Technique Report: TR-IIS-01-011

A Purely Image-Based Approach to Augmenting Panoramas with Object Movies

Yi-Ping Hung[1,2], Chu-Song Chen[1], Gregory Y. Tang[2], Yu-Pao Tsai[1], Chen-Fu Huang[1,2], Szu-Wei Lin[1,2], Chih-Han Yu[1,2]

[1] Institute of Information Science, Academic Sinica, Taipei, Taiwan

[2] Department of Computer Science and Information Engineering
National Taiwan University, Taipei, Taiwan

Abstract

In this paper, we propose an easy-to-use approach to augmenting a panorama with object movies, in such a way that the inserted foreground objects remain visually coherent with the panoramic background. The proposed method is purely image-based in the sense that it does not have to reconstruct the 3D geometric model of the real object to be inserted in the panorama. Based on the proposed method, we have implemented a system for authoring and for browsing augmented panoramas. Experimental results have shown that the composite images rendered by our user-friendly system is visually plausible.

1. Introduction

This work concerns with the implementation of a special class of augmented reality, which we refer to as augmented panoramas. In augmented panoramas, some foreground objects are inserted into a background scene recorded in the form of panoramic images. This background scene can be either static or dynamic. If the background scene is static, we can first acquire a sequence of overlapping images and then produce a panoramic image by using image mosaicking techniques [3][13]. If the background scene to be recorded is a dynamic one, we can capture it with the omni-directional cameras [4][16]. In fact, the panoramic background can also be a computer-generated scene. The advantage of using panorama to record the virtual scene is that the rendering can be real-time no matter how complex the virtual environment is.

In the past, most researches on augmented reality have focused on integrating 3D virtual objects with a sequence of real video images. The major reason for using the on-line acquired video as the background to be augmented is that it has many potential applications, such as surgery in medical applications and maintenance in engineering applications. However, there exist some applications in entertainment or education where the background is, or can be chosen to be, mostly static and we do not need to use video data that require a large amount of storage space and transmission bandwidth. In those cases, panorama is a good choice that does not sacrifice the requirement of interactivity. One interesting application is for the Internet shopping or the virtual mall, where different commodities can, in different seasons or different categories, be exhibited on the same counter of an existing and well-decorated department store.

To integrate a virtual object with a panorama in a visually coherent way is not a difficult task if the panoramic image to be augmented is synthesized by using a 3D geometric model of the environment. However, if an object movie (OM), which is in fact a set of multi-perspective images of an real object, is to be inserted into a panoramic image taken from the real world while no 3D environment model is available, then the problem of correct registration and shadow rendering is not as trivial.

In this paper, we propose an easy-to-use approach to augmenting a panorama with OMs, in such a way that the inserted foreground objects remain visually coherent with the panoramic background. The major contribution of this paper is that the proposed method is *purely image-based* in the sense that it does not have to reconstruct the 3D geometric model of the real object to be inserted in the panorama. That is, it can avoid the tedious 3D reconstruction problem while still allowing the user to view both the panorama and the inserted objects from different perspectives, interactively and with correct 3D perception and shadowing effect.

This paper is organized as follows. Section 2 explains the problem we are dealing with. Section 3 introduces the image-based method used to integrate the OMs with the panoramas in a visually coherent manner. Section 4 describes some experimental results. Finally, Section 5 gives a conclusion and some directions for future work.

2. Problem Description

Our goal is to insert OMs into some places in a given panorama. The places for inserting the OMs should be specified by the user, or more precisely, by the author of the OM - augmented panorama. In most cases, the places for insertion are planar surfaces, such as the tabletop, the floor, the wall, and the ceiling. (If the surfaces for placing the objects on, and/or for casting the shadows on, are not planar, our method will have to be mentioned accordingly, as discussed in Section 5.)

Consider the example shown in Figure 1, where the two selected places for inserting OMs are planar surfaces. Figure 1(a) shows the entire panoramic image of the panorama to be augmented. Figures 1(b) and 1(c) show the two places selected from Figure 1(a) for the insertion. Suppose our task is to insert a Winnie into Figure 1(b) and a Kitty into Figure 1(c). Then, the two snapshots shown in Figure 2 exemplify the desired augmented result. Notice that the object to be inserted into the panorama is recorded in the form of OM. Hence, we would like to allow the user to manipulate the object in front of a panoramic background, e.g., to rotate the object in order to see it from another perspective, as illustrated in Figure 3.

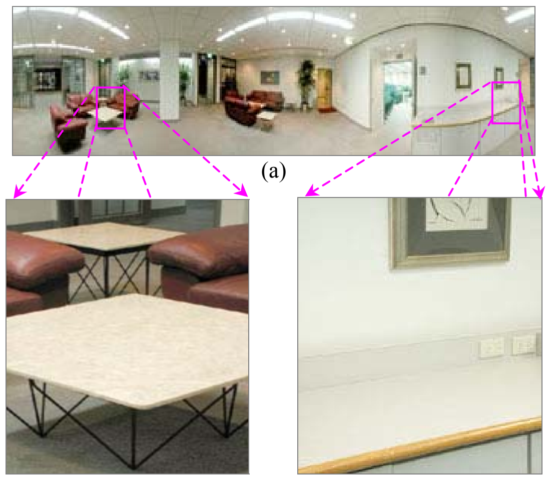
(a) (b) (c)

Figure 1: Figure (a) shows a complete panoramic image for augmentation. Figures (b) and (c) exemplify the two places selected for inserting object movies.

