Light Field Rendering for Large-Scale Scenes

Hiroshi Kawasaki Katsushi Ikeuchi Masao Sakauchi
Institute of Industrial Science, University of Tokyo
6–4–1 Komaba, Meguro-ku, Tokyo, 153-8505, Japan
{h-kawa,sakauchi}@sak.iis.u-tokyo.ac.jp, ki@cvl.iis.u-tokyo.ac.jp

Abstract

In this paper, we present an efficient method to synthesize large-scale scenes, such as broad city landscapes. To date, model based approaches have mainly been adopted for this purpose, and some fairly convincing polygon cities have been successfully generated. However, the shapes of real world objects are usually very complicated and it is infeasible to model an entire city realistically. On the other hand, image based approaches have been attempted only recently. Image based methods are effective for realistic rendering, but their huge data sets and restrictions on interactivity pose serious problems for an actual application. Thus, we propose a hybrid method, which uses simple shapes such as planes to model the city, and applies image based techniques to add realism. It can be performed automatically through a simple image capturing process. Further, We also analyze the relationship between error and number of needed images to reduce the data size.

1 Introduction

To create a large-scale virtual environment, such as entire town or city, is one of the most important applications in computer vision and computer graphics. So far, model-based rendering (MBR) is usually adopted for this purpose. A MBR uses the geometry and surface attributes of objects to construct images from a given viewpoint. Thus, if we want to synthesize realistic environments with MBR, precise geometry and an accurate reflectance model is required. However, in practice, it is still difficult to acquire a reasonable level of detail of the surface reflectance model and geometries.

On the other hand, the image-based rendering (IBR) technique has recently become a major research topic in CV and CG, due to IBR's great potential for photo-realistic image synthesis. However, for practical use of IBR, little research has been done and few actual applications have been developed. One significantly important reason for this is its requirement of a huge data size.

Based on these facts, we propose a hybrid method, which uses simple geometric primitives to model the city, and apply the IBR technique to add realism. This method uses significantly smaller data sets than traditional IBR methods without degradation of quality. Further, it can be performed automatically through a simple image capturing process with a single scan along a street with a vehicle mounted multiple-configured cameras or single omnidirectional camera.

This paper is organized as follows. In Section 2, we explain the basic idea of our method with regards to related works, and in Section 3, describe the algorithm and the characteristics. Section 4 presents results of experiments using a real scene of a landscape of the city. Section 5 contains our conclusions regarding this method.

2 Technical Overview

Generating a 3-D virtual world directly from real scene images without analyzing a reflectance model or using an explicit 3-D model is a promising technique. This method, referred to as image-based rendering (IBR), creates new views by resampling those prerecorded pixels in a timely manner. "Aspen Movie Map"[9] was the pioneering work of this IBR technology. Another representative system based on IBR is "QuickTime VR"[4]. In terms of rendering large-scale scenes, Hirose's [6] and Takahashi et.al.'s [14] works are similar to our own. Takahashi et.al. applied a light field technique to generate new views and enlarge the possible view area. However, they still had singular directions which couldn't be rendered correctly, resulting in severe distortions in synthesized images. Also, their image capturing path was limited to a straight line. In terms of using the light field, our method is very similar to theirs, though our method provides a solution to overcome these limitations.

The light fields is one of the key concepts of IBR. The most general light field can be described by a 7D plenoptic function[1]. Ignoring time and wavelength,"Plenoptic Modeling"[10] is a continuous 5D plenoptic function. "Lumigraph"[5], "Light Field Rendering"[8] and "Ray-space method"[11] systems efficiently parameterize this 7D function into 4D. "Rendering with Concentric Mosaics"[13] is a 3D plenoptic function which, as its name suggests, creates concentric mosaics. Although these works successfully reduce the 7D data sets to a reasonable size, the viewing space is greatly restricted.

In terms of the data size of light field, “Plenoptic sampling”[3] reveals the tradeoff relationship between geometrical complexity relative to the focal length and the number of the needed images to synthesize images with comparable quality. Considering a large-scale scene, particularly an entire town, the geometrical variation is naturally very large, and therefore a huge number of original images are needed if we vary the viewing depth. Though our method is based on the light field, we efficiently use simple geometrical models to synthesize the images and successfully reduce the required number of sampled images more than every other related work to date. In the following section, we explain the details of our method.

