In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'96)*, San Francisco, June 1996, pp. 852–858

Stereo Vision for View Synthesis

Daniel Scharstein*
Department of Computer Science
Cornell University
Ithaca, NY 14853-7501, USA
schar@cs.cornell.edu

Abstract

We propose a new method for view synthesis from real images using stereo vision. The method does not explicitly model scene geometry, and enables fast and exact generation of synthetic views. We also re-evaluate the requirements on stereo algorithms for the application of view synthesis and discuss ways of dealing with partially occluded regions of unknown depth and with completely occluded regions of unknown texture. Our experiments demonstrate that it is possible to efficiently synthesize realistic new views even from inaccurate and incomplete depth information.

1 Introduction

Stereo vision has been one of the earliest and most thoroughly investigated topics in the computer vision community [3]. Although numerous stereo vision systems have been developed that exhibit good performance in restricted environments or for specific tasks, the "general stereo vision problem" is far from being solved. Among the most notorious problems in stereo vision are repetitive patterns and textureless areas, and the presence of depth discontinuities and half-occluded regions.

In this paper we re-evaluate stereo vision for the application of view synthesis. While in traditional stereo the desired output is a 3-D description of the observed scene, in the application of view synthesis the desired output are realistic-looking images of the scene as it would appear from novel viewpoints. This problem might appear to be at least as hard as the original problem, since the generation of new views obviously involves knowledge of the scene geometry. Certain common scenarios, however, can yield correct synthetic views even though the underlying geometry is ambiguous, while other scenarios can yield consistent synthetic views that look realistic to an observer (and thus meet our goal), even though they are based on incorrect geometry.

Many authors have used the generation of new views by means of 3-D reconstruction and reprojection to illustrate the performance of their stereo and structure from motion algorithms. In these cases, the image is typically projected (texture-mapped) onto the recovered three-dimensional surface (which might be a triangulation of sparse feature points), and then rendered from a vastly different viewpoint, to emphasize the errors in computed depth and allow an evaluation of the accuracy of the method. Scene points of unknown geometry and previously invisible scene points are usually displayed in black.

In contrast to this, our proposed application of view synthesis from stereo data has very different goals: Instead of emphasizing errors in the recovered geometry, we are interested in generating realistic new views with minimal visual artifacts. This restricts new viewpoints to be reasonably close to the existing ones. Also, we can not tolerate "black holes" due to regions of unknown geometry or texture, but we have to deal explicitly with these cases by making "educated guesses". To allow real-time applications, such as virtual reality, new views need to be synthesized efficiently by warping the existing images based on depth information, rather than by explicit model building and rerendering.

In this paper, we present a new method for view synthesis which addresses these issues. Our method is based on a rectification step that both aids in stereo matching and allows an easy formulation of fast exact view synthesis. The method incorporates ways of dealing with partially occluded regions of unknown depth and with completely occluded regions of unknown texture, which are issues not addressed in most previous approaches.

2 Related work

View synthesis from real images is a topic that has received much recent interest. Several authors have proposed synthesizing views from stereo data. Ott *et al.* [10] create a virtual center view from two off-center images. Laveau and Faugeras [8] describe constructing a new view directly from weakly calibrated images. McMillan and Bishop [9] use cylindrical stereo on two panoramic views created by mosaicing. Szeliski and

*This work was supported by NSF grant IRI-9057928.

Kang [13] synthesize new views using spline-based image registration. Kumar *et al.* [7] describe parallax-based view reprojection for constructing 3D-corrected mosaics. Fuchs *et al.* [4] and Kanade *et al.* [5] describe systems for virtual teleconferencing and "virtualized reality" based on re-rendering real images that have been mapped onto polygonal meshes, which in turn are computed from hand-edited depth maps acquired by multiple-baseline stereo.

