
International Journal of Computer Vision 58(2), 139–163, 2004

Extracting View-Dependent Depth Maps from a Collection of Images

SING BING KANG AND RICHARD SZELISKI
Microsoft Research, One Microsoft Way, Redmond, WA 98052, USA
sbkang@microsoft.com
szeliski@microsoft.com

Received May 16, 2002; Revised October 21, 2002; Accepted December 9, 2002

Abstract. Stereo correspondence algorithms typically produce a single depth map. In addition to the usual problems of occlusions and textureless regions, such algorithms cannot model the variation in scene or object appearance with respect to the viewing position. In this paper, we propose a new representation that overcomes the appearance variation problem associated with an image sequence. Rather than estimating a single depth map, we associate a depth map with *each* input image (or a subset of them). Our representation is motivated by applications such as view interpolation and depth-based segmentation for model-building or layer extraction. We describe two approaches to extract such a representation from a sequence of images.

The first approach, which is more classical, computes the *local* depth map associated with each chosen reference frame independently. The novelty of this approach lies in its combination of shiftable windows, temporal selection, and graph cut optimization. The second approach simultaneously optimizes a set of self-consistent depth maps at multiple key-frames. Since multiple depth maps are estimated simultaneously, visibility can be modeled explicitly and disparity consistency imposed across the different depth maps. Results, which include a difficult specular scene example, show the effectiveness of our approach.

Keywords: stereo correspondence, multi-view stereo, occlusions, view-dependent texture maps, view-dependent depth maps, image-based rendering

1. Introduction

Stereo correspondence, 3D reconstruction, and motion estimation have long been central research problems in computer vision. Early work was motivated by the desire to recover depth maps and coarse shape models for robotics and object recognition applications. More recently, depth maps obtained from stereo and correspondence maps obtained from motion have been combined with texture maps extracted from input images in order to create realistic 3D scenes and environments for virtual reality and virtual studio applications (McMillan and Bishop, 1995; Szeliski and Kang, 1995; Kanade et al., 1996; Blonde et al., 1996), as well as for motion-compensated prediction in video processing applications (Le Gall, 1991; Lee et al., 1997; de Hann and Beller, 1998). Unfortunately, the quality and resolution of most of today's algorithms falls quite short of that demanded by these new applications, where even isolated errors in correspondence become readily visible when composited with synthetic graphical elements.

One of the most common errors made by these algorithms is a mis-estimation of depth or motion near occlusion boundaries. Traditional correspondence algorithms assume that every pixel has a corresponding pixel in all other images. Obviously, in occluded regions, this is not so. Furthermore, if only a single depth or motion map is used, it is impossible to predict the appearance of the scene in regions which are occluded (Fig. 1). Other problems include dealing with untextured or regularly textured regions, and with

Figure 1. Slice through a motion stereo sequence spatio-temporal volume. A standard estimation algorithm only estimates the motion at the center frame (⇒), whereas our multi-view approach produces several additional estimates (→). A layered motion model would use two (or more) layers to describe this motion, whereas a volumetric approach would assign one voxel to each "streak".

viewpoint-dependent effects such as specularities or shading.

One novel approach to tackling these problems is to build a *disparity space* or 3D volumetric model of the scene (Yang et al., 1993; Bobick and Intille, 1999; Collins, 1996; Scharstein and Szeliski, 1998; Seitz and Dyer, 1999; Szeliski and Golland, 1999; Saito and Kanade, 1999). The scene volume is discretized, often in terms of equal increments of disparity. The goal is then to find the voxels which lie on the surfaces of the objects in the scene. The benefits of such an approach include the equal and efficient treatment of a large number of images (Collins, 1996), the possibility of modeling occlusions (Bobick and Intille, 1999), and the detection of mixed pixels at occlusion boundaries (Szeliski and Golland, 1999). Unfortunately, discretizing space volumetrically introduces a large number of degrees of freedom and leads to sampling and aliasing artifacts. To prevent a systematic "fattening" of depth layers near occlusion boundaries, variable window sizes (Kanade and Okutomi, 1994), shiftable windows (Okutomi et al., 2002), or iterative evidence aggregation (Scharstein and Szeliski, 1998) can be used. Sub-pixel disparities can be estimated by finding the the analytic minimum of the local error surface (Tian and Huhns, 1986; Matthies et al., 1989) or using gradient-based techniques (Lucas and Kanade, 1981), but this requires going back to a single depth/motion map representation.

Another active area of research is the detection of parametric motions within image sequences (Wang and Adelson, 1994; Irani et al., 1995; Sawhney and Ayer, 1996; Black and Jepson, 1996; Weiss and Adelson, 1996; Weiss, 1997). Here, the goal is to decompose the images into sub-images, commonly referred to as *layers*, such that the pixels within each layer move with a parametric transformation. For rigid scenes, the layers can be interpreted as planes in 3D being viewed by a moving camera, which results in fewer unknowns (Baker et al., 1998). This representation facilitates reasoning about occlusions, permits the computation of accurate out-of-plane displacements, and enables the modeling of *mixed* or *transparent* pixels. Unfortunately, initializing such an algorithm and determining the appropriate number of layers is not straightforward, and may require sophisticated optimization algorithms such as expectation maximization (EM) (Torr et al., 2001).

Thus, all current correspondence algorithms have their limitations. Single depth or motion maps cannot represent occluded regions not visible in the reference image and usually have problems matching near discontinuities. Volumetric techniques have an excessively large number of degrees of freedom and have limited resolution, which can lead to sampling or aliasing artifacts. Layered motion and stereo algorithms require combinatorial search to determine the correct number of layers and cannot naturally handle true three-dimensional objects (they are better at representing "cardboard cutout" or *shallow* scenes (Sawhney and Hanson, 1991)). Furthermore, none of these approaches can easily model the variation of scene or object appearance with respect to the viewing position.

In this paper, we propose a new representation that overcomes most of these limitations. Rather than estimating a single depth (or motion) map, we associate a depth map with *each* input image (or some subset of them, Fig. 1). We define a *depth image* as a depth map with texture (i.e., color and depth per pixel). Furthermore, we try to ensure consistency between these different depth image estimates using a *depth compatibility* constraint and reason about occlusion relationships by computing pixel *visibilities*.

To generate this representation, we propose two methods. The first method computes a depth map for a single reference image using multiple input images. The depth map is computed using shiftable windows and view selection, followed by global optimization with smoothness. To produce the multiple depth image representation, this method has to be applied multiple times independently, each time for a different reference view. As an alternative, we also propose another method that computes all the depth maps simultaneously with visibility handling.

Our new approach is motivated by several target applications. One application is *view interpolation*,

where we wish to generate novel views from a collection of images with associated depth maps. The use of multiple depth maps and images allows us to model partially occluded regions and to model view-dependent effects (such as specularities) by blending images taken from nearby viewpoints (Debevec et al., 1996). Another potential application is *motion-compensated frame interpolation* (e.g., for video compression, rate conversion, or de-interlacing), where the ability to predict bi-directionally (from both previous and future keyframes) yield better prediction results (Le Gall, 1991). (See Szeliski (1999) for more details on how our multi-view framework applies to general 2D motion estimation and compensation.) A third application is as a low-level representation from which segmentation and layer extraction (or 3D model construction) can take place.

1.1. Previous Work on Stereo

A substantial amount of work has been done on stereo matching; recent surveys can be found in Dhond and Aggarwal (1989), Szeliski and Zabih (1999) and Scharstein and Szeliski (2002). Stereo can generally be described in terms of the following components: matching criterion, aggregation method, andwinner selection (Scharstein and Szeliski, 1998; 2002).

1.1.1. Matching Criterion. The matching criterion is used as a means of measuring the similarity of pixels or regions across different images. A typical error measure is the RGB or intensity difference between images (these differences can be squared, or robust measures can be used (Black and Rangarajan, 1996)). Some methods compute subpixel disparities by computing the analytic minimum of the local error surface (Tian and Huhns, 1986; Matthies et al., 1989) or using gradient-based techniques (Lucas and Kanade, 1981; Shi and Tomasi, 1994; Szeliski and Coughlan, 1997). Birchfield and Tomasi (1998) measure pixel dissimilarity by taking the minimum difference between a pixel in one image and the interpolated intensity function in the other image.

1.1.2. Aggregation Method. The aggregation method refers to the manner in which the error function over the search space is computed or accumulated. The most direct way is to apply search windows of a fixed size over a prescribed disparity space for multiple cameras (Okutomi and Kanade, 1993) or for verged camera configuration (Kang et al., 1995). Other approaches use adaptive windows (Okutomi and Kanade, 1992), shiftable windows (Arnold, 1983; Bobick and Intille, 1999; Tao et al., 2001; Okutomi et al., 2002), or multiple masks (Nakamura et al., 1996). Another set of methods accumulates votes in 3D space, e.g., the space sweep approach (Collins, 1996) and voxel coloring and its variants (Seitz and Dyer, 1999; Szeliski and Golland, 1999; Kutulakos and Seitz, 2000). More sophisticated methods take into account occlusion in the formulation, for example, by erasing pixels once they have been matched (Seitz and Dyer, 1999; Szeliski and Golland, 1999; Kutulakos and Seitz, 2000), by estimating a depth map per image ((Szeliski, 1999) and this paper), or using prior color-based segmentation followed by iterative analysis-by-synthesis (Tao et al., 2001).

1.1.3. Optimization and Winner Selection. Once the initial or aggregated matching costs have been computed, a decision must be made as to the correct disparity assignment for each pixel $d(x, y)$. Local methods do this at each pixel independently, typically by picking the disparity with the minimum aggregated value. Multiresolution approaches have also been used (Bergen et al., 1992; Hanna, 1991; Szeliski and Coughlan, 1997) to guide the winner selection search. Cooperative/competitive algorithms can be used to iteratively decide on the best assignments (Marr and Poggio, 1979; Scharstein and Szeliski, 1998; Zitnick and Kanade, 2000).

Dynamic programming can be used for computing depths associated with edge features (Ohta and Kanade, 1985) or general intensity similarity matches. These approaches can take advantage of one-dimensional ordering constraints along the epipolar line to handle depth discontinuities and unmatched regions (Geiger et al., 1992; Belhumeur, 1996; Bobick and Intille, 1999). However, these techniques are limited to two frames.

Fully global methods attempt to find a disparity surface $d(x, y)$ that minimizes some smoothness or regularity property in addition to producing good matches. Such approaches include surface model fitting (Hoff and Ahuja, 1986), regularization (Poggio et al., 1985; Terzopoulos, 1986; Szeliski and Coughlan, 1997), Markov Random Field optimization with simulated annealing (Geman and Geman, 1984; Marroquin et al., 1987; Barnard, 1989), nonlinear diffusion of support at different disparity hypotheses (Scharstein and Szeliski, 1998), graph cut methods

(Roy and Cox, 1998; Ishikawa and Geiger, 1998; Boykov et al., 2001), and the use of graph cuts in conjunction with planar surface fitting (Birchfield and Tomasi, 1999).

1.1.4. Layers and Regions. Some approaches use layers to handle scenes with possible textureless regions and large amounts of occlusion. One of the first techniques, in the context of image compression, uses affine models (Wang and Adelson, 1994). This was later further developed in various ways: smoothness within layers (Weiss, 1997), "skin and bones" (Ju et al., 1996) and additive models (Szeliski et al., 2000) to handle transparency, and depth reconstruction from multiple images (Baker et al., 1998).

1.1.5. Dealing with Occlusions. While occlusions are usually only explicitly handled in the dynamic programming approaches (where semioccluded regions are labeled explicitly), some techniques have been developed for reasoning about occlusions in a multiple-image setting. These approaches include using multiple matching templates (Nakamura et al., 1996; Okutomi et al., 2002), voxel coloring and its variants (Seitz and Dyer, 1999; Szeliski and Golland, 1999; Kutulakos and Seitz, 2000), estimating a depth map per image ((Szeliski, 1999) and this paper), and graph cuts with the enforcement of unique correspondences (Kolmogorov and Zabih, 2001).

1.2. Overview

In this paper, we present two main techniques for recovering view-dependent depth maps from multiple images. The first is more classical, where only a single depth map is computed locally (Section 2) at one time. In this case, visibility computation can be either implicit or explicit. The second computes multiple depth maps simultaneously, which allows visibility reasoning to be explicit (Section 3). (The extension of this approach to general 2-D motion can be found in Szeliski (1999).)

