Layered Depth Panoramas

Ke Colin Zheng[1], Sing Bing Kang[2]
Michael F. Cohen[2], Richard Szeliski[2]
[1]University of Washington, Seattle, WA
[2]Microsoft Research, Redmond, WA

Abstract

Representations for interactive photorealistic visualization of scenes range from compact 2D panoramas to data-intensive 4D light fields. In this paper, we propose a technique for creating a layered representation from a sparse set of images taken with a hand-held camera. This representation, which we call a layered depth panorama (LDP), allows the user to experience 3D by off-axis panning. It combines the compelling experience of panoramas with limited 3D navigation. Our choice of representation is motivated by ease of capture and compactness. We formulate the problem of constructing the LDP as the recovery of color and geometry in a multi-perspective cylindrical disparity space. We leverage a graph cut approach to sequentially determine the disparity and color of each layer using multi-view stereo. Geometry visible through the cracks at depth discontinuities in a frontmost layer is determined and assigned to layers behind the frontmost layer. All layers are then used to render novel panoramic views with parallax. We demonstrate our approach on a variety of complex outdoor and indoor scenes.

1. Introduction

A single photograph of a scene is just a static snapshot with limited field of view captured from a single viewpoint. Many techniques have been proposed to extend the ways in which a scene can be visualized by taking multiple photographs. These range from creating 2D panoramas from a few photographs (to extend the field of view) to creating 4D lightfields from a large number of images (to provide extensive freedom to explore a scene, with expensive capture and storage requirements).

In this paper, we present a system that asks little more of the user than capturing a simple panorama from a sparse set of images with a hand-held camera. We provide a result that is only fractionally larger than a simple 2D panorama, yet affords the ability to view the result with both the wide field-of-view of panoramas and enough parallax between objects at different depths to create a more visceral sense of immersion in the scene.

The capture process is much like that for a traditional panorama in which a sparse set of images is taken about a single center of projection to avoid parallax. However, we instead require the user to merely hold the camera at arm's length to capture the parallax induced when moving the camera along an arc. We automatically recover a layered representation [2] in which multiple depths may exist for a single line of sight. Such a representation was called a layered depth image (LDI) [23]. Because our representation is a layered analogue of the panorama, we refer to it as layered depth panorama (LDP). The LDP removes the fundamental limitations of 2D mosaics by supporting viewpoint translation with reasonable extra cost in memory and computation. When viewed from any single point-of-view, the LDP appears like a normal 2D panorama; when the viewpoint moves off its center, the LDP exhibits motion parallax, thus providing a more immersive 3D experience.

We next review previous work in image-based scene modeling and stereo matching. In Section 3, we describe how to compute the LDP for our novel representation. We present some experiments on a few real world examples in Section 4. We conclude the paper with a discussion of our results and a list of topics for future research.

2. Previous work

Our system builds on several algorithms previously developed for image-based rendering and stereo reconstruction. In this section, we review relevant work in these areas.

2.1. Image-based modeling and rendering

The techniques described below are specialized instances of *image-based rendering* [17, 13, 8, 6], where the goal is to create novel views from a collection of input images.

2D Panoramas. 2D panoramas are constructed by stitching together a collection of images taken from the same center of projection [31, 29]. They support viewing the scene

from this point in any desired direction. Panoramas can be captured with or without a tripod, and can be automatically stitched [5]. However, they do not support view translation; this deprives users of motion parallax, which is an important cue in 3D scene perception.

Concentric mosaics. If we constrain the camera motion to planar concentric circles, we obtain a 3D plenoptic function called *concentric mosaics* [24]. Such mosaics can be formed by compositing slit images taken at different locations along each circle [18]. Like 2D panoramas, concentric mosaics do not require recovering geometric and photometric scene models. Moreover, they provide a much richer viewing experience by allowing users to move freely in a circular region and to observe significant parallax and lighting changes. However, there is a problem associated with not using appropriate geometry: the vertical scaling in the reconstructed views can appear incorrect. Also, concentric mosaics are much more data intensive than 2D panoramas, and require special hardware such as a motorized tripod with an extended arm.

Layered Depth Images. Layered depth images (LDIs) are images that have potentially more than a single depth/color pair at each pixel [23]. These allow the scene to be rendered from multiple adjacent points of view without the introduction of either *cracks* (holes in the reconstruction) or spurious surfaces (that look like rubber sheets). When the pixels are organized into a small number of *layers*, the resulting representation can be efficiently rendered on a GPU [33].

