SCENE RECONSTRUCTION FROM MULTIPLE CAMERAS

Richard Szeliski

Vision Technology Group
Microsoft Research
One Microsoft Way
Redmond, WA 98052-6399

ABSTRACT

This paper reviews a number of recently developed stereo matching algorithms and representations. It focuses on techniques that are especially well suited for stereoscopic and 3-D imaging applications such as novel view generation and the mixing of live imagery with synthetic computer graphics. The paper reviews some recent approaches to the classic problem of recovering a depth map from two or more images. It then describes a number of newer representations (and their associated reconstruction algorithms), including volumetric representations, layered plane-plus-parallax representations, and multiple depth maps.

1. INTRODUCTION

Stereo matching, which is one of the oldest problems in computer vision, now appears to be a maturing research area. Real-time stereo matching, which a few years ago required special-purpose hardware [3, 5], is now implementable on regular personal computers (see [5] for some references). Depth maps computed with such systems can now be used as basic building blocks for higher level processes such as background subtraction and tracking.

But is stereo really a solved problem? Consider, for example, one of the more recent applications of real-time stereo matching: the ability to composite live video with synthetic computer graphics using the process of *z-keying* [3]. Or, consider the ability to film or photograph a scene or activity from multiple views, and to then look at the same scene from novel viewpoints, i.e., *virtualized reality* [4]. These applications are certainly very exciting, but is the quality of existing algorithms adequate for their use in real production environments?

Judging from the results in recent papers, it appears that we are not there yet. The reconstructions produced by today's algorithms still often leave a "halo" of background pixels clinging to the foreground object. Furthermore, even if a stereo algorithm were to assign a correct depth to each pixel in an image, it would still fail to correctly handle *mixed pixels*, i.e., pixels whose color is a combination of foreground and background colors (which occur at nearly all pixels along a depth discontinuity). Similar problems arise if the stereo matching is used to create a 3-D broadcast stream that could then be viewed stereoscopically or with interactive viewpoint control.

2. DEPTH MAPS

The classical problem of computing a dense depth map from two or more images has been extensively studied. The goal of stereo matching can be stated as follows: For each (x, y) location in disparity space, find the disparity d that aligns corresponding locations in the input images (ignoring, for now, the possibility that pixels may be occluded). In traditional area-based correlation, the quality of a match is measured by comparing windows centered at corresponding locations, for example, using the sum of squared intensity differences (SSD).

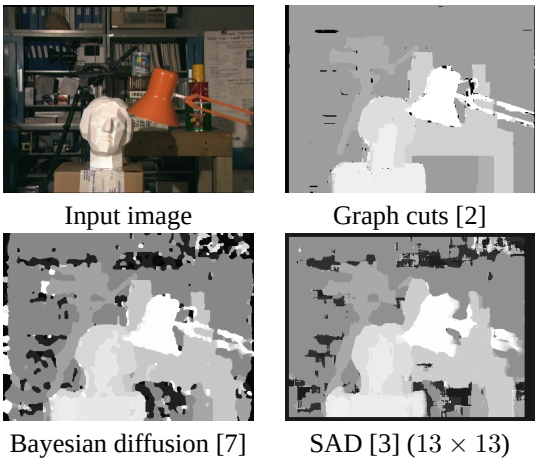
A more general way of characterizing area-based algorithms is the following [7]:

1. For each disparity under consideration, compute a per-pixel matching cost.
2. Aggregate support spatially (e.g., by summing over a window, or by diffusion).
3. At each pixel (x, y), find the best matching disparity d based on the aggregated support.
4. Compute a sub-pixel disparity estimate (optional).

More details on these four steps and their variants can be found in [9]. Additional information on depth map computation can be found in [6].

Optimization (regularization) approaches start with the same computation of matching costs as area-based techniques, but then add a controlled smoothness penalty (prior) on the disparity field $d(x, y)$. A variety of optimization algorithms can then be used to find a good solution to this problem [7, 2]. A more detailed survey of these techniques can be found in [9].

