Rendering with non-uniform approximate concentric mosaics

Jinxiang Chai[1], Sing Bing Kang[2], and Heung-Yeung Shum[3]

[1] The Robotics Institute, Carnegie Mellon University,
Pittsburgh, PA 15213, USA
[2] Vision Technology Group, Microsoft Research,
Redmond, WA 98052, USA
[3] Microsoft Research,
Beijing, China

Abstract. In this paper, we explore the more practical aspects of building and rendering concentric mosaics. First, we use images captured with only *approximately* circular camera trajectories. The image sequence capture can be achieved by holding a camcorder in position and rotating the body all around. In addition, we investigate the use of variable input sampling and fidelity of scene geometry based on the level of interest (and hence quality of view synthesized) on the objects in the scene. We achieve the tolerance for minor perturbations about the exact circular camera path and variable input sampling by using and analyzing a variant of the Hough space of all captured rays. Examples using real scenes are shown to validate our approach.

1 Introduction

Image-based rendering (IBR) has become a popular approach for modeling and rendering a virtual environment. While the conventional means of rendering uses a 3D model (with possibly a complicated photometric model), image-based rendering directly interpolates novel views from captured images. If the input images are captured sparsely in the space, establishing correspondences may still be necessary. However, if the input images are densely captured, direct view interpolation will suffice.

In theory, one needs only to capture a complete plenoptic function [1, 7] in order to synthesize a novel image from any viewpoint and at any viewing direction. However, a complete plenoptic function is at least 5D, which includes 3D spatial location and 2D ray directions at any point. If free space is assumed, the plenoptic function can be reduced to 4D, as shown in the lumigraph [2] and light field rendering [6]. However, for modeling a virtual environment, the size of the database for the light field is usually massive because it has to sample four dimensions.

Recently, concentric mosaics [11] has been proposed to sample a virtual environment where the viewpoints are constrained on a planar surface. It has been shown in [11] that a novel view can be generated from a sequence of images captured from a camera rotated off-center along a circular path. A *linear pushbroom camera model* is assumed [3] (as is with our work). In other words, the camera model used comprises a stack of parallel perspective views perpendicular to the y-axis, with each perspective view representing a horizontal scanline. While vertical distortion exists as a result of using this camera model, the synthesized images show good rendering quality with the help of constant depth correction and bilinear interpolation.

However, there are at least two disadvantages associated with the current concentric mosaics work. First, it requires a capturing rig that is bulky. It is much more practical if a user can hold a camcorder in a position and rotate his body around to capture the necessary images. Second, it is desirable to capture the environment with variable sampling rates and fidelities. For example, it is intuitive that more samples should be taken at regions that are deemed more interesting. It also makes more sense to make more samples at areas that is highly textured and where depth variation is significant.

This paper addresses the above two practical issues in concentric mosaic building and rendering, namely using hand-held camera to acquire images and variable input sampling. The input sequences of images are captured using a hand-held camera, and recovery of the camera pose is accomplished using a structure from motion algorithm. However, we do not explicitly build a 3D model from the input images (e.g., generate 3D panoramic models from stereo [4]). To handle the variable sampling resolution, we propose a new representation we call called *signed Hough space* that enables uniform sampling and efficient computation in the ray space.

1.1 Previous work

There has been significant work done on image-based rendering using large quantities of input images. The pioneering work on the lumigraph [2] and light-field rendering work [6] have spawned a number of related work. Two of the more notable ones are the concentric mosaic [11] and the stereo panorama [8]. There are also others who use the approach of generating 3D panoramic models [4], or computing panoramic depth as a means for rendering [7, 12].

1.2 Outline of paper

The remainder of this paper is organized as follows. We describe our new representation called *signed Hough space* in Section 2. In Section 3, we give a summary of the least-squares method to extract camera pose from a sequence of tracked images. Once camera poses are known, the input data is mapped to the new representation space. Issues with rendering with approximate concentric mosaics using the new representation is discussed in Section 4. Experimental results using synthetic and real images are shown in Section 5. We conclude this paper in Section 6.

2 Signed Hough space

Our image-based approach is based on reusing captured rays from input images to reconstruct an image at a novel viewpoint. An important problem in image-based rendering is the representation, namely, how to represent the rays that are captured. For example, the lumigraph is a particular way of sampling the ray space using a 4D two-plane parameterization. Concentric mosaics sample the space using three parameters, i.e., the rotation angle, radius and vertical field of view.