3 Algorithm and Theory

As already stated in previous section, our method utilizes geometric primitives. Generally speaking, estimating depth from images is usually difficult and our purpose is to create the city automatically, therefore we model the city with simple geometric primitives such as rectangular planes perpendicular to the ground. Since simple texture mapping often lacks realism, we apply light field representation on the plane to add it back. In other words, with regard to actual implementation, we generate a view-dependent texture image for each plane and render with a traditional MBR method with texture mapping. In this section, first, we describe the image capturing method, then explain how to model the city and apply a light field technique to the plane respectively. We also evaluate the method and describe how we render the novel images and reduce the data size .

3.1 Image Capturing process

If the camera position of the captured images is not limited on the straight line, the image capturing process is straightforward. We simple put an omni-directional video camera on the roof of a vehicle equipped with GPS and a gyro-scope sensor, and scan the city. When we generate light field data from these captured images, we need an accurate camera position and a view direction correlated to the image. Therefore, the image and other sensor’s data should be synchronized precisely.

With regard to the image capturing device, we assume two types of omni-directional cameras as defined in the following.

3.1.1 Single Camera with Hyperboloidal Mirror

The simplest and easiest method for capturing panoramic images is to use an single camera with hyperboloidal mirror[12]. This type of camera has two foci and a single effective viewpoint (see Fig.1-(a)). From the sensed omni-directional image, we can generate pure perspective images, and, thus can make panoramic images from the omni-directional image.

Using one omni-directional camera, we can take images with 360 degrees in a horizontal direction; those images cover the northern hemisphere of a viewing sphere. The images cover the directions above the image plane of the omni-directional camera.

3.1.2 Cylindrically-Configured Multiple Cameras

Another method for capturing panoramic images is to arrange some cameras cylindrically as shown in Fig.1-(b) . These cameras’ optical axes intersect at one point. So these perspective images are projected to cylindrical coordinates, and these cylindrical images are stored at each location.

Hereafter ‘panoramic image’ means both of these two images.

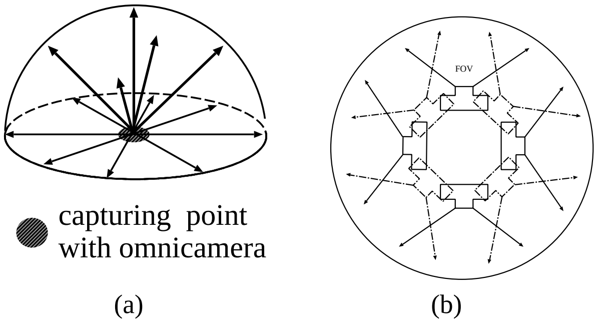


Figure 1. (a)Capturing with an omni-directional camera (b)Configuration of cameras to obtain panoramic images

3.2 Plane-based Data Structure

Because our purpose is to synthesize large-scale scenes, and such scenes naturally consist of a large number of complicated geometrical primitives, reasonable simplification and good approximation of the geometry is crucial. In this paper, we assume that the city consists of simple geometrical primitives such as rectangular planes perpendicular to the ground. We classify this plane into three types, as follows (see also Fig.2).

- Parallel plane · · · This plane is parallel to the vehicle’s running direction and usually represents the front face of buildings.

- Vertical plane · · · This plane is perpendicular to the vehicle’s running direction and usually represents the side of buildings and the front face of buildings which face the street vertical to the running direction.

- Infinite plane · · · After construction of the parallel and vertical plane, if the camera’s view direction does not intersect any existent planes, we make an infinite plane

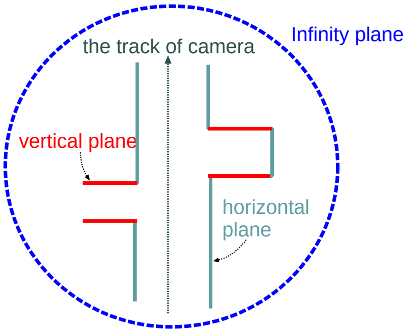


Figure 2. Geometrical model of city

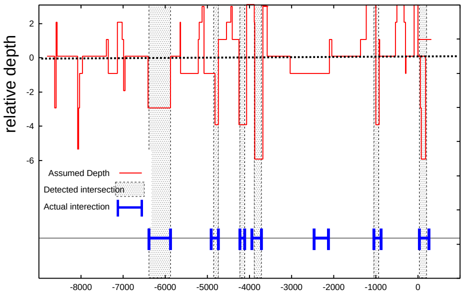


Figure 3. Estimated depth.

in that direction. If the route of the vehicle is straight, and there are no buildings ahead, the infinite plane is created in the running direction; this direction is defined as singular direction in [14].