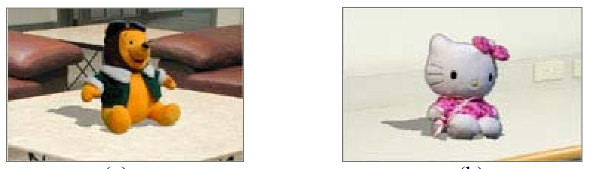
(a) (b)

Figure 2: Two snapshots of the desired augmented panorama, which corresponds to Figures 1(b) and 1(c), respectively.

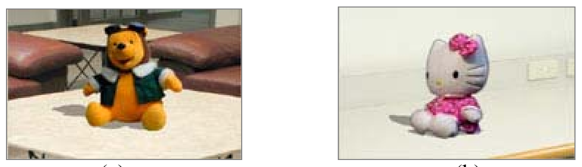
(a) (b)

Figure 3: Another two perspectives of the objects by letting the user rotate the objects.

2.1. Object Movies and Novel View Generation

In this work, the multi-perspective images contained in an OM are captured by using a digital camera mounted on a swing arm. By using this image capture device, we know the pan and tilt angles, θ and ϕ, of the viewing direction associated with each captured image (though the angle values need not to be very precise with our method). In this setup, the optical axis of the camera roughly points toward the *rotation center*, which is the intersection point of the pan axis (for rotating the object) and the tilt axis (for rotating the camera), as shown in Figure 4. When photographing an OM, this rotation center will reside at a certain point with respect to the object being photographed. This point will then become the rotation center of the object when a user is browsing the OM, and will be denoted by C_O.

In this paper, we define the object reference frame (ORF) to be the coordinate system locating at C_O, with its x-axis pointing to the camera while the camera is taking the first image (i.e., $\theta = 0$ and $\phi = 0$) and its z-axis pointing upward.

A few samples of the multi-perspective images that comprise the OM of Winnie are shown in Figure 5. It can be seen from

Figure 5 that the sampled images were captured at discrete viewpoints. If we increase the number of the sampled images, the playback of the OM will become smoother but the amount of image data will also increase accordingly. One approach to solving this dilemma is to adopt the techniques for novel view generation (or view interpolation), which uses a finite set of sampled images to create an infinite number of intermediate views.

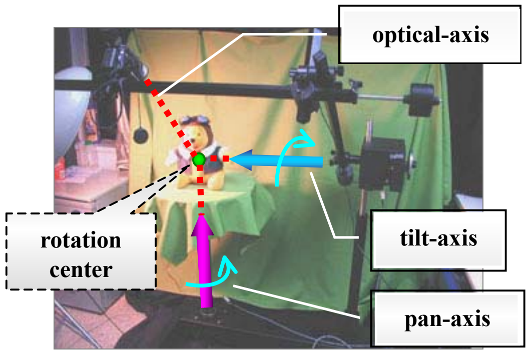

Figure 4: The experimental setup for acquiring images of object movies.

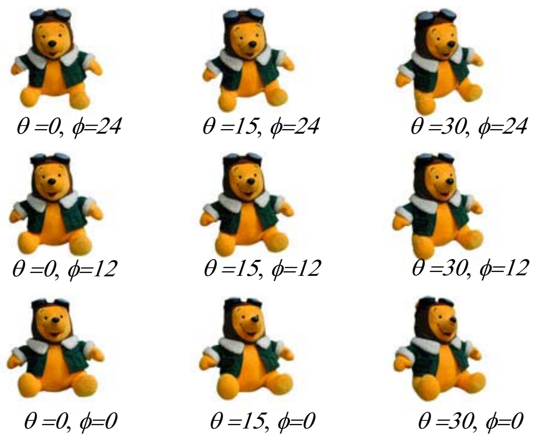

Figure 5: A few samples of the multi-perspective images that comprise an object movie of Winnie.

In augmented panoramas, novel view generation is desired not only because the playback of an OM can be smoother given the same amount of acquired images, but also because the intermediate views are indispensable, as explained below. Consider Figure 6, where our initial intention is to insert the object at point Q_1. In this case, the viewing direction should be $\overrightarrow{C_P Q_1}$, where C_P is the viewer center of the panorama. Hence, the tilt angle for photographing the object should be ϕ_1. Now, suppose we change our mind and would like to move the object closer to the viewer, e.g., move the object to point Q_2. Then the system will have to create a new image of the object that is supposed to be seen when photographing this object from the new tilt angle ϕ_2. The same situation happens if we would like to put the object on a higher or lower tabletop instead. Also, the pan angle will vary if we would like to move the object further left or further right. This is the reason why an OM to be integrated with a panorama should be a 360°×360° object movie, or at least a 360° × 180° one. For an OM that was taken only with a fixed tilt angle (i.e., a single-axis-rotation OM), visual coherence may not be able to achieved when integrating with panoramas.

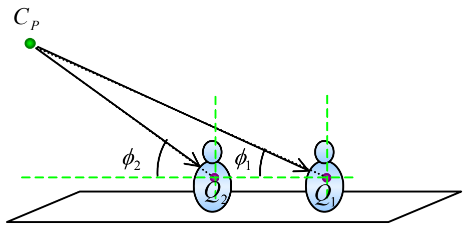

Figure 6: Moving the object around will change the viewing perspective of the object.

