant. A brightness correction function, *Brightness*(), has been implemented to reduce the visibility of the seam. Pixel-wise color similarity test was perform between the user selected rectangular region, denoted as U, and the obtained image region R_b from the bth frame. For each pixel position, if the color similarity of two pixels in U and R_b is above a threshold, then the intensity value of the pixel in U will be taken into account. An average intensity value can be obtained through this process. Then, the intensity of image region R_b will be adjusted according to the obtained average intensity value. The output of function *Brightness*(), is the result after intensity adjustment.

Finally, a linear blending scheme is also applied to further reduce the rectangular artifact. An image border region of B (obtained by the algorithm) is defined to perform the color blending. The width of the border was set to five pixels. A linear color blending function is used to adjust the pixels' colors within the border region. Figure 5(middle) illustrates an example where the moving pedestrians have been removed. (a) shows the original panoramic image, (b) shows an intermediate result after the region has been replaced, (c) is the result after brightness correction, and (d) is the result after color blending. The process can be done repeatedly, each time enclosing a different region, until the refinement result is satisfactory.

4 Stereo Video Texture Synthesis

The method for generating a video texture based on a single image sequence has been reported in [6]. Due to page limitation, this section focuses on synthesizing the 'stereo' video texture. Because it is impossible by any way to construct a pair of "geometrically-correct" stereo video textures from a single video sequence acquired by rotating a camera. Therefore, once the left/right video texture has been created, another video texture (for stereo views) must be generated by image synthesis technique.

Given the left video texture, the algorithm used to synthesize the right video texture is presented in Algorithm 3. Assuming that the left video texture consists of k frames, and they are denoted as L_i for $i = 1, 2, \ldots, k$. A function *DynamicArea*() has been performed to partition the image region into two subregions. One is still background region and the other is dynamic object region. Function *DynamicArea*() analyzes the intensity variation of each pixel among L_i for all i. If the intensity variation of the considered pixel is below a threshold, then it is assigned as a background region pixel, otherwise, it is assigned as a dynamic object region pixel. The resulting mask M is a binarized image.

Algorithm 3 : Video texture synthesis

Input: Image frames L_i for $i = 1, 2, \ldots, k$.
Output: Image frames R_i for $i = 1, 2, \ldots, k$.

```
M = DynamicArea(L_i for all i ),      % M is a mask
[L, R] = Cropping(L_l, P_l, P_r),     % P_l and P_r are the left and the right panoramic images
A = Parallax(L, R),                   % A is a 2D array storing the parallax information
for i = 1 to k do
   L_i = UpdatingLeftFrame(L_i, M, L),
   R_i = CreatingRightFrame(R),
   R_i = UpdatingRightFrame(L_i, R_i, M, A),
end
```

Next, a left and a right images are cropped from the left and the right panoramic images, respectively, based on the given left video texture location. These two images L and R are used for calculating the image parallax (or disparity) information. The first step in the function *Parallax*() is to apply the SIFT (Scale-invariant Feature Transform) feature detection on the two input images. Then, the SIFT-based matching is used to obtain the image disparity values. The result is stored in a 2D array A.

Before generating the right video textures, the input left video textures goes through an updating function, called *UpdatingLeftFrame*(). The reason for this updating is to avoid image "shaking" effect while visualizing them through the developed player. For each frame L_i, image pixels outside of the determined mask M will be replaced by the pixel at the same coordinates in image L. Each frame of the right video texture, R_i, is first created by an identical image from image R. Then, the function *UpdatingRightFrame*() will be performed to synthesize the parallax for the dynamic object regions. For each R_i, image pixels within the mask M will be replaced by the pixel at the same coordinates in image L_i taking into account the disparity information in A. The procedures and example are illustrated in Fig. 6.

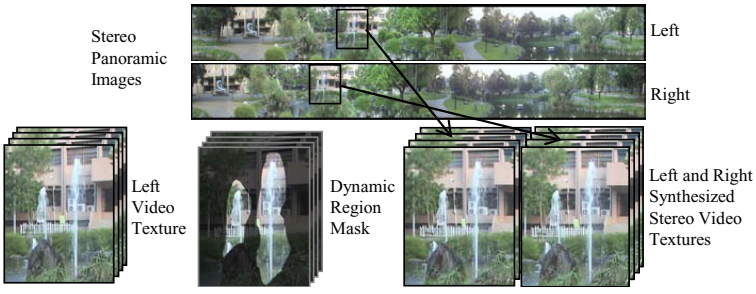


Fig. 6. The procedures of stereo video texture generation.

Table 1. Information of the input videos for generating the video textures

	Original sequence	Video texture	Image size	Computation time
Fountain 1	105 frames	27 frames	382 × 209 pixels	136 seconds
Straws	80 frames	22 frames	180 × 210 pixels	121 seconds
Fig. 6	223 frames	25 frames	181 × 289 pixels	132 seconds
Fountain 2	124 frames	28 frames	255 × 115 pixels	38 seconds
Leaves	165 frames	24 frames	158 × 111 pixels	83 seconds

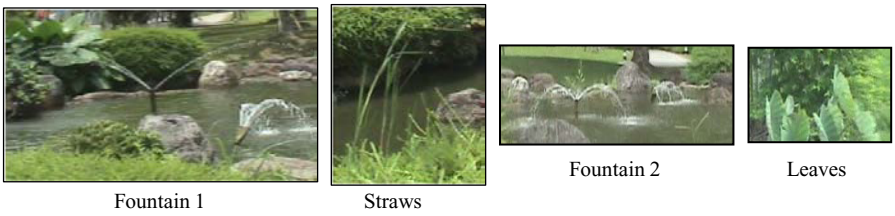


Fig. 7. Four video texture examples. The video size and the computation time are given in Table 1.