Figure 1 shows the results of applying three different matching algorithms on a multi-camera stereo data set provided by the University of Tsukuba. You can readily see



Input image | Graph cuts [2]

Bayesian diffusion [7] | SAD [3] (13×13)

Figure 1: Results for several stereo algorithms on the University of Tsukuba imagery.

that the two energy minimization-based approaches (Graph cuts and Bayesian) have much crisper depth discontinuities, compared with the sum of absolute differences (SAD) technique.

3. VOLUMETRIC REPRESENTATIONS

Unfortunately, a depth map representation is unable to represent and hence render partially occluded background regions. This is due to our insistence (enforced during the winner-take-all stage) that only a single depth value be assigned to each pixel in the reference image.

What if we were to relax this assumption? What if in addition to being able to have several depth along each ray in the reference image, we also represented the colors of these pixels and their (potentially partial) opacities? In principle, we should be able to represent and reason about partially occluded pixels, and to correctly estimate the color values of mixed pixels. These are the intuitions that led to the development of our volumetric stereo reconstruction algorithm [11].)

Our algorithm starts by performing the same matching cost computation, aggregation, and winning depth value selection as described in the previous section. However, instead of insisting that every pixel in the reference image pick a winning depth, we only select depth values that have a good match (good aggregated evidence), using a threshold to mark other pixels as currently "unassigned" (0 opacity).

Once we have an initial (x, y, d) volume containing estimated RGBA (color and 0/1 opacity) values, we can reproject this volume into each of the input cameras. After the re-projection step, we refine the disparity estimates by preventing visible surface pixels from voting for potential disparities in the regions they occlude. More precisely, we build an (x, y, d, k) *visibility map*, which indicates whether a given camera k can see a voxel at location (x, y, d) [11]. Only visible pixels at a given depth now participate in computing the per-pixel matching costs. This results in a better set of stereo matches, and the process can be iterated.

While the above process of computing visibilities and refining disparity estimates will in general lead to a higher quality disparity map (and better quality mean colors, i.e., texture maps), it will not recover the true colors and transparencies in *mixed pixels*, e.g., near depth discontinuities, which is one of the main goals of this research. Therefore, in the second phase of our algorithm, we adjust the opacity and color values to match the input images (after re-projection), while favoring continuity in the color and opacity values [11].

The volumetric approach is a much more powerful representation for dealing with partially occluded regions and mixed pixels. Unfortunately, this power comes at the expense of two problems: the depth are quantized, which can lead to aliasing effects, and the representation has a very large number of degrees of freedom, which makes it difficult to find the optimal solution.

4. LAYERED REPRESENTATIONS

To overcome these problems, we draw some inspiration from recent work in layered motion estimation [12]. Here, the goal is to decompose the images into *layers*, such that the pixels within each layer move in a manner consistent with a parametric transformation. A example of such a transformation is the 8–parameter homography (collineation) that describes the motion of a rigid planar patch as either it or the camera moves.

While existing techniques have been successful in detecting multiple independent motions, layer extraction for scene modeling has not been fully developed. One fact that has not been exploited is that, when simultaneously imaged by several cameras, each of the layers implicitly lies on a fixed plane in the 3D world. Another omission is the proper treatment of transparency. With a few exceptions [10], the decomposition of an image into layers that are partially transparent has not been attempted.

In our own work [1], we have developed a framework for reconstructing a scene as a collection of approximately planar layers. Each of the layers has an explicit 3D plane equation and is recovered as a *sprite*, i.e. a colored image with per-pixel opacity (transparency). To model a wider range of scenes, a per-pixel depth offset relative to the plane is also added.

Our estimation framework consists of two parts. In the first part, we assume boolean opacities to get a first approximation to the structure of the scene, i.e., we introduce boolean masks B_{kl} which denote the pixels in image I_k that are images of points on layer L_l. Once we have estimates of the masks, we immediately compute masked input im-

(a) (b) (c)

Figure 2: Layered stereo results: (a) middle (of five) images; (b) the five layer sprites; (c) novel view with residual depth (note the "rounding" of the people).

ages $M_{kl} = B_{kl} \cdot I_k$, and use these to compute the layer motion and out-of-plane parallax. In the second part of our framework, we use the initial estimates of the layers made by the first part as input into a re-synthesis algorithm that refines the layer sprites L_l, including real-valued opacities α_l. This second step requires a generative or forward model of the image formation process. A more detailed description of our algorithm can be found in [1, 9].