There are many methods available for novel view generation. Light Field Rendering [11] and Lumigraphs [5] are two famous ones, but their large memory requirements make them impractical for real applications at this moment, especially for those requiring Internet transmission. Another approach tries to reconstruct some geometric model of the object based on the consistency with the image information, such as Visual Hull [10], Photo Hull [9], and Minimal Photo Hull [14]. This approach seems to be quite promising for reducing the memory size of an OM, but it is not a purely image-based approach and hence the camera parameters has to be known accurately for reconstructing the 3D shape. In this work, we adopt disparity-based view morphing (DBVM) for generating the novel views whenever necessary. Details of DBVM are given in the appendix. In the following, we focus our discussion on how to maintain visual coherence in an OM - augmented panorama including perspective coherence and shadow coherence.

3. Augmenting a Panorama with Object Movies

To insert an OM into a panorama, our system requires the user to first select a 3D shadow reference frame (SRF) in the panorama. It is this reference frame that defines where the shadow of the object is supposed to be projected onto. Section 3.1 describes how the user can specify this 3D shadow reference frame easily with a simple interactive way, based on a 2D dewarped view of the panorama. Once the shadow reference frame is determined, the remaining tasks can be divided into three parts: (i) rendering the background layer, (ii) rendering the shadow layer, and (iii) rendering the object layer. An example of the three layers is given in Figure 7. Notice that the SRF should be specified before defining the ORF, because the former should be attached to some existing surface in the panorama. However, the shadow can only be rendered after the ORF is determined, because it is a projection of the object whose position depends on the ORF. Hence, we will first explain the rendering of object layer in Section 3.2, and then the rendering of shadow layer in Section 3.3. As to the rendering of background layer, one can easily map the panoramic texture to a 3D model for fast rendering, and hence will not be described here [13].

3.1 Selecting a Shadow Reference Frame

Not much work has been done in merging 3D objects with a single 2D image. In [2], Chen et al proposed a cuboid-based method for inserting 3D virtual objects into a single 2D image in a geometrically consistent way. In that work, we assumed that the intrinsic camera parameters were unknown, and hence the user had to specify six vertices of a cuboid in order to let the system compute the geometric relationship between the 3D cuboid and the 2D image. Recently, Chen and Hsieh modified the above method for the integration 3D graphical objects with a single panorama [1]. Because the intrinsic camera parameters (i.e., the image center and the normalized effective focal length) for any given dewarped view of a panorama can be computed directly from the dewarping process, the problem became simpler and it turned out that only four vertices of the

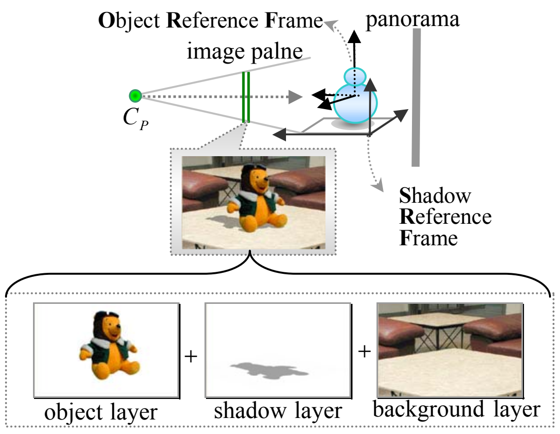

Figure 7: The desired augmented view is composed of three layers of images: the background layer, the shadow layer, and the object layer.

cuboid needed to be specified by the user.

This paper can be considered as an extension of our previous work described in [1], but dealing with the more difficult problem where the inserted object is recorded in the form of a 360° × 360° object movie and does not have a known 3D geometric model. The methods used for rendering the object and the shadow are completely different from those used in the previous paper, and will be described in Sections 3.2 and 3.3. As to the selection of the shadow reference frame (SRF), the adopted method is exactly the same as the one used in [1] and will only be briefly reviewed below.

At the beginning, the user is given a dewarped view of the panorama augmented with a dangling SRF, as shown in Figure

8(a). As a convention, the *x-y* plane of the SRF is the planar surface to which the inserted object is supposed to attach. Based on the dewarped view, the user can then drag the four endpoints of the reference frame to any image points until the shaded half-cuboid complies with the user's 3D perception, as illustrated in Figure 8(b).

Once the user has specified a SRF in the dewarped panoramic view, our system can use this information to compute the geometric transformation between this SRF and the panorama reference frame (PRF), which is a global reference frame centered at C_P, up to an unknown scaling factor. This means that the system can determine the rotation matrix but cannot determine the translation vector completely. For example, in Figure 9(a), it cannot tell whether cuboid A or cuboid B is the correct answer. Fortunately, this does not affect our augmentation process since both answers will produce the same image. Therefore, we can simply choose an arbitrary one, e.g., choose the one whose origin O is 1000 units away from C_P. Another byproduct of this computation is that we can determine the length of $\overline{OX}$, $\overline{OY}$ and $\overline{OZ}$, again, up to a scale.