Based on these assumptions, we can make the simple geometrical model of the urban city from captured image sequences.

3.2.1 Acquiring Planes from Video

In the past, various research has been conducted regarding the acquisition of 3-dimensional (3D) information of a townscape. For example, if a camera is fixed in the direction of movement and analyzes the cross section of the spatiotemporal image, this image is called EPI(epipolar plane image) [2], and is suitable for urban cities where most structures consist of planes. However, in an actual city environment, it is difficult to acquire 3D information due to many obstacles such as telephone poles or trees, or buildings which consist of complicated structures other than planar surfaces.

In this paper, we use extended EPI method [7] which is specialized for urban cities. With this method, the actual building's boundaries are detected robustly; these boundaries describe straight lines on the EPI plane. Therefore, the detected boundaries can be used as the side of the rectangular plane.

The depth information of the plane acquired by this method is shown in Fig.3. Although this method still produces some error, it can acquire depth information to a satisfactory degree. In particular, deep depths such as intersections can be detected with high stability by setting up a good threshold.

3.3 Light Field Representation

3.3.1 Light Field Data Construction

Given a series of panoramic images on a car trail, we can construct a light field data as follows. First, we employ a perspective projection of the image at the point P, as illustrated in Fig.4 to the estimated plane of the city and we can easily create a texture image for the estimated planes. Since the panoramic images are usually densely sampled by video camera, a large number of images are created for an identical plane; these images are dependent on the viewing direction. Therefore, secondly, we create another database to record the view direction information for each texture image. Certainly, if the assumed depth is correct and the surface model is simple as lambertian, we only need a few textures for the plane. Even if this is not the case, the texture data is usually oversampled, so thirdly, we try to reduce the large number of texture images without degrading the quality of the synthesized image.

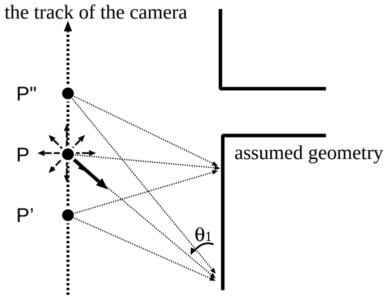


Figure 4. Construction of light field data

3.3.2 Light Field Data

The light field data consists of set of images and accompanying data which contain view information to identify the specific image. Fig.5 shows the actual light field data of our method.

3.4 Rendering

Once the light field data is obtained, we can easily construct a novel view from arbitrary viewpoints. Consider the case of rendering a novel image from the point P, as illustrated in Fig.6. For constructing the view from P, we need rays around P from R_s to R_e as shown in the figure. The ray across P also crosses the planes of the city model, so we

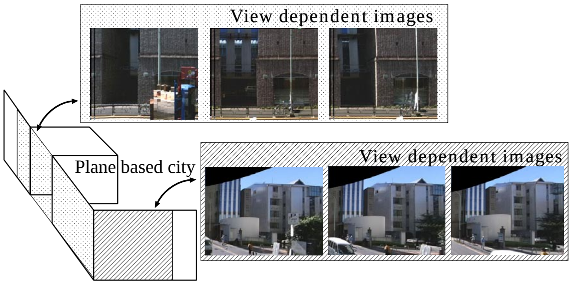


Figure 5. Light field data with view-direction data

can substitute these rays for those across the plane's texture images dependent of the view direction. Therefore, by finding the slits of the planes via calculating the view-directions corresponding to the rays across P and then collecting those slits, we can synthesize a view-dependent texture for the plane. After synthesizing a view-dependent texture, a simple MBR can be applied for final rendering. Another merit of using MBR is that we can easily add another object in the virtual environment.

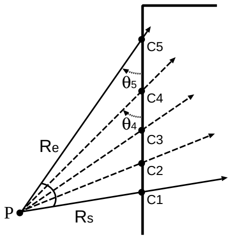


Figure 6. Rendering novel view

3.5 Distortion and Error Evaluation

In the synthesized images, two main distortions exist. The first is the vertical distortion and is a well-known distortion frequently encountered with similar IBR methods [13, 14]. The second is the horizontal distortion, derived from sparse sampling; we discuss this distortion in next section, as it is closely related to image reduction.

According to "Plenoptic sampling"[3], even if we can not estimate the correct depth, we can render correct images by using a large number of sampled images. However, with our image capturing process, we cannot increase the number of sample images in the vertical direction. Therefore, just as the concentric mosaics[13] and Takahashi's work[14], this system has an inevitable vertical distortion when the estimated depth is incorrect.