In other work, intermediate views are created using image interpolation: Chen and Williams [1] use image interpolation to speed up rendering in computer graphics. Werner *et al.* [14] use view interpolation to generate new views close to existing ones. Seitz and Dyer [12] derive criteria under which image interpolation yields the correct synthetic view. Katayama *et al.* [6] describe view generation based on the interpolation of epipolar-plane images.

Some of the above methods yield the correct view only for special viewing configurations, while others rely on affine projection models. Few existing methods deal with occlusion issues. Our method handles not only occlusion, but also yields the exact views under the full perspective model due to a special rectification step, although it is in spirit an interpolation method.

3 Synthesizing a new view

In this section we show how a new, *virtual* view I_3 can be synthesized from two *reference views* I_1, I_2. We will develop coordinate transforms that enable us to formulate view synthesis as linear disparity interpolation, allowing fast generation of new views by a local warping algorithm. Note that we solve the exact view synthesis problem as opposed to other work in which the term "view interpolation" refers to an approximation of the correct synthetic view.

3.1 Three-view rectification

We will assume that the geometry of the two existing views is known, either by explicit calibration, or by self-calibration [2], and that the desired configuration of the third (virtual) camera is specified relative to the existing two.[1]

For convenience, we will choose our global coordinate system such that all three focal points $\mathbf{c_1}$, $\mathbf{c_2}$, $\mathbf{c_3}$ lie in the plane $Z = 0$. In particular

$$\mathbf{c_1} = [0\ 0\ 0]^T, \quad \mathbf{c_2} = [1\ 0\ 0]^T, \quad \mathbf{c_3} = [X_S\ Y_S\ 0]^T,$$

where the subscripts S indicate the synthetic view.

We use projective coordinate transforms $\mathbf{T_i}$, $(i = 1, 2, 3)$, to project the original images onto the plane $Z = 1$, yielding the rectified images I_i'. That is, a point $\mathbf{q_i} = (u_i, v_i)^T$ in image I_i is projected to $\mathbf{q_i}' = (u_i'/w_i', v_i'/w_i')^T$, with

$$\begin{bmatrix} u_i' \\ v_i' \\ w_i' \end{bmatrix} = \mathbf{T_i} \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix}, \quad \text{and} \quad \mathbf{T_i} = \left[\begin{array}{c|c|c} \mathbf{r_i} & \mathbf{s_i} & \mathbf{o_i} - \mathbf{c_i} \end{array}\right],$$

where $\mathbf{r_i}$, $\mathbf{s_i}$ are the unit vectors and $\mathbf{o_i}$ is the origin of the original coordinate system of image I_i. The reprojection of I_i to I_i' based on $\mathbf{T_i}$ can be done using a fast projective image warping algorithm [15].

In the resulting rectified geometry, all three cameras have identical parameters, all image planes coincide, and all three coordinate systems are oriented the same way.[2] Now, in images I_1', I_2', I_3', a scene point $\mathbf{P} = (X_P, Y_P, Z_P)^T$ has the coordinates

$$\mathbf{p_1} = \begin{bmatrix} \frac{X_P}{Z_P} \\ \frac{Y_P}{Z_P} \end{bmatrix}, \quad \mathbf{p_2} = \begin{bmatrix} \frac{X_P - 1}{Z_P} \\ \frac{Y_P}{Z_P} \end{bmatrix}, \quad \mathbf{p_3} = \begin{bmatrix} \frac{X_P - X_S}{Z_P} \\ \frac{Y_P - Y_S}{Z_P} \end{bmatrix}.$$

Its *disparity*, i.e., its offset in position between images I_1' and I_2' is $d = -1/Z_P$. Thus, we can specify $\mathbf{p_3}$, the image coordinates of $\mathbf{P}$ in the virtual view, as a linear combination of disparity d and the position of the virtual camera $(X_S, Y_S)^T$:

$$\mathbf{p_3} = \mathbf{p_1} + d \begin{bmatrix} X_S \\ Y_S \end{bmatrix}. \tag{1}$$

3.2 Rendering a new view

Given that we have a dense disparity map d_{12} between images I_1' and I_2' (an issue which we will discuss in Section 4), Equation 1 yields a fast way of synthesizing any new view at $(X_S, Y_S)^T$ based on *forward mapping*. There are two issues that need to be dealt with: resolving visibility and filling *holes*.