In this section, we present a new approach to represent non-uniform concentric mosaics from a large collection of images taken along an approximate circle. The major issue in choosing a representation for non-uniform plenoptic sampling is how to parameterize the space of oriented lines. We consider a good choice of parameterization of oriented rays to have the following characteristics:

– *Efficient calculation*. The computation of the position of oriented ray from its parameter space, and vice versa, should be fast.
– *Uniform sampling*. The sampling within the spatial and directional spaces should be uniform. This is to avoid potential problems in rendering.
– *All inclusive*. All possible oriented rays in the space should be represented, with no exceptions.

Note 1.
Duality. Reciprocal behavior should exist between the destination (within a panorama in view space), and source (a geometric point with its radiance in Cartesian space). In other words, analysis would proceed exactly the same if the destination and source are switched.

It is obvious that light field representation using the two-plane parameterization cannot satisfy the third item. Rays that are parallel or do not intersect the slabs are not represented. In our case, rays at all orientations and positions can be included in our representation.

Note 2. For simplicity, we first describe the representation of oriented rays in 2D Cartesian space, and then we will extend it to 3D space for the representation of approximate concentric mosaics.

One of the ways that we can visualize the population of rays available is to construct the usual Hough space which uses the normal (r, θ) parameterization. However, rays are directional, and the conventional Hough space is unable to distinguish rays that have the same equation by are of opposite directions. We solve this by using the right-hand rule: A ray that is directed in an anti-clockwise fashion about the coordinate center is labeled positive, otherwise it is labeled negative. “Positive” rays have positive r values, i.e., (r, θ), while “negative” rays have negative r values, i.e., $(-r, \pi + \theta)$. Figure 2.1 shows four different rays in 2D space and their corresponding points in the signed Hough space.

An attractive feature of this representation is the duality between points and sinusoids in both Cartesian and signed Hough space. Figure 2 shows examples of common projections are represented in signed Hough space. For example, panoramic visibility at a point in Cartesian space (Figure 2(a)) is represented as a sampled sinusoidal curve in the parameter space. A concentric mosaic (Figure 2(b)) is mapped to a horizontal line in the signed Hough space, while parallel projections (Figure 2(c)) are mapped to a vertical line in the signed Hough space.

Note 3. Specifically, the bundle of all rays emitted by a 3D geometric point in Cartesian space also takes the shape of a sampled sinusoidal curve featured by its space location (r_0, θ_0). Thus, the captured perspective scene can be easily transformed into the parameter space. Rendering a novel view in the scene is equivalent to extracting a partial sinusoidal curve from the signed Hough space. Interestingly, computing the depth of

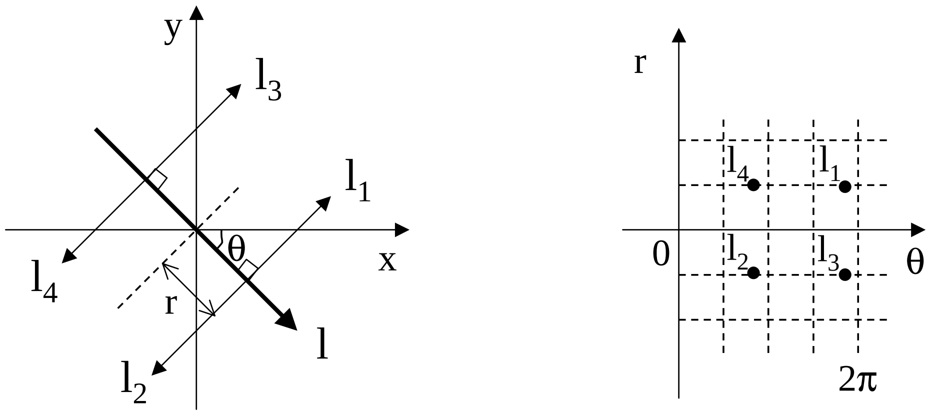
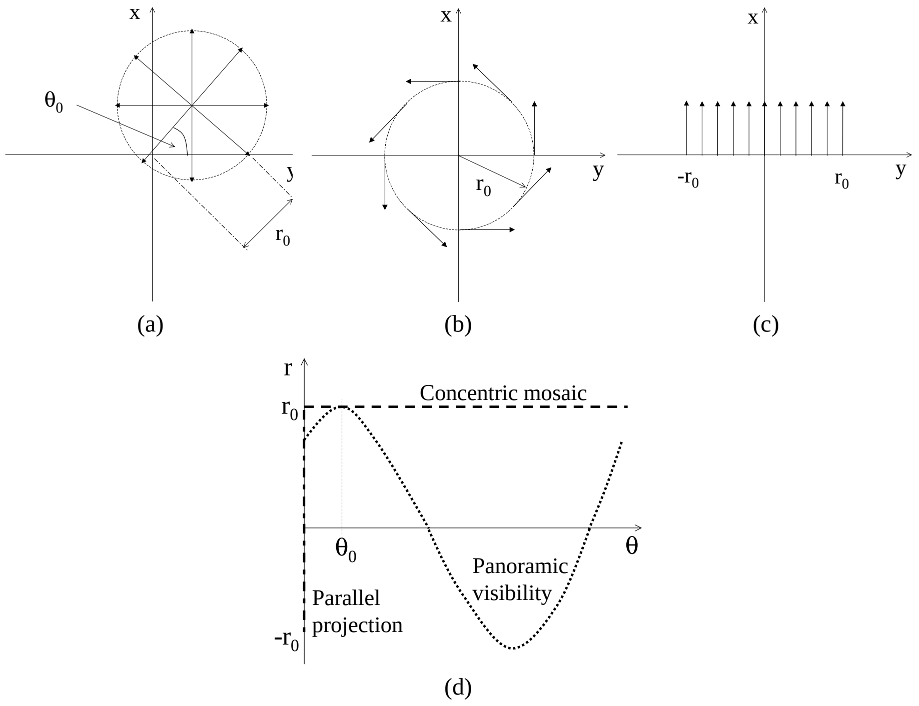