Image-based editing. Image-based editing [25] bypasses the difficult modeling process by manually specifying geometry. 3D models are built by segmenting images into sprites that are mapped to separate planes. Image-based editing techniques take advantage of human knowledge of the scene, which allows them to maximize the 3D effect while minimizing the amount of depth data. However, manual geometry specification is slow and tedious, requiring more efficient user interfaces. In an earlier work [2], a semi-automated stereo matching approach was proposed to create such *layered scene descriptions*. Efficient image-based rendering techniques for such representations have also been developed [23, 33].

Dense depth map. Laser rangefinding technologies acquire dense, accurate depth maps that can be converted into high-quality models. Bahmutov *et al.* [1] model a real world scene using depth enhanced panoramas. Such panoramas with per-pixel depth are acquired using they call a *model camera*, which is a structured-light acquisition device. These methods produce good geometry but suffer from long acquisition times and high equipment cost. Passive stereo-based reconstruction techniques (described below) can capture the scene more quickly, since only a few images are required, but usually do not produce as high-quality a result.

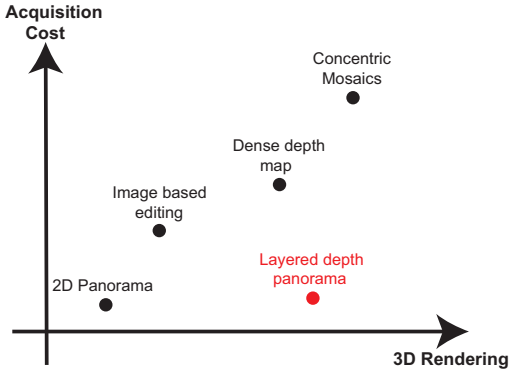

Figure 1. Cost/quality tradeoff for various modeling techniques.

Figure 1 shows the relative tradeoff between acquisition cost and rendering quality for various image-based modeling and rendering techniques. Our goal is to develop a solution that has almost as low acquisition cost as 2D panoramas, yet produces similar 3D effects as concentric mosaics. We leverage state-of-the-art stereo-based scene reconstruction techniques to achieve this goal.

2.2. Stereo-based scene reconstruction methods

The problem of reconstructing a scene from multiple cameras has received a lot of attention in the last few years [19, 20].

In voxel-based approaches, the scene is represented as a set of 3D voxels, and the task is to compute the visibility as well as the color of each voxels. One major limitation of voxel coloring [22] is that "hard" decisions concerning voxel occupancy are made as the volume is traversed. Because the data is ambiguous, such decisions can easily be wrong, and there is no easy way to correct them.

Szeliski and Golland [30] applied an iterative framework to solve not only for depth, but for color and transparency as well. They iteratively aggregate visibility evidence to refine the reconstruction. Since this problem is grossly underconstrained and difficult to optimize, their results were not that encouraging. Baker *et al.* [2] proposed an alternative approach, where only a small number of layers is hypothesized and recovered (see also [3, 15]).

Kolmogorov and Zabih [12] took an approach that yielded excellent results for stereo [19], namely energy minimization via graph cuts, and generalized it to solve the scene reconstruction problem. They treat the input images symmetrically, handle visibility properly, and impose spatial smoothness while preserving discontinuities. Their work is one example of approaches that recover *multiple depth maps* simultaneously in the context of a global optimization framework [28, 11, 33, 7].

Stereo matching has also been applied to panoramic images, both in the case of panoramas taken from widely sep-

arated viewpoints [10], and for panoramas taken as concentric mosaics [14]. In both of these cases, a single dense depth map with a large field of view is directly computed from the multi-perspective panoramas. In our work, we create *layered depth panoramas*, which can represent multiple depths at each pixel in a panoramic coordinate system. Furthermore, we optimize the estimated colors using results from recent image stitching algorithms.

3. Approach

In this section, we describe the construction of the layered depth panorama (LDP) from multiple images, and how novel views are rendered.

3.1. The Layered Depth Panorama

A layered depth image (LDI) differs from a normal image in that each pixel stores one or more pixels along the line of sight represented by the pixel. The front element in the layered depth pixel samples the first surface seen along that line of sight; the next element samples the next surface seen along that line of sight, etc. Our LDP uses the same underlying concept adapted to a cylindrical parameterization to accommodate larger fields of view.

A hand-held camera held at arm's length captures views to simulate an off-center camera rotation, from which we construct the LDP. As in an LDI, for the LDP we also select a 2D array of rays to form a layered representation. We would like these rays to be close to the captured rays, and we also wish to cover a wide field of view in order to maximize the range of viewpoints the user can virtually explore.

The original images are gathered roughly along a circular arc in space with a radius roughly equal to an arm's length. We first determine the arc that most closely follows the camera path. This establishes a cylindrical coordinate system (see Figure 2). The 2D array of rays is formed by the set of rays that pass through the arc and simultaneously lie in planes that pass through the center of the arc. In other words, these are the rays that would form the center vertical scanlines of cameras with optical centers lying on the arc and facing outward. This is a cylindrical version of pushbroom panoramas [21].