Figure 2 shows some results of applying our algorithm to five images from a 40-image stereo data set taken at a graphics symposium. Figure 2(a) shows the middle input image, Figure 2(b) shows the recovered sprites, and Figure 2(c) shows the same sprite collection seen from a novel viewpoint (well outside the range of the original views), with residual depth correction. The gaps in Figure 2(b) correspond to parts of the scene that were not visible in any of the five input images.

To summarize, the layered approach to 3D reconstruction represents the scene as a collection of approximately planar layers. Each layer consists of a plane equation, a layer sprite image, and a residual depth map. The framework exploits the fact that each layer implicitly lies on a fixed plane in the 3D world. This is both the algorithm's strength (using a compact description) and its weakness (it is limited to scenes where objects are "cutouts with relief").

5. MULTIPLE DEPTH MAPS

In our most recent work, we have been investigating an alternative to volumetric and layered representations that can also represent and reason about semi-occluded regions. Rather than estimating a single depth map, we associate a depth map with *each* input image (or some subset of them) [8]. Furthermore, we try to ensure consistency between these different estimates using a *depth compatibility* constraint, and reason about occlusion relationships by computing pixel *visibilities*. Our representation can be used as is for image-based rendering (view interpolation) applications, or it can be used as a low-level representation from which segmentation and layer extraction (or 3D model construction) can take place.

To formulate the multi-view stereo problem, we take the matching costs for all reference images and sum them together. This *brightness compatibility* term, which measures the degree of agreement in brightness or color between corresponding pixels, can be written as

$$\mathcal{C}(\{\mathbf{x}_s\}) = \sum_{s \in S} \sum_{t \in \mathcal{N}(s)} w_{st} \sum_{\mathbf{x}_s} v_{st}(\mathbf{x}_s) \rho\left(I_s(\mathbf{x}_s) - I_t(\mathbf{x}_t)\right).$$

The images I_s form the set S of *keyframes* (or *key-views*) for which we will estimate a depth estimate $d_s(\mathbf{x}_s)$. The decision as to which images are keyframes is problem-dependent, much like the selection of I and P frames in video compression.

Images $I_t, t \in \mathcal{N}(s)$ are *neighboring frames* (or views), for which we require that corresponding pixel brightnesses (or colors) agree. The pixel coordinate $\mathbf{x}_t$ corresponding to a given keyframe pixel $\mathbf{x}_s$ with depth d_s can be computed according to a rigid motion model. The constants w_{st} are the *inter-frame weights* that dictate how much neighboring frame t will contribute to the estimate of d_s. Corresponding pixel brightness or color differences are passed through a robust penalty function ρ. The visibility factor $v_{st}(\mathbf{x}_s)$, which encodes whether pixel $\mathbf{x}_s$ is *visible* in image I_t, can be computed by comparing corresponding depth values, i.e., checking whether $d_t(\mathbf{x}_t) \leq d_s(\mathbf{x}_s)$.

The cost function used in [8] consists of two additional terms. The controlled *depth compatibility* constraint, enforces *mutual consistency* between depth estimates at different neighboring keyframes. The controlled *depth smoothness* constraint, encourages the depth maps to be piecewise smooth. The shape of this robust penalty function is affected by the brightness/color difference between neighboring pixels (see [8] for details).

Our algorithm operates in two phases. During an initialization phase, we estimate the depths independently for each keyframe. Since we do not yet have any good motion estimates for other frames, the depth compatibility term is ignored, and no visibilities are computed (i.e., $v_{st} = 1$). In the second phase, we enforce depth compatibility and compute visibilities based on the current collection of depth estimates $\{d_s\}$. Details on the optimization algorithm can be found in [8].