A visibility decision needs to be made whenever two different points map to the same location in the new view. A key advantage of the rectified geometry is that visibility can be resolved automatically by simply mapping the pixels to their new positions in the correct sequence, since the front-to-back order is the same for all three views. The correct mapping sequence depends only on image coordinates and not on the depth values and has the effect that closer pixels are mapped later, thus automatically overwriting pixels further away.

Holes in the new view occur if the new viewpoint uncovers previously invisible scene points. We have

[1] In the case of "pure" weak calibration, i.e., where we only know the fundamental matrix relating the epipolar geometries, specifying the new viewpoint presents a problem [8]. We thus assume that we have at least a rough estimate of the full (external) calibration (see Section 6).

[2] Rectification is commonly done for stereo vision from two images, yielding coinciding epipolar lines. The rectifying plane has to be parallel to the baseline, but its orientation is arbitrary. We have taken advantage of this fact and have chosen a plane that is parallel to *all three* baselines, yielding pairwise coinciding epipolar lines between all three images.

to distinguish carefully between sampling gaps due to the forward mapping process, and real holes caused by occlusion boundaries in the disparity map. Sampling gaps occur when the (small) disparity difference between adjacent pixels is amplified in the remapping process. The same is true for holes, except that the disparity difference that causes the hole corresponds to a depth discontinuity. Since depth maps are discrete, it might not be obvious to distinguish between the two cases. One possibility is to impose a disparity gradient limit acting as a threshold.

Given that we have distinguished between depth discontinuities and small disparity differences, we can counteract sampling gaps by increasing the sampling rate depending on the magnitude of the offset vector $(X_S, Y_S)^T$. This results in "stretching" the visual surface. We need a different approach to deal with holes, however, since we do not want to stretch surfaces across depth discontinuities.

Before discussing how holes can be filled explicitly, we will show how the size and number of holes can be reduced by combining the information from both reference images. Using two symmetric disparity maps d_{12} and d_{21}, we can warp each image I_1, I_2 separately, yielding two synthetic images $I_{3,1}$, $I_{3,2}$ for the same new viewpoint. Although displaying the identical view, these two images can differ in the following ways: (1) The global intensities can be different due to different camera characteristics of the original two cameras; (2) the quality can be different due to the different distortions created by the two warps; (3) the holes (i.e., previously invisible scene points) are at different positions.

To compensate for the first two effects, it is useful to blend the intensities of the two images, possibly weighing the less-distorted image more (e.g., by using weights proportional to the distances of the virtual viewpoint to the reference viewpoints). If there is only a hole in one of the two images, it can be filled using intensity information from the other image.[3] If both images have a hole at the same position, we need to fill it explicitly, which will be discussed in the next section. It is advisable to perform a global intensity correction before the images are combined, in order to avoid visual artifacts resulting from filling holes from single images.

3.3 Filling holes

Holes in the synthesized view occur when the new viewpoint reveals previously invisible scene points. We have seen that only holes that occur at the same position in both images need to be filled explicitly. Such coinciding holes correspond to scene points invisible from both cameras, which are quite likely observed from "extrapolated" viewpoints outside the original baseline, but are unlikely for "interpolated" viewpoints in between the reference viewpoints. The reason is illustrated in Figure 1: two different objects have to "conspire" in order for coinciding holes to occur in intermediate views. We can not exclude this case in natural environments, and thus holes can never be avoided completely.

[3]Note that we can only fill holes from single images if we have disparity estimates for partially occluded pixels, which will be discussed in Section 4.2.

