

Efficient Dense Depth Estimation from Dense Multiperspective Panoramas *

Yin Li, Chi-Keung Tang
Computer Science Department, HKUST
Hong Kong, P.R.C.
{liyin,cktang}@cs.ust.hk

Heung-Yeung Shum
Microsoft Research, China
Beijing, P.R.C.
hshum@microsoft.com

Abstract

In this paper we study how to compute a dense depth map with panoramic field of view (e.g., 360 degrees) from multiperspective panoramas. A dense sequence of multiperspective panoramas is used for better accuracy and reduced ambiguity by taking advantage of significant data redundancy. To speed up the reconstruction, we derive an approximate epipolar plane image that is associated with the planar sweeping camera setup, and use one-dimensional window for efficient matching. To address the aperture problem introduced by one-dimensional window matching, we keep a set of possible depth candidates from matching scores. These candidates are then passed to a novel two-pass tensor voting scheme to select the optimal depth. By propagating the continuity and uniqueness constraints non-iteratively in the voting process, our method produces high-quality reconstruction results even when significant occlusion is present. Experiments on challenging synthetic and real scenes demonstrate the effectiveness and efficacy of our method.

1 Introduction

Computing a dense depth map with a large field of view (e.g., 360 degrees) has many applications such as large environment navigation. One way to achieve it is to merge reconstruction results from traditional stereo of two regular images with limited field of view. However, in complex real scenes, the accumulation error can quickly add up, which may fail this straightforward alternative miserably. Another problem with traditional stereo is that the resulting depth map is sparse, since matching is usually performed on a limited number of feature points. Such a sparse depth map may be inadequate for some applications requiring photorealism.

An alternative approach is to apply stereo algorithms to panoramic images, bypassing the need of merging intermediate representation. In [3], a multi-baseline stereo algorithm is proposed that employs omni-directional panoramic images. But the epipolar constraints are no longer straight lines [6]. Most recently, multiperspective panoramas [14] have also been proposed to reconstruct large environments. Unlike conventional images, multiperspective panoramas capture parallax effects, as

each column of pixels is taken from a different perspective point. Optimal configurations for such stereo setups are also studied in [12]. It has been shown in [14] that the imaging geometry of multiperspective panoramas can be greatly simplified for depth reconstruction.

To improve the reconstruction accuracy, multiple images can be used [8, 2, 11, 10, 4]. It has been shown that by using multi-baseline stereo, match ambiguities can be reduced and precision can be improved as well. See [8] for an inspiring discussion. However, the computation cost involved in using dense samples (e.g., in [3, 12, 14]) may be an issue. For example, [10] presents a maximum-flow formulation of the general N -camera stereo problem that produces a dense disparity map. The minimum cut of a graph is the desired disparity surface. While it does not use an iterative minimization scheme, as noted in [10], its time complexity is $O(n^2 d^2 \log(nd))$, where n is the total number of image pixels, and d is the depth resolution (although the average case has lower complexity). Similarly, the sweeping algorithms are computationally expensive as well.

In this paper, we present an efficient algorithm to compute a panoramic depth map from dense multiperspective panoramas. Our system is similar to that in [14] where multiperspective panoramas re-sampled from a dense sequence of images are used for stereo reconstruction. However, we use a dense sequence of (hundreds) concentric mosaics whereas only several mosaics are used in [14]. By taking advantage of the inherent linearity property in the imaging geometry, we derive approximate epipolar plane images (EPI [1]). Therefore, stereo reconstruction can be obtained by applying a $1D$ matching window to the EPI. Our algorithm runs in linear time and space with respect to the total number of pixels. The $1D$ matching window is efficient; and our algorithm does not need iterate for each pixel (e.g. [15]).

Similar to conventional stereo algorithms, depth reconstruction from dense multiperspective panoramas must also deal with depth discontinuities and occlusions. A common solution is to adopt a constrained functional optimization problem (e.g. [9]), such as relaxation and dynamic programming. In [14], for example, a cylinder sweep stereo is proposed for multiperspective panoramas, and a post-processing regularization step is applied to obtain smooth depth map. Owing to the inherent incompatibility of smoothness and discontinuity information [5], however, it is difficult to represent occlusions in a single continuous objective function. Moreover, functional optimization is usually

*This work is supported by the University Grant Council: Area of Excellence in Information Technology Grant, and the Research Grant Council of the Hong Kong Special Administrative Region, China under grant number HKUST 6426/00E.