In the case of extracting a single depth map, we propose two complementary approaches to better deal with occlusions in multi-view stereo matching. The first approach (Section 2.2) uses not only spatially adaptive windows, but also selects a temporal subset of frames to match at each pixel. The second approach (Section 2.3) uses a global (MRF) minimization approach based on graph cuts that explicitly models occluded regions with a special label. It also reasons about occlusions by selectively freezing good matching points and erasing these from the set of pixels that must be matched at depths farther back. In our case, we combine both approaches into a single system. We also demonstrate a more efficient hierarchical graph cut algorithm that works by overloading disparity labels at the first stage and restricting search at the subsequent stage.

In Section 3, the problem of extracting multiple depth maps from multiple images is directly cast as a global optimization over the unknown depth maps. Robust smoothness constraints are used to constrain the space of possible solutions. In Section 4, we show an application of view interpolation using view-dependent depth maps, and describe how we render this representation in a seamless and photorealistic manner. In Section 5, we discuss various performance issues, followed by some concluding remarks.

2. Computing a Single Depth Map from Multiple Images

In this section, we describe our approach to computing a single high-quality depth map from a sequence of images. First, we define the multi-view stereo algorithm. Then, we describe our approach, which consists of using spatially adaptive windows and temporal selection in conjunction with graph cut optimization, as shown in Fig. 2.

2.1. Problem Formulation

In a multi-view stereo problem, we are given a collection of images $\{I_k(x, y), k = 0 \ldots K\}$ and associated camera matrices $\{\mathbf{P}_k, k = 0 \ldots K\}$. $I_0(x, y)$ is the *reference image* for which we wish to compute a *disparity map* $d(x, y)$ such that pixels in $I_0(x, y)$ project to their corresponding locations in the other images when the

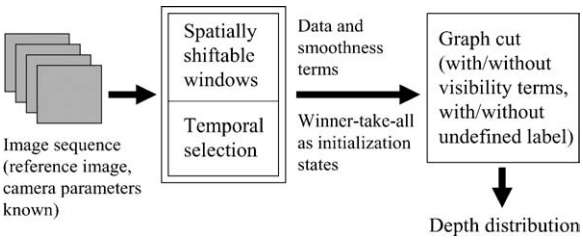

Figure 2. Overview of our approach for computing single depth maps.

correct disparities are selected. Note that the issue of camera calibration is outside the scope of our paper, as there are textbooks that describe the various calibration techniques. The camera parameters associated with the image sequences we use in our paper (i.e., flower garden, symposium, and University of Tsukuba sequences shown in Figs. 6, 10 and 11, respectively) were computed elsewhere and assumed accurate enough for our work.

In the classic forward-facing multi-baseline stereo configuration (Okutomi and Kanade, 1993), the camera matrices are such that disparity (inverse depth) varies linearly with horizontal pixel motion,

$$\hat{I}_k(x, y, d) = I_k(x + b_k d(x, y), y), \tag{1}$$

where $\hat{I}_k(x, y, d)$ is image I_k warped by the disparity map $d(x, y)$. In a more general (plane sweep) multi-view setting (Collins, 1996; Szeliski and Golland, 1999), each disparity corresponds to some plane equation in 3D. Hence, the warping necessary to bring pixels at some disparity d into registration with the reference image can be represented by a homography $H_k(d)$,

$$\hat{I}_k(x, y, d) = H_k(d) \circ I_k(x, y), \tag{2}$$

where the homography can be computed directly from the camera matrices $\mathbf{P}_0$ and $\mathbf{P}_k$ and the value of d (Szeliski and Golland, 1999). In this paper, we assume the latter generalized multi-view configuration, since it allows us to reconstruct depth maps from arbitrary collections of images. (Note that this approach can also be generalized to other sweep surfaces, such as cylinders (Shum and Szeliski, 1999).)

Given the collection images warped at all candidate disparities, we can compute an initial *raw* (unaggregated) matching cost

$$E_{\text{raw}}(x, y, d, k) = \rho(I_0(x, y) - \hat{I}_k(x, y, d)), \tag{3}$$

where $\rho(\cdot)$ is some (potentially) robust measure of the color or intensity difference between the reference and warped image (see, e.g., Scharstein and Szeliski (1998, 2002) for some comparative results with different robust metrics). In this paper, we use a simple squared color difference in our experiments.

The task of stereo reconstruction is then to compute a disparity function $d(x, y)$ such that the raw matching costs are low for all images (or at least the subset where a given pixel is visible), while also producing a "reasonable" (e.g., piecewise smooth) surface. Since the raw matching costs are very noisy, some kind of spatial aggregation or optimization is necessary. The two main approaches used today are local methods, which only look in a neighborhood of a pixel before making a decision, and global optimization methods.

2.2. *Local Techniques*

The simplest aggregation method is the classic sum of sum of squared distances (SSSD) formula, which simply aggregates the raw matching score over all frames

$$E_{\text{SSSD}}(x, y, d) = \sum_{k \neq 0} \sum_{(u,v) \in \mathcal{W}(x,y)} E_{\text{raw}}(u, v, d, k), \tag{4}$$

where $\mathcal{W}(x, y)$ is an $n \times n$ square window centered at (x, y). This can readily be seen as equivalent to a convolution with a 3D box filter. This also suggests a more general formulation involving a general convolution kernel, i.e., the *convolved squared differences*

$$E_{\text{CSD}}(x, y, d) = W(x, y, k) * E_{\text{raw}}(x, y, d, k), \tag{5}$$

where $W(x, y, k)$ is an arbitrary 3D (spatio-temporal) convolution kernel (Scharstein and Szeliski, 1998).

After the aggregated errors have been computed, local techniques choose the disparity with the minimum SSSD error, which measures the degree of photoconsistency at a hypothesized depth. The best match can also be assigned a local confidence computed using the variance (across disparity) of the SSSD error function within the vicinity of the best match (Matthies et al., 1989).

While window-based techniques work well in textured regions and away from depth discontinuities or occlusions, they run into problems in other cases. Figure 3 shows how a symmetric (centered) window may lead to erroneous matching in such regions. Two ways of dealing with this problem are spatially shiftable windows and temporal selection.

2.2.1. Spatially Shiftable Windows. The idea of spatially shiftable windows is an old one that has recently had a resurgence in popularity (Levine et al., 1973; Arnold, 1983; Nakamura et al., 1996; Bobick and Intille, 1999; Tao et al., 2001; Scharstein and Szeliski, 2002; Okutomi et al., 2002). The basic idea is to try several windows that include the pixel we

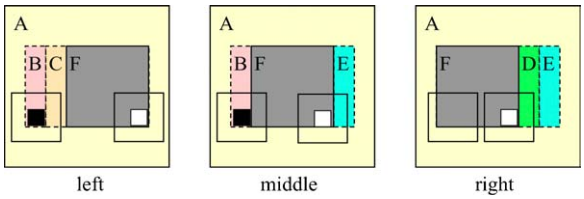

Figure 3. A simple three-image sequence (the middle image is the reference image), with a frontal gray square F, and a stationary background. Regions B, C, D, and E are partially occluded. A regular SSD algorithm will make mistakes when matching pixels in these regions (e.g., the window centered on the black pixel in B), and also in windows straddling depth discontinuities (the window centered on the white pixel in F).

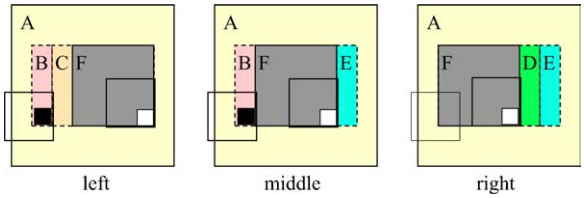

Figure 4. Shiftable windows help mitigate the problems in partially occluded regions and near depth discontinuities. The shifted window centered on the white pixel in F now matches correctly in all frames. The shifted window centered on the black pixel in B now matches correctly in the left image. Temporal selection is required to disable matching this window in the right image.

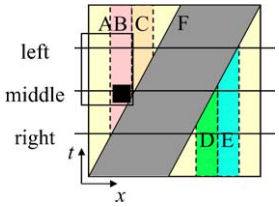

Figure 5. The spatio-temporal diagram (epipolar plane image) corresponding to the previous figure. The three images (middle, left, right) are slices through this EPI volume. The spatially and temporally shifted window around the black pixel is indicated by the rectangle, showing the the right image is not being used in matching.

are trying to match, not just the window centered at that pixel (Fig. 4). (When using square windows, finding the best matching shifted window can be computed by passing a min-filter over the original SSD scores (Scharstein and Szeliski, 2002; Okutomi et al., 2002).) This approach can improve the matching of foreground objects near depth discontinuities (so long as the object is not too thin), and also handle background regions that are being disoccluded rather than occluded (the black pixel in the middle and left image of Fig. 4).

To illustrate the effect of shiftable windows, consider the flower garden sequence shown in Fig. 6. The effect of using spatially shiftable windows over all 11 frames is shown in Fig. 7 for 3×3 and 5×5 window sizes. As can be seen, there are differences, but they are not dramatic. The errors seen can be attributed to ignoring the effects of occlusions and disocclusions.

2.2.2. Temporal Selection. Rather than summing the match errors over all the frames, a better approach would be to pick only the frames where the pixels are visible. Of course, this is not possible in general without resorting to the kind of visibility reasoning present in volumetric (Seitz and Dyer, 1999; Szeliski and Golland, 1999; Kutulakos and Seitz, 2000) or multiple depth map approaches (Section 3), and also in the multiple mask approach of Nakamura et al. (1996) and Okutomi et al. (2002). However, often a semi-occluded region in the reference image will only be occluded in the predecessor or successor frames, i.e., for a camera moving along a continuous path, objects that are occluded along the path in one direction tend to be seen along the reverse direction. (A similar idea has recently been applied to optic flow computation (Sun et al., 2000).) Figure 4 shows this behavior. The black pixel in region B and its surrounding (shifted) square region can be matched in the left image but not the right image. Figure 5 show this same phenomenon in a spatio-temporal slice (epipolar plane image). It can readily be seen that temporal selection is equivalent to shifting the window in time as well as in space.

Temporal selection as a means of handling occlusions and disocclusions can be illustrated by considering selected error profiles depicted in Fig. 9. Points such as A, which can be observed at all viewpoints, work without shiftable windows and temporal selection. Points such as C, which is an occluding point, work better with shiftable windows but do not require temporal selection. Points such as B, however, which is occluded in a fraction of the viewpoints, work best with both shiftable windows and temporal selection.

Rather than just picking the preceding or succeeding frames (one-sided matching), a more general variant would be to pick the best 50% of all images available. (We could pick a different percentage, if desired, but 50% corresponds to the same fraction of frames

Figure 6. 1st, 6th, and 11th image of the eleven image flower garden sequence used in the experiments. The image resolution is 344 × 240.

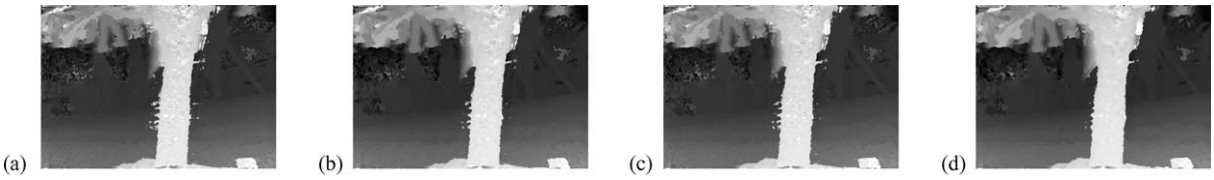

Figure 7. Comparison of results, 128 disparity levels: (a) 3 × 3 non-spatially perturbed window, (b) 5 × 5 non-spatially perturbed window, (c) 3 × 3 spatially perturbed window, (d) 5 × 5 spatially perturbed window. Darker pixels denote distances farther away.