Parallel and vertical plane Actual buildings and objects in the real world have complicated structures and we simplify these geometries with planes, and thus the depth differences between real objects and estimated planes create distortions. Fig.7 represents a vertical plane containing the viewing direction. In this figure, we define the image capturing position as the coordinate origin, Z_{view} as view position, Z_{actual} as the object's position and $Z_{virtual}$ as the estimated depth of the plane, respectively. We also define θ_{view} as the zenith angle of the view direction and θ_{diff} as the error angle of the view, respectively. θ_{diff} can be calculated as:

$$\theta_{diff} = arctan(\frac{1+r_o-r_n}{(1+r_o)(1+r_n)}tan(\theta_{view})) - \theta_{view} \quad (1)$$

where $r_o = \frac{Z_{virtual}-Z_{actual}}{Z_{actual}-Z_{view}}$ and $r_n = Z_{view}/Z_{actual}$.

To evaluate the error, we insert actual values into this function. These values are defined as follows.

- allowable error angle as 0.5 $deg.$
- maximum field of view as 60 $deg.$
- $-0.2 < r_n < 0.2$. (our supposed system is a navigation system and the viewing position is inside the street)

Then θ_{diff} describes the curves in Fig.8(a), with various values of r_o. In order to confine θ_{diff} within ±0.5 $deg.$, we set the maximum r_o to 0.1. Therefore, the vertical distortion of the parallel plane usually does not affect the synthesized image by the geometrical detail of the buildings, but if the depth of the building itself is incorrect, distortion occurs.

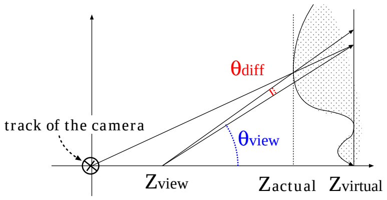
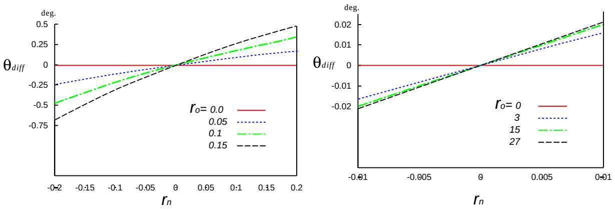


Figure 7. Vertical distortion



(a) (b)

Figure 8. Evaluation of vertical error

Infinite plane With our method, we define the geometry which exists further away from the threshold as an infinite plane, therefore we put other values to r_n for evaluation.

- $-0.01 < r_n < 0.01$. (infinite plane is far away from the current viewing area)

Results are shown in Fig.8(b). In any case of r_o, θ_{diff} is small enough, and therefore the vertical distortion of the infinite plane usually does not affect the synthesized image if the actual object is located reasonably far away from the camera.

3.6 Reduction of View-dependent Image

As mentioned in the previous section, there is a good potential to reduce the number of view-dependent images without degrading the quality of the synthesized image. In this paper, we apply a simple method to remove the over-sampled images as follows.

1. Define one of the view-dependent texture images as standard; at first, we adopt the image from a viewing direction is perpendicular to the plane

2. Calculate the correlation value between the standard image and adjacent image, and if the value is larger than threshold, remove that image

3. Iterate **2**, until the correlation value is smaller than threshold value. Then set this image as the new standard

4. Iterate whole process for all view-dependent ($\theta > \theta_{thre}$) texture images.

5. Remove all texture images for($\theta < \theta_{thres}$).

Threshold (θ_{thres}) in **4, 5** can be calculated as follows. Fig.9 represents a horizontal plane. In this figure, we define the image capturing position as the origin, and other parameters are the same as Fig.7. This time, we define θ_{view} as the azimuth angle of the view direction and θ_{diff} as the error of view angle, respectively. θ_{diff} can be calculated as:

$$\theta_{diff} = arctan(\frac{(1 + r \cdot tan(\theta_{thres}))tan(\theta_{view})}{1 + r \cdot tan(\theta_{view})}) - \theta_{view} \quad (2)$$

where $r = b/a$ and θ_{thres} represents the threshold mentioned above. To evaluate the error, we insert values into this function defined as follows.