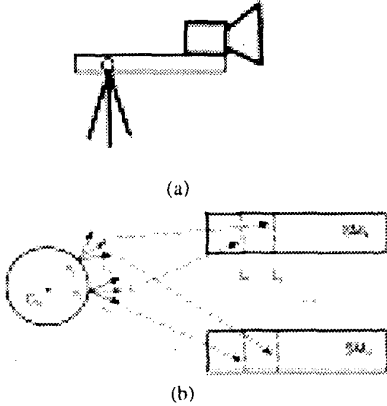


Figure 1 The camera setup: (a) acquisition, and (b) concatenation process.

implemented as an iterative algorithm, and thus, initialization, convergence, and parameter dependence are problematic. In this paper, we apply tensor voting [7] to impose the continuity and uniqueness constraints, while preserving depth discontinuities. Other approaches (e.g., Zitnick and Kanade [15]) have been proposed. But they have higher complexity than tensor voting, since the process has to be iterated for each pixel.

It is worth to note that the use of 1D matching windows in our stereo algorithm will inevitably run into the aperture problem. Our solution is to keep a set of possible inverse depth maps at the initial reconstruction stage; the aperture problem is then solved by an adaptive smoothing criterion in the first pass of tensor voting, which also removes wrong matches and handles depth discontinuities. The uniqueness constraint is then applied by the second pass of tensor voting, so that the inverse depth with maximum directional support is our output.

The outline of this paper is as follows: we first describe a practical camera setup (section 2) to capture our dense image sequence. We then describe our 1D gradient matching algorithm (section 3), and the use of tensor voting to vote for the maximum depth non-iteratively (section 4). We analyze the time and space complexities of our method (section 5). Finally, we present results on complicated simulated environment as well as real data.

2 Dense Multiperspective Panoramas

In this section, we describe a sweeping camera setup that captures a dense set of multiperspective panoramas, and state two properties of this imaging geometry.

2.1 The camera setup

Figure 1 shows a camera setup used to capture our multiperspective panoramas. This setup is the same as the one used in [13]. We swing an off-the-shelf camera mounted on a rotating bar looking outward. The rotation speed is kept constant. Images are sampled at equal time interval during the rotation. Corresponding columns of pixels across the sampled image sequence are concatenated to form a multiperspective panorama. An image sequence of F frames of size $W \times H$ can be concatenated into (up to) W panoramas of size $F \times H$. Some sample

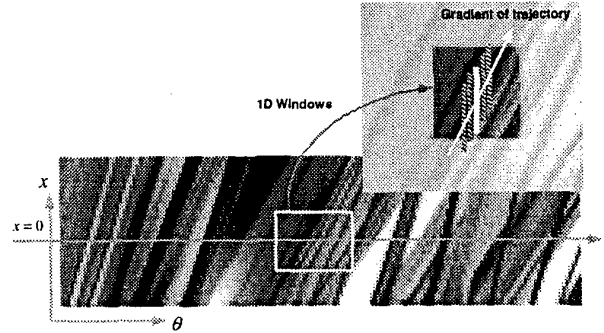


Figure 2 An example EPI from the real scene of balcony in 9. Note the linear patterns. A 1D window is used to estimate the line gradient, which indicates inverse depth.

panoramas can be seen in Figure 8 and Figure 9.

2.2 Two properties of the imaging geometry

The imaging geometry of multiperspective panoramas with a planar rotating camera was first introduced in [14]. We now present two important properties of the imaging geometry, especially with the epipolar geometry. More details can be found in the appendix.

- *Horizontal epipolar geometry.* The imaging geometry can be well approximated by horizontal epipolar geometry so that corresponding image points lie on the same scanlines across the captured image subsequence. An epipolar plane image, or EPI, is shown in Figure 2. It is obtained by concatenating corresponding scanlines where x indicates pixel location and θ represents the rotation angle of the camera. (See Figure 3 for more details.)
- *Linearity.* A straight line in an EPI indicates the locus or trajectory of an image point. We restate Equation (8), which is derived in the appendix:

$$K_{epi} = 1 + \frac{1}{D}$$

where K_{epi} is a line gradient or slope, and $\frac{1}{D}$ is the inverse depth. Note that K_{epi} is independent of x and θ .

Based on these properties, we conclude that matching requires only 1D search in a single EPI. Specifically, matching can be implemented as 1D convolution using a *constant* 1D search window. No rectification is needed. Because Equation (8) is linear, we can quantize the inverse depth uniformly, without any bias or negligence.

3 Dense Depth Estimation from EPI

Here, we estimate K_{epi} so that it can be plugged into Equation (8) for depth estimation.

3.1 Gradient Estimation in EPI

By construction (Figure 7 in appendix), an EPI is indexed by x and θ (Figure 2). Let $I(x, \theta)$ be an EPI. Given any θ , we define an 1D window, $W(\theta)$, to be

$$W(\theta) = \{I(x_i, \theta) | x_i \in [-w, w] \text{ for some integer } w\}. \quad (1)$$

Typical value of w is 5.

Suppose we slide this 1D window along a direction (Figure 2), and compute the consistency of pixel colors between this 1D window and the overlapping pixels. K_{epi} is therefore equal to the direction that produces the maximum consistency.