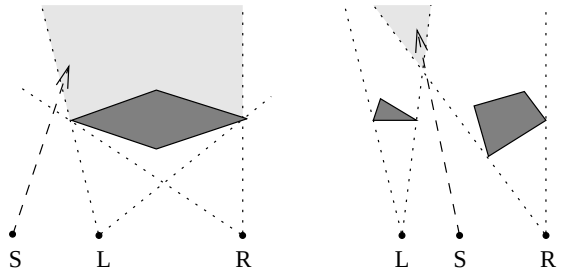

Figure 1: A synthetic view S generated from two views L and R has holes due to the exposure of previously invisible scene points. Single objects can cause holes only for views outside the original baseline (left), while multiple objects can conspire to create holes even in intermediate views (right).

Dealing with this situation involves synthesizing texture for the newly visible areas. An easy way of filling these holes is to spread the intensities of the neighboring pixels, but this often yields "blurry" regions. For the experimental results shown in Section 5, holes are filled by mirroring the intensities in the scanline adjacent to the hole, which gives noticeably better results than simple intensity spreading. It is very important to prevent intensities from being spread across occlusion boundaries, since holes are usually created by a close object that has uncovered part of the scene, and now bounds the hole on one side. The new texture should be based only on existing intensities on the close side of these boundaries, to avoid "smearing" of foreground and background. More sophisticated texture synthesis methods based on statistical measures of neighboring intensities are clearly possible.

3.4 The algorithm

In summary, we can synthesize a new view I_3 from images I_1, I_2 using the following algorithm:

1. Compute rectified images I_1', I_2' using projective transforms $\mathbf{T_1}$, $\mathbf{T_2}$.
2. Using a stereo algorithm that explicitly handles occlusion, compute dense disparity maps $d_{12}(i,j)$ and $d_{21}(i,j)$ between images $I_1'(i,j)$ and $I_2'(i,j)$.
3. Compute new images $I_{3,1}'$ and $I_{3,2}'$ by mapping points from $I_1'(i,j)$ to $I_{3,1}'(i + X_S d_{12}(i,j), j +$

$Y_S d_{12}(i,j))$ and from $I'_2(i,j)$ to $I'_{3,2}(i + X_S d_{21}(i,j), j + Y_S d_{21}(i,j))$.

4. Adjust the intensities of images $I'_{3,1}$ and $I'_{3,2}$, and combine them into image I'_3, filling single holes in the process.
5. Fill the remaining holes in I'_3 using texture synthesis.
6. Compute final "de-rectified" image I_3 from I'_3 using inverse transform $\mathbf{T_3}^{-1}$.

Note that if many views need to be synthesized from the same original image pair, the first two steps of the algorithm, i.e., rectification and stereo matching, only need to be performed once. Even if the new views lie in different planes, which requires a new rectification step, the disparity map does not need to be recomputed, but can simply be reprojected.

Stereo matching is the most time-intensive step of the algorithm. It takes at least $O(ND)$ time, where N is the number of pixels and D is the number of disparities. Rectification and view generation can be accomplished much faster, essentially in constant time per pixel ($O(N)$). This enables interesting applications, such as "low-cost virtual reality", where a single server with high computing power provides images and disparities in real time, and a large number of clients can generate different viewpoints requiring only little computing power.

Note that the view synthesis algorithm based on explicit rectification becomes impractical for viewing directions close to parallel to the *tri-focal plane*, the plane containing the three camera centers. The reason is that explicit rectification for these directions results in distortions and large image sizes. New views could still be generated using a slower general rendering algorithm.

4 Re-evaluating stereo

In the previous section we have seen that we can efficiently generate new views from a stereo correspondence map relating two rectified images. In this section we will discuss the requirements on a stereo algorithm whose output is to be used for view synthesis.

One basic difference of stereo for view synthesis to many other applications is that we need a dense depth map. During the mapping step, we can neither utilize information about certainties of depth estimates nor about unmatched points, since every pixel in the image needs to be mapped to a new position. This has two consequences: we want the stereo algorithm to pick canonical solutions that create minimal artifacts where there are multiple or ambiguous depth interpretation; and we have to make extra assumptions about the disparities of unmatched points.