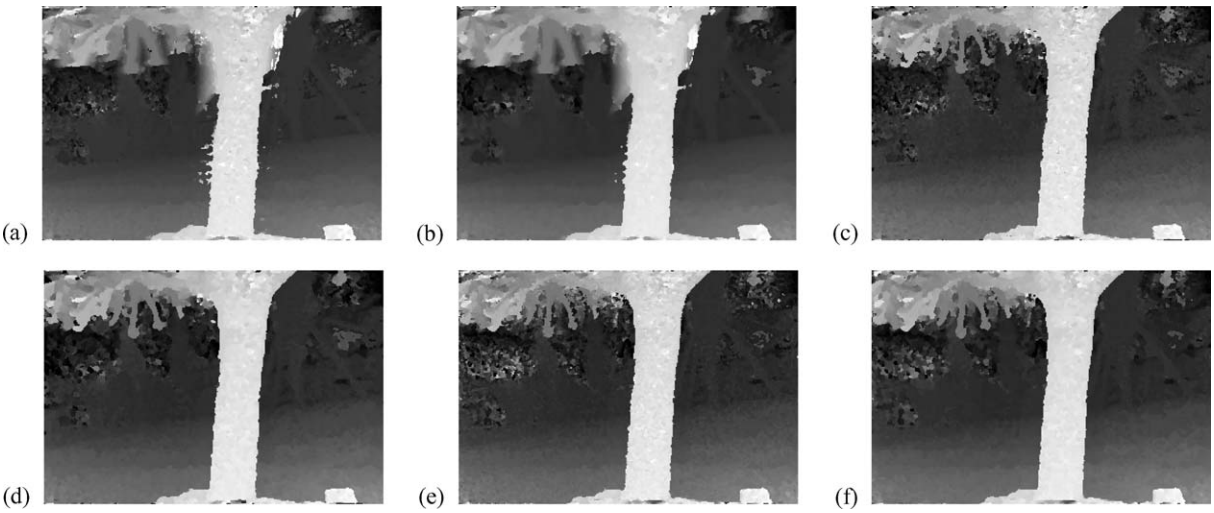

Figure 8. Comparison of results (all using spatially perturbed window, 128 disparity levels): (a) 3 × 3 window, using all frames, (b) 5 × 5 window, using all frames, (c) 3 × 3 window, using best 5 of 10 neighboring frames, (d) 5 × 5 window, using best 5 of 10 neighboring frames, (e) 3 × 3 window, using better half sequence, (f) 5 × 5 window, using better half sequence. Darker pixels denote distances farther away.

as choosing either preceding or succeeding frames.) In this case, we compute the local SSD error for each frame separately, and then sum up the lowest values (this is called *sorting summation* in Satoh and Ohta, 1996). This kind of approach can better deal with objects that are intermittently visible, i.e., a "picket fence" phenomenon.

We have experimented with both variants, and found that they have comparable performance. Figure 8 shows the results on the flower garden sequence. As can be seen, using temporal selection yields a dramatic improvement in results, especially near depth discontinuities (occlusion boundaries) such as the edges of the tree. Similar improvements can also be observed in Fig. 12 for the symposium and Tsukuba sequences.

In addition to experimenting with spatially shiftable windows and temporal selection, we have also developed an adaptive window that does better at filling in

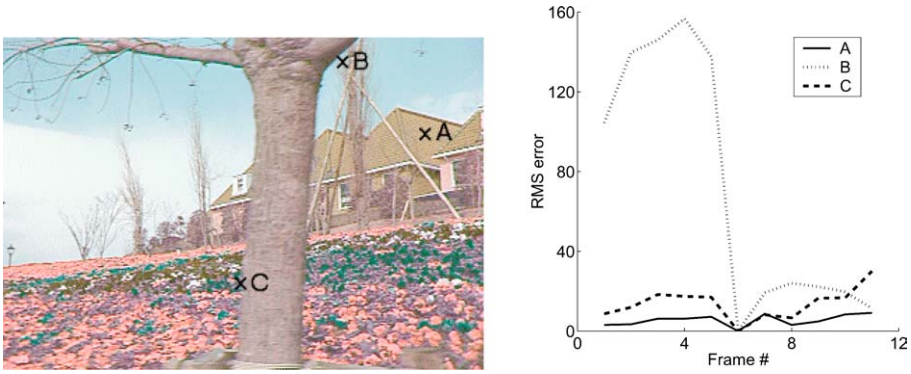

Figure 9. Error profiles for three points in reference image. A: point seen all the time, B: point occluded about half the time, C: occluding point. Left: Reference image, Right: Error graph at respective optimal depths with respect to the frame number (frame #6 is the reference).

Figure 10. Another example: 5-image symposium sequence, courtesy of Dayton Taylor. The 1st, 3rd, and 5th images are shown.

Figure 11. Another example: a 5-image sequence, courtesy of the University of Tsukuba. The 1st, 3rd, and 5th images are shown.

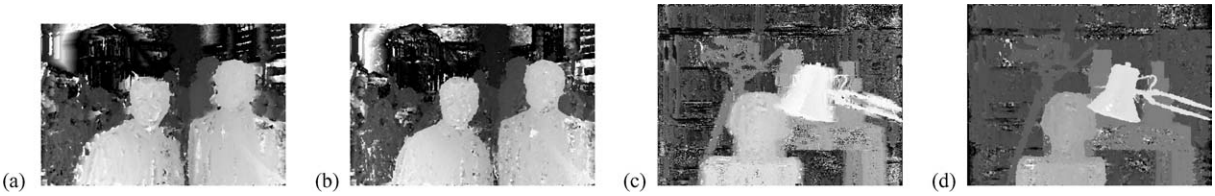

Figure 12. Local (5×5 window-based) results for the symposium and Tsukuba sequences: (a) and (c) non-spatially perturbed (centered) window; (b) and (d) using better half sequence.

textureless regions of the scene (Kang et al., 2001a, 2001b). However, since this approach does not work as well as the global technique described next, we do not describe it in this article.

2.3. Global Techniques

The second general approach to dealing with ambiguity in stereo correspondence is to optimize a global energy function. Typically, such a function consists of two terms,

$$E_{\text{global}}(d(x, y)) = E_{\text{data}} + E_{\text{smooth}}. \tag{6}$$

The value of the disparity field $d(x, y)$ that minimizes this global energy is chosen as the desired solution. Because of the tight connection between this kind of global energy and the log-likelihood of a Bayesian model using Markov Random Fields, these methods are also often called Bayesian or MRF methods (Geman and Geman, 1984; Belhumeur, 1996; Boykov et al., 2001).

The *data* term E_{data} is just a summation of the local (aggregated or unaggregated) matching costs, e.g.,

$$E_{\text{data}} = \sum_{(x,y)} E_{\text{SSSD}}(x, y, d(x, y)). \tag{7}$$

Because a smoothness term is used, spatial aggregation is traditionally not used, i.e., the window $W(x, y)$ in the SSSD term is a single pixel (but see, e.g., Bobick and Intille (1999) for a global method that starts with a window-based cost measure, as well as the results described in this paper).

The smoothness term E_{smooth} measures the piecewise-smoothness in the disparity field,

$$E_{\text{smooth}} = \sum_{(x,y)} \left[s^h_{x,y} \phi(d(x, y) - d(x + 1, y)) + s^v_{x,y} \phi(d(x, y) - d(x, y + 1)) \right]. \tag{8}$$

The smoothness potential $\phi(\cdot)$ can be a simple quadratic, a delta function, a truncated quadratic, or some other robust function of the disparity differences (Black and Rangarajan, 1996; Boykov et al., 2001). The smoothness strengths $s^h_{x,y}$ and $s^v_{x,y}$ can be spatially varying (or even tied to additional variables called line processes (Geman and Geman, 1984; Black and Rangarajan, 1996)). The MRF formulation used by Boykov et al. (2001) makes $s^h_{x,y}$ and $s^v_{x,y}$ monotonic functions of the local intensity gradient, which greatly helps in forcing disparity discontinuities to be coincident with intensity discontinuities.

If the vertical smoothness term is ignored, the global minimization can be decomposed into an independent set of 1D optimizations, for which efficient dynamic programming algorithms exist (Geiger et al., 1992; Belhumeur, 1996; Bobick and Intille, 1999). Many different algorithms have also been developed for minimizing the full 2D global energy function, e.g., Geman and Geman (1984), Poggio et al. (1985), Terzopoulos (1986), Szeliski and Coughlan (1997), Scharstein and Szeliski (1998), Roy and Cox (1998), Ishikawa and Geiger (1998) and Boykov et al. (2001).

In this section, we propose two extensions to the graph cut formulation introduced by Boykov et al. (2001) in order to better handle the partial occlusions that occur in multi-view stereo, namely explicit occluded pixel labeling and visibility computation. We also describe a hierarchical disparity computation method that improves the efficiency of the graph cut algorithm.

2.3.1. Explicit Occluded Pixel Labeling. When using a global optimization framework, pixels that do not have good matches in other images will still be assigned some disparity. Such pixels are often associated with a high local matching cost, and can be detected in a post-processing phase. However, occluded pixels also tend to occur in contiguous regions, so it makes sense to include this information within the smoothness function (i.e., within the MRF formulation).

Our solution to this problem is to include an additional label d_{occl} that indicates pixels that are either outliers or potentially occluded. A fixed penalty E_{occl} is associated with adopting this label, as opposed to the local matching cost associated with some other disparity label. The penalty should be set to be somewhat higher that the largest value observed for correctly matching pixels. The smoothness term for this label is a delta function, i.e., a fixed penalty Φ_{occl} is paid for every non-occluded pixel that borders an occluded one.

The matching cost in (4), (5), and (7) can be rewritten as

$$E'_{\text{SSSD}}(x, y, d) = \begin{cases} E_{\text{SSSD}}(x, y, d) & \text{if } d \neq d_{\text{occl}} \\ E_{\text{occl}} & \text{if } d = d_{\text{occl}} \end{cases}. \tag{9}$$

Figure 13. Effect of using the undefined label for 11-frame flower garden sequence (64 depth levels, no visibility terms, using best frames): (a) Reference image is 1st image, (b) Reference image is 6th image, (c) Reference image is 11th image. The undefined label is black, while the intensities for the rest are bumped up for visual clarity.

Meanwhile, the smoothness potential $\phi()$ in (8) is modified to

$$\phi'(p-q)=\begin{cases}\phi(p-q) & \text{if } p \text{ and } q \neq d_{\text{occl}}\\ \Phi_{\text{occl}} & \text{if } p \text{ or } q = d_{\text{occl}},\ p \neq q.\\ 0 & \text{if } p = q = d_{\text{occl}}\end{cases} \tag{10}$$

The occlusion penalty term E_{occl} was set to 18, and the fixed smoothness penalty Φ_{occl} set to 10 in our experiments.

Examples of using such a label can be seen in Fig. 13. The black regions are classified as the occluded regions. Unfortunately, this approach sometimes fails to correctly label pixels in occluded textureless regions, since these pixels may still match correctly at the frontal depth. In addition, the optimal occluded label penalty setting depends on the amount of contrast in a given scene.

2.3.2. Visibility Reasoning. An idea that has proven to be effective in dealing with occlusions in volumetric (Seitz and Dyer, 1999; Szeliski and Golland, 1999; Kutulakos and Seitz, 2000) or multiple depth map (Section 3) (Szeliski, 1999) approaches is that of visibility reasoning. Once a pixel has been matched at one disparity level, it is possible to "erase" that pixel from consideration when considering possible matches at disparities farther away from the camera. This is the most principled way to reason about visibility and partial occlusions in multi-view stereo. However, since the algorithms cited above make independent decisions between pixels or frames, their results may not be optimal.

To incorporate visibility into the global optimization framework, we compute a visibility function similar to the one presented in Szeliski and Golland (1999). The visibility function $v(x, y, d, k)$ can be computed as a function of the disparity assignments at layers closer than d. Let $o(x, y, d') = \delta(d', d(x, y))$ be the *opacity* (or indicator) function, i.e., a binary image of those pixels assigned to level d'. The *shadow* $s(x, y, d', d, k)$ that this opacity casts relative to camera k onto another level d can be derived from the homographies that map between disparities d' and d

$$s(x, y, d', d, k) = \left(H_k(d)H_k^{-1}(d')\right) \circ o(x, y, d'). \tag{11}$$

(We can, for instance, use bilinear resampling to get "soft" shadows, indicative of partial visibility.) The visibility of a pixel (x, y) at disparity d relative to camera k can be computed as

$$v(x, y, d, k) = \prod_{d'<d} (1 - s(x, y, d', d, k)). \tag{12}$$

Finally, the raw matching cost (3) can then be replaced by

$$E_{\text{vis}}(x, y, d, k) = v(x, y, d, k)\rho(I_0(x, y) - \hat{I}_k(x, y, d)). \tag{13}$$

The above visibility-modulated matching score thus provides a principled way to compute the goodness of a particular disparity map $d(x, y)$ while explicitly taking into account occlusions and partial visibility. For any given labeling $d(x, y)$, we can compute the opacities, shadows, and visibilities, and then sum up the visibility-modulated matching scores (13) to obtain the final global energy (6). Unfortunately, it is not obvious how to minimize such a complicated energy function.