Fig. 1. Definition of the ray space we captured to reconstruct the 3D geometry. Each oriented ray in Cartesian space (at left) is represented by a sampled point in the signed Hough space.



(a)

(b)

(c)

(d)

Fig. 2. Three typical viewing setups and their associate sampled curve in signed Hough space. (a) Panoramic visibility at a point in 2D Cartesian space, (b) A concentric mosaic, (c) Parallel projection, and (d) Their respective sampled curves in the signed Hough space.

scene can also be defined as a curve fitting problem that is constrained by a specific BRDF model.

3 Rendering using handheld sequential images as input

The previous work on concentric mosaic [11] uses images from a camera with a perfectly circular trajectory using a motorized setup. We extend this work to a more practical level by allowing visualization from *approximate* concentric mosaics. The input images can be captured from a hand-held camera that is moved through an approxiately circular trajectory.

3.1 Computing structure from motion

Building the approximate concentric mosaic requires accurate camera poses associated with the input images. To do this, we first calibrate the camera to extract intrinsic parameters using the method described in [15]. Subsequently, we automatically track point features in the image sequence using Shi and Tomasi's tracker [10]. Their tracker uses an affine model and a Hessian-based measure of the local texturedness to determine removal and addition of point features at each frame.

Once the point tracks are available, we apply the iterative least-squares minimization technique based on Levenberg-Marquardt on these point tracks [14] to recover camera motion. For completeness, we provide a brief description of this algorithm.

Structure and motion are solved simultaneously to minimize the difference between the 2-D track points and the 3-D object points projected into 2-D. The Levenberg-Marquardt algorithm [9], a standard iterative least-squares solver, is used to minimize the objective function

$$\mathcal{C}(\mathbf{a}) = \sum_i \sum_j c_{ij} |\mathbf{u}_{ij} - \mathbf{f}(\mathbf{a}_{ij})|^2, \tag{1}$$

where $\mathbf{u}_{ij}$ is the measured point feature location, $\mathbf{f}(\mathbf{a}_{ij})$ is the predicted projected point,

$$\mathbf{a}_{ij} = (\mathbf{p}_i^{\mathbf{T}}, \mathbf{m}_j^{\mathbf{T}}, \mathbf{m}_g^{\mathbf{T}}) \tag{2}$$

and $\mathbf{c}_{ij}$ is a measure of confidence of the position, based on the amount of local texture at the point.

The vector **a** contains the 3-D points $\mathbf{p}_i$ for each point i, the local motion parameters $\mathbf{m}_j$ for each frame j, and the global motion and camera intrinsic parameters $\mathbf{m}_g$. The function $\mathbf{f}(\mathbf{a}_{ij})$ is the projective function that maps the point $\mathbf{p}_i$ to the image j, using the camera position and the camera intrinsic parameters.

For each iteration, the Levenberg-Marquardt algorithm finds an approximate Hessian matrix **A** and gradient vector **b**, which is used to solve for an increment $\delta\mathbf{a}$ towards the minimum. The equation solved is

$$(\mathbf{A} + \lambda \mathbf{I})\delta\mathbf{a} = -\mathbf{b}, \tag{3}$$

where λ is a time-varying stabilization factor and **I** is the identity matrix.