- allowable error angle as 0.5 $deg.$

- maximum field of view as 60 $deg.$

- $r < 0.5$(our method can robustly detect the large depth)

Then this function describes the curves in Fig.10(a)and(b). In Fig.10(b), a $20deg.$ curve is described under the allowable error angle $0.5deg.$, therefore the threshold can be defined as $20deg.$ this time.

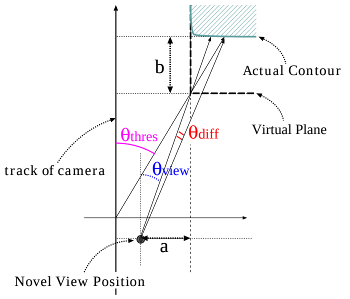
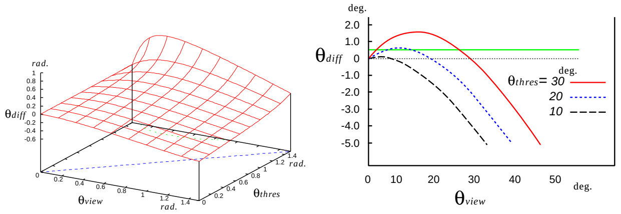


Figure 9. horizontal distortion



(a) (b)

Figure 10. Evaluation of the threshold:(b) is a $y - x$ plane of (a)

Fig.11 shows an example of the correlation value for the planes. There is a dent in the middle of the buildings where the correlation value is lower than other locations. Therefore, we cannot remove the images for middle planes.

We applied this method to the actual data set and even though we defined a high threshold, the data size was reduced from 335MB images to to 74MB(22%) without any compression algorithm.

4 Experiments and Demonstrations

We performed several experiments to test the effectiveness of our method. In the following two experiments, we use an outdoor scene, located in the landscape of the town Shinanomachi, with panoramic images captured by multiple cameras arranged cylindrically on the car, as stated in sec.**3.1.2**. Fig12 is one of those synthesized panoramic images. The car moves along a public road in Shinanomachi, and captures images. At the same time, the capturing positions of those images are recorded by GPS. Sec.**4.3** shows a snapshot of the actual application for the MR system.

4.1 Parallel plane

In this experiment, we rendered the novel images of the parallel plane. New viewing positions and other object lo-

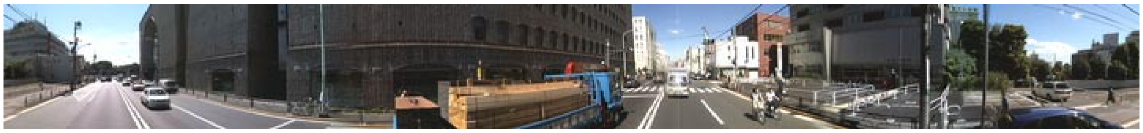


Figure 12. Sample image of input panorama sequence

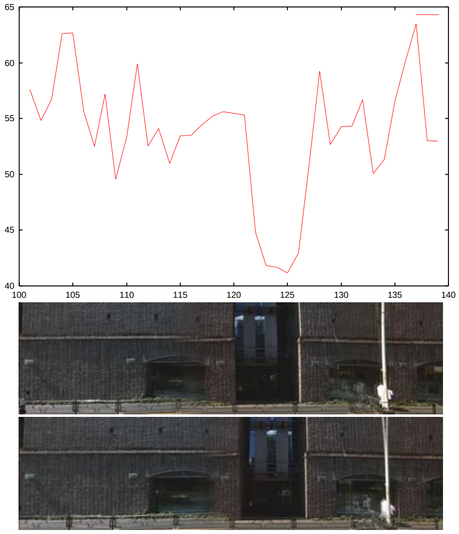


Figure 11. Correlation between 2 images

cations are shown in Fig.13(d). Note that these images are all rendered, and are not from the image capturing line. Fig.13(a),(b),(c) and (e) shows a series of rendered images viewed from the virtual camera positions and directions by our proposed method. When see the images(a)(b) and (c), we can see that the pole is rendered at the right position, dependent on the camera position. Fig.13(e') shows a rendered image with Takahashi's method[14] with the same camera position and direction as (e). Comparing these two results, the differences are apparent. Besides, there are none of the typical artifacts with our method, although there are vertical distortions and discontinuities at the slit joints caused by the sampling frequency in Fig.13(e').

4.2 Vertical plane

In this experiment, we rendered novel images of the vertical plane. New viewing positions and other object locations are shown in Fig.14(d). Fig.14(a),(b) and (c) show a series of rendered images viewed from the virtual running line by our proposed method. In these figures, we can see that the pole and signboard are rendered at a certain place, which is dependent on the camera position. Fig.14(e) shows a rendered image with Takahashi's method. Similar to the result of the parallel plane, we observe no conspicuous artifacts in the image with our method, although they are abundant in Takahashi's.