The 2D array of rays are parameterized by (θ, v) (see Figure 2). θ is the angle (or position) along the arc through which all rays pass. v is the vertical position where the ray pierces a cylinder a *unit* distance from the arc (i.e., a cylinder with radius one greater than the radius of the arc). Discretizing θ and v defines the resolution of the LDP. Each discrete position, (θ, v), defines a 3D ray, also referred to as a *pixel* in each layer of the LDP.

Our goal is to recover the (possibly multiple) depths d associated with each pixel.

Depth, d, is discretized at increments proportional to $1/d$, to ensure that *disparity* increments are similar. Each discrete position (θ, v, d) represents a *voxel* in the cylindrical volume.

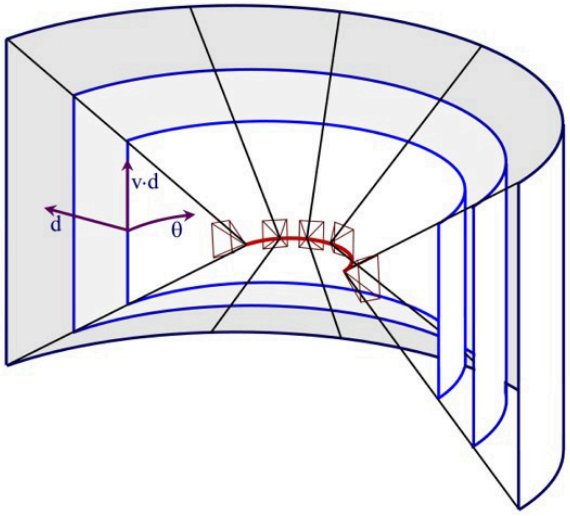

Figure 2. The original cameras lie along an arc shown in red. This arc defines the spine of the rays in the LDP and defines a 3D volume of concentric cylinders.

The LDP consists of a set of layers L_i, where i is the layer index. For each ray, (θ, v), we determine the depth d of objects intersected by the ray. We also determine the color, c, for the intersection points. Thus, $L_i(\theta, v) = (d, c)$ indicates the pixel with coordinate (θ, v) on the i-th layer has color c, and is at depth d. In other words, the voxel at (θ, v, d) is on a surface with color c; and it is the i-th colored voxel along the ray from the camera arc to (θ, v, d). The first layer is dense (i.e., there is a well defined d value for every (θ, v) pixel). Layers behind the first layer are kept as small as possible, just enough to fill holes seen through depth discontinuities when viewed from along the camera arc.

Our goal is to represent the scene with an LDP such that all the input images can be explained. We achieve this by sequentially solving each layer in the LDP from front to back with multi-view stereo. We begin with a cylindrical plane sweep algorithm to generate an initial disparity space image [9] (DSI). Later, the DSI is further refined based on visibility inferred from the LDP. We leverage state-of-the-art optimization techniques with improved matching costs and smoothness constraints to reconstruct each layer. The algorithm works in a iterative fashion, where we modulate the DSI based on reconstructed geometry from LDP and we update the LDP based on the improved DSI.

3.2. Cylindrical Plane Sweep Algorithm

Plane-sweep and space coloring/carving stereo algorithms support multi-image matching, enable reasoning about occlusion relationships, and are more efficient than traditional correlation-based formulations. Rather than searching for corresponding windows across images as in traditional stereo matching algorithms, plane sweep algorithms consider each candidate disparity as defining a plane in space and project all images to be matched onto that plane, using a planar perspective transform (homography).

We have generalized plane sweep stereo to perform a multi-image cylinder sweep stereo reconstruction. All images are projected onto cylinders at various depths d. A per-pixel robust variance of the collection of input pixels that map to an output pixel is first computed. These are then aggregated spatially using an efficient convolution algorithm (a moving average 5×5 box filter). Finally, we use aggregated shiftable windows, essentially seeking the lowest variance within ± 1 pixel and select the lowest value. This last step improves the performance of matching near depth discontinuities [11].

Thus, for every location (θ, v, d) in the cylindrical volume, we have $\mu(\theta, v, d)$ and $\phi(\theta, v, d)$, where μ is the median color and ϕ is the robust variance. This forms our raw disparity space image (DSI), the initial matching cost. Later, we will describe how we iteratively update the DSI based on visibility information.

3.3. Optimization

For each layer L_i, we solve for its color and its depth separately.