(a) (b) (c)

Figure 3: Results of multi-view stereo algorithm: (a) depth estimate for first frame; (b) warped (resampled) images without visibilities; (c) with visibility computation.

Figure 3 shows some representative results from running our algorithm. The depth map estimated by the algorithm is shown in Figure 3a. Figure 3b shows the results of warping the last image to the first image, based on the depth computed in the first image. Figure 3c shows the same warped image with invisible pixels flagged as black. Notice how the algorithm correctly labels most of the occluded pixels to the right of the two people's heads.

The multi-view stereo matching framework described in this section produces estimates for a subset of the input images, thereby representing depth in partially occluded regions and explicitly modeling the variation in appearance between different views. Compared with the volumetric and layered representations, the multiple depth map representation is potentially not as compact (although it can be more compact than the search space of the volumetric technique), nor does it correctly model mixed pixels (because the concept of opacity is not built in). The representation does, however, capture the variation in appearance between different view (for tasks such as novel view generation), and can also be used to *bootstrap* a more parsimonious representation such as 3D layers.

6. DISCUSSION AND CONCLUSIONS

In this paper, I have presented four different representations for stereo matching. A single depth map, the traditional representation used for matching, is a very compact and useful representation that can yield good results when the amount of occlusion is not large, i.e., when the surface is smoothly varying (e.g., a human face) and the range of viewpoints is limited The volumetric representation (with partial opacities) can be used to represent and reason about partially occluded regions and mixed pixels. Unfortunately, it also has many degrees of freedom, which makes it tricky to find the best reconstruction. Layered representations have the same advantages as volumetric ones, and are potentially more compact, and hence easier to recover. However, determining the best number of planes and the correct pixel assignment is a tricky problem, which we are currently trying to solve. Finally, multiple depth maps can be used to obtain some of the same advantages with respect to partially occluded regions, and also to model the variation in appearance between viewpoints. Unfortunately, they are not guaranteed to have consistent representations of 3D shape, and also do not correctly predict the appearance of mixed pixels.

Thus, we see that all of the representations suggested so far have their limitations. Still, a tremendous amount of progress has been made in recent years in obtaining better and better stereo reconstructions. I expect that by re-visiting issues in representation, we will be able to make even further progress, and thereby expand the utility and applicability of multi-camera stereo-based reconstruction techniques.

References

[1] S. Baker, R. Szeliski, and P. Anandan. A layered approach to stereo reconstruction. In *CVPR'98*, pp. 434–441, June 1998.

[2] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. In *ICCV'99*, page 377-384, September 1999.

[3] T. Kanade et al. A stereo machine for video-rate dense depth mapping and its new applications. In *CVPR'96*, pp. 196–202, June 1996.

[4] T. Kanade, P. W. Rander, and P. J. Narayanan. Virtualized reality: constructing virtual worlds from real scenes. *IEEE MultiMedia Magazine*, 1(1):34–47, Jan-March 1997.

[5] R. Kimura et al. A convolver-based real-time stereo machine (SAZAN). In *CVPR'99*, v. 1, pp. 457–463, June 1999.

[6] R. Kumar et al. 3D Manipulation of Motion Imagery. In *ICIP-2000*, September 2000.

[7] D. Scharstein and R. Szeliski. Stereo matching with nonlinear diffusion. *Int. J. Comp. Vis.*, 28(2):155–174, July 1998.

[8] R. Szeliski. A multi-view approach to motion and stereo. In *CVPR'99*, v. 1, pp. 157–163, June 1999.

[9] R. Szeliski. Stereo algorithms and representations for image-based rendering. In *British Machine Vision Conference (BMVC'99)*, v. 2, pp. 314–328, September 1999.

[10] R. Szeliski, S. Avidan, and P. Anandan. Layer extraction from multiple images containing reflections and transparency. In *CVPR'2000*, June 2000.

[11] R. Szeliski and P. Golland. Stereo matching with transparency and matting. *Int. J. Comp. Vis.*, 32(1):45–61, 1999.

[12] J. Y. A. Wang and E. H. Adelson. Representing moving images with layers. *IEEE T. Im. Proc.*, 3(5):625–638, September 1994.