4.3 Omni-directional Camera Image

Fig.15-(a) and (b) are images of our campus captured by the HyperOmni-directional camera[12]. From this camera's characteristics, we can easily generate a perspective image from omni-directional image. Fig.15-(c) is a part of perspective image which was generated from Fig.15-(a).

By using this camera, we can capture the whole northern hemisphere at one time, therefore all of the tall buildings can be captured; the upper story of the buildings can not be captured by cylindrically-configured cameras. Fig.15(d)(e) and (f) are the rendered images from novel view positions. We can see that the top of the tall building is successfully rendered as well as the parallax of the objects.

4.4 3D City

The mixed reality (MR) system is now one of the most promising applications for vision research. We will show a sample application for the MR system, which is effectively achieved by using our light field data and rendering method. Fig.16 shows the sample snapshot of a virtual city and we can see that the texture of the singular direction and side view of buildings is successfully rendered in the same scene.

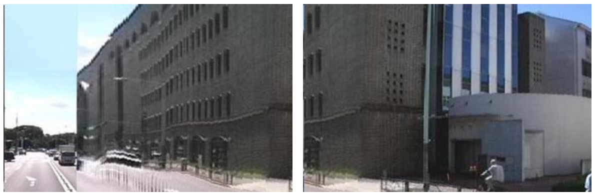
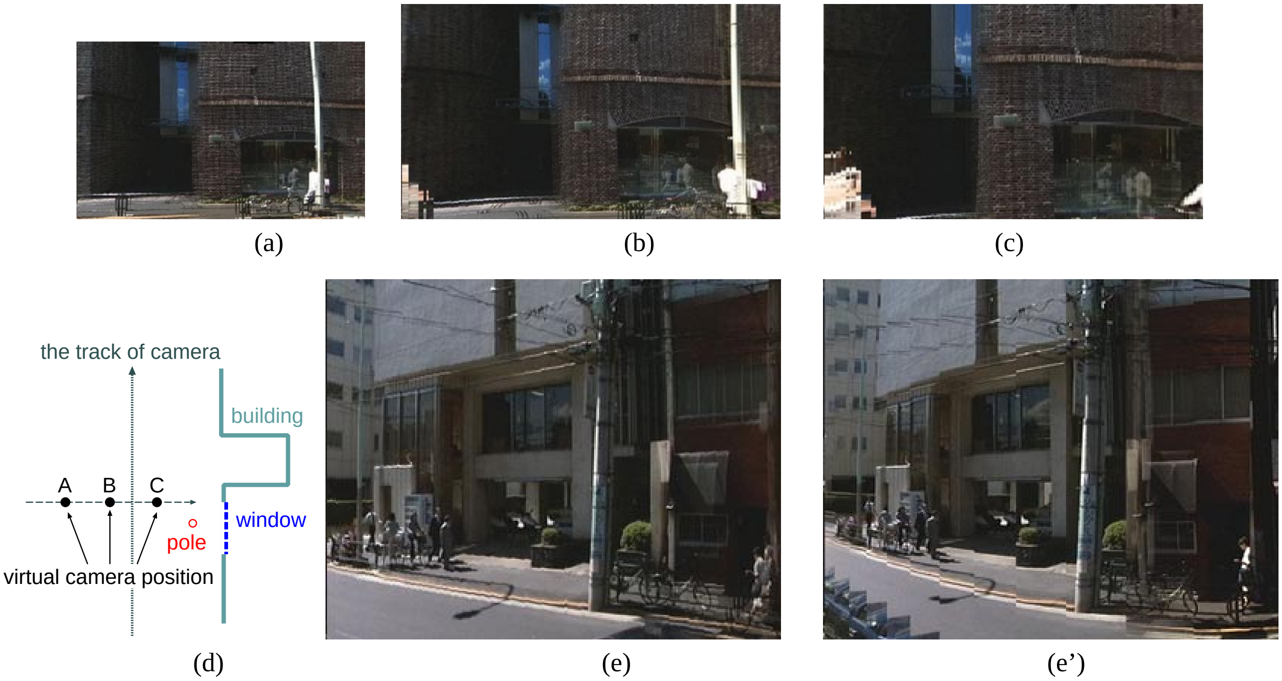


Figure 16. Snap shot of Virtual 3D map

5 Conclusion

In this paper, we proposed the hybrid method of light field technique and model based approach to reconstruct a novel view of large-scale scenes. This method, using simple geometry, can significantly reduce the image data size without degrading the quality of the synthesized image and can also modify the vertical distortions effectively. Further, our method, unlike traditional IBR, utilizes geometrical models and is more suitable for MR systems.