The above computation is referred to as the problem of *camera pose estimation* in the computer vision community. In particular, given a trihedral with the angles between each pair of lines being 90° in the 3D space, this problem can be converted into a simpler problem of solving a second-order polynomial equation [15]. The two solutions for the second-order polynomial equation are related to the illusion of Necker cuboid. As illustrated in Figure 9, the solution corresponding to the right-hand rule is a convex half-cuboid, while the one corresponding to the left-hand rule is a concave half-cuboid. As mentioned above, our system will overlay a shaded half-cuboid on the dewarped panoramic view, so that the user can easily adjust the half-cuboid interactively until it complies with the user's 3D perception. If the user chooses the left-hand rule while still dragging the four endpoints of the half-cuboid using the same way as that is used in Figure 8(b), then

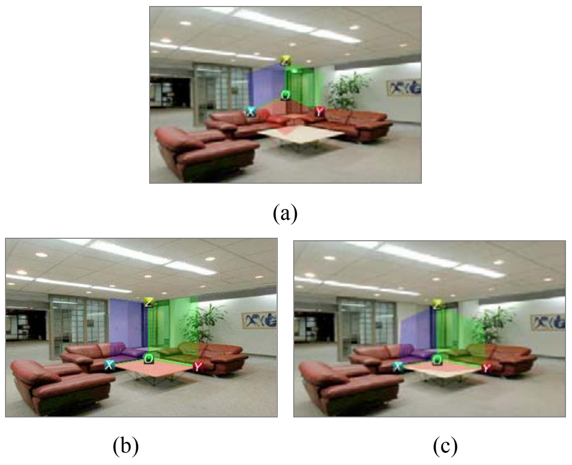

Figure 8: An example of selecting the shadow reference frame based on a dewarped view of a panorama. (b) and (c) are the convex and concave solution that are related to illusion of Necker cuboid.

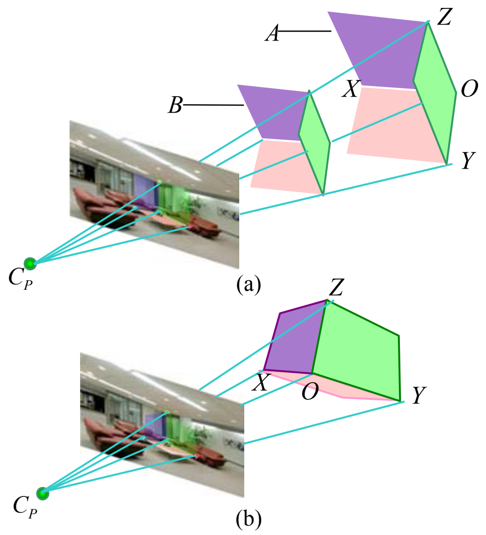

Figure 9: 3D illustration of the two solutions shown in Figure 8: (a) the convex case, and (b) the concave case. Figure (a) also illustrates that the translation can only be determined up to a scale.

he will see the shaded half-cuboid shown in Figure 8(c), which is not visually coherent with the perceived panoramic view.

3.2. Rendering the Object Layer

Our approach to rendering the object layer is to place a *correct textured plate* of the object in the *correct place* with respect to the shadow reference frame (notice that the geometric relationship between this shadow reference frame and the 3D panorama reference frame has been determined, as described in Section 3.1). Once this is done, we can then render the object layer simply by projecting the textured plate onto the image plane. Therefore, the questions are: "what is the correct textured plate" and "where is the correct place"? Let us first define the object-viewing direction and the object-observation direction. The object-viewing direction, **n**, is the direction pointing from the viewer (i.e., the center of panorama in our case) to the rotation center of the observed object. In general, this vector can be described with respect to either the SRF or the PRF (since the relation between these two reference frames is independent of the scaling, the translation, or the rotation of the object to be inserted). In the following discussion, we let the object-viewing direction be described with respect to the SRF. The object-observation direction, $\mathbf{n}'$, refers to the viewpoint (or viewing direction) that we use to observe the object, and is always described with respect to the ORF defined in Section 2.1. In fact, the object-viewing direction should be denoted by $\mathbf{n}_j$, where j varies as the object is dragged around or scaled up and down. As to the object-observation direction, a better notation is $\mathbf{n}'_{ji}$, which implies that this direction not only depends on the translation and scaling of the object (indexed by the variable j) but also depends on the rotation of the object manipulated by the user (indexed by the variable i). Ideally, we would like to let

$$\mathbf{n}'_{ji} = \mathrm{T}_{\mathrm{ORF}}^{\mathrm{SRF}}(\ \mathbf{n}_j\ ;\ \mathrm{ORF}(i)\ ) \tag{1}$$

where $\mathrm{T}_{\mathrm{ORF}}^{\mathrm{SRF}}(\ \cdot\ ;\ \mathrm{ORF}(i)\ )$ is the coordinate transformation from the SRF to the ORF(i). If the object center (i.e., C_O defined below) happens to be projected onto the view center (i.e., $\mathbf{m}_O$ defined below), then $\mathbf{n}_j$ will become $\mathbf{n}_0$, which is referred to as the optical axis. If the ORF is also at its initial orientation, then ORF(i) will become ORF(0). In this case, we have $i = 0$ and $j = 0$, which implies $\mathbf{n}'_{00} = \mathrm{T}_{\mathrm{ORF}}^{\mathrm{SRF}}(\ \mathbf{n}_0\ ;\ \mathrm{ORF}(0)\ )$.