The elements of the Hessian **A** are approximated as the product of partial derivatives with respect to **a**:

$$\mathbf{A} = \sum_i \sum_j 2c_{ij} \frac{\partial \mathbf{f}^{\mathbf{T}}(\mathbf{a}_{ij})}{\partial \mathbf{a}_{ij}} \frac{\partial \mathbf{f}(\mathbf{a}_{ij})}{\partial \mathbf{a}_{ij}^{\mathbf{T}}}, \tag{4}$$

and the gradient vector **b** is

$$\mathbf{b} = \sum_i \sum_j 2c_{ij} \frac{\partial \mathbf{f}^{\mathbf{T}}(\mathbf{a}_{ij})}{\partial \mathbf{a}_{ij}} \mathbf{e}_{ij}, \tag{5}$$

where $\mathbf{e}_{ij} = \mathbf{u}_{ij} - \mathbf{f}(\mathbf{a}_{ij})$ is the position error.

Note 4. For our application of rendering with approximate concentric mosaics, we would also like to constrain the camera motion to a simple planar motion from general rigid motion. The structure from motion algorithms would be more robust with the reduction in the number of parameters.

Once we have obtained the camera poses using the tracker and subsequent structure from motion algorithm, we can then map all the input rays associated with the cameras to the signed Hough space for subsequent rendering.

4 Rendering from the signed Hough space

By resampling the input rays into the signed Hough space, we can achieve the tolerance for minor perturbations about the exact camera poses. These camera parameters may not be perfectly recovered from the above structure from motion algorithms. In the new space, we improve rendering quality by designing optimal interpolation filters. We analyze various interpolation filters, including parallel interpolation and constant depth interpolation along r and θ directions. Furthermore, multi-resolution rendering (i.e., zoom in and out of objects/regions of interest) can also be easily implemented in the new representation space.

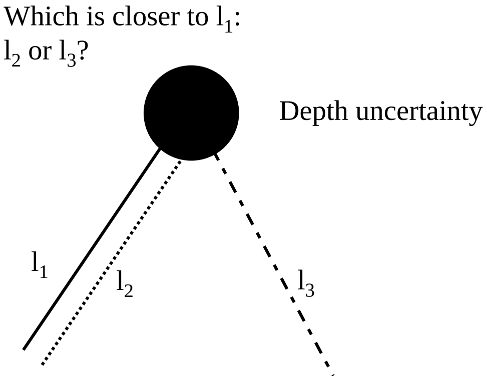


Fig. 3. Ambiguous definition of closest ray.

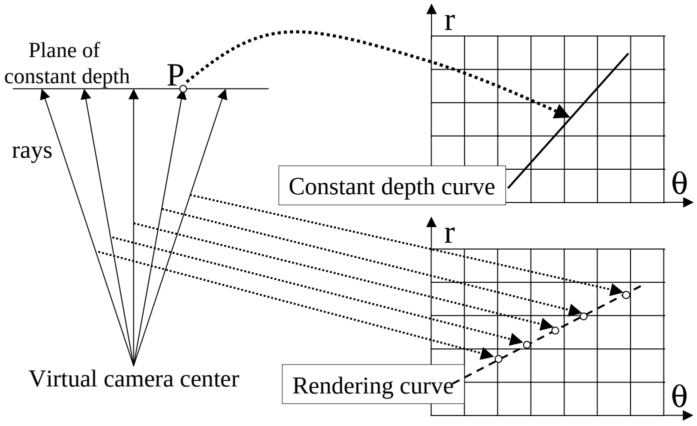
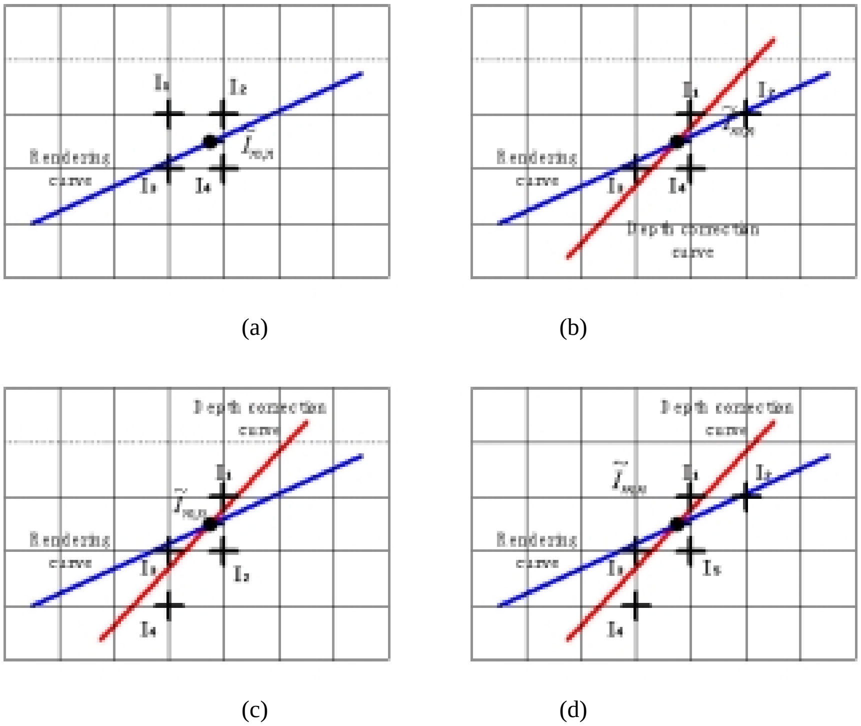


