3D SCENE EDITING
BY RAY-SPACE PROCESSING

Lei LV, Tomohiro YENDO*, Masayuki Tanimoto*,Toshiaki FUJII***

*Graduate School of Engineering,
Nagoya University,
Furo-cho, Chikusa-ku, Nagoya, 464-8603,JAPAN
E-mail: lvlei@tanimoto.nuee.nagoya-u.ac.jp
{yendo,tanimoto}@nuee.nagoya-u.ac.jp

**Graduate School of Science and Engineering,
Tokyo Institute of Technology,
2-12-1 Ookayama, Meguro-ku, Tokyo, 152-8550 JAPAN
E-mail:fujii@it.ss.titech.ac.jp

ABSTRACT

In this paper we focus on EPI (Epipolar-Plane Image), the horizontal cross section of Ray-Space, and we propose a novel method that chooses objects we want and edits scenes by using multi-view images.

On the EPI acquired by camera arrays uniformly distributed along a line, all the objects are represented as straight lines, and the slope of straight lines are decided by the distance between objects and camera plane. Detecting a straight line of a specific slope and removing it mean that an object in a specific depth has been detected and removed. So we propose a scheme to make a layer of a specific slope compete with other layers instead of extracting layers sequentially from front to back. This enables an effective removal of obstacles, object manipulation and a clearer 3D scene with what we want to see will be made.

Keywords: Ray-Space, EPI, 3D Scene Editing

1. INTRODUCTION

The data acquired by multiple cameras from multiple viewpoints can be parameterized in a single function called the plenoptic function introduced by Adelson and Bergen in [1] with the goal of describing what one sees from an arbitrary viewpoint in space. It can therefore be characterized with seven dimensions namely the viewing location and direction, wavelength and time. A particular case is acquired by fixing time and wavelength and reducing the viewpoint to a line. Under this circumstance, the plenoptic function reduces to a 3-dimensional function also know as the EPI volume [2]. On the EPI, the object is represented as a straight line, with a slope inversely proportional to its depth. This character makes the removal of lines with specific slope and the removal of objects same meaning and a coherent function for processing all the images simultaneously in 3D possible.

EPI was first analyzed in the original work of Bolles et al. in [2] where it was shown that 3D information of a scene can be recovered by finding lines in the EPI. In [7], it is suggested that the segmentation of EPI into coherent regions can be beneficial for numerous applications such as scene interpretation, object manipulation, obstacle removal and compression [8]. In order to segment the data, the authors introduce the notion of EPI tube. An EPI tube is obtained by gathering a collection of lines in the EPI volume that have similar slope or belong to the same layer. And In previous research [6], whether all the pixels in a specific slope belong to a single object or not is judged from threshold of variance. This method makes some parts of images where pixels have drastic variation blurred and details are lost.

In this paper we propose a scheme to make a layer of a specific slope compete with other layers instead of extracting layers sequentially from front to back. Moreover, we divide an EPI into several parts so we can process these parts independently in order to avoid the influence of other parts. When we process all the images taken from camera array, we use Ray-Space method [5].

This paper is organized as follows: In Section 2 we give a brief introduction to Ray-Space method. We then describe the EPI tube detection algorithm we proposed in Section 3. In Section 4, we talk about how to edit a 3D scene by EPI extraction. The results based on real data will be shown in Section 5. Finally, we conclude in Section 6.

2. RAY-SPACE METHOD

Ray-Space method is an Image Based Rendering (IBR) technique to generate arbitrary views. In Ray-Space, we describe three-dimensional space by using ray information transmitted in space without considering the object's geometry information. We can choose parameters to describe the ray based on our need. But here, we consider a parameter suitable to the situation in which camera array is uniformly distributed along a line. First of all, we assume that light radiates straight without attenuation in three-dimensional space. Then, one ray is possible to describe uniquely according to four parameters, position (*x*, *y*) and direction (*θ*, *φ*). Assuming the function that represents optical strength of this ray to be *f*, we can represent ray information by *f*(*x, y, θ, φ*).