3.3.1 Depth for the First Layer

Our first goal is to assign to each pixel $p = (\theta, v)$ on the first layer, L_1 a label corresponding to the depth (or disparity) of the first intersection along the ray (θ, v). Later, we will use almost the same formulation for the back layers. We formulate the problem of finding the disparity map for each layer as a global optimization. The objective is to find a disparity function d that minimizes a global energy given by

$$E(d) = E_{\text{data}}(d) + \lambda \cdot E_{\text{smooth}}(d). \tag{1}$$

The data term, $E_{\text{data}}(d)$, measures how well the disparity function d agrees with the input images. Using the disparity space formulation,

$$E_{\text{data}}(d) = \sum_{p \in (\theta, v)} \phi(p, d(p)), \tag{2}$$

where ϕ is the matching cost (robust variance) in the DSI. Recall that p represents the ray direction (θ, v) in our cylindrical coordinates.

The smoothness term $E_{\text{smooth}}(d)$ encodes the smoothness assumptions which in our case encourages a piece-wise smooth result:

$$E_{\text{smooth}}(d) = \sum_{(p,q) \in \mathcal{C}} \rho_d(d_p, d_q) \rho_I(\mu(p, d_p), \mu(q, d_q)), \tag{3}$$

where $\mathcal{C}$ is the set of 4-connected neighbors in (θ, v), d_X is the depth at X, ρ_d is a monotonically increasing function of disparity difference, and ρ_I is a monotonically decreasing function of intensity differences that lowers smoothness costs at high intensity gradients.

$$\rho_d(d_p, d_q) = min(|d_p - d_q|, c_1), \tag{4}$$

$$\rho_I(\mu(p, d_p), \mu(q, d_q)) = e^{c_2 |\mu(p,d_p) - \mu(q,d_q)|}, \tag{5}$$

We use $c_1 = 2.0$ and $c_2 = -0.01$ for all of our examples. Our smoothness term encourages disparity discontinuities to coincide with intensity/color edges, which accounts for some of the good performance of global optimization stereo approaches. Note because we do not have an image taken from the virtual camera, we approximate the intensity with the median from the DSI.

We balance the data term and the smoothness term using a constant $\lambda = 2.0$ for all examples. Once the global energy has been defined, we use the graph cut alpha-expansion algorithm of Boykov *et al.* [4] to solve for the label d.

3.3.2 Color

In addition to depth, we also recover the color for each pixel in each layer of the LDP. Since we are reconstructing a global texture for each layer, we will make the simplifying Lambertian assumption and determine a single color for each entry in the LDP.

We solve for the color of each entry in a layer in a similar fashion to the disparity determination, by leveraging a label optimization procedure. In this step, the labels identify the input image from which to pull the pixel color to assign to the layer. This avoids blurring caused by simply blending all input pixels that project to a particular voxel.

For each voxel $V = (\theta, v, d)$, we find all the input images that see it by reprojecting the voxel location back into those images accounting for visibility between the voxel and input cameras [30]. The indices of the visible input cameras form the set of candidate labels, and we find the best labeling once again using graph cuts.

As before, we define cost functions to express the desired properties of a labeling l:

$$E'(l) = E'_{\text{data}}(l) + \lambda \cdot E'_{\text{smooth}}(l). \tag{6}$$

The data term $E'_{\text{data}}(l)$ reflects the property that each pixel p in the scene should be imaged from a viewpoint l_p that is

most aligned with the virtual camera. It is specified as

$$E'_{\text{data}}(l) = \sum_{p \in (\theta, v)} \cos^{-1}(p \cdot (V - C_{l_p})), \qquad (7)$$

where V is the position of the voxel and C_{l_p} is the center of camera l_p. This forces using rays from the closest cameras to compute the color of the ray p.

$E'_{\text{smooth}}(l)$ measures how well the labeling agrees with our smoothness assumption. $E'_{\text{smooth}}(l)$ has the form

$$E'_{\text{smooth}}(l) = \sum_{(p,q) \in \mathcal{C}} \eta_l(l_p, l_q) \cdot \eta_d(d_p, d_q), \qquad (8)$$

where $\mathcal{C}$ is the set of 4-connected neighbors in (θ, v), η_l is a monotonically increasing function of the distance between cameras, and η_d is a monotonically decreasing function of disparity differences that lowers smoothness costs at high depth discontinuities.

$$\eta_l(l_p, l_q) = min(|l_p - l_q|, c_3), \qquad (9)$$

$$\eta_d(d_p, d_q) = e^{c_4 |d_p - d_q|}, \qquad (10)$$

We use $c_3 = 2.0$ and $c_4 = -0.1$ for all of our examples. Our smoothness term encourages the camera labeling to align with depth discontinuities. Again, we use the same graph cut alpha-expansion algorithm to compute the labeling l.