4.1 Areas of uniform intensities

Our basic observation is that uniform regions, which traditionally have been a problem in stereo, can yield the same views largely independent of the underlying depth interpretation. Thus, the *aperture problem*, which states that the local displacements can only be recovered in the direction of the intensity gradient, does not affect the synthesized view. This observation is in agreement with Seitz and Dyer [12], who show that pure interpolation of views is a well-posed problem under the additional assumption of monotonicity (which excludes occlusion). They propose a view interpolation algorithm that matches and shifts uniform patches of intensity as a whole.

While it is necessary to impose the strong constraints of monotonicity and strict interpolation to prove this result, we argue that uniform patches usually do not create visual artifacts in the new view as long as their boundaries are matched correctly. Since our algorithm relies on explicit depth maps, we can assign disparities to uniform regions by simply interpolating the disparities from the boundaries. A different way of achieving a similar effect is to treat uniform areas the same way as unmatched points.

Of course, there are viewpoints for which incorrect views will be generated, even if the underlying disparity map is a canonical interpretation of an ambiguity. In general, this is true for any viewpoint from which an additional (real) camera could be used to disambiguate the possible depth interpretation. In other words, if an error in the computed disparities could be detected with an additional camera, then the view from this point can reveal the error. This includes not only ambiguities due to the aperture problem, but also repetitive patterns.

Resolving ambiguities is not always necessary, however, as there are many situations in which the canonical interpretation is incorrect, but the corresponding geometry is consistent, and the error is not apparent in the new view. That is, even though adding an extra camera would yield a different depth map (and different synthetic views), this is not necessary to convey a convincing three-dimensional structure. This is also true for the problem of filling holes: adding extra cameras reduces the number of holes, and thus less "guessing" of textures is required, increasing the accuracy of the synthesized view. However, adding extra cameras might not be economical, since it requires a more complicated calibration procedure.

4.2 Dealing with partial occlusion

Besides regions of uniform intensity, we also have to deal with *partially occluded* regions that are visible from only one camera. Figure 2 shows an example of such a case. Note that we have intensity information but no depth information for the points only visible

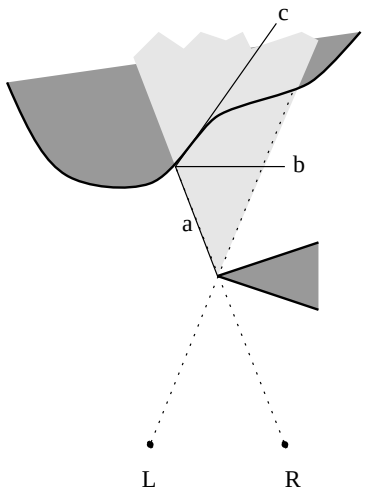

Figure 2: Partial occlusion: The lightly-shaded region is only visible from the left camera, giving rise to different possible depth interpretations.

from the left camera. If we want to generate new views to the left of the right camera position, we either have to make assumptions about the depth of these half-occluded points, or we will be left with holes in the new image. While the latter approach can be taken [10], we do not want to discard the intensity information of the partially occluded points. Thus, we have to assign explicit depth to these points.

Obviously, assigning depth has to rely on heuristics, as there are an infinite number of possible depth interpretations (corresponding to the lightly shaded region in Figure 2). However, there are a number of obvious choices: (**a**) interpolating the depth values between the points of known depth, (**b**) assuming constant depth, or (**c**) assuming constant depth gradient. Interpolating depth (**a**) is usually a bad choice since it assumes an unlikely viewing position of the right camera. Assuming constant slope (**c**) seems like a good idea, but it is difficult to reliably estimate the depth gradient from a discrete noisy disparity map. Also, since the half-occluded regions are usually fairly narrow, constant slope and constant depth assumptions often result in minimal differences. Thus, assuming constant depth (**b**) is easiest and most stable, and has also produced good results in our experiments.