$$f(x,y,\theta,\phi), -\pi \le \theta \le \pi, -\frac{\pi}{2} \le \phi \le \frac{\pi}{2} \qquad (1)$$

We consider camera array as an equipment to capture this ray information. We regard a camera as the equipment that collects rays, not a recorder of images. To understand better, we consider the Ray-Space *f*(*x*, *θ*) where we ignore vertical coordinate *y* and *φ*. Fig 1 shows the relation between a real space and a Ray-Space. In Fig.1, camera array is uniformly distributed along a line, and information recorded by these cameras is saved as Ray-Space data by arranging according to a specific rule. Rays pass through the reference plane at position *x*, with the direction *θ* in real space. θ_1 and θ_2 are defined in the direct way of angles. Assuming that image pixel is *u* and camera position is (*X*,*Z*), equation(2) show us this relation from geometric relation. Arranging captured images straightly constitutes Ray-Space.

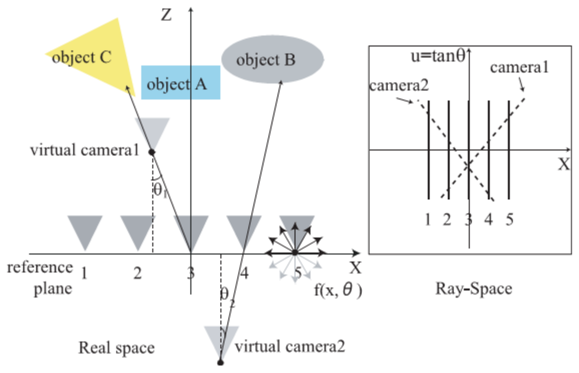


Fig.1: Relation Between Real Space and Ray-Space

$$u = \frac{1}{Z}(X - x) = \tan\theta \qquad (2)$$

Next, we explain how to generate an arbitrary image. Captured image of camera (*X*,*Z*) is to collect rays pass through the camera. Equation(3) show us the relation between these rays gatherings and pixels.

$$x = X + Z\tan\theta, u = \tan\theta \qquad (3)$$

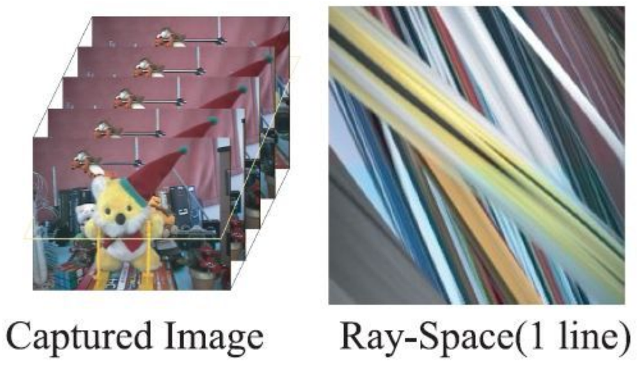


Fig.2: Relation Between Captured Image and Ray-Space

Given the camera position and ray direction, we can settle on the transit position on the reference plane. This is equivalent to hew by straight line in Ray-Space. Moreover, transforming captured image from real space to Ray-Space becomes possible by arranging images in order as Fig.2. In Fig.2, the left side is arranged images, and the right side is Ray-Space, horizontal cross section(EPI) from arranging images captured.

3. PROPOSED METHOD

3.1 Acquiring EPI

EPI is the horizontal cross section of Ray-Space. So if we take photography on a 3D scene for Ray-Space, we can acquire EPI simultaneously. In fact, if we take out one line in the same height of every pictures taken by the cameras parallel with each others and arrayed by the same interval, and put them from the top down orderly like Fig.3, EPI has been acquired. So we can conclude that the height of EPI is the same with number of pictures, and the width of EPI is the same with width of pictures taken. And the number of EPIs we can get is the same with the height of the pictures taken.