Let θ_0 be the location of the 1D window centered at $x = 0$ (reference image). To compute color consistency, we compute sum of squared difference, or SSD , with direction K at θ

$$SSD(K, \theta)|_{\theta_0} = \sum_{i=-w}^w [I(x_i + (\theta - \theta_0)K, \theta) - I(x_i, \theta_0)]^2 \quad (2)$$

With multiple images, we adopt $SSSD$ (sum of SSD) for the reference image at θ_0 in a neighborhood of size M (typically set to be 5) to compute color consistency as

$$SSSD(K)|_{\theta_0} = \sum_{n=-M}^M SSD(K, \theta_n)|_{\theta_0}. \quad (3)$$

3.2 Computing Potential Inverse Depth Image

If we have approximate knowledge on the minimum and maximum depth of the scene (e.g., in [13]), the range of K can be determined by Equation (8). We perform uniform quantization: $K_n = K_{min} + \frac{2n-1}{2N}(K_{max} - K_{min})$, $n = 1 \dots N$, where K_{min} and K_{max} are the minimum and maximum K_{epi} corresponding to the maximum and minimum depth, respectively.

For each 1D window at θ , we compute all $SSSD(\cdot)$ for each quantized K_n . Define¹ $P(\theta, K_n) = 1 - SSSD(K_n)|_{\theta}$. Thus, the larger $P(\theta, K_n)$, the more probable that the depth corresponding to K_n is our solution. We normalize $P(\theta, K_n)$ so that it ranges from 0 to 1:

$$\bar{P}(\theta, K_n) = \frac{P(\theta, K_n)}{\sum_{i=1}^N P(\theta, K_i)}, n = 1 \dots N \quad (4)$$

Each \bar{P}_θ is a depth belief vector along the line of sight, where $\bar{P}_\theta = \{\bar{P}(\theta, K_n) | n = 1 \dots N\} = [\bar{P}(\theta, K_0), \bar{P}(\theta, K_1), \dots, \bar{P}(\theta, K_N)]^T$. If we concatenate all \bar{P}_θ vectors, we obtain a 2D potential inverse depth image (several are shown in Figure 3), where the brightest locations indicate the most probable inverse depth silhouette (curve).

3.3 Extracting Inverse Depth Surface

By now, we know how to produce a 2D potential inverse depth image from an EPI. By juxtaposing all 2D potential inverse depth images resulted from their respective EPI's along the Y -direction (Figure 3), a 3D potential inverse depth image $\bar{P}(Y, \theta, K_n)$ is obtained. Analogous to the 2D case, if we make the intensity level at each voxel (Y, θ, K_n) of this 3D map be proportional to $\bar{P}(Y, \theta, K_n)$, the depth silhouette, as given by the brightest locations, will indicate the most probable inverse depth surface. The problem of depth estimation can thus be translated into one of extracting this surface (S) from the 3D potential depth image assuming the scene is opaque:

$$S = \bigcup_{Y, \theta} \{(Y, \theta, K_n) | \bar{P}(Y, \theta, K_n) \geq \bar{P}(Y, \theta, K_i), i = 1 \dots N\} \quad (5)$$

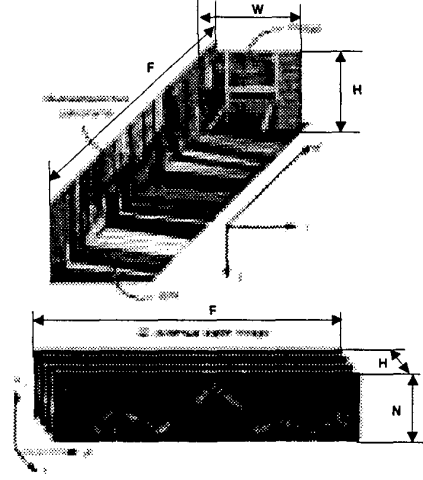


Figure 3 An EPI is produced by extracting a horizontal slice as shown. Each EPI corresponds to a 2D potential inverse depth image of a scanline in the corresponding multiperspective panorama. A 3D potential inverse image is produced by stacking together all the 2D images along the Y -direction.

For every (Y, θ) , we output the voxel with the maximum $\bar{P}(Y, \theta, K_n)$, among all the N candidates along the line of sight. Unfortunately, this straightforward algorithm only works well for locations with rich textures. For example, Figure 6(a) shows a (θ, K_n) slice depicting maximum $\bar{P}(\cdot)$. Note that outliers and depth discontinuities are clearly visible. To deal with these problems of outliers and occlusions, which are typical to traditional stereo matching as well. We propose to use tensor voting [7] to address them.