5 Experimental Results

The program was written in Borland C++ Builder 6.0. The experiments were performed on Windows XP (Service Pack 3) operating system running on a Intel(R) Core(TM) i7 CPU 920 2.67 GHz with 3G of RAM. The image resolution has been set to 480(width) × 720 (height) pixels with frame rate equal to 30 frames per second. The computation time for video registration and stereo panoramic image generation together is between 5 and 9 minutes depending on width of image strips assigned to perform registration. This is considered acceptable if comparing to panoramic video texture [1], which would take hours to process.

We have tested the algorithm on dynamic scene objects such as waves trees or leaves and water fountains. Examples used for experiments are shown in Fig. 7. The original fountain 1 sequence extracted from the input video consists of 105 frames of size 382 × 209 pixels. The resulting video texture has 27 frames. The video texture computation time for this example is 136 seconds. The video information of these examples and the computation times are summarized in Table 1. The aim is to produce video textures consisting of small number of image frames unless user has specified a preferred minimum video texture length. All these examples demonstrate the capability of the proposed approach to generate video textures that are consist of small number of image frames. In comparison to the panoramic video, the proposed approach requires much less memory storage to represent the animated scenes. Some results are also included in supplementary material.

6 Conclusion

This paper presented an approach for generating a pair of animated stereo panoramic images based on a single video sequence. The developed program has the following novel features: First, the input video sequence can be captured by a standard digital recorder, no special equipment or stereo camera is required. Second, it provides an user interface for eliminating the undesired moving foreground objects. Finally, the dynamic sceneries can be represented by seamless-looping stereo video textures to enhance the realistic impression of the virtual environment.

References

1. Agarwala, A., Zheng, C., Pal, C., Agrawala, M., Cohen, M., Curless, B., Salesin, D., Szeliski, R.: Panoramic video textures. In: Proc. SIGGRAPH 2005, Los Angeles, USA, pp. 821–827 (August 2005)
2. Chen, S.E.: QuickTimeVR - an image-based approach to virtual environment navigation. In: Proc. SIGGRAPH 1995, Los Angeles, California, USA, pp. 29–38 (August 1995)
3. Criminisi, A., Perez, P., Toyama, K.: Region filling and object removal by exemplar-based image inpainting. EEE Trans. Image Process. 13 (2004)
4. Finkelstein, A., Jacobs, C., Salesin, D.: Multiresolution video. In: Proc. SIGGRAPH 1996, New Orleans, Louisiana, USA, pp. 281–290 (August 1996)
5. Huang, F., Klette, R.: Astereo panorama acquisition and automatic image disparity adjustment for stereoscopic visualization. Multimedia Tools and Applications 47(3), 353–377 (2010)
6. Huang, F., Yang, K.-M., Wei, Z.-J., Tsai, A., Tsai, J.-Y.: Animated Panorama from a Panning Video Sequence. In: Proc. IVCNZ 2010, Queenstown, New Zealand (November 2010)
7. Huang, F., Klette, R., Scheibe, K.: Panoramic Imaging: Sensor-Line Cameras and Laser Range-Finders. Wiley, West Sussex (2008)
8. Irani, M., Anandan, P.: Video indexing based on mosaic representation. In: Proc. IEEE 1986 (1986)
9. Kimber, D., Foote, J., Lertsithichai, S.: Flyabout: spatially indexed panoramic video. In: Proc. ACM MULTIMEDIA 2001, New York, NY, USA, pp. 339–347 (2001)
10. Kwatra, V., Schodl, A., Essa, I., Turk, G., Bobick, A.: Graphcut textures: image and video synthesis using graph cuts. In: Proc. SIGGRAPH 2003, San Diego, California, USA, pp. 277–286 (July 2003)
11. Nayar, S.: Catadioptic omnidirectional cameras. In: Proc. CVPR 1997, San Jaun, Puerto Rico, USA, pp. 482–488 (June 1997)
12. Neumann, U., Pintaric, T., Rizzo, A.: Immersive panoramic video. In: Proc. ACM MULTIMEDIA 2000, New York, NY, USA, pp. 493–494 (2000)
13. Peleg, S., Ben-Ezra, M.: Stereo panorama with a single camera. In: Proc. CVPR 1999, Fort Collins, Colorado, USA, pp. 395–401 (June 1999)
14. Schödl, A., Szeliski, R., Salesin, D.H., Essa, I.: Video textures. In: Proc. SIGGRAPH 2000, New Orleans, Louisiana, USA, pp. 489–498 (July 2000)
15. Shum, H.-Y., He, L.-W.: Rendering with concentric mosaics. In: Proc. SIGGRAPH 1999, Los Angeles, California, USA, pp. 299–306 (August 1999)
16. Wexler, Y., Shechtman, E., Irani, M.: Space-time completion of video. IEEE Trans. Pattern Anal. Mach. Intell. 29 (2007)