3.3.3 Depth and Color for Subsequent Layers

Computing the depth and color of layers beyond the first one proceeds almost exactly as for the first layer. One difference is that we first remove from consideration any voxels (θ, v, d) for which the depth d is less than or equal to the depth at the corresponding pixel in the first layer $L_1(\theta, v)$.

The robust variance of the intensity that projects to the remaining voxels is computed using median absolute variance (MAD). Note, however, that due to occlusion by the first layer, many voxels behind the first layer will no longer be visible to any input camera. In addition, voxels that are visible through the cracks induced by depth discontinuities will typically be visible in only one camera. The optimizations for depth and color then proceed as before for the second, and if desired, third layer.

3.3.4 Refinement of the LDP

Note that during determination of the first layer, we assumed all voxels were visible. However, even the first layer induces self occlusion; thus we can refine the process in an EM-like iteration [27]. Assuming the depths of the first layer are correct, we recompute the DSI taking into consideration visibility information. From this we recompute the first layer's geometry, and proceed through the complete construction of the LDP one more time.

4. Results

In this section, we demonstrate our technique on three examples (two outdoor and one indoor) with varying amounts of scene complexity. Results are shown in Figures 3-5. Reconstructions at the end of each figure show the benefits of multiple depth layers for avoiding holes vs. reconstructions of a single layer as in [14]. Readers are encouraged to check out the supplementary material, which contains the inputs and results at the original resolutions, as well as videos of rendering results.

All scenes were captured by the user holding the camera arm length away from the body and capturing approximately 20 (800×600 pixel) images along an arc with approximately 75% overlap. The scenes exhibit large depth variation, resulting in significant parallax and thus large occluded areas. Two of the scenes contain numerous small surfaces, such as leaves and branches for the outdoor scene and individual fruits and signs for the indoor scene.

The camera positions were estimated using an off-the-shelf structure from motion (SFM) system [26] which recovers both the intrinsic and the extrinsic parameters of the camera. We fit a circular arc to the recovered camera positions using least-squares.

We constructed LDPs with two different parameterizations. The first simple representation with all rays converging on a central point at the center of the camera arc. This is a cylindrical analogue to a Layered Depth Image. The second, and more successful parameterization, is the one described in the paper, which is a layered version of a cylindrical analogue to a "pushbroom" panorama. Figures 3 and 4 show how the cylindrical pushbroom results in a better depth estimate due to the rays being in better proximity to the original images. We used 16 depth labels and computed 2-3 layers depending on scene complexity. Each LDP computation took 1-2 hours on a 3.2GHz computer with 2GB memory.

The first and simplest example is shown in Figure 3. It depicts a couple sitting on a wall with more distant trees, buildings and a mountain in the background. This is typical of many "tourist" shots of a family member standing in front of a vista.

As expected, the depth map results show a clear separation between the foreground couple and the background. The first layer depicts the scene (with depth) similar to a view from the center of the arc. The second layer includes only those pixels hidden by the first layer that are revealed as one would move the viewpoint along the camera arc. The sparsity of this second layer shows the efficiency in the LDP representation while allowing a viewing experience depicting significant parallax (see the videos in supplementary materials).

The second and third scenes, a garden in front of a house (Figure 4), and a pile of melons in a market (Figure 5) are

significantly more complex. The reconstruction once again faithfully finds depths for both the first and hidden layers. The second layers are not as sparse in these scenes due to the many depth discontinuities; however, they are still quite sparse compared to a full panorama. A third layer was also generated for the second scene (see images in supplementary materials). It is much sparser than the second layer, yet is helpful for filling small cracks in rendering.

The size of an *uncompressed* LDP is less than twice as large as a normal panoramic image. The data includes the first layer which is equivalent in size to a panorama. The two depth layers are 4 bits each (for 16 disparity layers) but are spatially coherent. The second layer texture and depth typically contain significantly fewer pixels although there is some overhead encoding the sparsity.

5. Discussion

Our system currently computes each layer sequentially. Such ordering dependency decreases the robustness of the system if errors get introduced at an early stage. Iterative methods could potentially alleviate such problem by solving all layers simultaneously, although this would result in a higher computational cost.

We can achieve smoother looking results if we allow voxels to be partially opaque at the boundaries of objects. Adding a matting component as a post-processing step for each layer as was done in [33] would definitely help.

The back layers in our LDP representation are usually quite sparse, containing many small yet non-rectangular shaped regions. Standard compression techniques support such type of data, yet with some overhead. We expect to be able to exploit compression methods such as in [33], with the added benefit that each layer should help predict voxels seen in further layers. Finally, we are exploring faster rendering methods to take advantage of current graphics hardware to make the viewing experience more interactive.