It is apparent that the stereo algorithm needs to detect and correctly label partially occluded points, rather than assigning random disparities in these areas. One way of detecting occluded regions is to separately compute the two disparity maps d_{12} and d_{21}, which are required by the view synthesis algorithm, and then to label those points as occluded whose disparities disagree. This "two-pass" approach to dealing with occlusion is used in our current stereo algorithm.

5 Experimental Results

We tested our method on several image pairs. In this paper we show results for the KIDS image pair[4] which is already rectified, thus making explicit rectification unnecessary. The stereo matcher used here is an extension of a stereo matcher based on intensity gradients described in [11]. The algorithm favors disparity boundaries that coincide with intensity edges, and detects half-occluded regions by performing a consistency check as mentioned above. A "no-match" status is assigned to points that fail the consistency check, and also to areas that match with low confidence (e.g., areas of uniform intensities).

Once the correspondences have been established, new views can be generated very efficiently. Figure 3 shows results for the KIDS image pair. The first five rows show synthesized views from different positions along the baseline; the last row shows the disparity maps. The left original image is shown in row 2, the right image in row 4. Rows 1, 3, and 5 contain synthesized views corresponding to viewpoints to the left of, in between, and to the right of the original viewpoints respectively. The distance between adjacent viewpoints is half the baseline. In the left column, holes corresponding to previously invisible points are shown in black; in the right column these holes have been filled in from adjacent regions. As expected the center view has many fewer holes than the two extreme views, since in the center view most scene points are visible from at least one of the original views.

The bottom left image shows the computed disparities with respect to the left original image. The black areas correspond to unmatched regions, which are caused by either an inconsistency between the two disparity maps, indicating a half-occluded region, or by insufficient evidence for a clear match. The bottom right image shows the same disparity map after the disparities in the unmatched regions have been estimated using the constant disparity assumption described in Section 4.2. An intensity edge map has been overlaid to aid judging the quality of the stereo data.

It is easier to evaluate the performance of the method when the synthesized views are displayed in an animated movie sequence. We have found that simple synthesized movies can communicate an impressively high amount of scene structure, even if the underlying disparity map is of low quality. This clearly demonstrates the potential of view synthesis from stereo data for simple virtual reality applications. Most of the visual artifacts created by our current implementation are caused by incorrect stereo data, in particular by occlusion boundaries that were recovered incorrectly. Mismatched points due to uniform intensities, on the other hand, usually do not cause problems.

[4] Provided by Stephen Intille, MIT Media Lab.

6 Image-based scene representations

Synthesizing new views from a stereo pair can be seen as part of a bigger framework, in which a scene is represented by a graph consisting of images and correspondence maps [1, 4]. Each vertex in this graph represents a panoramic view from a physical location in the scene, which can be stored implicitly by a set of images and their viewing angles, or explicitly by mosaicing the images into a single composite image [7, 9, 13]. The edges in the graph are dense disparity maps between adjacent views computed by a stereo algorithm. This graph constitutes a local view-based representation of the scene geometry, and new views can be generated efficiently from a small number of nearby views using the techniques discussed above. We argue that if the sampling of reference images is reasonably dense, the instabilities of the image-based method have a relatively small effect, since we only need to deal with small changes in viewpoints.

Using only a small number of local images for view synthesis has the additional advantage that we only need to know the relative configurations between adjacent views, which do not need to be globally consistent. For example, images could be acquired with a hand-held camera and be labeled with rough global coordinates. Then, for each pair of adjacent images, the epipolar geometry could be recovered by self-calibration. Another advantage of using a small set of images over methods that combine image data from a wide range of viewing configurations is that common assumptions (such as Lambertian surfaces) are less easily violated.

7 Conclusion

We have proposed a new method for view synthesis from real images using stereo vision. In our approach, scene geometry is implicitly represented by correspondence maps acquired by stereo vision techniques.