4 Depth Estimation by Tensor Voting

In this section, a two pass algorithm based on tensor voting for depth estimation is described. Given the initial set of matches from the 3D potential inverse depth image $\bar{P}(Y, \theta, K_n)$, our objectives are to

1. remove noisy wrong matches, and infer smooth features which are possibly missed due the aperture problem associated with a 1D matching window,
2. infer the missing matches after noise removal, and compute the inverse depth with maximum support,

while preserving depth discontinuities in both cases. Two passes of tensor voting are used. The first pass propagates the continuity constraint to achieve step (1). After removing outlier matches, a reliable set of inverse depths is obtained. The second pass achieves step (2) by applying the uniqueness constraint. A large number of tensor votes is collected. The solution with maximum support along the line of sight is produced.

4.1 Terminologies of Tensor Voting

Tensor voting uses a second order symmetric *tensor* for data representation, and a *voting* methodology for data communication. Each input site is encoded as a tensor, propagating preferred direction in a neighborhood. In essence, we collect a large

¹Note that $SSSD$ should first be normalized to $[0, 1]$ by the window size.

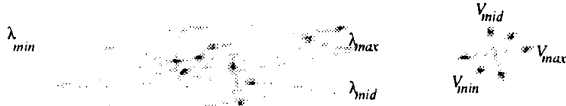


Figure 4 A second order symmetric tensor in 3D. The equivalent eigensystem is shown.

number of tensor votes at each input point in order to attenuate the effect of outlier noise, and analyze their direction consistency simultaneously. If there is a high agreement in normal direction, it indicates a high surface saliency. If there is a high disagreement in normal direction, it indicates a surface orientation discontinuity. If only a small number of inconsistent votes is received, the point should be an outlier. We now introduce terminologies which will be used in this section.

Representation as tensors A point in the 3D space can assume one of the followings: a surface patch, a discontinuity, or an outlier. A point on a smooth surface is very certain about its surface normal orientation (or stick tensor), while at a point junction at which surfaces intersect has absolute orientation uncertainty (indicated by a ball tensor). A second order symmetric tensor in 3D is used to represent this continuum. This tensor representation can be visualized as an ellipsoid (Figure 4). To describe it, we use an eigensystem with three unit eigenvectors \hat{V}_{max} , \hat{V}_{mid} , and \hat{V}_{min} and three eigenvalues $\lambda_{max} \geq \lambda_{mid} \geq \lambda_{min}$. $\lambda_{max} - \lambda_{mid}$ is used to indicate *surface saliency* [7].

Data communication by voting First, we encode the input into a set of *default tensors*: If the voxel contains an input point, we associate it with a 3D default ball tensor, having all $\lambda_{max} = \lambda_{mid} = \lambda_{min}$, and $\hat{V}_{max} = [1\ 0\ 0]^T$, $\hat{V}_{mid} = [0\ 1\ 0]^T$, and $\hat{V}_{min} = [0\ 0\ 1]^T$. Otherwise, if the voxel does not contain an input point, it is associated with a *zero tensor* (i.e. zero eigenvalues and zero eigenvectors). These input tensors *cast votes*, or are made to align (by translation and rotation), with predefined *voting fields*. In particular, we describe the *ball voting field* here, which is used for depth estimation in this paper. One slice on the x - y plane of this 3D tensor field is shown in Figure 5. It is a dense isotropic field without any orientation preference, which propagates all possible directions in a neighborhood with equal likelihood. The neighborhood size is determined by the scale of analysis, or equivalently, the size of the voting field.

When each input point has cast its tensor vote to its neighboring voxels by aligning with the ball voting fields, each voxel in the volume receives a set of tensor votes. These votes are collected, using *tensor addition*, as a 3×3 covariance matrix of second order moment collection of all the vote contribution. Upon eigensystem analysis, we obtain a generic saliency tensor or ellipsoid, encoding preferred normal orientation and discontinuity information by the stick and the ball tensors, respectively.

4.2 Pass One – Continuity Constraint

Recall that our input is a 3D potential depth map, where each voxel contains a measure $\bar{P}(\cdot)$.

In the first pass, S is first computed, where S is the set of voxel locations whose $\bar{P}(\cdot)$ is maximum among all values along the line of sight, as defined earlier in Equation (5). The algorithm is summarized as follows, along with a running example:



Figure 5 One slice of the 3D ball voting field, which propagates all directions in equal likelihood in a neighborhood.