6. Conclusion and future work

In this paper, we have developed a technique for creating a layered representation from a sparse set of images taken with a hand-held camera. This concise representation, which we call a layered depth panorama (LDP), allows the user to experience wide angle panoramas including the parallax associated with off-axis panning. The added 3D experience incurs a reasonable cost in terms of space efficiency (only about twice the size of the equivalent panorama). We formulate the problem of constructing the LDP as the recovery of color and geometry in a multi-perspective cylindrical disparity space. We introduce a new cylindrical pushbroom parameterization to closely follow the array of input images. Graph cut is leveraged to sequentially determine the disparity and color of each layer using multi-view stereo. As demonstrated both in the paper and the supplementary videos, our approach is able to achieve high quality results on a variety of complex outdoor and indoor scenes.

References

[1] G. Bahmutov, V. Popescu, and E. Sacks. Depth enhanced panoramas. In *Proceedings of the conference on Visualization '04*, page 598.11, 2004.

[2] S. Baker, R. Szeliski, and P. Anandan. A layered approach to stereo reconstruction. In *Proc. of CVPR'98*, pages 434–441, 1998.

[3] S. Birchfield and C. Tomasi. Multiway cut for stereo and motion with slanted surfaces. In *Proc. of ICCV'99*, pages 489–495, 1999.

[4] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. on PAMI*, 23(11), 2001.

[5] M. Brown and D. Lowe. Recognising panoramas. In *Proc. of ICCV'03*, volume 2, pages 1218–1225, 2003.

[6] C. Buehler, M. Bosse, L. McMillan, S. J. Gortler, and M. F. Cohen. Unstructured lumigraph rendering. *ACM Trans. Graph.*, pages 425–432, August 2001.

[7] P. Gargallo and P. Sturm. Bayesian 3D modeling from images using multiple depth maps. In *Proc. of ICCV'05*, volume 2, pages 885–891, 2005.

[8] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen. The Lumigraph. In *ACM Trans. Graph.*, pages 43–54, 1996.

[9] S. S. Intille and A. F. Bobick. Disparity-space images and large occlusion stereo. In *Proc. of ECCV'94*, volume 1, pages 179–186, 1994.

[10] S. B. Kang and R. Szeliski. 3-D scene data recovery using omnidirectional multibaseline stereo. *IJCV*, 25(2):167–183, November 1997.

[11] S. B. Kang and R. Szeliski. Extracting view-dependent depth maps from a collection of images. *IJCV*, 58(2):139–163, July 2004.

[12] V. Kolmogorov and R. Zabih. Multi-camera scene reconstruction via graph cuts. In *Proc. of ECCV'02*, pages 82–96, 2002.

[13] M. Levoy and P. Hanrahan. Light field rendering. In *ACM Trans. Graph.*, pages 31–42, 1996.

[14] Y. Li, H.-Y. Shum, C.-K. Tang, and R. Szeliski. Stereo reconstruction from multiperspective panoramas. *IEEE Trans. on PAMI*, 26(1):45–62, 2004.

[15] M. Lin and C. Tomasi. Surfaces with occlusions from layered stereo. *IEEE Trans. on PAMI*, 26(8):710–717, 2004.

[16] D. Lowe. Distinctive image features from scale invariant keypoints. *IJCV*, 2004.

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

[18] S. Peleg, M. Ben-Ezra, and Y. Pritch. Omnistereo: Panoramic stereo imaging. *IEEE Trans. on PAMI*, 23(3):279–290, 2001.

[19] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *IJCV*, 47(1-3):7–42, 2002.