Fig. 4. Rendering and depth correction curves.



(a)

(b)

(c)

(d)

Fig. 5. Different bilinear interpolation filters. (a) Parallel bilinear interpolation, (b) Bilinear interpolation with constant depth correction along angular direction, (c) Bilinear interpolation with constant depth correction along radius direction, and (d) Bilinear interpolation with constant depth correction along both directions. Note that the horizontal axis is that of θ while the vertical axis is that of r.

Given a set of non-uniform concentric mosaics collected from a camera moving non-uniformly along an approximately circular path, we can render any novel view. The rendered views are constrained by the camera trajectory, similar to concentric mosaics where viewpoints of the rendering camera are constrained by the capturing circle.

Rendering a new image at any viewpoint becomes the problem of extracting a sinusoidal curve in the signed Hough volume. However, due to the discretization of the signed Hough volume, interpolation techniques have to be carefully chosen in order to obtain high quality rendering results.

Before we describe the interpolation techniques, let us make a couple of definitions, with the help of Figure 4. All the rays for a given virtual camera map into what we call a *rendering curve*. If the depth correction is specified, any given ray will intersect at a known point, say P. P then maps onto the *depth correction curve* in ray space.

To continue, a good interpolation filter should make use of depth information. However, when no information about the scene geometry is available, the parallel bilinear filter (e.g., [11]) is commonly used to interpolate the rendering rays. It works by assuming all of the scene points are located in infinity, as shown in Figure 5(a). In this particular case, the four closest ray bins I_1, I_2, I_3, and I_4 are used to compute the color of the virtual ray indicated by $\tilde{I}_{m,n}$.

Bilinear interpolation and constant depth assumption can be used to improve the quality of rendered images. With the constant depth assumption, all of the objects seen by the camera are deemed to be located along a simple surface such as a cylinder. As with any assumption on scene depth, the issue is how to choose the closest points to reconstruct the rendered point.

The definition of “closest” points is ambiguous if no accurate depth information is known. Consider, for example, the question as to which of the rays, l_2 or l_3, is "closer" to ray l_1? The notion of closeness makes sense only if the object distance is known, even approximately. The interpolation techniques shown in Figure 5(b)-(d) uses specified depth corrections to decide which ray bins to use. As an example as to how the ray bins are chosen for interpolation, consider the case of constant depth correction along the angular direction, as shown in Figure 5(b). First, the intersections between the depth correction curve and horizontal rows closest to the virtual ray $\tilde{I}_{m,n}$ are computed. The sampling ray bins are those just on each horizontal side of these intersections. Similar reasoning can be applied to Figure 5(c) and (d).

5 Experiments

Unlike most capture setups for image-based rendering, the image capture process here is very simple. Specifically, a single camera is moved by hand to rotate along an approximate circular path. In our experiments, a total number of 1864 images of a real scene is captured. The image size is 360×288. Only 530 frames are used to recover camera poses using our SFM algorithm. Two input images are shown in Figure 7(a)(b) where a number of feature points are tracked for the SFM algorithm. As shown in Figure 6, the rotation and translation parameters are recovered fairly well.

Using the estimated camera motion, we transform the input images into our signed Hough space. The binning process is based on nearest neighborhood. The new parameter space has the resolution of 230×310 in radial and angular dimensions. The signed Hough space can also be examined to see if it can be represented with coarser discretization by checking the density of ray occupancy. Downsampling has the benefit of compactness. In addition, we have applied vector quantization compression to our database to further reduce its size; in our example, the reduced size is about 4MB.