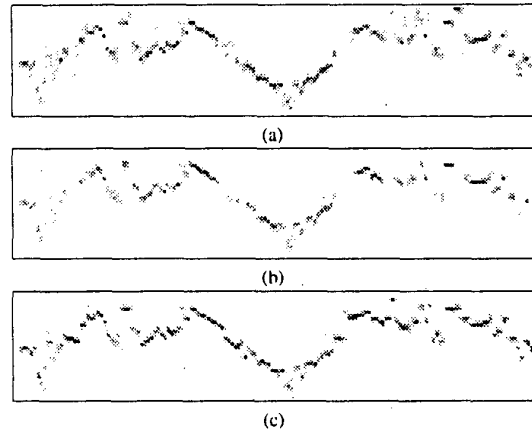


Figure 6 Running example. (a) the candidate set S with maximum SSSD along the line of sight. (b) outlier removal and discontinuity preservation by applying the smoothness constraint. (c) missing details are filled in by applying the uniqueness constraint.

1. Compute S . Encode S into a set of default ball tensors. All eigenvalues are made equal to its $\bar{P}(\cdot)$ (Figure 6(a)).
2. Compute V , the set of voxel locations whose associated $\bar{P}(\cdot) \geq p_1$, and $S \cap V = \emptyset$. The choice of $0 \leq p_1 < 1$ (e.g. $p_1 = 0.01$) is not critical, since voxel locations in V only cast votes but do not collect votes. We also encode V into a set of ball tensors, with all the eigenvalues equal to their respective $\bar{P}(\cdot)$'s.
3. The encoded S and V vote with the ball voting field.
4. S collects votes by tensor addition. The resulting eigensystem is computed.
5. A subset of points in S , whose normalized surface saliencies exceeds some p_2 , is obtained, Figure 6(b).

The choice of p_2 (typical value of p_2 is 0.1) is not critical either, since we collect votes in *every* voxel location in the 3D image in pass 2. Figure 6(a) and (b) respectively depicts the S before and after pass 1. Note that both smooth structures and depth discontinuities are preserved simultaneously, while most of the outliers are eliminated.

Let $\bar{S} \subset S$ be the filtered set shown in Figure 6(b). It provides more reliable evidence. In pass 2, we “densify” the whole 3D volume using \bar{S} by computing a generic tensor vote at all quantized inverse depths.

4.3 Pass Two – Uniqueness Constraint

In pass 2, we apply the uniqueness constraint along the line of sight, and vote for the *maximum inverse depth*: the inverse depth that receives the maximum support from the \bar{S} .

1. Each point in \bar{S} is initially encoded as a ball tensor, with the three eigenvalues set to its surface saliency $ssal = \lambda_{max} - \lambda_{mid}$, which is obtained from its generic saliency tensor inferred at each point after the first pass. By doing so, voters with higher surface saliency are more preferred, since it indicates a higher likelihood that the point should lie on the underlying inverse depth surface.
2. Now, each encoded ball casts ball vote in its neighborhood to “densify” the whole 3D volume. For every (Y, θ) , we compute *all* N tensor votes, received at $(Y, \theta, K_1), (Y, \theta, K_2), \dots, (Y, \theta, K_N)$. A voxel not in \bar{S} will assume a zero tensor initially.
3. When the whole (Y, θ, K_n) volume has collected all non-zero votes, we apply the uniqueness constraint: for each (Y, θ) , we return $K_{Y, \theta}$, that receives the maximum support, or the largest surface saliency along the line of sight:

$$K_{Y, \theta} = \{K_n | ssal(Y, \theta, K_n) \geq ssal(Y, \theta, K_i), i = 1 \dots N\} \quad (6)$$

Figure 6(c) shows one slice of our result. Note that each column consists of only one solution that corresponds to the maximum inverse depth.

5 Complexity Analysis

We analyze the time and space complexities of our algorithm in this section. Let:

- F = horizontal dimension of the panorama (section 2)
- H = vertical dimension of the panorama (section 2)
- N = total number of quantized K_n 's (section 3)
- w = size of the 1D window (Equation (1))
- M = size of neighborhood for computing SSSD (section 3)
- k = size of neighborhood used in tensor voting
- S = the set of maximum $\bar{P}(\cdot)$ (Equation (5))

Since our algorithm does not have any additional space requirement during the computation process, the total space complexity is $O(FHN)$, i.e., the size of the 3D potential depth image (Figure 3). For 1D matching, since $w, M \ll F$, and w, M are constants, each estimation takes only $O(1)$ time. Therefore, the total time complexity for 1D matching is $O(FHN)$.

Tensor voting takes $O(k)$ time per input token [7]. In our case, since we have dense information, typical size of k is 2 (i.e., very small). Therefore, each voting operation essentially takes $O(1)$ time. In the first tensor voting pass, we perform $O(|S|)$ voting operations. Since $|S| = FN$, the time complexity for the first pass is $O(FN)$. In the second pass, we compute a tensor vote for every voxel location in the 3D potential inverse depth image. So, the time complexity is $O(FHN)$.

Therefore, our algorithm runs in linear space and time in total. The constant factor is small, since we use a small voting kernel and do not iterate for each pixel. Typical running time for $F = 1500$, $H = 128$, and $N = 100$ takes about 60 minutes on a Pentium-III 550 MHz.