Consider the image acquisition for an OM. Figure 10(a) shows that an image of the 3D object is taken at its nominal 3D position, where C_C represents the center of the camera. Let C_O denote the rotation center of the object, and $\mathbf{m}_O$ be its image projection. With the image acquisition setup described in Section 2.1, $\mathbf{m}_O$ should be the center of the acquired image (however, reasonable deviation is gracefully allowed). Figure 10(b) represents the textured plate corresponding to the nominal object-observation direction $\mathbf{n}'_{00}$. Figures 10(c) and 10(d) show another imaging configuration with object-observation direction $\mathbf{n}'_{0i}$ and its corresponding textured plate. Next, we would like to determine an initial position of the object, i.e., the position before the user begins to drag the object around. Recall that the length of $\overline{OX}$, $\overline{OY}$ and $\overline{OZ}$ can be determined in Section 3.1. Let N_c and N_r be the number of rows and columns of the acquired image, respectively. In this paper, we set the width of the textured plate to be $\overline{OX}$ and set the height of the textured plate to be $\overline{OX} \cdot N_r / N_c$. Let b be the lowest row of the object seen in the image, as illustrated in Figure 10 (a), and let a be the row number of $\mathbf{m}_O$. Then, the center of the 3D textured plate is initially set to be $C_O(j) = (\overline{OX}/2,\ \overline{OY}/2,\ \overline{OX} \cdot |a-b| / N_c)$, as illustrated in Figure 11. The purpose of this assignment is to let the lowest contour point of the object touch the x-y plane of the shadow reference frame. It is a good approximation but does not ensure that the bottom of the object can align with the x-y plane. Fortunately, the user can always adjust it interactively with the help of the shadow rendered in Section3.3. In summary, the above step determines the translation between the SRF and the ORF by specifying or adjusting the origin of the ORF, $C_O(j)$, with respect to the SRF. As to the initial orientation of the object, we can arbitrarily align the nominal view-observation direction, n'_{00}, toward the negative y-axis of the SRF, as illustrated in Figure 11. This assignment defines the orientation of the ORF with respect to the SRF.

At this time, if we simply put at $C_O(j)$ the textured plate corresponding to the nominal view-observation direction, $\mathbf{n}'_{00}$, and let this textured plate facing toward the y-axis, as illustrated in Figure 12(a), the result will not be visually coherent with the background panorama. The reason is that the object-viewing direction pointing from C_P to $C_O(j)$, which is now $\mathbf{n}_j$, is not the same as the object-observation direction of the textured plate, which is $\mathbf{n}'_{00}$. The remedy is to adopt the textured plate corresponding to the object-viewing direction, and facing it toward C_P, as illustrated in Figure 12 (b). That is, the texture to

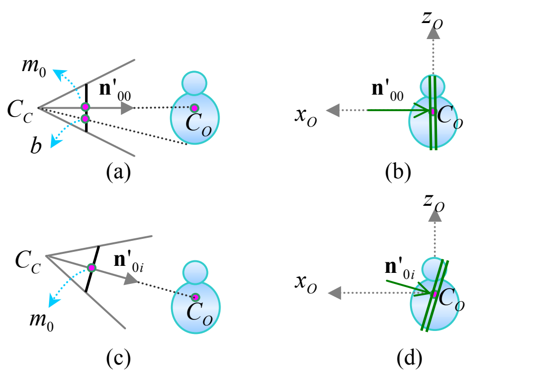

Figure 10: Image acquisition for an object movie from: (a) the nominal object-observation direction, $\mathbf{n}'_{00}$, and from: (c) another object-observation direction, $\mathbf{n}'_{0i}$. Figures (b) and (d) show the textured plate corresponding to (a) and (c).

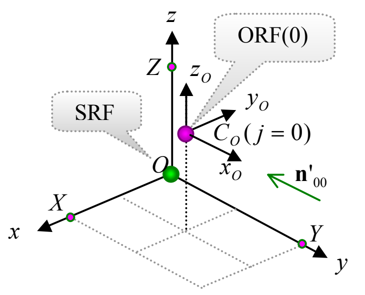

Figure 11: Initial position and orientation of the object reference frame.

be used should be the one corresponding to (θ_j , ϕ_j), as illustrated in Figures 12(b) and 12(d), instead of the one corresponding to ($\theta = 0$, $\phi = 0$), as illustrated in Figures 12(a) and 12(c).

After the above initial setup, the user can see a reasonable image of the object rendered at a reasonable place. If the user does not like the position or the size of the object he perceives, he can then drag the object around, or scale the object up and down, in an interactive way. Notice that the operations of object translation and scaling will shift the object center, $C_O(j)$ in effect. Hence, the object-viewing direction, $\mathbf{n}_j$, has to be modified accordingly. By using equation (1), the updated $\mathbf{n}_j$ can then determine a new object-observation direction, $\mathbf{n}'_{j0}$, which will be used to retrieve the correct image for texture mapping.

In the above description, the subscript *0* in $\mathbf{n}'_{j0}$ implies that the object has not been rotated until now (or that it has been rotated back to its initial orientation). At this moment, if the user begins to rotate the object around its current rotation center, $C_O(j)$, the ORF attached to the object will then be rotated accordingly from ORF(*0*) to another ORF(*i*). Again, by using equation (1), we can compute the new object-observation direction, $\mathbf{n}'_{ji}$, which will then be used to retrieve another image having the correct perspective for texture mapping. This object-observation direction, $\mathbf{n}'_{ji}$, can be decomposed into two components, This object-observation direction, $\mathbf{n}'_{ji}$, can be decomposed into two components, that is, (θ_{ji}, ϕ_{ji}) = (θ_j, ϕ_j) + (θ_i , ϕ_i), where (θ_j, ϕ_j) depends only on the position of the object center, and (θ_i , ϕ_i) depends on the orientation of the object. Notice that, with our current implementation, the object scaling will also affect the object center.