Figure 7(c,d) show two rendered images. Note the significant parallax changes around the monitor in the middle and through the window on the right. Four different interpolation techniques have been applied to render the new images, as shown in Figure 8. These techniques are parallel interpolation, depth correction around radial direction, depth correction around angular direction, and depth correction with both radial and angular directions, respectively. Among these techniques, depth correction along radial direction produces the best rendering result, whereas depth correction along angular direction is the worst. Because angular sampling is much denser than radial sampling in the original images, interpolation along radial direction is effective. In fact, the angular direction is over-sampled. Depth correction along both directions produces comparable rendering result as with depth correction along radial direction only. Parallel interpolation has better rendering result than depth correction along angular direction because parallel interpolation is in fact along the radial direction, albeit at the infinite radius.

With the new parameter space, we can also render images in different resolutions. Figure 9 shows the results of zooming in and zooming out. Notice the appropriate changes in apparent size of the bunny. In general, there are two approaches to obtain the zoom-in effect. First, we can sample the areas of interest more densely than others. But multi-resolution representations should be applied for efficiently storing the data. Second, depth information can be used to improve the resolution. Higher resolution of output images can be achieved with more accurate depth information. The depth information can be obtained by either vision reconstruction techniques or human interaction. For example, Figure 9(b) is obtained with a different depth specified by the user than the depth used in Figure 9(a).

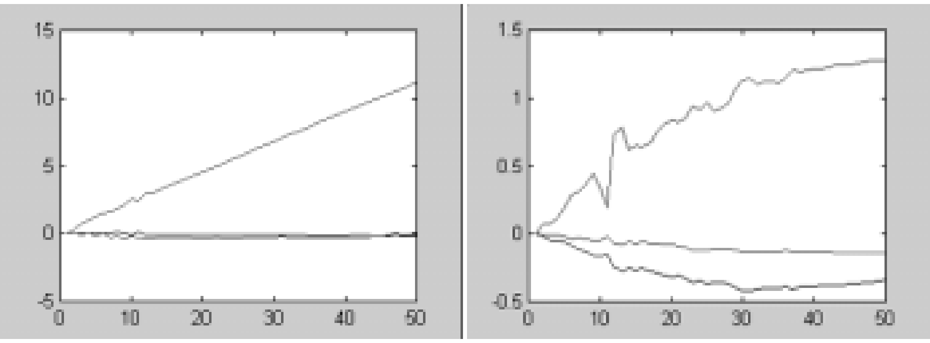


Fig. 6. Camera poses estimated using structure from motion algorithms. Left: Graph depicting the variation in rotation (in degrees) about the y, x, and z axes (curves from top to bottom). Right: Graph depicting the variation in translation along the x, y, and z axes (curves from top to bottom).