6 Experiments

We perform experiments on some challenging synthetic and real data to evaluate our method. In all our experiments, $w = 5$, and $M = 5$. Figures 8(a) and (b) show a 360° multiperspectice panorama for a synthetic *Virtual Room* and its corresponding dense depth map by our method. The multiperspectice panorama (a) is then reprojected to a novel view point where occlusions between objects (e.g., teapot, ball) and the wall are clearly visible. Due to the cylindrical mapping, the walls appear curved. Using the depth map shown in Figure 8(d), the teapot can be observed from a novel viewpoint at a lower viewing angle. To demonstrate the high-quality reconstruction of the virtual room, we show the top-down view and the side-top view of the Euclidean reconstruction in Figures 8(f) and (g), respectively. Note that reconstructed four walls are nearly perpendicular, and four objects keep their respective shapes very well. Note that our reconstruction result is much better than that of a previous work [14] on a similar scene.

Figure 9 shows the results on complex real scene, with severe depth discontinuities and camera noise. Figure 9(a) shows a multiperspectice panorama, and Figure 9(b) shows its corresponding depth map by our method. In Figure 9(c), a reprojected depth map from a novel view shows the good quality of our reconstruction. Pay special attention to the middle of the panorama where the wall under windows is curved due to cylindrical mapping. Texture mapped views of the wall and windows displayed in (d) and (e) demonstrated that our method performs well even under significant occlusion. We want to point out once again that it is a very challenging scene with abundant textureless regions and mirror reflections. Figures 9(f) and (g) show the Euclidean reconstruction of the real scene from top-down view and top-side view, respectively. The shape of the interior environment can be clearly observed from the rectangle shape.

7 Conclusion

In this paper, we have proposed an efficient algorithm for computing a dense depth with large field of view (e.g. 360°). To reduce ambiguities and increase precision, we make use of significant data redundancy inherent in a set of dense multiperspectice panoramas. The issue of computational efficiency is solved by our 1D matching algorithm, which is made possible by the linearity constraint in our approximate EPIs. Using tensor voting, we address the aperture problem with an adaptive smoothing criterion which preserves discontinuities, and deals with occlusions, missing data, and outlier matches. This criterion is implemented by properly propagating the continuity and uniqueness constraints, non-iteratively in a neighborhood. We have obtained significant improvement (c.f. [14]) without compromise in computation cost.

8 Appendix

Although similar results have been obtained in [14], the derivation below focuses more on epipolar geometry. Illustrated in Figure 7 is a more detailed geometry of our imaging system. Let O be the center of *sweeping circle*, which is the loci of all optical centers of the rotating camera. The plane on which the

sweeping circle lies is called the *sweeping plane*. The radius of the sweeping circle is assumed to be one, and camera is assumed to be normalized.

Suppose there exists a 3D point P visible to the camera at C_1 (Figure 7). Define P' to be its perpendicular projection onto the sweeping plane. Define C_0 to be the intersection between the sweeping circle and OP' , and θ to be the angular displacement $\angle C_0OC_1$.

Let the 3D coordinates of P be $(X\ Y\ Z)^T$ w.r.t. the camera coordinate system, and let the image coordinates of P be $(x\ y)^T$. Note that each camera coordinate system is obtained by rotating the world coordinate system about its Y -axis by θ , and then to the camera location.

Note, by construction, the Y -axes of all camera coordinate systems are parallel, which further implies that Y , or the Y -coordinate of the point P measured w.r.t. the respective camera coordinate systems, are the *same*.

Since we have normalized camera, $y = Y/Z$. Let $\rho = OP'$. We have $Z = \rho \cos \theta - 1$:

$$\frac{dy}{d\theta} = \frac{d(Y/Z)}{d\theta} = -\frac{Y}{Z^2} \frac{dZ}{d\theta} = \frac{Y}{Z} \frac{\rho}{Z} \sin \theta \quad (7)$$

Therefore, y remains almost constant across our image subsequence (i.e., $\frac{dy}{d\theta} \rightarrow 0$) when the following conditions are satisfied: (1) θ is sufficiently small, (2) image point $(x\ y)^T$ is not too far away from the central scanline, and (3) scene is not too close to the sweeping circle.