It is worthwhile to mention that, because the image acquired in Section 2.1 always has its up vector pointing to the north pole, so will be the image retrieved with $\mathbf{n}'_{ji}$, or (θ_{ji}, ϕ_{ji}). If the user would like to rotate the object arbitrarily with three degrees of freedom, the textured plate will have to perform a roll motion. That is, the system should rotate the retrieved image around the object-observation direction, $\mathbf{n}'_{ji}$. As an example, Figure 13(a) shows a sequence of rendered images without performing the roll motion, while Figure 13(b) shows some results that allow the roll motion. Details of the implementation will not be explained here due to the space limitation.

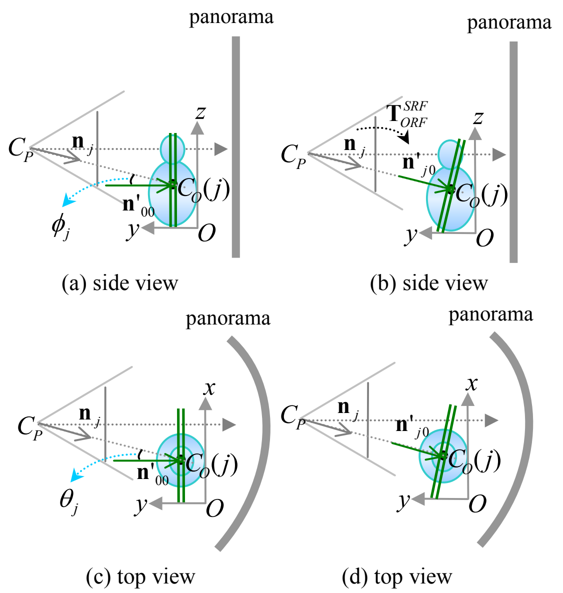

Figure 12: This figure illustrates the initial orientation of the textured plate and the selected image for texture mapping. The setup shown in (a) and (c) will gives visually inconsistent results, while (c) and (d) show the preferred configuration.

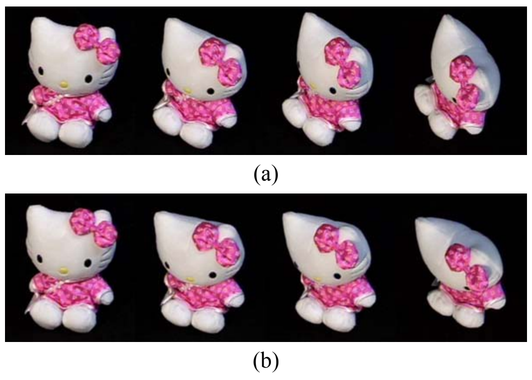

Figure 13: Images rendered (a) without the roll motion, and (b) with the roll motion, where the images in (b) is a rotated version of (a).

3.3. Rendering the Shadow Layer

The next problem is how to generate a realistic shadow of a rotating or translating object without using any 3D geometric model of the object. Similar to Section 3.2, our approach is to put the correct shadow map at the correct position with respect to a user-specified SRF. To simplify the problem, we assume that the shadow to be generated is produced by a parallel light source with lighting direction, $\mathbf{n}_L$, as illustrated in Figure 14. By using equation (1) and the current object orientation described by ORF(i), this lighting direction can be transformed into $\mathbf{n}'_{Li}$, which is the object-observation direction for the viewer locating at the light source. We use $\mathbf{n}'_{Li}$ to retrieve the image that the light source is supposed to see, and use the alpha-channel of this image as the shadow map to cast the shadow on the surfaces where the shadow is chosen to appear, e.g., the *x-y* plane of the SRF in most situation. Sometimes, the *y-z* or *x-z* plane may also be chosen to show shadow, especially when the object is inserted at a place near a wall.

4. Experiments

We have developed a system for inserting OMs in panoramas, which consists of two modules: the *authoring module* and the *browsing module*. With the authoring module, the user can easily insert multiple OMs in any places in a panorama and then produce an augmented panorama. Given this augmented panorama, other users can then use the browsing module to look around the panoramic environment and browse the inserted OMs interactively and realistically.

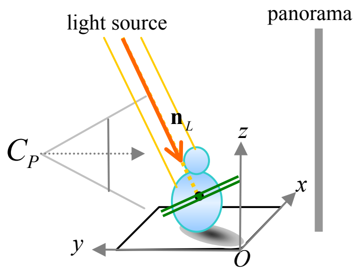

Figure 14: Illustration of casting shadow for an object movie.

4.1 Authoring Module

To create an augmented panorama containing OMs, the user first inputs a panorama, chooses the places where he wants to insert OMs, and then begins to use the simple interactive manner described in Section 3.1 to specify the shadow reference frames for the OMs. Figure 15 displays a few snapshots of an authoring example that has been recorded in a demo video clip submitted with this paper. Figure 15(a) shows the specified shadow reference frame, where both the *x-y* plane (the counter top) and the *x-z* plane (the wall) are chosen to show the shadow. Figure 15(b) shows the initial result after putting the Kitty on the counter without any human adjustment. It can be seen that both the position and perspective of the inserted Kitty seem to be already quite reasonable. To view the result more carefully, we can zoom into the augmented image. We can then adjust the position and intensity of the light source to produce a desired shadow. Figure 15(c) shows the augmented image after zooming in and rendering a shadow. Now we can move the object around, and the system will automatically adjust the perspective and the size of the object image. From Figure 15(d), we can see that the system has rendered a better image, different from the one shown in Figure 15(c). If the user scales the object up and down, the object can maintain its contact with the counter and the shadow remain to be visually plausible, as demonstrated in Figures 15(e) and (f).