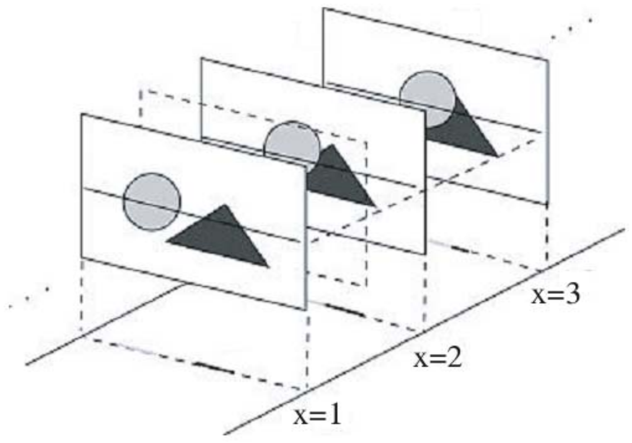


Fig.3 How We Acquire EPI

3.2 About EPI

As we talked about above, EPI is the gathering of many straight lines with different slopes, for example Fig.4. These lines represent the objects, and the loci of lines represent the change of position of objects respectively. This change is occurred among all the pictures. So we can be aware that if the object is far away from the camera plane, change of position among pictures will be small, and slope of line will be inclined to vertical. If the object is close to the camera plane, change of position among pictures will be large and slope of line will be inclined to horizontal. In this research, we will take full vantage of this characteristic to detect the line having any slope so that we can detect the object in any depth, and we can manipulate objects we want and we can edit multi-view images to fill our need to get the image just obtaining what we want to see.

3.3 Proposed Method

As we see, occlusions and disparity are closely related since a point with a larger disparity (closer to the camera plane) will occlude a point that is further away. In previous research, the way that computing variance of all the pixels in a specific slope and compare it with a threshold is used to detect lines and judge whether all the pixels in a specific belong to single object or not. However, this method suffers from serious mistake when there is much noise or the part of object occluded has drastic. Generated images become blurred and details are lost.

To solve this problem, we propose a scheme to make a layer of a specific slope compete with other layers having different slopes instead of extracting layers sequentially from front to back using threshold judgment.

The characteristic of EPI is that all the objects are represented by EPI tubes. The slopes of the lines are decided by the depth (distance to camera plane) of the object which lines represent. So, if we are successful to detect the line with an arbitrary slope, it means that we are successful to detecting the object which exists in a real space.

All the pixels in a line of EPI tube is the same thing in different place. Since they are all the same, finding a parameter which measure similarity of all members will help us make clear that whether all the pixels we check belong to a single object.

The parameter often used to measure similarity and difference is variance. The variance is defined as follows. Here, $x_1, x_2, ..., x_N$ is the intensity of each pixel in a specific slope, and N is the total number of pixels counted following that slope from top down. $\bar{x}$ is the average of all x.

$$\sigma^2 = \frac{1}{N}\sum_{k=1}^{N}(x_k - \bar{x}) \tag{4}$$

Before we compute variance, we need to do a pre-work. That is to find how many kinds of slopes exist in the image, and value of these slopes are necessary as well.

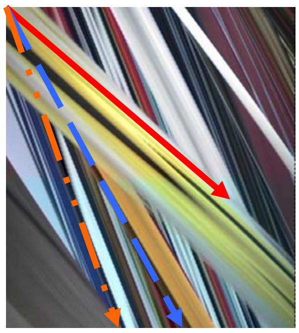


Fig.4: Characteristic of EPI

Then beginning with a pixel in line, we will calculate the variance of all pixels following the slopes we got, like Fig.5. When we use one slope data, we will get a variance of all pixels in that slope. *N* kinds of slopes have *N* kinds of value of variance. When we finish calculating variance of all kinds of slope, we will compare them and find the minimum. The slope corresponding to minimum will be saved and used next in that spot next. Don not forget to scan all pixels from left to right in line one to cover all pixels in image.