(a) (b) (c)

(d) (e) (e’)

Figure 13. (a)(b)(c)(e): Rendered images by our method,(d):camera position. (e’): rendered images by other method

To capture images, we simply install a omni-directional video camera onto a vehicle and scan along a street freely. Once the light field data is generated, the rendering process is straightforward. We only make view-dependent textures for each plane and render with common model based techniques.

To test the effectiveness of our proposed method, we conducted several experiments using real world panorama image sequences. The result of the experiments show the effectiveness of our proposed method to render novel images without distortions.

Future work includes the development of a driving simulator for the entire city of Tokyo using this method for ITS purposes.

References

[1] E. H. Adelson and J. Bergen. *The Plenoptic function and the elements of early vision*. MIT Press Cambridge, MA, 1991.

[2] R. Bolles, H. Baker, and D. Marimont. Epipolar plane image analysis:an approach to determining structure from motion. *Int.J.of Computer Vision*, 1:7–55, 1987.

[3] J. Chai, X. Tong, S.-C. Chan, and H.-Y. Shum. Plenoptic sampling. *ACM SIGGRAPH*, pages 307–318, July 2000.

[4] S. E. Chen. Quicktime vr - an image-based approach to virtual enviroment navigation. In *Proceedings of ACM SIGGRAPH’95*, pages 29–38, 1995.

[5] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen. The lumigraph. *ACM SIGGRAPH*, pages 43–54, 1996.

[6] M. Hirose and E. Takaaki. Building a virtual world from the real world. In *Proceedings of International Symposium on Mixed Reality*, pages 183–197, Mar. 1999.

[7] H. Kawasaki, T. Yatabe, K. Ikeuchi, and M. Sakauchi. Construction of a 3D city map using EPI analysis and DP matching. In *Asian Conference on Computer Vision*, volume 2, pages 1149–1155, Jan. 2000.

[8] M. Levoy and P. Hanrahan. Light field rendering. *ACM SIGGRAPH*, pages 31–42, 1996.

[9] A. Lippman. Movie-maps. an application of the optical videodisc to computer graphics. In *Proceedings of ACM SIGGRAPH ’80*, pages 32–43, 1990.

[10] L. McMillan and G. Bishop. Plenoptic modeling: An image-based rendering system. *ACM SIGGRAPH*, pages 39–46, 1995.

[11] T. Naemura and H. H. et al. Ray-based creation of photo-realistic virtual world. In *Virtual Reality and MultiMedia (VSMM’97)*, pages 59–68, 1997.

[12] Y. Onoe, K. Yamazawa, H. Takemura, , and N. Yokoya. Telepresence by real-time view-dependent image generation from omnidirectional video streams. In *Computer Vision and Image Understanding,Vol.71,No.2*, pages 154–165, 1998.

[13] H.-Y. Shum and Li-Wei-He. Rendering with concentric mosaics. *ACM SIGGRAPH*, pages 299–306, 1999.

[14] T. Takahashi, H. Kawasaki, K. Ikeuchi, and M. Sakauchi. Expanding possible view points of virtual environment using panoramic images. In *Computer Vision and Pattern Recognition*, volume 2, pages 296–303, June 2000.