4.2 Browsing Module

The browsing module allows the user to view the augmented panorama produced by the authoring module described in Section 4.1. Figures 2 and 3 have shown a few snapshots of the results. One problem concerns with the different resolutions between the panorama and the OMs. Because a panorama needs to cover a larger area than a OM, as can be seen from Figure 16(a), its resolution is usually much lower than the OM when we zoom into the interesting object to play the OM. This incompatibility of image resolution may make the panorama look like a separate background from the foreground object. To solve this problem without a overwhelming increase in the memory size of the panorama, we simply took a few closer shot for the places where we planned to put small objects (small in a relative sense) when photographing the images for panorama production. Since the images for closer view and the images for producing the panorama are all taken at the same camera nodal point, the relationship between each pair of images is a homography. Also, the relation between a closer view and a dewarped panoramic view is also a homography. Therefore, we can estimate the projection matrix between these two images by specifying four pairs of corresponding points. Once the homography matrix is known, we can smoothly hop from one image to the other in the following way.

Let the wider view of the panoramic environment that the user is currently watching be represented by a pan angle θ_c, a tilt angle ϕ_c, and a focal length f_c, as shown in Figure 16(a). Based on the estimated homography matrix, our authoring system first projects the image center and the four corners of the narrower-view image onto the dewarped panoramic image and then determines the pan angle θ_n, the tilt angle ϕ_n and the focal length f_n in panoramic environment after zooming into the close view. Next, we place a textured plate, specified by (θ_n, ϕ_n) and f_n and texture-mapped by the close-view image, in front of the panoramic model, as shown in Figure 15(c). During the hopping, we interpolate the pan angle, tilt angle and focal length from (θ_c, ϕ_c, f_c) to (θ_n, ϕ_n, f_n) and vary the alpha value from completely transparent to completely opaque. Figure 16(b) show an intermediate result as we hop from Figure 16(a) to the Figure 15(c). A video clip demonstrating the browsing and smooth hopping in the augmented panorama is also included in this submission.

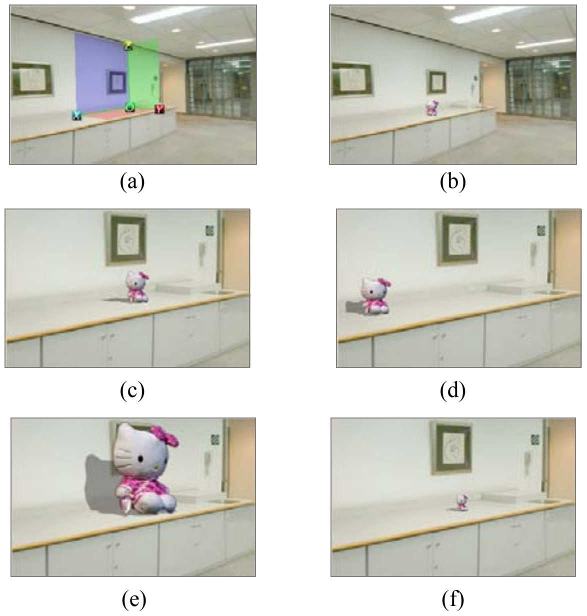

Figure 15: A few snapshots of a video clip demonstrating the authoring of an augmented panorama.

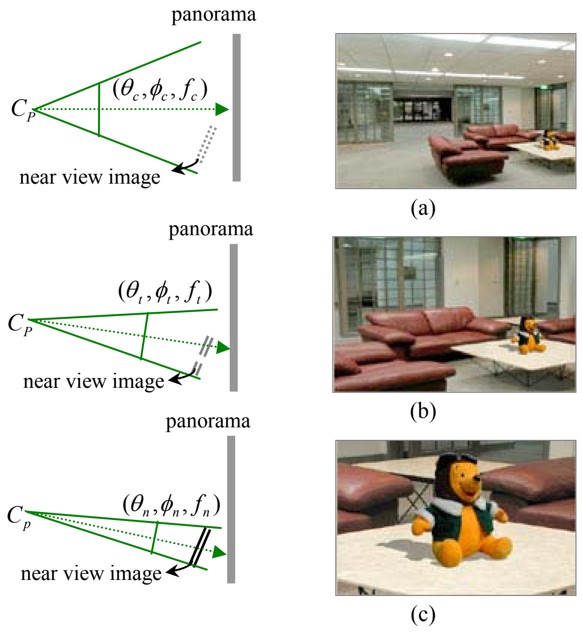

Figure 16: Smooth hopping from a wider view to a closer view. (a) *t=0.0*; (b) *t=0.6*, and (c) *t=1.0*

5. Conclusion and Future Work

In this paper, we have introduced an image-based approach to augmenting a panorama with OMs. The main focus of this paper is on how to maintain the visual coherence of the augmented panorama without reconstructing the 3D geometric models of the objects to be inserted. To achieve this goal, our method first selects the correct textured plate and shadow plate, and puts them in the correct place with correct orientations. Then, three layers of images: the background layer, the shadow layer, and the object layer, are rendered and combined to form the composite image required to show the augmented panorama. Based on the proposed method, we have implemented an easy-to-use system for authoring and for browsing augmented panoramas. Experimental results have shown that the composite images rendered by our system is visually plausible.