4. EXPERIMENT OF 3D SCENE EDIT

4.1 Experiment Condition

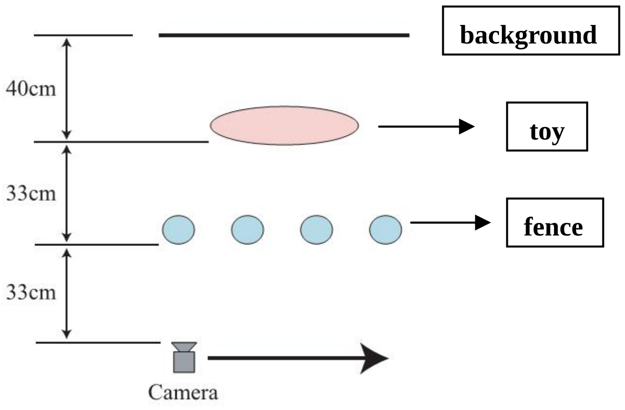
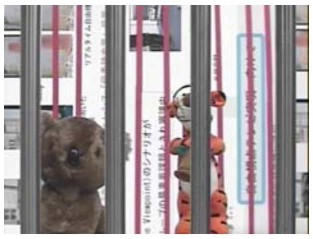


Fig.5: Experiment Condition

In our research we will process Ray-Space directly and we assume that we have previously acquired the Ray-Space. So we need to take photography densely, and use a lot of pictures. This time in our experiment we used the system shown in Fig.5. While camera is moving along the horizontal direction, we take pictures in the interval of every 0.33mm. The scene is like that: there are some fences in the most front, next two stuffed toys in the middle, and there is background behind. Totally we took 480 pictures at the size of 640×480.So we know that will get 480 slices of EPI at the size of 640×480.

4.2 Object Removal



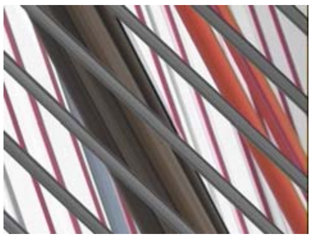


Fig.6: One Picture Captured

Fig.7: One EPI

One picture taken is shown in Fig.6. And one EPI corresponding to this scene (at height of 480) is shown in Fig.7. Our first goal is to get rid of obstacle fence in the front. Of course, we need to find EPI tube representing fence and remove them in all EPI.

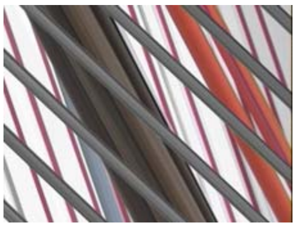
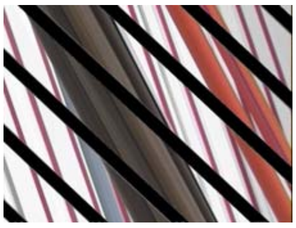



Fig.8: Detecting Fence Fig.9: EPI without Fence

There are 3 kinds of slopes in this scene (fence, toy, background).Firs, beginning with one pixel in first line, we compute variance using 3 kinds of slopes like Fig.7. Then we compare 3 kinds of variance to make sure which one is fence. When we finish scanning from left to right, we have found the range of fence. We can see the EPI which fence is taken off in Fig.8.

Until now, we have succeeded to get rid of fence. However, the spot which has been taken off just becomes black and has nothing. So we have to find the content it is supposed to be and fill it. Here we need to calculate variance using slope of toy and background again so as to avoid influence of fence. We use the result to which spots need to be filled with “background” and which spots need to be filled with “toy”, like Fig.9. And we use average of all pixels in that slope to fill in the blank.

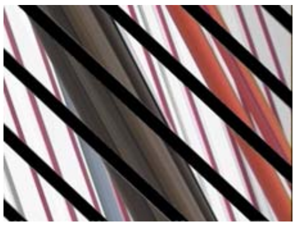
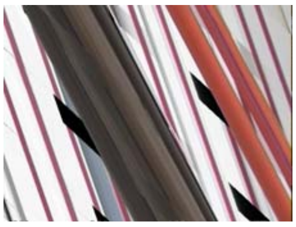



Fig.10: Decide How to Fill Fig.11: Blank is Filled

In the Fig.10, we can see that most of pixels which need to be filled have been filled. But there is still some part we have to fill, and now we know they belong to background so we use average of pixels of background to fill. We can see final result the fence has been got rid of in the image in Fig.11.