One possibility would be to start with all pixels visible, and to then run the usual graph-cut algorithm. From the initial $d(x, y)$ solution, we could recompute

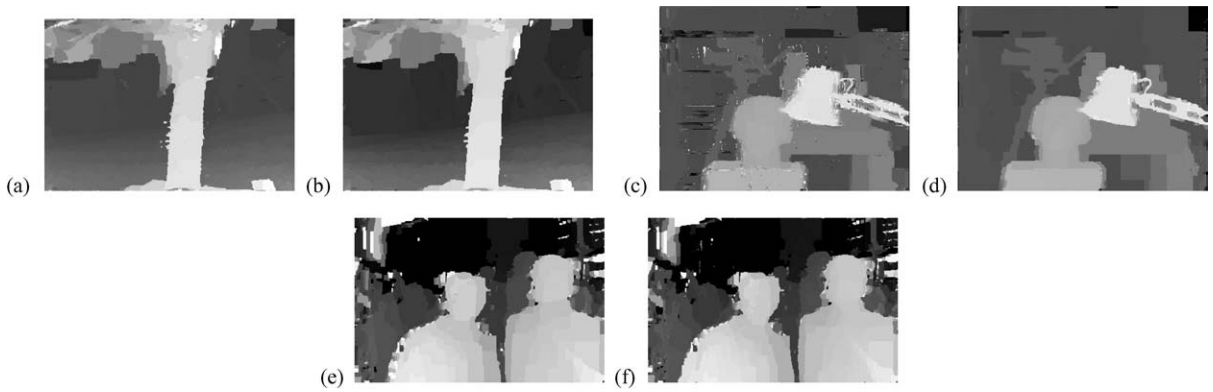

Figure 14. Effect of applying incremental visibility-based graph cuts: (a, c, e) results using all frames; (b, d, f) results using all frames and visibility. Note that we do not show the results involving the best half sequence (or best frames) data terms because there is no significant improvement in adding visibility reasoning.

visibilities, and then re-optimize the modified energy function. Unfortunately, this process may not converge, since the energy function is being modified from iteration to iteration, and the visibilities assumed for one iteration may be undone by a re-assignment of labels in that iteration.

The alternative we have come up with (inspired by Chou's Highest Confidence First algorithm (Chou and Brown, 1990)) is to progressively commit the best-matching depths (i.e., freeze their labels) and apply graph cut on the remaining pixels. This approach is related to the voxel coloring work (Seitz and Dyer, 1999), where voxels are tagged from front to back. However, in our approach, the best 15% of the pixels (based on the current visibility-modulated matching score (13)) whose depths have been computed by the graph cut are frozen. The visibility function and matching costs are then recomputed, which may affect costs at more distal voxels. Within each iteration, graph-cut labeling effectively takes into account neighboring pixels' preferences and tries to make the disparity function piecewise-smooth, whereas the voxel coloring approach only uses per-pixel photo-consistency. After 12 iterations, the remaining uncommitted pixels are frozen at their best value.

Figure 14 shows the results of adding visibility reasoning to the graph cut algorithm when starting with *all* frames as the data cost (no temporal selection). The improvement is significant for the Tsukuba sequence. Note that we do not show the results involving the best half sequence (or best frames) data terms because there is no discernable improvement in adding visibility reasoning. This suggests that shiftable windows coupled with temporal selection handle the occlusion problem well.

2.3.3. Hierarchical Disparity Computation. While the graph cut algorithm and its variants can produce very good results, the problem of computing the exact minimum via graph cuts is NP-hard (Veksler, 1999). Furthermore, the complexity of the approximating α-β swap algorithm is quadratic in the number of labels. As a result, we need to keep the number of labels (in the form of disparities) to a minimum.

To reduce the severity of this problem, we first solve the graph cut using a smaller number of labels, and then solve for an assignment at the desired final resolution level. In the first phase, each *overloaded* label represents a range of disparity values, as indicated in Fig. 15. The cost function associated with a label is

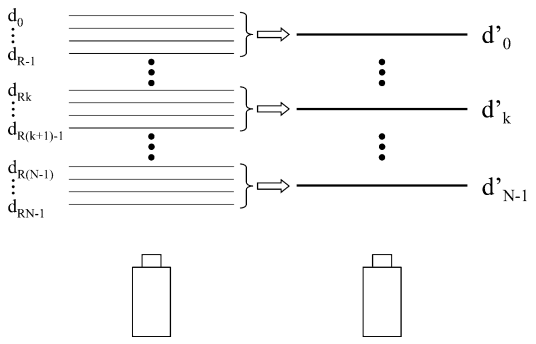

Figure 15. Overloaded disparity space. There are N disparity levels in the lower resolution (overloaded) space (right) and RN disparity levels in the original higher resolution space (left).

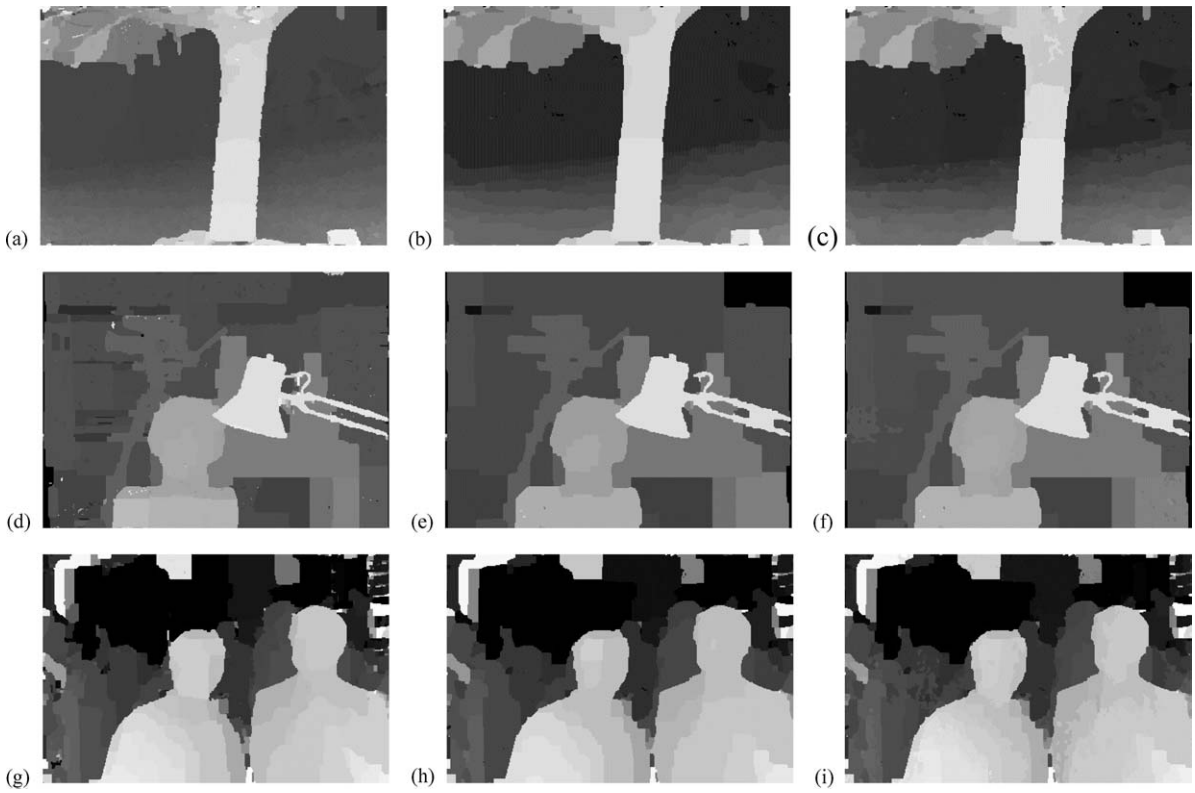

Figure 16. Results of using hierarchical graph cut: (a, d, g) Results using full depth resolution graph cut (128 levels), (b, e, h) Intermediate coarse results using hierarchical graph cut (32 levels), (c, f, i) Final results using hierarchical graph cut (128 levels). The results in the last two columns look very similar, but the third column has better depth resolution (more gray levels).

the minimum of the costs associated with its range of disparity values, i.e.,

$$C^*(u, v, d'_k) = \min_{d_i \in d'_k} C(u, v, d_i). \tag{14}$$

Label swapping at this stage uses the full range of the coarse levels. In the subsequent refinement stage, we use higher resolution disparity levels in the graph-cut algorithm. However, swapping between disparity labels is now only permitted within its previous range and its immediate neighbors.

Results using the proposed hierarchical graph cut can be seen in Fig. 16. In these sets of experiments, we represent a coarse disparity level with four original disparity levels (reducing the number of levels from 128 to 32 initially). The results obtained are comparable to those with the full resolution graph cut. While there appears to be some degradation of quality in the recovered depth maps, especially for the University of Tsukuba sequence, the visual reconstruction remains very good (see Figs. 17 and 18). The degradation is due to the early commitment of depth, as can be seen in Figs. 16(b, e, h).

The timings for the winner-take-all and graph cut portions of our stereo algorithm can seen in Table 1. The resolution of the 11-frame flower garden sequence is 344 × 240, while that for the 5-frame University of Tsukuba sequence is 384 × 288, and that for the 5-frame symposium sequence is 384 × 256. The results were produced using a PC with a 1 GHz processor, with 128

Table 1. Timings for the three sequences (all in "minutes:seconds"). Note that in each case, the graph cut algorithm is iterated four times for convergence.

Operation	Flower garden	Tsukuba	Symposium
Winner-take-all	2:19	2:20	2:27
Graph cut (full)	48:40	56:50	59:32
Graph cut (hierarchical)	11:03	14:05	13:24

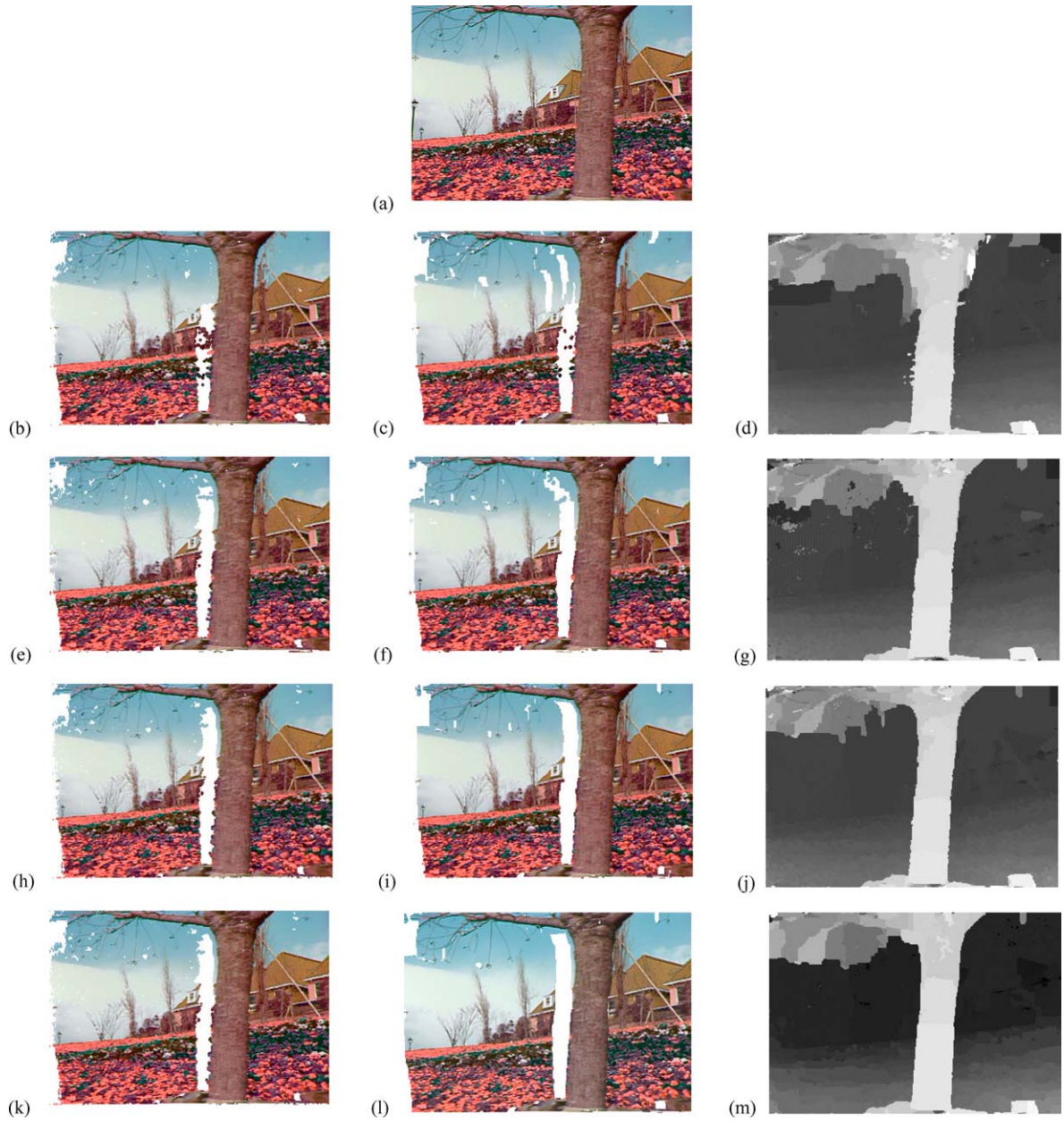

Figure 17. Examples of view reconstruction using results of graph cut (3×3 window used): (a) 1st frame of original 11-frame sequence (5th frame is the reference), (b, c) Reconstructed view using all frames (before and after graph cut), (e, f) Reconstructed view using best frames (before and after graph cut), (h, i) Reconstructed view using best half sequence (before and after graph cut), (k) is the same as (h), (l) Reconstructed view after hierarchical graph cut, (d, g, j, m) are the respective depth maps after graph cut.

disparity levels and maximum neighborhood span of 5 frames. For the hierarchical graph cut (with $N_o = 4$), each overloaded label represents four original labels. The timings for each sequence are reduced by a factor ranging from 4.0 to 4.4.