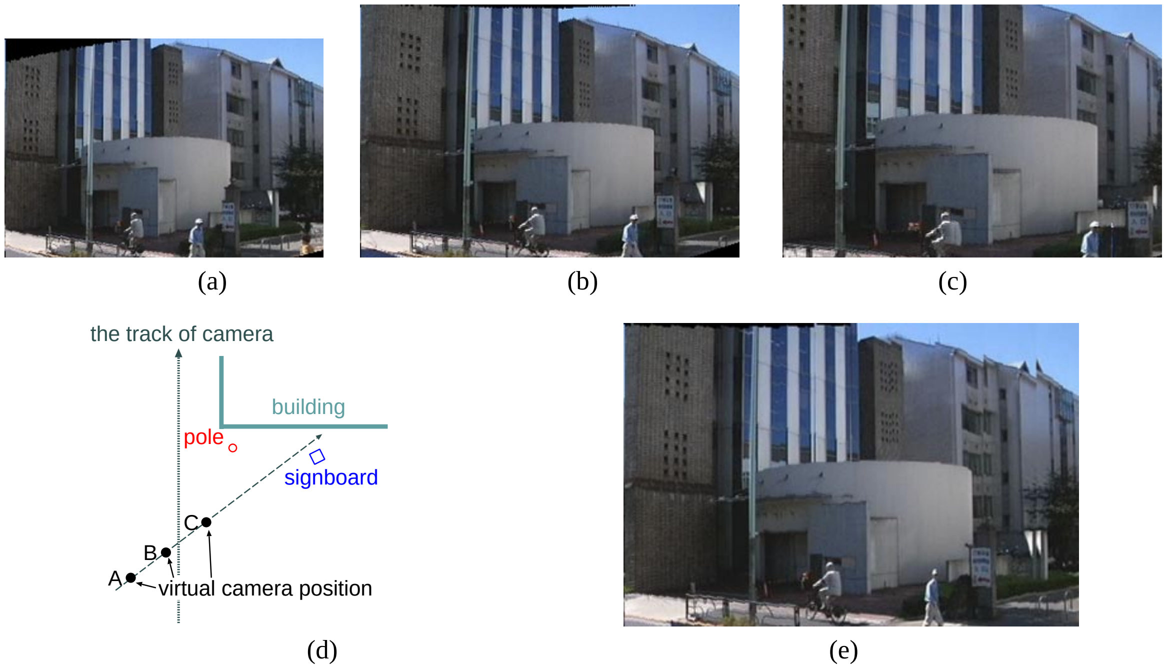


Figure 14. (a)(b)(c): Rendered images by our method,(d):camera position. (e): rendered images by other method

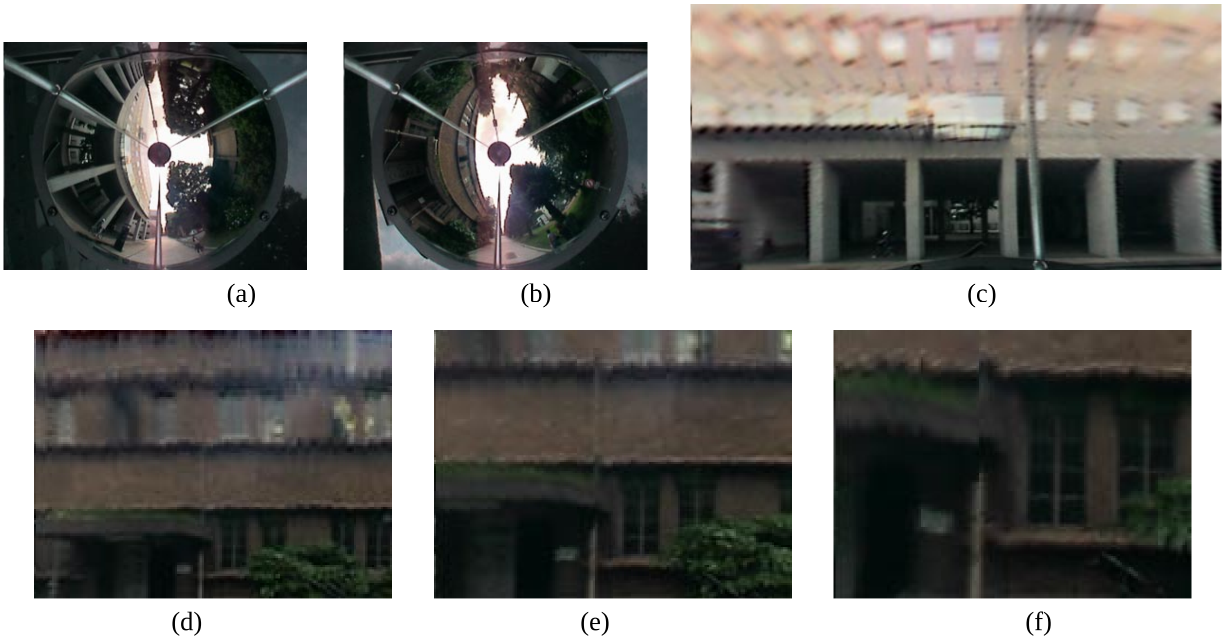


Figure 15. (a)(b): Captured images, (c): Perspective transformed image, (d)(e)(f): rendered images by our method. Note that the relation between the bush and the window is changing, dependent on the camera position.