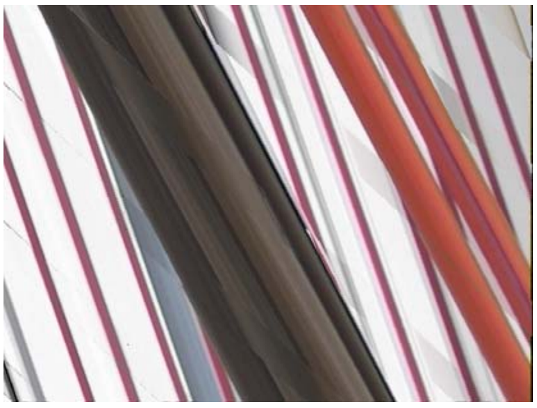


Fig.12: EPI Which Has Been Edited

4.3 Edit EPI Freely

Of course, what we can do is not just to take off the fence. EPI is good tool for us to get 3D information. Here what I want to propose is to edit EPI freely. By doing that we can actually choose what we want to see in the picture and what we do not want can be thrown away. So when we enjoy the free-viewpoint image, we can not only choose the viewpoint we like but also the object we like as well.

For example, if we just want to see the koala, we can use the method mentioned above just changing the value of slope. Because the koala and tiger are in the same depth, you have to be careful that you have to limit your scan range to avoid taking koala simultaneously.

For example, if we would like the toys to be away but the fence to stay in the picture, it is OK. First we have to take fence away temporarily because it is in the front. Keeping it in the picture will make us unable to edit other object behind. After we finishing editing other objects. Of course, the case that left side is koala and right side is fence is totally realizable. What we have to do is limit the scan arrange.

Because when we generate EPI, we arrange pictures orderly by the position of camera, getting different image in different viewpoint is also possible by editing some parts of all the EPIs only. For example, if we just edit top half part of EPIs, we can just get pictures just half of them changed. Where need to be changed, but where need not, is just your call.

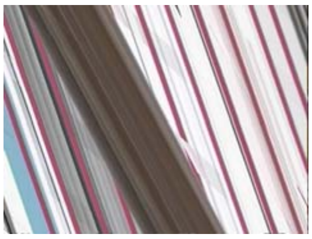
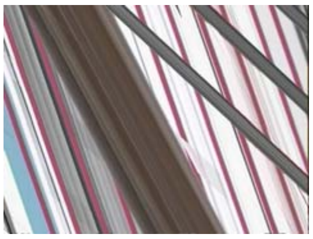



Fig.13: EPI without Tiger and Fence Fig.14: EPI with Koala and Half of Fences

4.4 Recoverable and Not Recoverable

In some occasion, in fact we can not get back what we want. That is because when we take photos, we have not got the information we want.

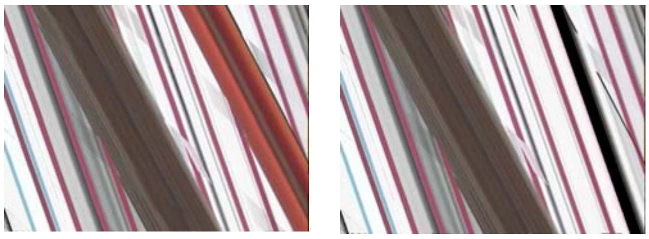


Fig.15: Part We Take off Can not be Recovered Completely

So even if we can take off what we want to throw away, maybe we are unable to fill in those blanks and just have to leave them black. Expanding range of taking photos may help us to solve this problem.

5. RESULT

We can see the comparison between previous method and the method I proposed below. And we can find that my method is more effective in editing EPI. And when we generate free-viewpoint image using edited EPI, my method is able to make us get a clearer scene.