2.4. *Discussion of Results*

Figures 17 and 18 show view reconstruction results on the flower garden and Tsukuba sequences. (View reconstruction results for the symposium sequence

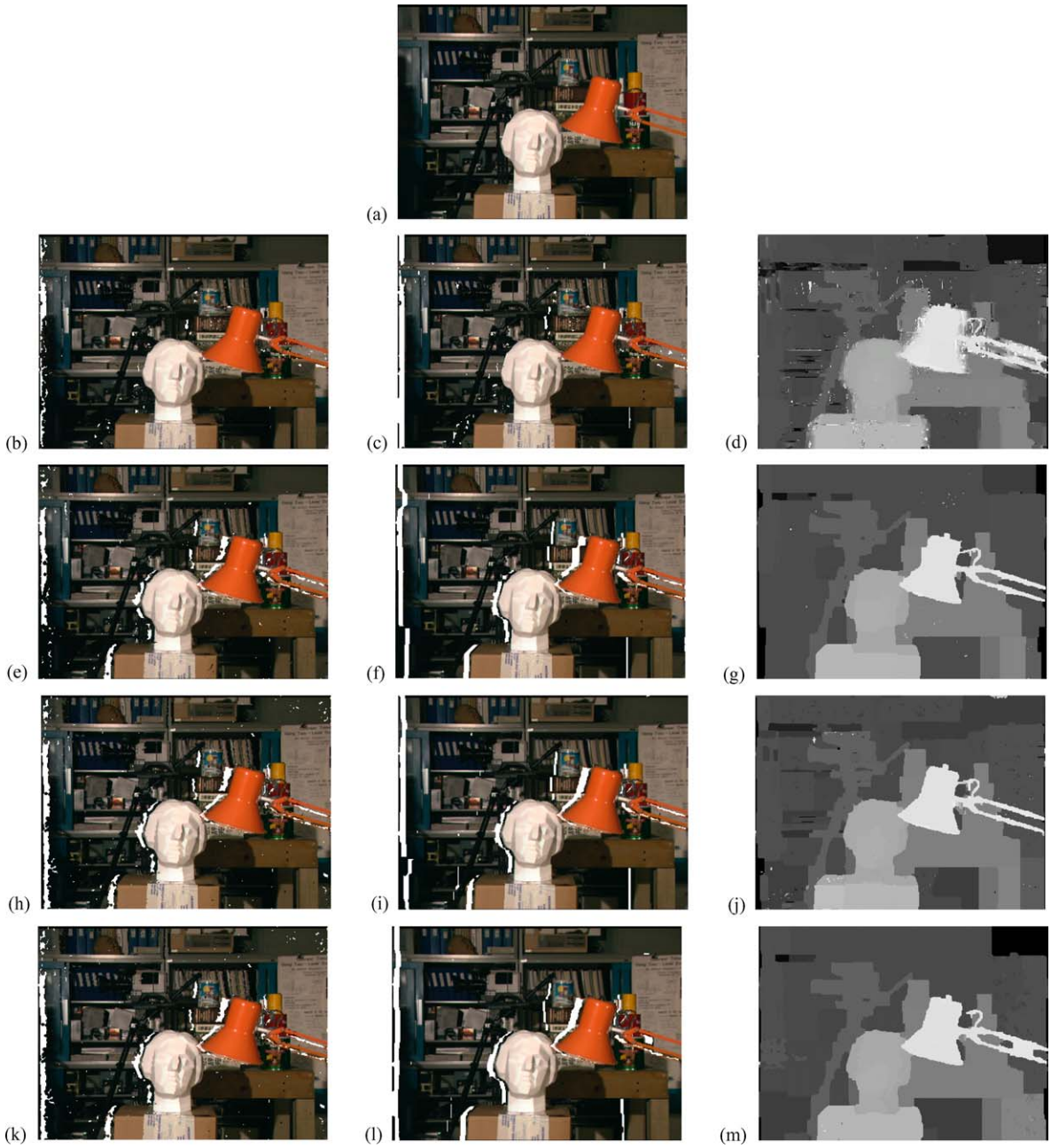

Figure 18. Examples of view reconstruction using results of graph cut (3×3 window used): (a) 5th frame of original 5-frame sequence (3rd frame is the reference), (b, c) Reconstructed view using all frames (before and after graph cut), (e, f) Reconstructed view using best frames (before and after graph cut), (h, i) Reconstructed view using best half sequence (before and after graph cut), (k) is the same as (h), (l) Reconstructed view after hierarchical graph cut, (d, g, j, m) are the respective depth maps after graph cut.

can be found in Kang et al. (2001b).) Note that the white cracks observed in the reconstructed views are caused by transferring adjacent pixels with depth discontinuities.

While our algorithm is targeted specifically towards multi-view stereo (with sequences of more than two frames), it also produces reasonable results in two frame situations. The temporal selection component

is inoperative in this case. Our results compare favorably with Boykov et al. (2001) (see Kang et al. (2001b)).

3. Computing Multiple Depth Maps Simultaneously

In the previous section, a single depth map was computed using a combination of shiftable windows, temporal selection, and optimization using graph cut. This is the traditional representation for computing stereo. In this section, we describe a new technique that computes multiple depth maps simultaneously from a collection of images. We refer to this as the multi-view stereo reconstruction framework.

Our multi-view framework is motivated by several requirements. These include the ability to accurately predict the appearance of novel views or in-between images and the ability to extract higher-level representations such as layered models or surface-based models. Therefore, our goal is to estimate a collection of depth maps associated with several images, such that other images in the input collection can be predicted based on these estimates (Fig. 1).

As before (see Section 2.1), we are given a collection of images $\{I_k(x, y), k = 0 \ldots K\}$. First, we select some set S of *keyframes* (or *key-views*) for which we will estimate depth estimates $\{d_l,\ l \in S\}$. The decision as to which images are keyframes is problem-dependent, much like the selection of *I* and *P* frames in video compression (Le Gall, 1991). For 3D view interpolation, one possible choice could be a collection of *characteristic views*. If view-dependent effects (such as specularities) are present, then more key views might be required (see Section 4).

Since we now have a collection of reference frames, we need to extend the definition of the warped images given in (1)–(2) to

$$\hat{I}_k^l(x, y, d) = I_k(x + (b_k - b_l)d(x, y), y) \qquad (15)$$

or

$$\hat{I}_k^l(x, y, d) = H_k^l(d) \circ I_k(x, y), \qquad (16)$$

where the homography H_k^l can be computed from the camera matrices $\mathbf{P}_l$ and $\mathbf{P}_k$ and the value of d (Szeliski and Golland, 1999).

The raw matching score given in (3) now depends on l as well,

$$E_{\text{raw}}(x, y, d, k, l) = \rho\big(I_l(x, y) - \hat{I}_k^l(x, y, d)\big), \quad (17)$$

as does the visibility-modulated matching score (13),

$$E_{\text{vis}}(x, y, d, k, l) = v(x, y, d, k, l)\rho\big(I_l(x, y) - \hat{I}_k^l(x, y, d)\big). \qquad (18)$$

(we will define $v(x, y, d, k, l)$ below).

The new global data term (replacing (7)) is therefore the summation over all keyframes and all pixels of the visibility-modulated matching scores corresponding to the current disparity estimates $d_l(x, y)$, i.e.,

$$E_{\text{data}} = \sum_{l \in S} \sum_{k \in \mathcal{N}(l)} w_{kl} \sum_{(x,y)} E_{\text{vis}}(x, y, d, k, l). \quad (19)$$

Comparing (19) with (7) shows that we now optimize over multiple depth maps simultaneously. (See Kolmogorov and Zabih (2002) for more recent work based on this framework.) Images $I_k, k \in \mathcal{N}(k)$ are *neighboring frames* (or views), for which we require that corresponding pixel colors agree. The constants w_{kl} are the *inter-frame weights* that control how much neighboring frame k will contribute to the estimate of d_l. (Note that we could set $w_{kl} = 0$ for $k \notin \mathcal{N}(l)$ and replace $\sum_{k \in \mathcal{N}(l)}$ with $\sum_k$. Also, a more geometrically plausible weighting would reflect the degree of similarity between the viewing rays on a pixel-by-pixel basis (Gortler et al., 1996; Buehler et al., 2001), but we have not implemented this idea.)

The complete cost function we use for multi-frame matching consists of three terms (compare with (6) in Section 2.3),

$$E_{\text{global}} = E_{\text{data}} + E_{\text{smooth}} + E_{\text{compat}} \qquad (20)$$

where E_{data} measures the *brightness compatibility*, i.e., the raw differences in corresponding pixel intensities or colors, E_{smooth} measures the *disparity smoothness*, and E_{compat} measures the temporal *disparity compatibility*, i.e., the agreement between disparity estimates in different frames. Note that E_{compat} is the important additional term in the cost function that is not present in the case of the single reference depth map. Since E_{data} has been described above in (19) and E_{smooth} is

an analogous multiple depth map extension of (8) that involves summing over all depth maps l, we only describe E_{compat} here.

The controlled disparity compatibility constraint is given by

$$E_{\text{compat}} = \sum_{l \in S} \sum_{k \in S} \tilde{w}_{kl} \sum_{(x,y)} v(x, y, d, k, l) \rho_C \big(d_l(x, y) - \hat{d}_k^l(x, y, d)\big). \tag{21}$$

This constraint enforces *mutual consistency* between disparity estimates at different neighboring keyframes, i.e., frames for which $\tilde{w}_{kl}$ is non-zero.

The warped disparity field $\hat{d}_k^l(x, y, d)$ can be computed in a manner analogous to the warped intensity image (16). However, if the plane sweeps used to define the different disparity fields d_l are not coincident, we may have to re-scale the d_k values by a projective transformation (Shade et al., 1998). For a scene with objects far enough away or for cameras arranged in a plane perpendicular to their optical axes (as in our current experiments), the inverse depths to corresponding pixels are close enough that this is not a problem.

The definition of the visibility function $v(x,y,d,k,l)$ is even simpler than in the case of a single depth map ((12) in Section 2.3.2). Once we have computed the warped (resampled) depth map $\hat{d}_k^l(x, y, d)$ as described above, we can simply compare the two depth maps and set

$$v(x, y, d, k, l) = \big(\big(d_l(x, y) - \hat{d}_k^l(x, y, d)\big) \leq \delta\big), \tag{22}$$

where δ is a threshold to account for errors in estimation and warping. This is because if (x, y) is visible in image k, the values of $d_l(x, y)$ and $\hat{d}_k^l(x, y, d)$ should be the same. If (x, y) is occluded, then $d_l(x, y) < \hat{d}_k^l(x, y, d)$ (assuming $d = 0$ at infinity and positive elsewhere in front of the camera). We set $v(x, y, d, k, l) = 0$ whenever the pixel corresponding to (x, y) is outside the boundaries of I_k. (We can think of the camera body as being the occluder, in this case.)

3.1. Estimation Algorithm

With our cost framework in place, we now describe our estimation algorithm, which combines ideas from hierarchical estimation (Quam, 1984; Bergen et al., 1992), correlation-style search (Matthies et al., 1989; Kanade and Okutomi, 1994), and sub-pixel motion/disparity estimation (Lucas and Kanade, 1981; Matthies et al., 1989).