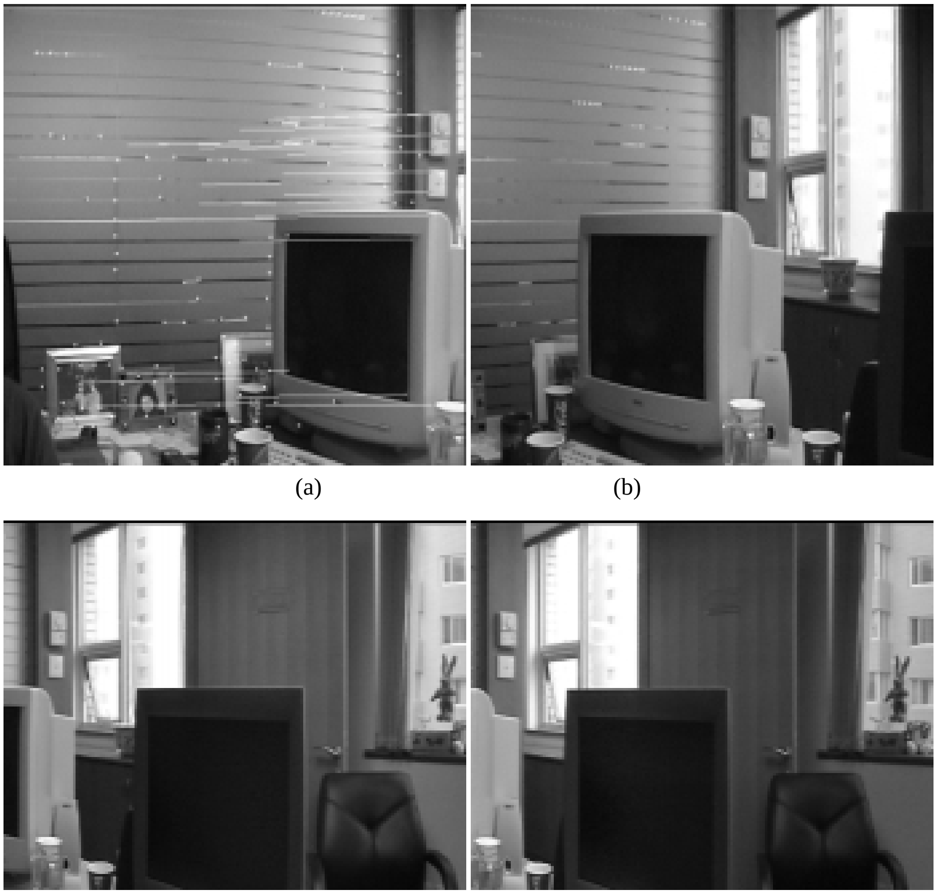


Fig. 7. Rendering with non-uniform concentric mosaics. (a,b) Two frames in the input image sequence, and (c,d) Two rendered images with significant parallax change.

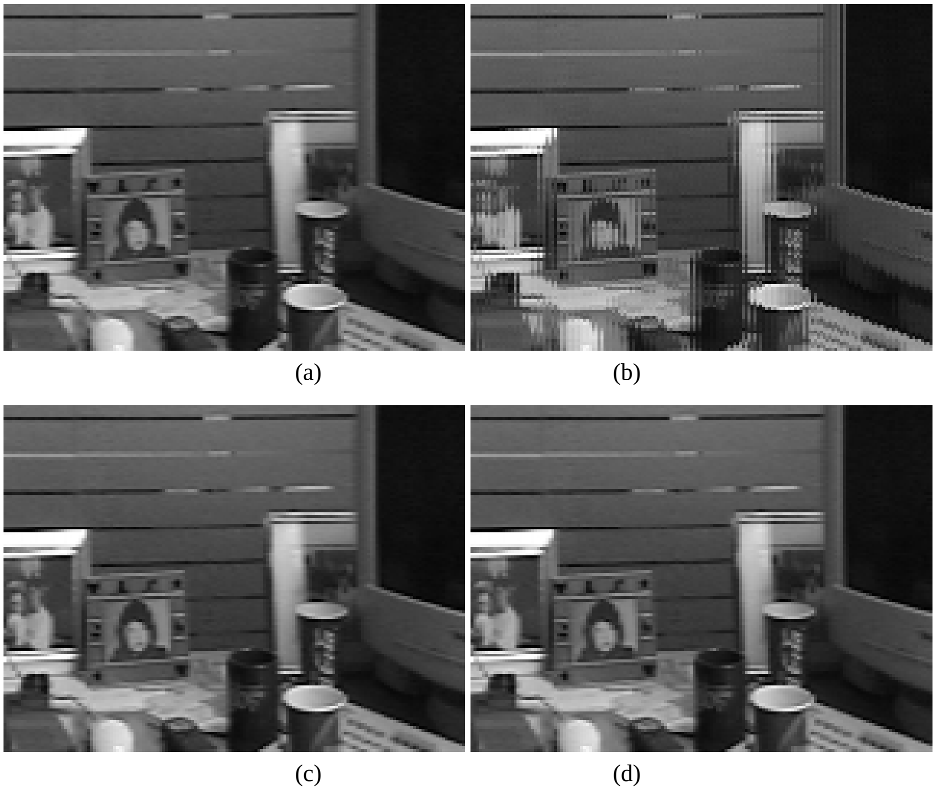


Fig. 8. Results of using different bilinear interpolation filters. (a) Parallel bilinear interpolation, (b) Bilinear interpolation with constant depth correction along angular direction, (c) Bilinear interpolation with constant depth correction along radius direction, and (d) Bilinear interpolation with constant depth correction along both directions.

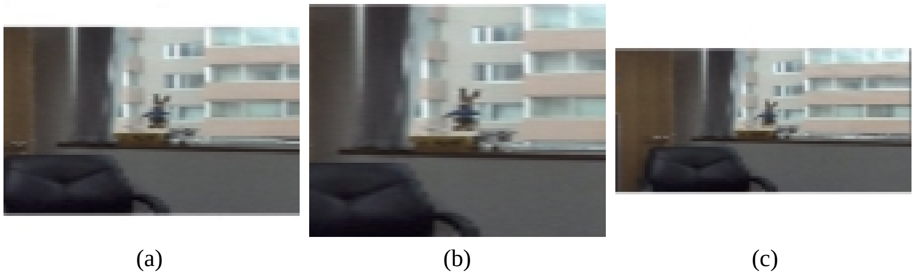


Fig. 9. Results of zooming in and out. (a) No zoom, (b) Zooming in with a factor of 0.75, and (c) Zooming out with a factor of 1.25. Note the size of change of the bunny.