Our method can be easily extended to display stereo OMs in a stereo panorama[8]. Also, it can be used to insert objects into a panoramic video if we can track the cuboid structure in the panoramic video. Other future work includes casting shadow on non-planar surfaces and composing illumination-consistent augmented panoramas.

Appendix. Disparity-Based View Morphing for Novel View Generation

An important feature of our image-based approach is that no 3D geometric model of the object is needed. However, if a certain perspective of the object has not been recorded in the image acquisition stage, we can try to approximate it with the acquired images. The simplest method is to use the nearest neighbor directly, which usually gives satisfactory results, except for some jerky effect when the original image sampling is not dense enough. When this happens, we can adopt the disparity-based view morphing (DBVM) technique [6] [7] to generate an approximation of the novel view.

DBVM is an extension of View morphing that was originally proposed in [12] for generating realistic transition views for lateral camera motion. In this work, we use the DBVM to generate the novel view by using three nearest non-collinear neighboring views that are acquired in Section 2.1. DBVM can be divided into four steps: pre-warping, disparity estimation, linear morphing, and post-warping.

In the pre-warping step, we rectify the three nearest views to three parallel views without changing their projection centers. Let C1, C2 and C3 be the projection centers of the three views. The directions of the parallel views is chosen to be the normal of triangle $\triangle C_1C_3C_2$ and the x-axis parallel to vector $\overrightarrow{C_1C_2}$. In the step of disparity estimation, we adopt the method used in [6], except that an energy term of anchor points is added into the energy function for global optimization. The anchor points are either correspondence selected manually or highly reliable match obtained automatically.

It can be derived that the disparity transition is linear after the image rectification. Consider the position of the three view centers, and all the correspondences in the three views, as affine frames. We specify the view center of novel view using barycentric coordinates according to the affine frame of three views, and apply the same barycentric coordinates to affine frames of correspondences to warp source views. After warping the three views, we use the method proposed in [6] to blend the warped images. In the fourth step, we apply the barycentric coordinates of novel view to calculate its camera parameter according to the original three views, and get camera parameter of the novel view. Thus, the post-warping can be simply performed by the inverse process of pre-warping.

References

1. C. S. Chen and W. T. Hsieh, "Composition of 3D graphic objects and panorama," *Proceedings of ICAT2000*, Taipei, Taiwan, pp. 207-214, 2000.
2. C. S. Chen, C. K. Yu, and Y. P. Hung, "New Calibration-free approach for augmented reality based on parameterized cuboid structure," *Proceedings of ICCV'99*, pp. 23-37, Corfu, Greece, 1999.
3. S. C. Chen, "QuickTime VR: an image-based approach to virtual environment navigation", *Proceedings of SIGGRAPH'95*, pp. 29-38, 1995.
4. J. Gluckman, and S. K. Nayer, "Ego-motion and omnidirectional cameras," *Proceedings of ICCV'98*, pp. 999-1005, 1998.
5. S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen, "The lumigraph," *Proceedings of SIGGRAPH'96,* pp. 43-54, 1996.
6. H. C. Huang, C. C. Kao, Y. C. Lin and Y. P. Hung, "Disparity-based view interpolation for multiple-perspective stereoscopic displays," *Proceedings of SPIE Conference on Stereoscopic Displays and Virtual Reality Systems VII*, San Jose, California, Vol.3957, pp.102-113, January 2000.
7. H. C. Huang, S. H. Nain, Y. P. Hung, and T. Cheng, "Disparity-based view morphing – a new technique for image-based rendering," *Proceedings of VRST'98*, pp. 9-16, Taipei, Taiwan, November 1998.
8. H. C. Huang, Y. P. Hung, "Panoramic Stereo Imaging System withAutomatic Disparity Warping and Seaming," *Graphical Models and Image Processing*, 60(3), pp. 196-208, May 1998.
9. K. Kutulakos and S. M. Seitz, "A Theory of Space Carving," *Proceedings of ICCV'99*, pp. 307-314, 1999.
10. A. Laurentini, "The Visual Hull Concept for Silhouette-based Image Understanding," *IEEE. Transactions on PAMI,* pp. 150-162, 1994.
11. M. Levoy and P. Hanrahan, "Light Field Rendering," *Proceedings of SIGGRAPH'9*6, pp.31-42, 1996.
12. S. M. Seitz and C. R. Dyer, "View morphing," *Proceedings of SIGGRAPH'96*, pp.21-30, 1996.
13. R. Szeliski and H.-Y Shum, "Creating full view panoramic image mosaics and texture-mapped models," *Proceedings of SIGGRAPH'97*, pp. 251-258, 1997.
14. C. M. Tsai and C. S. Chen, "Finding Minimal Photo Hull for Image-based Rendering by Carving Space with Progressively Stricter Consistent Criterion," *Proceedings of IAPR Workshop on MVA*, pp.35-38, Tokyo, Japan, 2000.
15. Y. Wu, S. Iyengar, R. Jain, "A New Generalized Computational Framework for Finding Object Orientation Using

Perspective Trihedral Angle Constraint," *IEEE Transactions on PAMI*, Vol. 16, No. 10, pp. 961-975, 1994.

16. Y. Yagi, Y. Nishizawa, and M. Yachida, "Map-Based Navigation For A Mobile Robot With Omnidirectional Image Sensor COPIS," *IEEE transaction on Robotics and Automation*, 11(5), pp. 630-648,1995.