Our algorithm operates in two phases. During an initialization phase, we estimate the depth maps independently for each keyframe. Since we do not yet have any good depth estimates for other frames, the disparity compatibility term E_{compat} is ignored, and no visibilities are computed (i.e., $v(x, y, d, k, l) = 1$). In the second phase, we enforce disparity compatibility and compute visibilities based on the current collection of disparity estimates $\{d_l\}$. A more detailed description of our algorithm can be found in Szeliski (1999).

To compute the initial set of depth maps, we use a hierarchical (coarse-to-fine) algorithm similar to Bergen et al. (1992). (We could just as well use the graph cut algorithm described in Section 2, but our implementation for multi-view stereo matching was done prior to our work on temporal selection and graph cuts.) Hierarchical matching results in an efficient algorithm, since fewer pixels are examined at coarser levels. It can also potentially results in better quality estimates, since a wider range of depths can be searched and a better local minimum can be found.

Within each level, we use correlation-style search, i.e., we evaluate several disparity hypotheses at once, and then locally pick the one that results in the lowest local cost function. At the very first iteration, we disable smoothness constraints, and then enable them and reduce the amount of spatial aggregation for later iterations. To obtain depth estimates with better accuracy, we compute a *fractional* depth estimate by fitting a quadratic cost function to the cost function values around the minimum and analytically computing its minimum (Matthies et al., 1989). We ignore the results of fractional disparity fitting if the distance of the analytic minimum from the discrete minimum is more than a half-step.

Once we have computed an initial set of disparity estimates $\{d_l\}$, we can now compute visibilities $v(x, y, d, k, l) = 1$ and add in the disparity compatibility constraint E_{compat}. The multi-view estimation algorithm can be repeated several times, at each iteration obtaining better estimates of depth and visibility. We currently perform this *sweeping* through the keyframes, instead of performing a single global estimation, because it is easier to implement and

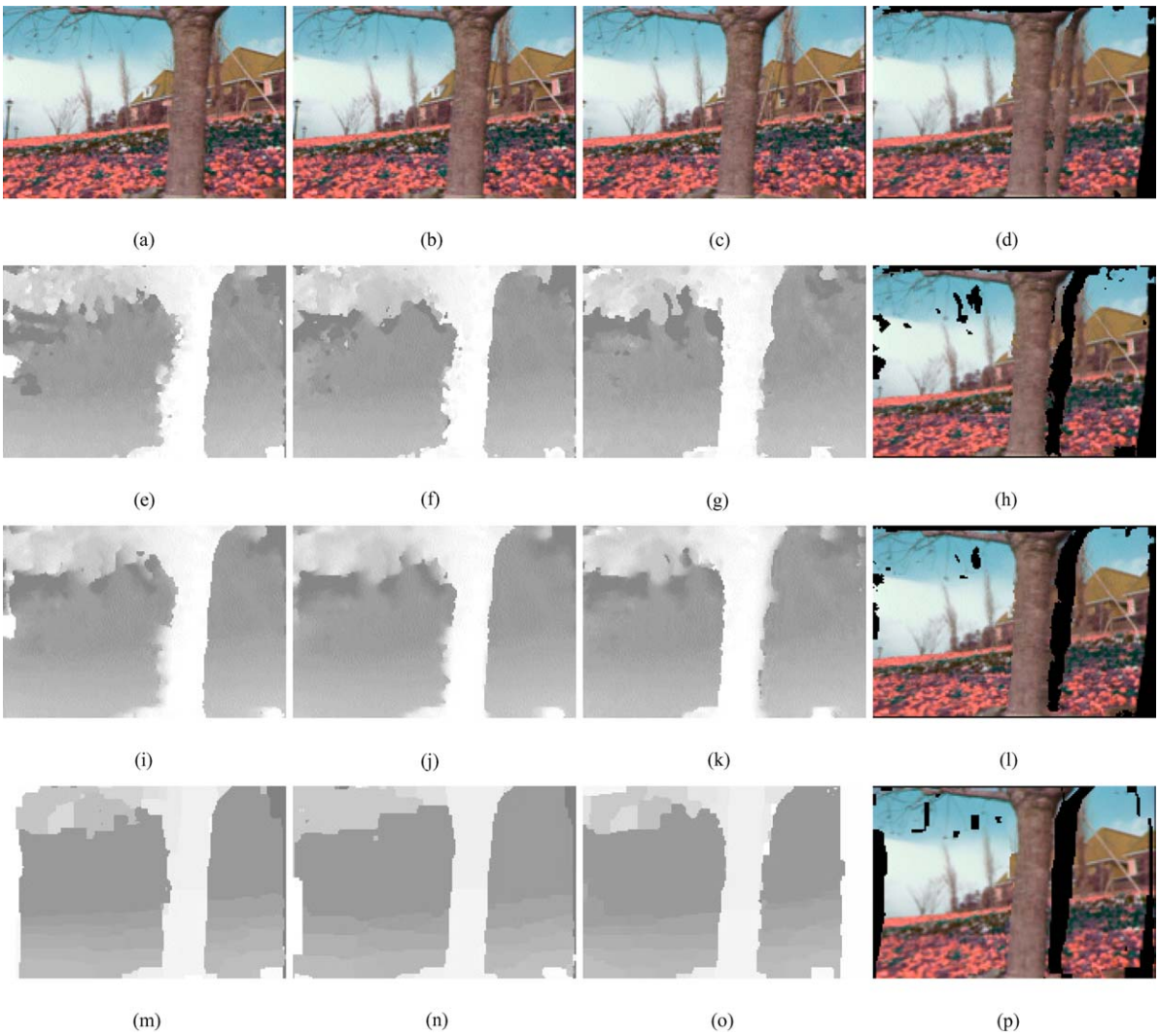

Figure 19. Results on the flower garden sequence: (a–c) first, second, and fourth (last) frame; (e–g) initial depth estimates; (i–k) refined (multi-view) depth estimates. Warped (resampled) images: (d) after initial estimate; (h) with visibility computation; (l) with refined estimates; (m–p) results obtained using the algorithm with a single reference view (Section 2).

requires less memory. An alternative would be to perform a full spatio-temporal regularization.

3.2. *Experiments*

We have applied our multi-view matching algorithm to a number of image sequences. Figures 19 and 20 show some representative results and illustrate some of the features of our algorithm.

In both sets of figures, images (a–c) show the first, middle, and last image in the sequence (we used the first 4 even images from the flower garden sequence and 5 out of 40 images from the symposium sequence). The depth maps estimated by the initial, independent analysis algorithm are shown in images (e–g). The final results of applying our multi-view estimation algorithm with smoothness, disparity compatibility, and visibility estimation are shown in images (i–k). Notice the improved quality of the estimates obtained with the multi-view estimation algorithm, especially in regions that are partially occluded. For example, in Fig. 19, since the tree is moving from right to left, the occluded region is to the left of the tree in the first image, and

Figure 20. Results on the symposium sequence: (a–c) first, third, and fifth (last) frame; (e–g) initial depth estimates; (i–k) refined (multi-view) depth estimates. Warped (resampled) images: (d) after initial estimate; (h) with visibility computation; (l) with refined estimates; (m–p) results obtained using the algorithm with a single reference view (Section 2).

to the right of the tree in the last one. Notice how the opposite edge of the trunk (where disocclusions are occurring) looks "crisp".

Image (d) in both figures shows the results of warping one image based on the depth computed in another image. Displaying these warped images as the algorithm progresses is a very useful way to debug the algorithm and to assess the quality of the depth estimates. Without visibility computation, image (d) shows how the pixels in occluded regions draw their colors somewhere from the foreground regions (e.g., the tree trunk in Fig. 19 and the people's heads in Fig. 20).

Images (h) and (l) show the warped images with invisible pixels flagged as black (the images were generated after the initial and final estimation stages, and hence correspond to the depth maps shown to their left). Notice how the algorithm correctly labels most of the occluded pixels, especially after the final estimation. Notice, also, that some regions without texture such as the sky sometimes erroneously indicate occlusion. Using more smoothing or adding a check that occluder and occludees have different colors could be used to eliminate this problem (which is actually harmless, if we are using our matcher for view interpolation or motion prediction applications).

Figures 19(m–p) and 20(m–p) also show the results of applying the independent disparity estimation algorithm described in Section 2. Compared to the results in the third row, we see that these results have crisper boundaries on *both* sides of depth discontinuities, e.g., on both sides of the tree in Fig. 19. This is because temporal selection is being used. They also sometimes have fewer errors (less variability) in the textureless regions, because the graph cut minimization is being used. On the other hand, because disparity compatibility is not being enforced, some of the results in the textureless regions are worse (e.g., the blue sky in the upper left corner of Fig. 20(p). The graph cut results also exhibit more blockiness, e.g., less smooth variation in slanted regions, such as the flower beds in Fig. 19.

The simultaneous depth map computation method is related to the method in Section 2 in that both can be used to produce our proposed representation of view-dependent depth images. These methods were developed independently. As described earlier, the current method optimizes by sweeping through the views, and at each view, correlation-style search is performed (with visibility handling), with the lowest local cost function chosen. While it would be possible to apply graph cut instead, this would require a huge memory footprint and would be much more computationally expensive.

4. Rendering View-Dependent Depth Images

One of the primary applications for view dependent depth maps is photorealistic view interpolation using multiple *depth images* (i.e., textured depth maps). We believe that this representation is a good choice to represent many kinds of scenes, including scenes with non-diffuse effects such as reflections and specularities (Fig. 23) (Swaminathan et al., 2002; Tsin et al., 2003).

To render a novel view from a collection of depth images, we treat each depth image as a single *sprite* and render it the same as described in Shade et al. (1998). This is accomplished using a two-step rendering algorithm based on the forward transfer equation

$$\begin{bmatrix} w_2 x_2 \\ w_2 y_2 \\ w_2 \end{bmatrix} = \mathbf{H}_{1,2} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} + d_1 \mathbf{e}_{1,2}, \tag{23}$$

where $\mathbf{H}_{1,2}$ is the homography that maps the source viewpoint to the target viewpoint, and $\mathbf{e}_{1,2}$ is the *epipole*. The rendering algorithm consists of the following two steps. First, we forward map the depth map d_1 using (23) to generate the depth map d_2 associated with the new view. (We do a small amount of morphological hole filling to close small gaps arising from the forward splatting process.) Then, we inverse map (with interpolation) the color image, using the homography and per-pixel parallax to compute the source pixel address in the sprite, i.e., using (23) with 1 and 2 interchanged (Shade et al., 1998).

In rendering multiple depth images, we render each depth image into an image accumulator that keeps track of the weighted sum of the color and sum of the weights. In the experiments shown in this paper, we chose the nearest two reference depth images, but we have also experimented with using the three nearest. The global weights applied to each reference depth image are computed based on degree of visual overlap, described as follows.

Suppose we wish to compute the visual overlap between depth images D_i (with viewpoint V_i) and virtual viewpoint V_v. Let N_i be the number of points of D_i seen from V_v and ξ_i be the proportion of the image occupied at V_v by the reprojected D_i. The visual overlap is then defined to be

$$\omega_i = \frac{\xi_i N_i}{N_I}, \tag{24}$$

where N_I is the number of pixels per depth image (Fig. 21). To simplify and speed up the weight computation, we use a fronto-parallel plane with average depth instead of the full depth distribution.

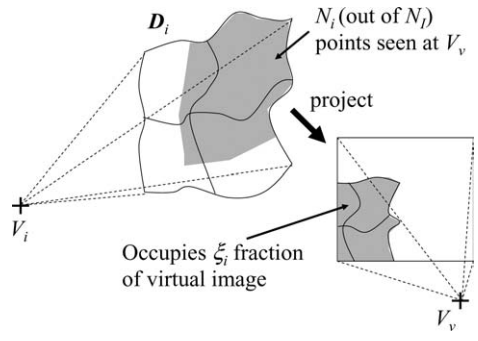

Figure 21. Illustration of visual overlap. $\mathbf{D}_i$ is the ith depth image, with V_i the center of the ith camera. V_v is the center of the virtual camera.

Figure 22. Comparison between actual and interpolated frames for the flower garden, Tsukuba, and symposium sequences. (a, c, e) Actual frame, (b, d, f) Interpolated frame. Here, only two neighboring depth images are used in the interpolation. The insets in (c) and (d) highlight the gap phenomena caused by misestimating depths of textureless regions while the insets in (e) and (f) show the result of rendering actifacts.