In the following, we derive a linear relationship between the gradient of the trajectory and inverse depth. Refer to Figure 7 again. By applying the law of sines to $\triangle OCP'$, we have $\frac{\rho}{\sin(\pi-\alpha)} = \frac{1}{\sin(\alpha-\theta)}$. Since $x = X/Z = \tan \alpha$, we have $x(\cos \theta - \frac{1}{\rho}) = \sin \theta$. Differentiating both sides of this equation w.r.t. θ , we obtain $\frac{1}{\rho} = \cos \theta - \frac{\cos \theta + x \sin \theta}{K_{epi}}$, where $K_{epi} = \frac{dx}{d\theta}$. Thus, K_{epi} is equal to the gradient of the trajectory on the X - θ plane (or equivalently, the EPI). If $\theta \rightarrow 0$ and $x \rightarrow 0$, we have $\frac{1}{\rho} = 1 - \frac{1}{K_{epi}}$ after first order approximation. Now, let $D = C_0P'$, which is equal to the depth of P from sweeping circle. Clearly, $D = \rho - 1$. Finally:

$$K_{epi} = 1 + \frac{1}{D} \quad (8)$$

Acknowledgment

The authors thank Sing Bing Kang for his many constructive suggestions. We would also like to thank Gang Xu, Tao Feng, Zhou Chen Lin for many fruitful discussions while the first author was at Microsoft Research, China.

References

- [1] R. C. Bolles, H. H. Baker, and D. H. Marimont. Epipolar-plane image analysis: An approach to determining structure from motion. *International Journal of Computer Vision*, 1:7–55, 1987.
- [2] R. T. Collins. A space-sweep approach to true multi-image matching. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'96)*, pages 358–363, San Francisco, California, June 1996.

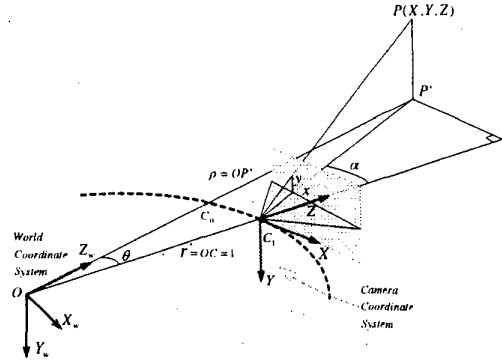
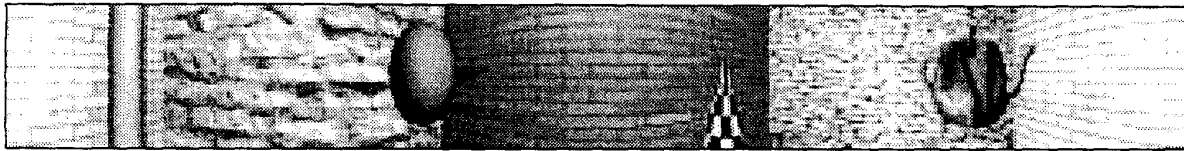
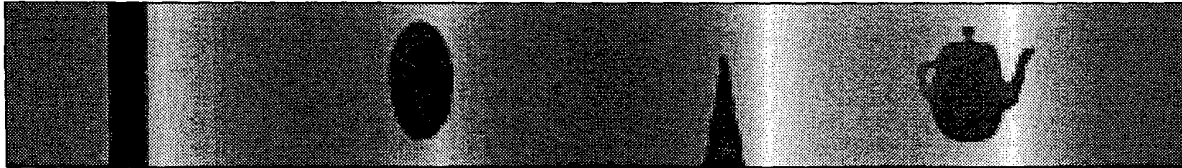


Figure 7 A more detailed description of the imaging geometry of multiperspective panoramas.

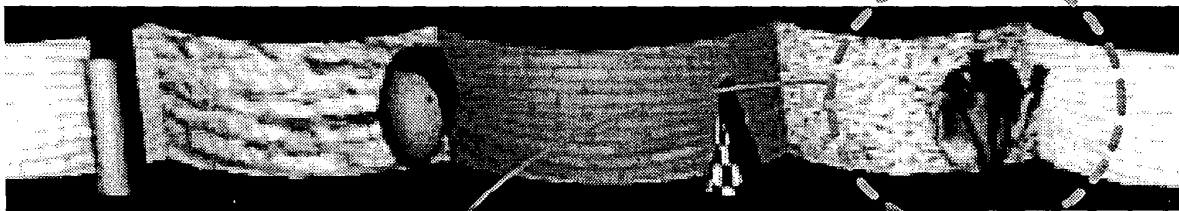
- [3] S. B. Kang and R. Szeliski. 3-D scene data recovery using omnidirectional multibaseline stereo. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'96)*, pages 364–370, San Francisco, California, June 1996.
- [4] K. N. Kutulakos and S. M. Seitz. A theory of shape by space carving. In *Seventh International Conference on Computer Vision (ICCV'99)*, pages 307–314, Corfu, Greece, September 1999.
- [5] M.-S. Lee and G. Medioni. Inferring segmented surface description from stereo data. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition 1998 (CVPR'98)*, Santa Barbara, California, pages 346–52, June, 1998.
- [6] L. McMillan and G. Bishop. Plenoptic modeling: An image-based rendering system. In *Proc. SIGGRAPH'95*, pages 39–46, 1995.
- [7] G. Medioni, M. Lee, and C. Tang. *A Computational Framework for Feature Extraction and Segmentation*. Elseviers Science, Amsterdam, 2000.
- [8] M. Okutomi and T. Kanade. A multiple baseline stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(4):353–363, April 1993.
- [9] L. Robert and R. Deriche. Dense depth map reconstruction: a minimization and regularization approach that preserves discontinuities. In *Fourth European Conference on Computer Vision (ECCV'96)*, 1996.
- [10] S. Roy and I. Cox. A maximum-flow formulation of the n-camera stereo correspondence problem. In *IEEE International Conference 1998 (ICCV'98)*, Bombay, India, pages 492–499, January 1998.
- [11] S. M. Seitz and C. M. Dyer. Photorealistic scene reconstruction by space coloring. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'97)*, pages 1067–1073, San Juan, Puerto Rico, June 1997.
- [12] H. Shum, A. Kalai, and S. Seitz. Omnivergent stereo. In *ICCV99*, pages 22–29, 1999.
- [13] H.-Y. Shum and L.-W. He. Rendering with concentric mosaics. In *Proc. SIGGRAPH'99*, pages 299–306, 1999.
- [14] H.-Y. Shum and R. Szeliski. Stereo reconstruction from multiperspective panoramas. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (ICCV'99)*, pages 14–21, 1999.
- [15] C. Zitnick and T. Kanade. A cooperative algorithm for stereo matching and occlusion detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-22(7):675–684, 2000.