A prime advantage of using stereo for view synthesis is that a rectified disparity map yields a simple and fast way of generating new views based on local image warping. A disadvantage is the limited available information about scene geometry, requiring strategies to deal with points of unknown geometry or intensity due to occlusion. We have proposed possible ways of dealing with both problems. We have also discussed the requirements on stereo algorithms imposed by the application of view synthesis, and have seen that some traditional problems, such as the aperture problem, have less weight in view synthesis.

Our experiments demonstrate that it is possible to efficiently synthesize realistic new views even from inaccurate and incomplete depth information, thus meeting our goal of creating convincing impressions of three-dimensional structure.

Acknowledgements

Thanks to Stephen Intille for providing the KIDS images, and to Amy Briggs, Dan Huttenlocher and Ramin Zabih for helpful comments and many fruitful discussions.

References

[1] S. E. Chen and L. Williams. View interpolation for image synthesis. In *SIGGRAPH*, pages 279–288, 1993.

[2] R. Deriche, Z. Zhang, Q.-T. Luong, and O. Faugeras. Robust recovery of the epipolar geometry for an uncalibrated stereo rig. In *Europ. Conf. on Comp. Vision*, volume 1, pages 567–576, May 1994.

[3] U. R. Dhond and J. K. Aggarwal. Structure from stereo—a review. *IEEE Trans. on Systems, Man, and Cybern.*, 19(6):1489–1510, Nov./Dec. 1989.

[4] H. Fuchs et al. Virtual space teleconferencing using a sea of cameras. In *Intl. Symposium on Medical Robotics and Computer Assisted Surgery*, Sept. 1994.

[5] T. Kanade, P. J. Narayanan, and P. W. Rander. Virtualized reality: Concepts and early results. In *IEEE Workshop on Repres. of Visual Scenes*, pages 69–76, June 1995.

[6] A. Katayama, K. Tanaka, T. Oshino, and H. Tamura. A viewpoint dependent stereoscopic display using interpolation of multi-viewpoint images. *SPIE Stereoscopic Displays and Virtual Reality Systems II*, 2409:11–20, Feb. 1995.

[7] R. Kumar, P. Anandan, M. Irani, J. Bergen, and K. Hanna. Representation of scenes from collections of images. In *IEEE Workshop on Repres. of Visual Scenes*, pages 10–17, June 1995.

[8] S. Laveau and O. Faugeras. 3-D scene representation as a collection of images. In *Intl. Conf. on Pattern Recog.*, pages 689–691, Oct. 1994. INRIA Technical Report No. 2205.

[9] L. McMillan and G. Bishop. Plenoptic modeling: An image-based rendering system. In *SIGGRAPH*, pages 39–46, Aug. 1995.

[10] M. Ott, J. Lewis, and I. J. Cox. Teleconferencing eye contact using a virtual camera. In *INTERCHI '93*, pages 109–110, 1993.

[11] D. Scharstein. Matching images by comparing their gradient fields. In *Intl. Conf. on Pattern Recog.*, volume 1, pages 572–575, Oct. 1994.

[12] S. Seitz and C. Dyer. Physically-valid view synthesis by image interpolation. In *IEEE Workshop on Repres. of Visual Scenes*, pages 18–25, June 1995.

[13] R. Szeliski and S. B. Kang. Direct methods for visual scene reconstruction. In *IEEE Workshop on Repres. of Visual Scenes*, pages 26–33, June 1995.

[14] T. Werner, R. Hersch, and V. Hlaváč. Rendering real-world objects using view interpolation. In *Intl. Conf. on Comp. Vision*, pages 957–962, June 1995.

[15] G. Wolberg. *Digital Image Warping*. IEEE Computer Society Press, Los Alamitos, California, 1990.