The color at pixel (p, q) for the virtual view, $\mathbf{c}_{p,q,v}$, is computed using

$$\mathbf{c}_{p,q,v} = \frac{\sum_{i \in \mathcal{N}(v)} \omega_i \eta_{p,q,i} \mathbf{c}'_{p,q,i}}{\sum_{i \in \mathcal{N}(v)} \omega_i \eta_{p,q,i}}, \tag{25}$$

where $\mathbf{c}'_{p,q,i}$ is the color at (p, q) in the virtual view after warping D_i. $\eta_{p,q,i}$ is 1 if (p, q) is occupied; otherwise it is set to 0. $\mathbf{c}_{p,q,v}$ is computed using the nearest depth images $\mathcal{N}(v)$ (nearest two in our experiments), and is set to the "empty" color (currently black) if the denominator in (25) is zero (i.e., no part of any depth image is mapped to (p, q)).

Some rendering results are shown in Fig. 22. (These rendering results use outputs from our method described in Section 2.) Figure 22(a, c, e) are the actual 3rd frames of their respective sequences, and (b, d, f) were interpolated using depth images at the 2nd and 4th frames. The maximum neighborhood used to compute each depth image is 5 frames, i.e., to compute the depth map at the kth frame, frames $(k-5) \ldots (k+5)$ are used (subject to indexing limits). The interpolated views look very similar to the actual views, except that they look less sharp and contain a few gaps. These gaps are due to misestimation of depth in textureless areas that are locally photoconsistent. One way of dealing with these gaps is to perform automatic hole filling, which was not implemented in our version of rendering. We also observe some "ringing" at the borders of depth discontinuities (Fig. 22(f) closeup), but this is mostly due to the rendering artifacts caused

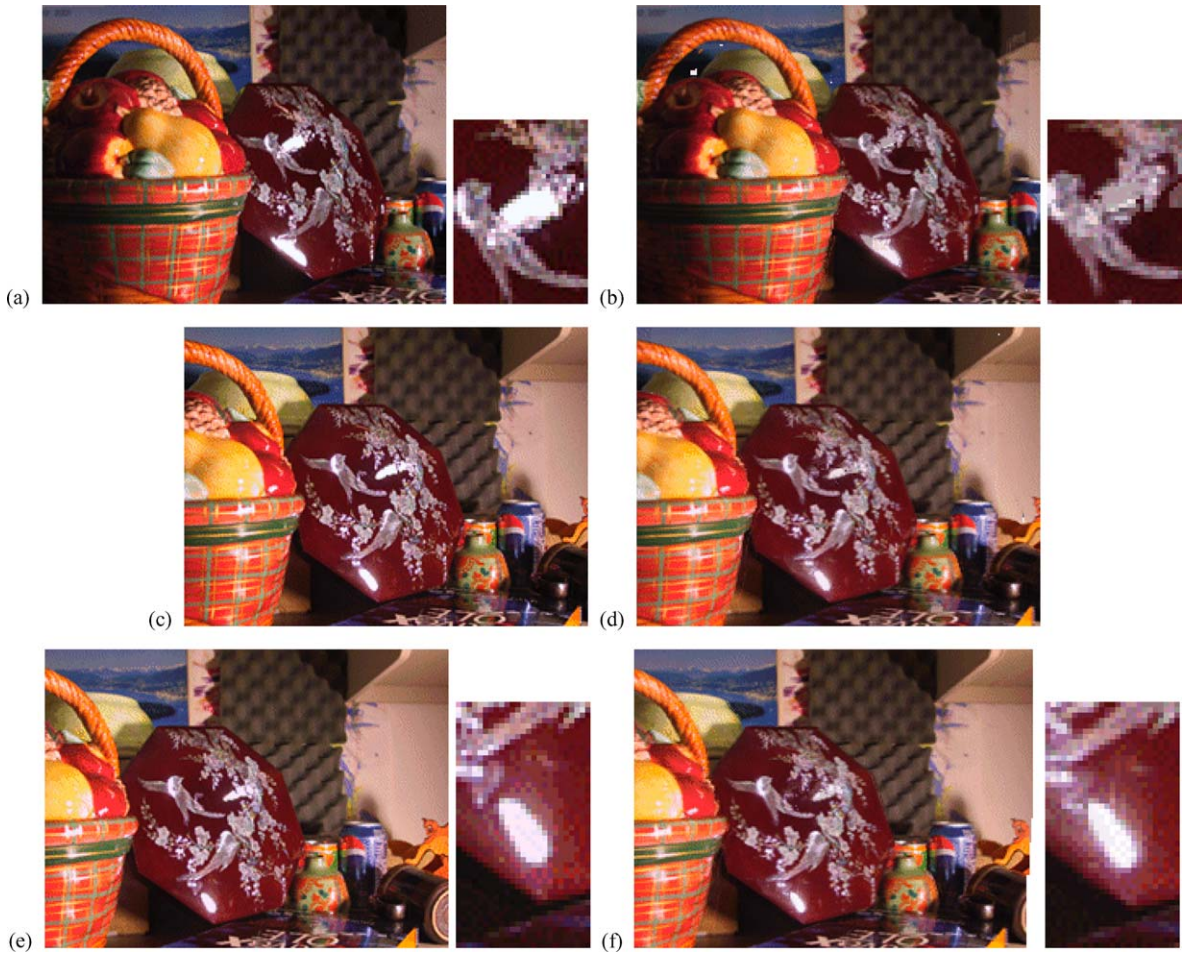

Figure 23. A more difficult sequence with specularities. The original sequence has 48 frames; the interpolated views (b, d, f) were generated using only 3 view-dependent depth images at frames 8, 24, and 39. The actual views, corresponding to frames 13, 36, and 42, are shown in (a, c, e). The insets in (a) and (b) show that certain highlights was not recreated well while the insets in (e) and (f) show that an example where a highlight was adequately synthesized.

by splatting in the forward mapping stage and dilation just prior to the inverse mapping stage.

Another example of why view-dependent depth images are necessary is shown in Fig. 23, which shows interpolated views of a scene with specular surfaces. Note that only *three* view-dependent depth images are used to represent the original 48 image sequence. As can be seen, many highlight areas were synthesized well, with a few exceptions (an example of which is shown in Fig. 23(a, b)). This problem can be alleviated by adding more view-dependent depth images. It is thus clear that using only a single depth image would not adequately represent the non-diffuse effects (without resorting to extracting the BRDF of the object surfaces).

5. Discussion

Over the years, researchers have developed many different types of representations and algorithms for recovering scene geometry from multiple images. These range from purely 3D representations, view-dependent textures (Debevec et al., 1998), and view-dependent 3D models (Pulli et al., 1997), which are more geometry-based, to Concentric Mosaics (Shum and He, 1999), Lumigraphs (Gortler et al., 1996), and Light Fields (Levoy and Hanrahan, 1996), which are more image-based. A description of the geometry-image tradeoff is given in Kang et al. (2000).

Table 2 compares a number of popular approaches to representing and rendering 3D scenes. The table lists

Table 2. Comparison of multi-view representations and some of their tradeoffs.

Representation	Space complex.	Comput. complex.	Discretization	View variation	Rendering
Voxels	*WHD*	*NWHD*	(x, y, d)	No	Voxel splatting
Layers	*WHL*	*NWHDL*?	$(x, y) \times L$	No	Two-pass rendering
3D surfaces	*WH*	*NWHI*?	Adaptive	No	Traditional CG
View-dep. depth images	*WHM*	$TMWHD^p$	(x, y)	Yes	Blended 2-pass rendr.

The constants in the space and computational complexity columns are: W: width; H: height; D: depth/disparity resolution; N: number of input images; L: number of layers; I: number of iterations; M: number of key views; T: size of temporal neighborhood, $T \sim O(N/M)$; p: exponent in graph cut algorithm (1–2).

the space and computational complexities of each approach (some of the computational complexities are uncertain, so they are tagged with a "?"), as well as the discretization involved in the representation, the support of view-dependent effects, and the rendering algorithms used. The space (storage) complexity of view-dependent depth images is relatively small, although not as compact as a single 3D model (which cannot model view-dependent effects). The computational complexity is similar to other multi-view stereo correspondence or refinement approaches. Additional advantages of our approach are that there is no need to discretize 3D space (which can result in rendering artifacts), nor is there a need to segment the images into coherent layers.

In choosing a representation, our goal was to provide adequately photorealistic view synthesis while maintaining a reasonable database size and permitting rendering at interactive speeds. Our proposed representation fulfills these requirements. It is related to the view-dependent 3D models used in Pulli et al. (1997), except that their models were created using accurate rangefinders and do not handle non-diffuse effects. Our technique is more practical and flexible, since we use only images and assume camera motion to be computed using standard structure from motion techniques.

Our representation can also be viewed as a form of compression of the 3D scene. With a judicious choice of the number and position of reference view-dependent depth images, we can generate the appearance of the captured 3D environment photorealistically. In this paper, we have not discussed the issue of depth image selection, but this is an area that we will be pursuing. One important question is determining the appropriate objective function to be used. Should a perceptually-based metric be used, or will a simple SSD-based error metric be adequate?

6. Conclusions

In this paper, we have proposed a view-dependent representation of 3D scenes, and shown how to extract such a representation from a sequence of images. We described two main approaches. The first is more traditional, computing the depth map associated with each chosen reference frame independently. The novelty of our approach lies in the combination of shiftable windows, temporal selection, and graph cut optimization. Shiftable windows enable object boundaries to be extracted well, while temporal selection is the key to (implicitly) handling occlusions. Finally, the graph cut algorithm handles, to a reasonable extent, textureless regions.

The second approach simultaneously optimizes a set of self-consistent depth maps at multiple key-frames. Since multiple depth maps are estimated simultaneously, visibility can be modeled explicitly, and indeed is used in the optimization function. In addition to the photoconsistency and smoothness terms, this approach also imposes disparity consistency across the different depth maps.

Results have shown that the first method produces significantly crisper boundaries. This can be attributed to the use of both shiftable windows and view selection in the matching process. On the other hand, it is more likely to produce less globally consistent depths at textureless regions, because disparity compatibility is not enforced across the different depth maps. In addition, the use of the graph cut for global optimization with smoothness tends to produce depth maps that are more blocky, e.g., less smooth variation in slanted regions such as the flower beds in Fig. 19.

Either approach can be used to produce a set of multi-view depth images, and both primarily rely on photoconsistency to produce results. The resulting representations capture both partially visible regions and

potential variations in local appearance across camera positions. The representation can therefore be used as a basis for novel view synthesis and interpolation applications.

Acknowledgment

We would like to thank Ramin Zabih and P. Anandan for initial discussions on stereo matching. Jinxiang Chai has been instrumental in implementing the graph cut algorithm used for extracting depth maps from multiple images.

References

Arnold, R.D. 1983. Automated stereo perception. Technical Report AIM-351, Artificial Intelligence Laboratory, Stanford University.

Baker, S., Szeliski, R., and Anandan, P. 1998. A layered approach to stereo reconstruction. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'98)*. Santa Barbara, pp. 434–441.

Barnard, S.T. 1989. Stochastic stereo matching over scale. *International Journal of Computer Vision*, 3(1):17–32.

Belhumeur, P.N. 1996. A Bayesian-approach to binocular stereopsis. *International Journal of Computer Vision*, 19(3):237–260.

Bergen, J.R., Anandan, P., Hanna, K.J., and Hingorani, R. 1992. Hierarchical model-based motion estimation. In *Second European Conference on Computer Vision (ECCV'92)*. Santa Margherita Liguere, Italy, pp. 237–252, Springer-Verlag.

Birchfield, S. and Tomasi, C. 1998. A pixel dissimilarity measure that is insensitive to image sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(4):401–406.

Birchfield, S. and Tomasi, C. 1999. Multiway cut for stereo and motion with slanted surfaces. In *Seventh International Conference on Computer Vision (ICCV'99)*. Kerkyra, Greece, pp. 489–495.

Black, M.J. and Jepson, A.D. 1996. Estimating optical flow in segmented images using variable-order parametric models with local deformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 18(10):972–986.

Black, M.J. and Rangarajan, A. 1996. On the unification of line processes, outlier rejection, and robust statistics with applications in early vision. *International Journal of Computer Vision*, 19(1):57–91.

Blonde, L. et al. 1996. A virtual studio for live broadcasting: The Mona Lisa project. *IEEE Multimedia*, 3(2):18–29.

Bobick, A.F. and Intille, S.S. 1999. Large occlusion stereo. *International Journal of Computer Vision*, 33(3):181–200.

Boykov, Y., Veksler, O., and Zabih, R. 2001. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239.

Buehler, C., Bosse, M., McMillan, L., Gortler, S.J., and Cohen, M.F. 2001. Unstructured Lumigraph Rendering. In *Proceedings of SIGGRAPH 2001*, pp. 425–432. ISBN 1-58113-292-1.

Chou, P.B. and Brown, C.M. 1990. The theory and practice of Bayesian image labeling. *International Journal of Computer Vision*, 4(3):185–210.