6 Discussion

Database acquisition for light-field-based IBR is usually a very laborious process and often require specialized (and thus expensive) equipment. Until drastic simplications are made to the acquisition process, IBR will remain beyond the reach of ordinary consumers. With our technique, however, such specialized equipment is not necessary. We have shown that we can provide high-quality visualization from a database created from images taken using just a hand-held camera that is manually moved along an approximately circular path.

We have also used the notion of variable sampling in our work. In areas where objects are less interesting to us, we can afford sparser input sampling and without (or with less accurate) depth information. This may not be very evident in our results, because our overall sampling is actually rather dense, even in the least densely sampled areas.

While the camera motion parameters are required to build the database for the concentric mosaic, absolute accuracy of these parameters are, in practice, not necessary. This is evidenced by our results. There are enhancements to our current SFM algorithm that we can make. Our SFM algorithm is currently too general. If we know that the motion is planar (or assumed planar), we can impose additional constraints in our algorithm, so that fewer parameters need to be computed. (In the handheld camera case, this may or may not be applicable.) Parameter recovery will be faster as well, especially when we are dealing with a large number of images and tracks.

7 Conclusions and future work

In this paper, we have proposed a practical method for capturing and rendering approximate and non-uniform concentric mosaics. The method does not require a specialized rig for image capture; manually moving a hand-held camera along an approximately circular path is sufficient. In addition, we introduced the *signed Hough space* to represent the captured rays. The extension to the conventional Hough space is necessary in order to encode rays with direction. For full 3D space of rays (i.e., using a normal perspective camera model instead of a pushbroom camera model), we can use an alternative representation based on *oriented projective geometry* [13]. This representation has been used to recover shape from silhouettes [5].

Judicious use of variable input sampling can be effective in making more optimal use of the available limited manual and rendering resources. This basically trades off fidelity of output with the level of interest. We intend to investigate this aspect more thoroughly.

Finally, we have describe different interpolation regimes and show the results of applying them. The bilinear interpolation with depth correction seems to work the best.

References

1. E. H. Adelson and J. R. Bergen. *Computation Models of Visual Processing*, chapter The plenoptic function and the elements of early vision. MIT Press, Cambridge, MA, 1991.

2. S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen. The lumigraph. *Computer Graphics (SIGGRAPH'96)*, pages 43–54, August 1996.
3. R. Gupta and I. H. Richard. Linear pushbroom cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(9):963–975, September 1997.
4. S. B. Kang and R. Szeliski. 3-d scene data recovery using omnidirectional multibaseline stereo. *International Journal of Computer Vision*, 25(2):167–183, November 1997.
5. K. N. Kutulakos. Shape from the light field boundary. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 53–59, Puerto Rico, June 1997.
6. M. Levoy and P. Hanrahan. Light field rendering. *Computer Graphics (SIGGRAPH'96)*, pages 31–42, August 1996.
7. L. McMillan and G. Bishop. Plenoptic modeling: An image-based rendering system. *Computer Graphics (SIGGRAPH'95)*, pages 39–46, August 1995.
8. S. Peleg and B. Ben-Ezra. Stereo panorama with a single camera. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 395–401, Fort Collins, CO, June 1999.
9. W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, Cambridge, England, second edition, 1992.
10. J. Shi and C. Tomasi. Good features to track. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 593–600, Seattle, WA, June 1994.
11. H.-Y. Shum and L.-W. He. Rendering with concentric mosaics. *Computer Graphics (SIGGRAPH'99)*, pages 299–306, August 1999.
12. H.-Y. Shum and R. Szeliski. Stereo reconstruction from multiperspective panoramas. In *International Conference on Computer Vision*, pages 14–21, Kerkyra, Greece, September 1999.
13. J. Stolfi. Oriented projective geometry. In *Annual Symposium on Computational Geometry*, pages 76–85, Waterloo, Canada, June 1987.
14. R. Szeliski and S. B. Kang. Recovering 3D shape and motion from image streams using nonlinear least squares. *Journal of Visual Communication and Image Representation*, 5(1):10–28, March 1994.
15. Z. Zhang. Flexible camera calibration by viewing a plane from unknown orientations. In *International Conference on Computer Vision*, pages 666–673, Kerkyra, Greece, September 1999.