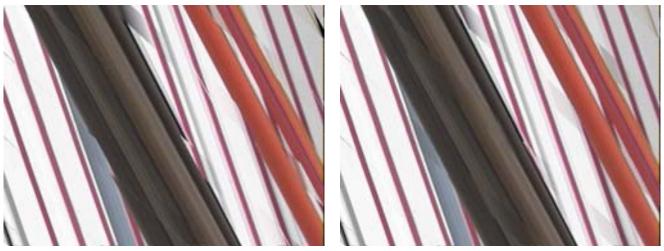


Fig.16: A Comparison of Edited EPI

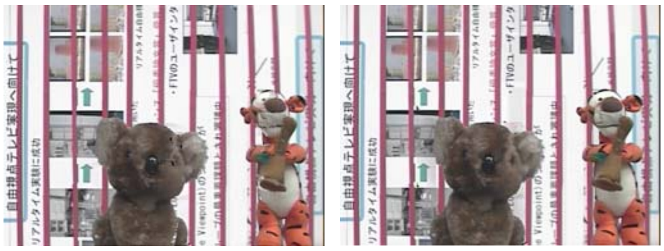


Fig.17: A Comparison of Generated Image

We can see the 258th and 322th picture in Fig 16.They are taken in the same scene. But now they have different contents. But it is a pity that what has been taken off can not be recovered completely.

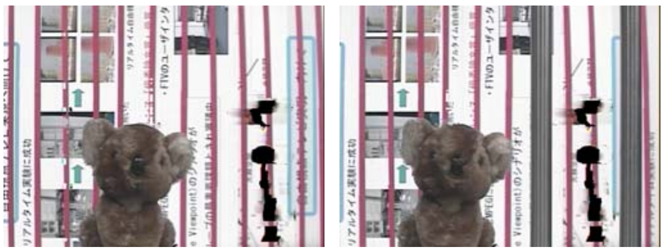


Fig.18:258th and 322th Picture Have Different Contents

6. CONCLUSION AND FUTURE WORK

In this paper we have proposed edit algorithm for 3D scene which is based on Ray-Space method, EPI, and 3D space continuity. Using epipolar geometry, we reduce the 3D problem of segmenting the EPI volume into a 2D curve evolution. The main contribution of this scheme is making us be able to edit EPI freely and generate 3D images not only with the viewpoint we want but also with the object in the place on right timing we want.

However, we believe the segmentation method has potential to be more accurate. In this experiment, we have just got 3 kinds of slope in the scene. In the future, we want find a way to edit EPI of a more complicated environment with many kinds of slopes. And this time we used 480 pictures in our processing. It costs a lot of time. So we also want to find a more accurate way by using fewer pictures.

7. REFERENCES

[1] E.H. Adelson and J.R. Bergen, "The plenoptic function and the elements of early vision," Computational Models of Visual Processing, M. Landy and J.A. Movshon, Eds. MIT Press, Cambridge, MA, 1991, pp. 3-20.

[2] R. Bolles, H.H. Baker, and D. Marimont, "Epipolar-plane image analysis: An approach to determining structure from motion," Int. Journal of Computer vision, vol. 1, pp. 7-55, 1987.

[3] J. Berent and P.L. Dragotti, "Segmentation of Epipolar Plane Image volumes with occlusion and dissocclusion competition, " IEEE Int. Workshop on Multimedia Signal Processing, pages 182-185, October 2006.

[4] J. Berent, P.L. Dragotti, "Plenoptic Manifolds: Exploiting Structure and Coherence in Multiview Images, " IEEE Signal Processing Magazine, vol. 24, no. 7, pp. 34-44, November 2007.

[5] T. Fujii, T. Kimoto, and M. Tanimoto, "Ray Space Coding for 3D visual communication, ", Proc. Picture Coding Symp.'96, vol. 2, pp. 447-451, 1996.

[6] R. Takano, T. Yendo, T. Fujii, and M. Tanimoto, "Scene Separation by Ray-Space Processing, ", The 2006 IEICE General Conference, pp. 67, 2006.

[7] A. Criminisi, S.B. Kang, R. Swaminathan, R. Szeliski, and P. Anandan, "Extracting Layers and Analyzing Their Specular Properties Using Epipolar-Plane-Image Analysis, ", Microsoft Research, Microsoft Corporation, Redmond, WA 98052, Tech. Rep. MSR-TR-2002-19, March 2002.

[8] J. Y. A. Wang and E. H. Adelson, "Representing Moving Images with Layers, ", IEEE Trans. On Image Processing Special Issue: Image Sequence Compression, vol.3, no. 5, pp. 625-638, September 1994.