(a)



(b)



(c)



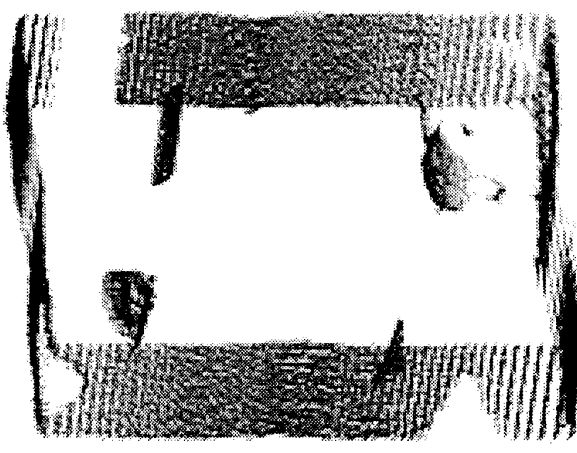
(d)



(e)

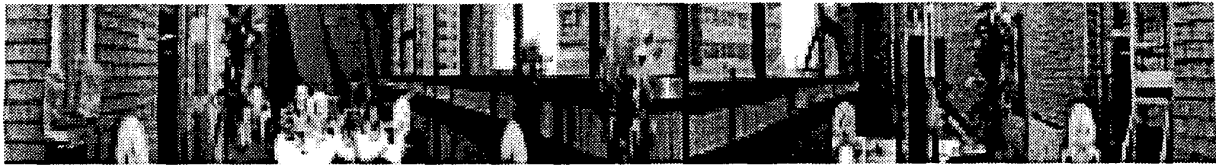


(f)



(g)

Figure 8 *Virtual Room: a synthetic scene. (a) and (b) A multiperspective panorama and its corresponding inverse depth map. (c) a novel view of the panorama (a) (d) depth map of the teapot. (e) a novel view of teapot reprojected with the depth map (d), (f) and (g) the reconstructed room from top-down and top-side views.*



(a)



(b)



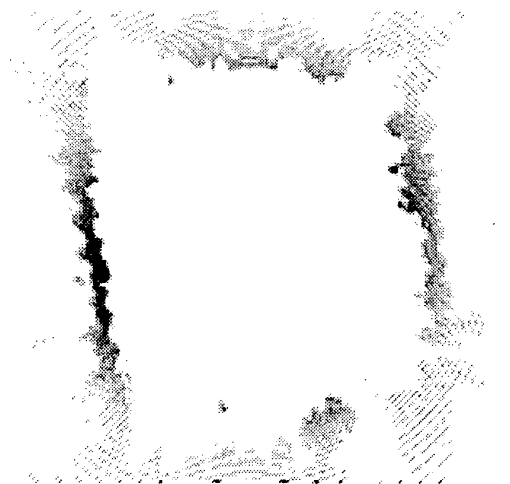
(c)



(d)



(e)



(f)



(g)

Figure 9 *Balcony: a real scene with depth discontinuities, textureless objects, and mirror reflection. (a) and (b) A multiperspective panorama, and its computed inverse depth map. (c) a novel view of the depth map (b), (d) and (e) projected views of part of windows and wall in the middle of (a), (f) and (g) the reconstructed balcony from top-down and top-side views.*