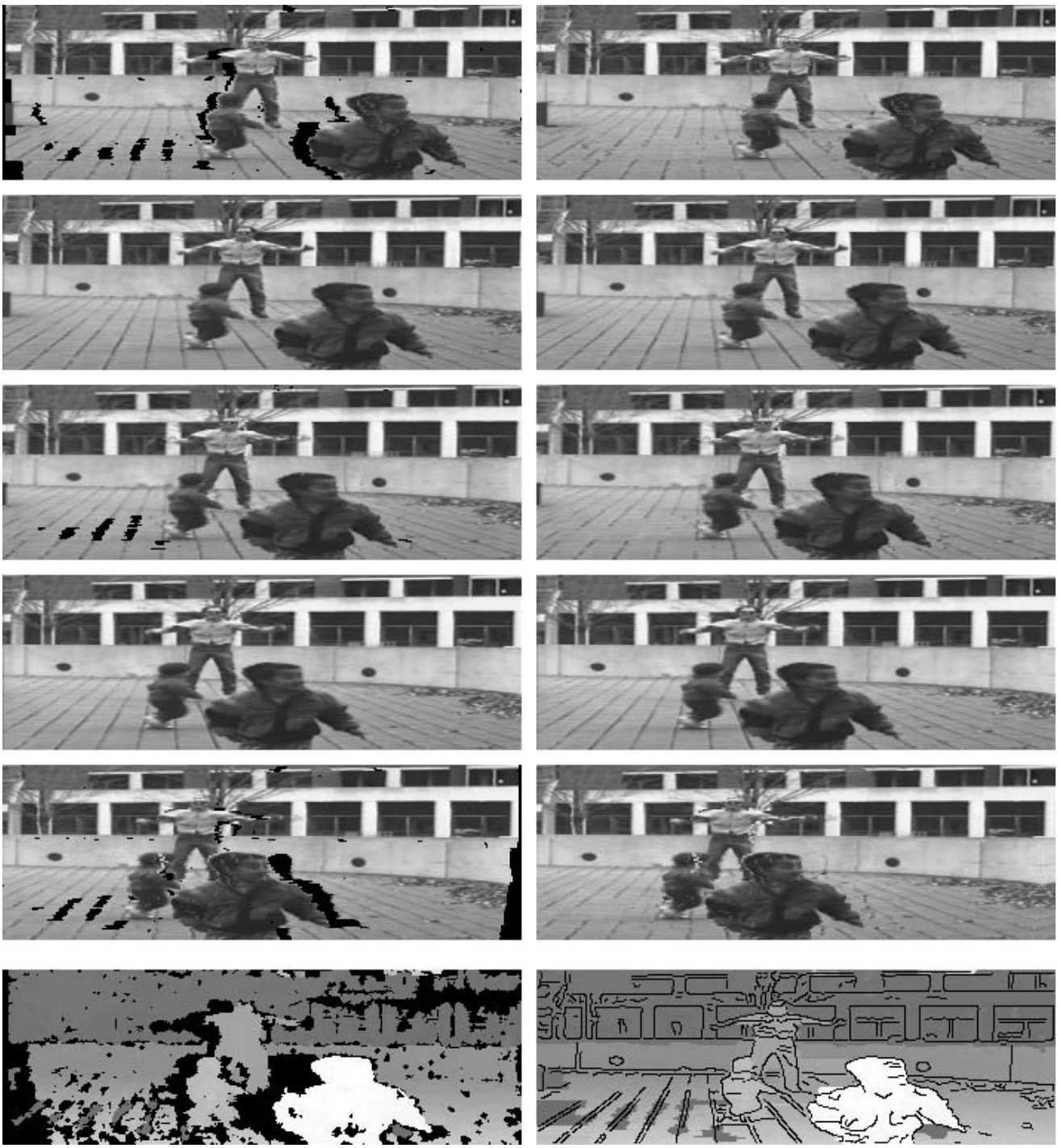

Figure 3: Sythesized views and disparities for the KIDS images. Rows 2 and 4 contain the original images. Rows 1, 3, and 5 contain synthesized views from different positions along the baseline. In the left column, the holes are shown in black, on the right they have been filled in. The bottom row shows the computed disparities with unmatched points (left), and the extrapolated disparities (right).