Collins, R.T. 1996. A space-sweep approach to true multi-image matching. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'96)*. San Francisco, California, pp. 358–363.

de Hann, G. and Beller, E.B. 1998. Deinterlacing—An overview. *Proceedings of the IEEE* 86(9):1839–1857.

Debevec, P.E., Taylor, C.J., and Malik, J. 1996. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. *Computer Graphics (SIGGRAPH'96)* pp. 11–20.

Debevec, P.E., Yu, Y., and Borshukov, G.D. 1998. Efficient view-dependent image-based rendering with projective texture-mapping. In *Eurographics Rendering Workshop 1998*, pp. 105–116. ISBN 3-211-83213-0. Held in Vienna, Austria.

Dhond, U.R. and Aggarwal, J.K. 1989. Structure from stereo—A review. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(6):1489–1510.

Geiger, D., Ladendorf, B., and Yuille, A. 1992. Occlusions and binocular stereo. In *Second European Conference on Computer Vision (ECCV'92)*. Santa Margherita Liguere, Italy, pp. 425–433, Springer-Verlag.

Geman, S. and Geman, D. 1984. Stochastic relaxation, Gibbs distribution, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(6):721–741.

Gortler, S.J., Grzeszczuk, R., Szeliski, R., and Cohen, M.F. 1996. The Lumigraph. In *Computer Graphics Proceedings, Annual Conference Series*. Proc. SIGGRAPH'96 (New Orleans): pp. 43–54, ACM SIGGRAPH.

Hanna, K.J. 1991. Direct multi-resolution estimation of ego-motion and structure from motion. In *IEEE Workshop on Visual Motion*. Princeton, New Jersey, pp. 156–162, IEEE Computer Society Press.

Hoff, W. and Ahuja, N. 1986. Surfaces from stereo. In *Eighth International Conference on Pattern Recognition (ICPR'86)*. Paris, France, pp. 516–518, IEEE Computer Society Press.

Irani, M., Anandan, P., and Hsu, S. 1995. Mosaic based representations of video sequences and their applications. In *Fifth International Conference on Computer Vision (ICCV'95)*. Cambridge, Massachusetts, pp. 605–611.

Ishikawa, H. and Geiger, D. 1998. Occlusions, discontinuities, and epipolar lines in stereo. In *Fifth European Conference on Computer Vision (ECCV'98)*. Freiburg, Germany, pp. 232–248, Springer-Verlag.

Ju, S.X., Black, M.J., and Jepson, A.D. 1996. Skin and bones: Multi-layer, locally affine, optical flow and regularization with transparency. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'96)*. San Francisco, California, pp. 307–314.

Kanade, T. et al. 1996. A stereo machine for video-rate dense depth mapping and its new applications. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'96)*. San Francisco, California, pp. 196–202.

Kanade, T. and Okutomi, M. 1994. A stereo matching algorithm with an adaptive window: Theory and experiment. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(9):920–932.

Kang, S.B., Szeliski, R., and Anandan, P. 2000. The geometry-image representation tradeoff for rendering. In *International Conference on Image Processing (ICIP-2000)*, vol. II. Vancouver, pp. 13–16.

Kang, S.B., Szeliski, R., and Chai, J. 2001a. Handling occlusions in dense multi-view stereo. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'2001)*. Kauai, Hawaii.

Kang, S.B., Szeliski, R., and Chai, J. 2001b. Handling occlusions in dense multi-view stereo. Technical Report MSR-TR-2001-80, Microsoft Research.

Kang, S.B., Webb, J., Zitnick, L., and Kanade, T. 1995. A multibaseline stereo system with active illumination and real-time image acquisition. In *Fifth International Conference on Computer Vision (ICCV'95)*. Cambridge, Massachusetts, pp. 88–93.

Kolmogorov, V. and Zabih, R. 2001. Computing visual correspondence with occlusions using graph cuts. In *Eighth International Conference on Computer Vision (ICCV 2001)*, vol. II. Vancouver, Canada, pp. 508–515.

Kolmogorov, V. and Zabih, R. 2002. Multi-camera scene reconstruction via graph cuts. In *Seventh European Conference on Computer Vision (ECCV 2002)*, vol. III. Copenhagen, pp. 82–96, Springer-Verlag.

Kutulakos, K.N. and Seitz, S.M. 2000. A theory of shape by space carving. *International Journal of Computer Vision*, 38(3):199–218.

Le Gall, D. 1991. MPEG: A video compression standard for multimedia applications. *Communications of the ACM*, 34(4):44–58.

Lee, M.-C. et al. 1997. A layered video object coding system using sprite and affine motion model. *IEEE Transactions on Circuits and Systems for Video Technology*, 7(1), 130–145.

Levine, M.D., O'Handley, D.A., and Yagi, G.M. 1973. Computer determination of depth images. *Computer Graphics and Image Processing*, 2(4):131–150.

Levoy, M. and Hanrahan, P. 1996. Light field rendering. In *Computer Graphics Proceedings, Annual Conference Series*. *Proc. SIGGRAPH'96* (New Orleans): pp. 31–42, ACM SIGGRAPH.

Lucas, B.D. and Kanade, T. 1981. An iterative image registration technique with an application in stereo vision. In *Seventh International Joint Conference on Artificial Intelligence (IJCAI-81)*. Vancouver, pp. 674–679.

Marr, D.C. and Poggio, T. 1979. A computational theory of human stereo vision. *Proceedings of the Royal Society of London*, B 204 301–328.

Marroquin, J., Mitter, S., and Poggio, T. 1987. Probabilistic solution of ill-posed problems in computational vision. *Journal of the American Statistical Association*, 82 (397):76–89.

Matthies, L.H., Szeliski, R., and Kanade, T. 1989. Kalman filter-based algorithms for estimating depth from image sequences. *International Journal of Computer Vision*, 3:209–236.

McMillan, L. and Bishop, G. 1995. Plenoptic modeling: An image-based rendering system. *Computer Graphics (SIGGRAPH'95)* pp. 39–46.

Nakamura, Y., Matsuura, T., Satoh, K., and Ohta, Y. 1996. Occlusion detectable stereo-occlusion patterns in camera matrix. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'96)*. San Francisco, California, pp. 371–378.

Ohta, Y. and Kanade, T. 1985. Stereo by intra- and inter-scanline search using dynamic programming. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-7(2):139–154.

Okutomi, M. and Kanade, T. 1992. A locally adaptive window for signal matching. *International Journal of Computer Vision*, 7(2):143–162.

Okutomi, M. and Kanade, T. 1993. A multiple baseline stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(4):353–363.

Okutomi, M., Katayama, Y., and Oka, S. 2002. A simple stereo algorithm to recover precise object boundaries and smooth surfaces. *International Journal of Computer Vision*, 47(1–3):261–273.

Poggio, T., Torre, V., and Koch, C. 1985. Computational vision and regularization theory. *Nature*, 317(6035):314–319.

Pulli, K. et al. 1997. View-based rendering: Visualizing real objects from scanned range and color data. In *Proceedings of the 8-th Eurographics Workshop on Rendering*. St. Etienne, France.

Quam, L.H. 1984. Hierarchical warp stereo. In *Image Understanding Workshop*. New Orleans, Louisiana, pp. 149–155, Science Applications International Corporation.

Roy, S. and Cox, I.J. 1998. A maximum-flow formulation of the N-camera stereo correspondence problem. In *Sixth International Conference on Computer Vision (ICCV'98)*. Bombay, pp. 492–499.

Saito, H. and Kanade, T. 1999. Shape reconstruction in projective grid space from large number of images. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'99)*, vol. 2. Fort Collins, pp. 49–54.

Satoh, K. and Ohta, Y. 1996. Occlusion detectable stereo—systematic comparison of detection algorithms. In *13th International Conference on Pattern Recognition (ICPR'96)*. Los Alamitos, pp. 280–286.

Sawhney, H.S. and Ayer, S. 1996. Compact representation of videos through dominant multiple motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8):814–830.

Sawhney, H.S. and Hanson, A.R. 1991. Identification and 3D description of 'Shallow' environmental structure over a sequence of images. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'91)*. Maui, Hawaii, pp. 179–185, IEEE Computer Society Press.

Scharstein, D. and Szeliski, R. 1998. Stereo matching with nonlinear diffusion. *International Journal of Computer Vision*, 28(2):155–174.

Scharstein, D. and Szeliski, R. 2002. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1):7–42.

Seitz, S.M. and Dyer, C.M. 1999. Photorealistic scene reconstruction by voxel coloring. *International Journal of Computer Vision*, 35(2):151–173.

Shade, J., Gortler, S., He, L.-W., and Szeliski, R. 1998. Layered depth images. In *Computer Graphics (SIGGRAPH'98) Proceedings*. Orlando, pp. 231–242, ACM SIGGRAPH.

Shi, J. and Tomasi, C. 1994. Good features to track. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'94)*. Seattle, Washington, pp. 593–600, IEEE Computer Society.

Shum, H.-Y. and He, L.-W. 1999. Rendering with concentric mosaics. In *SIGGRAPH'99*. Los Angeles, pp. 299–306, ACM SIGGRAPH.

Shum, H.-Y. and Szeliski, R. 1999. Stereo reconstruction from multiperspective panoramas. In *Seventh International Conference on Computer Vision (ICCV'99)*. Kerkyra, Greece, pp. 14–21.

Sun, S., Haynor, D., and Kim, Y. 2000. Motion estimation based on optical flow with adaptive gradients. In *International Conference on Image Processing (ICIP-2000)*, vol. I. Vancouver, pp. 852–855.

Swaminathan, R., Kang, S.B., Szeliski, R., Criminisi, A., and Nayar, S.K. 2002. On the motion and appearance of specularities in image sequences. In *Seventh European Conference on Computer Vision (ECCV 2002)*, Springer-Verlag, Copenhagen, pp. 508–523.

Szeliski, R. 1999. A multi-view approach to motion and stereo. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'99)*, vol. 1. Fort Collins, pp. 157–163.

Szeliski, R., Avidan, S., and Anandan, P. 2000. Layer extraction from multiple images containing reflections and transparency. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'2000)*, vol. 1. Hilton Head Island, pp. 246–253.

Szeliski, R. and Coughlan, J. 1997. Hierarchical spline-based image registration. *International Journal of Computer Vision*, 22(3):199–218.

Szeliski, R. and Golland, P. 1999. Stereo matching with transparency and matting. *International Journal of Computer Vision*, 32(1):45–61. Special Issue for Marr Prize papers.

Szeliski, R. and Kang, S.B. 1995. Direct methods for visual scene reconstruction. In *IEEE Workshop on Representations of Visual Scenes*. Cambridge, Massachusetts, pp. 26–33.

Szeliski, R. and Zabih, R. 1999. An experimental comparison of stereo algorithms. In *International Workshop on Vision Algorithms*. Kerkyra, Greece, pp. 1–19, Springer.

Tao, H., Sawhney, H., and Kumar, R. 2001. A global matching framework for stereo computation. In *Eighth International Conference on Computer Vision (ICCV 2001)*, vol. I. Vancouver, Canada, pp. 532–539.

Terzopoulos, D. 1986. Regularization of inverse visual problems involving discontinuities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(4):413–424.

Tian, Q. and Huhns, M.N. 1986. Algorithms for subpixel registration. *Computer Vision, Graphics, and Image Processing*, 35:220–233.

Torr, P.H.S., Szeliski, R., and Anandan, P. 2001. An integrated Bayesian approach to layer extraction from image sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(3):297–303.

Tsin, Y., Kang, S.B., and Szeliski, R. 2003. Stereo matching with reflections and translucency. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'2003)*, Madison, WI, pp. 702–709.

Veksler, O. 1999. Efficient graph-based energy minimization methods in computer vision. Ph.D. thesis, Cornell University.

Wang, J.Y.A. and Adelson, E.H. 1994. Representing moving images with layers. *IEEE Transactions on Image Processing*, 3(5):625–638.

Weiss, Y. 1997. Smoothness in layers: Motion segmentation using nonparametric mixture estimation. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'97)*. San Juan, Puerto Rico, pp. 520–526.

Weiss, Y. and Adelson, E.H. 1996. A unified mixture framework for motion segmentation: Incorporating spatial coherence and estimating the number of models. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'96)*. San Francisco, California, pp. 321–326.

Yang, Y., Yuille, A., and Lu, J. 1993. Local, global, and multilevel stereo matching. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'93)*. New York, New York, pp. 274–279, IEEE Computer Society.

Zitnick, C.L. and Kanade, T. 2000. A cooperative algorithm for stereo matching and occlusion detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(7):675–684.