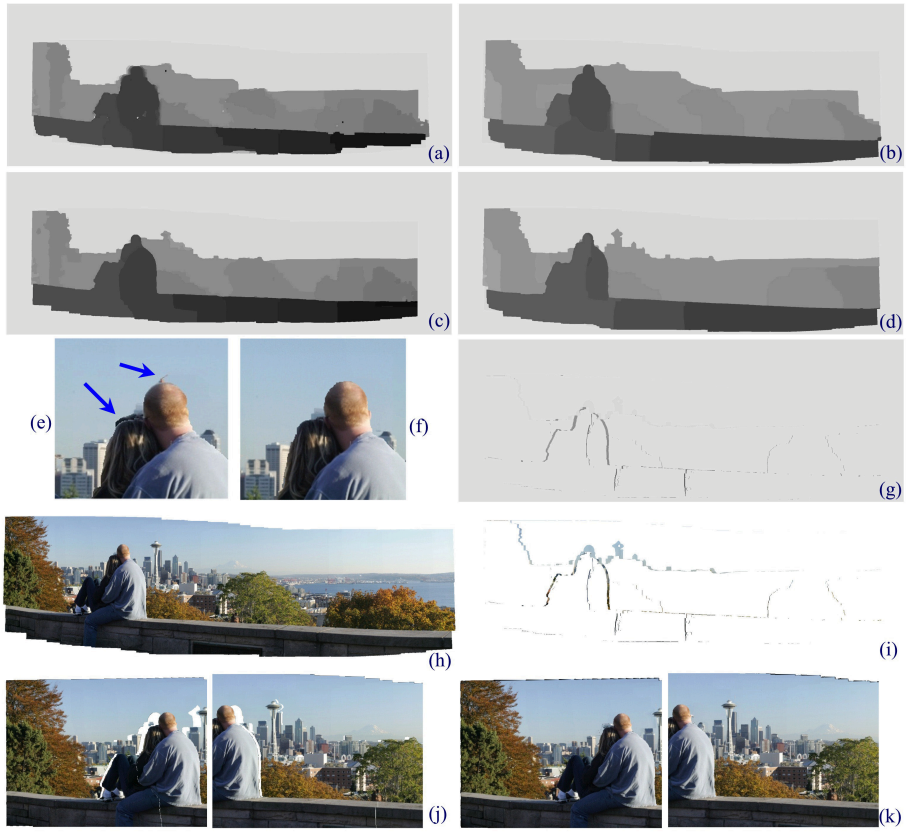

Figure 3. City scene and results: (a) and (b) show the first layer depths after one and two iterations generated from a global cylindrical center, (c) and (d) uses the arc-based pushbroom parameterization for depths, after one iteration and a second that accounts for visibility, (e) and (f) are details from the first layer textures comparison using the central and pushbroom parameterizations (note artifacts in the centered parameterization), (g) the second layer depth, (h) and (i) are the textures associated with the first and second layers, (j) two reconstructions from the first layer only showing obvious holes, and (k) the same reconstruction using two layers.

[20] S. Seitz et al. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Proc. of CVPR'06*, volume 1, pages 519–526, 2006.

[21] S. Seitz and J. Kim. The space of all stereo images. *IJCV*, 48(1):21–38, June 2002.

[22] S. M. Seitz and C. R. Dyer. Photorealistic scene reconstruction by voxel coloring. In *Proc. of CVPR'97*, pages 1067–1073, 1997.

[23] J. Shade, S. Gortler, L.-W. He, and R. Szeliski. Layered depth images. *ACM Trans. Graph.*, pages 231–242, 1998.

[24] H.-Y. Shum and L.-W. He. Rendering with concentric mosaics. *ACM Trans. Graph.*, 33:299–306, 1999.

[25] H.-Y. Shum, J. Sun, S. Yamazaki, Y. Li, and C.-K. Tang. Pop-up light field: An interactive image-based modeling and rendering system. *ACM Trans. Graph.*, pages 143–162, 2004.

[26] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: Exploring photo collections in 3D. *ACM Trans. Graph.*, 25:835–846, 2006.

[27] C. Strecha, R. Fransens, and L. V. Gool. Combined depth and outlier estimation in multi-view stereo. In *Proc. of CVPR'06*, volume 2, pages 2394–2401, 2006.

[28] R. Szeliski. A multi-view approach to motion and stereo. In *Proc. of CVPR'99*, volume 1, pages 157–163, 1999.

[29] R. Szeliski. Image alignment and stitching: A tutorial. *Foundations and Trends in Computer Graphics and Computer Vision*, 2(1), 2006.

[30] R. Szeliski and P. Golland. Stereo matching with transparency and matting. In *Proc. of ICCV'98*, pages 517–526, 1998.

[31] R. Szeliski and H.-Y. Shum. Creating full view panoramic image mosaics and texture-mapped models. *ACM Trans. Graph.*, pages 251–258, 1997.

[32] Z. Zhu and A. R. Hanson. 3D LAMP: A new layered panoramic representation. In *Proc. of ICCV'01*, volume 2, pages 723–730, 2001.

[33] C. L. Zitnick, S. B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski. High-quality video view interpolation using a layered representation. *ACM Trans. on Graph.*, 23(3):600–608, August 2004.

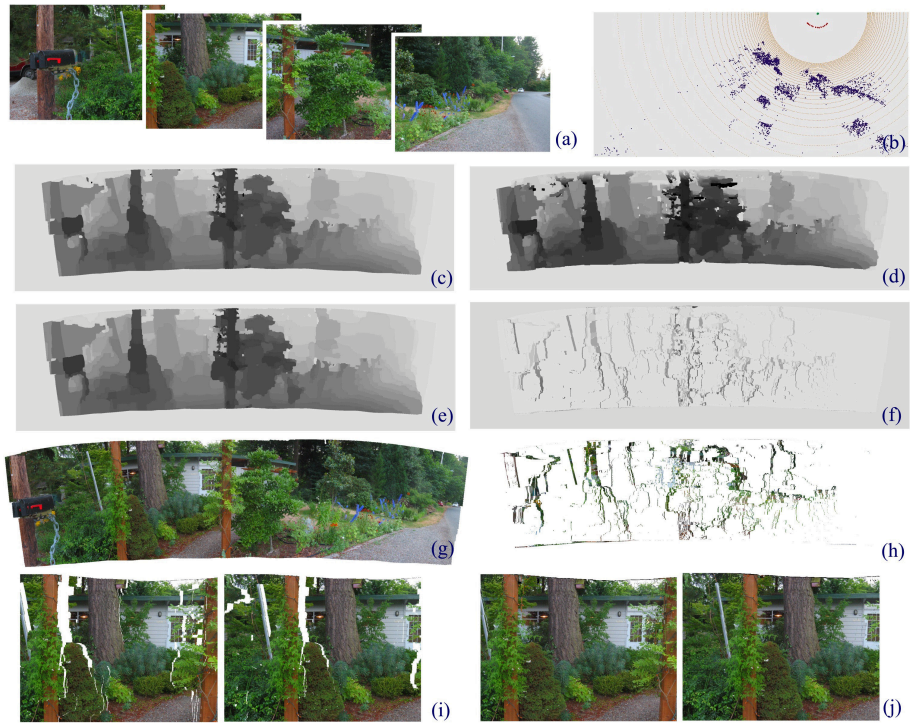

Figure 4. Garden scene and results: (a) are the input images (4 out of 14 shown here), (b) is a top-down view of camera positions and scene points recovered from SFM, (c) shows the front depth distribution after one iteration using our arc-based parameterization, (d) and (e) show the frontal depth distribution after multiple iterations and accounting for visibility, d) is generated from a global cylindrical center, and (e) uses our arc-based parameterization, (f) depths of second layer, (g) and (h) textures of first and second layer using our arc-based parameterizaton, (i) and (j) are two virtual views rendered with only the front layer (showing some holes), and all the layers.

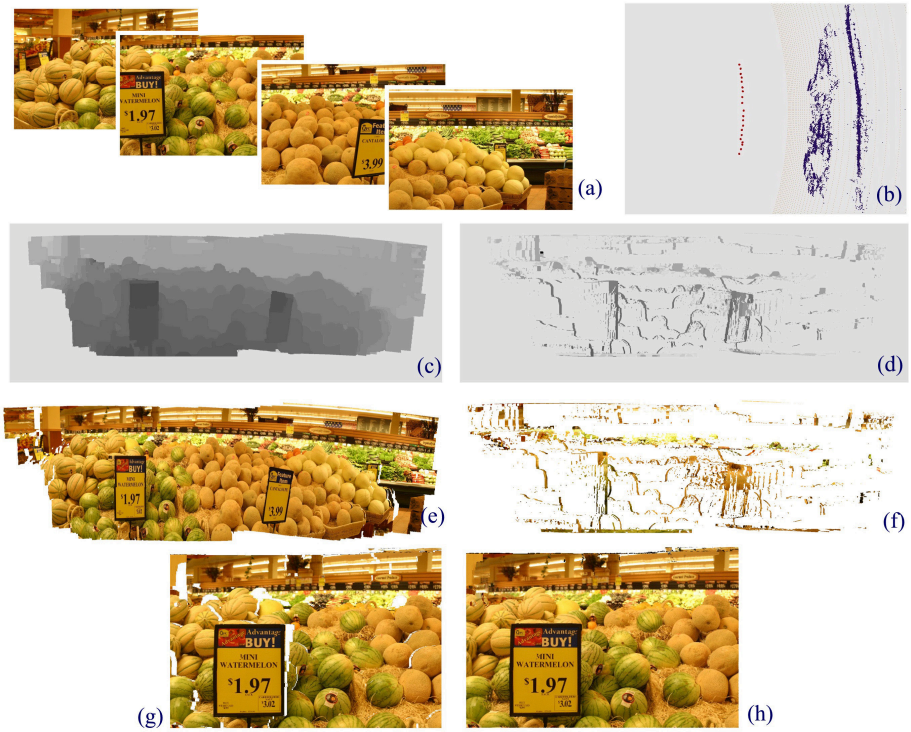

Figure 5. Market scene and results: (a) are the input images (4 out of 20 shown here), (b) camera positions and scene points recovered from SFM, (c) and (d) depth distributions of first and second layer, (e) and (f) texture of the first and second layer, (g) and (h) are virtual views rendered with only the front layer (showing some holes), and all the layers.