View Generation for Three-Dimensional Scenes from Video Sequences

Nelson L. Chang, *Student Member, IEEE*, and Avideh Zakhor *Member, IEEE*

***Abstract*—This paper focuses on the representation and view generation of three-dimensional (3-D) scenes. In contrast to existing methods that construct a full 3-D model or those that exploit geometric invariants, our representation consists of dense depth maps at several preselected viewpoints from an image sequence. Furthermore, instead of using multiple calibrated stationary cameras or range scanners, we derive our depth maps from image sequences captured by an uncalibrated camera with only approximately known motion. We propose an adaptive matching algorithm that assigns various confidence levels to different regions in the depth maps. Nonuniform bicubic spline interpolation is then used to fill in low confidence regions in the depth maps. Once the depth maps are computed at preselected viewpoints, the intensity and depth at these locations are used to reconstruct arbitrary views of the 3-D scene. Specifically, the depth maps are regarded as vertices of a deformable 2-D mesh, which are transformed in 3-D, projected to 2-D, and rendered to generate the desired view. Experimental results are presented to verify our approach.**

I. Introduction

In LIGHT of recent advances in technology, virtual environments have become an important tool in engineering, design, manufacturing, and many other areas. Especially important to the development of this growing field is the problem of arbitrary view generation (AVG), in which a novel view of a three-dimensional (3-D) scene is generated from its neighboring views.

Existing work in this area can be placed into three classes. In the first class, a full 3-D model of the scene is constructed by volumetric intersection and then reprojected in order to generate the desired view [11], [1], [4], [17]. The main difficulty with this approach is that of registering and combining the two-dimensional (2-D) information to generate a full 3-D model.

In the second class, views are generated by direct methods, without having to estimate structure directly. In [35] and [27], image mosaics are constructed by registering and reducing the set of input images into a single, larger resolution frame. This frame then serves as the representation of the scene. While this representation is useful for capturing the information generated by panning a given environment, it is difficult to generate an arbitrary view with this approach, since the necessary depth or structure information has not been estimated. Other researchers have considered exploiting certain invariants in the geometry of the problem [38], [33], [25]. This approach, however, correctly reconstructs only those points that lie in the intersection of the given views and not points that become uncovered. In the case of [24], despite being able to generate novel views from a set of reference views, one requires dense correspondences and the epipolar geometry to be known *a priori*. In addition, these kinds of approaches would require a considerable amount of computation to continuously update a user-defined viewpoint for a real-time application.

The third class of AVG algorithms attempts to deal with occluded/uncovered regions in the scene better than the second class while not resorting to a full 3-D representation of the first class. Generally, a set of depth surfaces is first estimated and then combined to generate the desired view. For example, Chen and Williams [10] measure range information and camera transformation to establish pixel correspondence and then apply morphing techniques to interpolate intermediate views. Similarly, Skerjanc and Liu [34] compute depth with known camera positions in order to synthesize intermediate pictures. Kanade *et al.* [22] estimate depth using a camera set-up with known camera geometry from which they estimate depth and generate new views.

In this paper, we address the problem of representing a static scene from a given image sequence and reconstructing the view from an arbitrary viewpoint. Our approach to AVG falls into the third category [7], [5], [6]. However, unlike existing techniques, we use a sequence of images captured by a hand-held, uncalibrated camcorder with translational motion confined to the x-y plane. We will assume the motion is primarily horizontal with some possible fluctuations in the vertical direction. Uncalibrated cameras with unknown position are used to avoid the difficult and time-consuming step of calibration, thereby increasing the flexibility of the image acquisition process. Our motivation for using a sequence of video images rather than a few still images is to improve the robustness of the depth estimation step. Wide availability of video cameras in today's research and commercial environment justifies their use in place of still cameras in many applications.

Our proposed approach consists of translating a camcorder by hand across several trajectories, including at different elevations, around an object in the scene to generate image sequences used to construct the depth maps. This simple

Manuscript received September 1, 1995; revised October 25, 1996. This work was supported by an Air Force Laboratory Graduate Fellowship PYI-NSF Grant MIP-9057466, an ONR Young Investigator Award N00014-92-J-1732, and Sun Microsystems. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Sarah Rajala.

The authors are with the Video and Image Processing Laboratory, Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA 94720 USA (e-mail: avz@eecs.berkeley.edu; http://www-video.eecs.berkeley.edu).

Publisher Item Identifier S 1057-7149(97)02463-9.

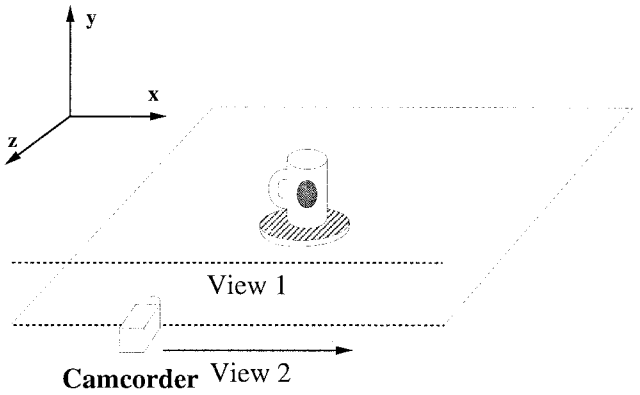

Fig. 1. Experimental set-up used to generate results.

imaging geometry is shown in Fig. 1. The motivation for not choosing rotation, or a combination of rotational and translational motion, is the sensitivity of depth reconstruction to these classes of motion, especially when the motion parameters are unknown. In addition, it is well known that depth reconstruction can be more accurate when the camera translates across an object, rather than when the camera translates toward or away from it. The idea is to estimate depth only at several prespecified locations, called "reference frames," by using their neighboring captured frames. Once the depth has been computed at reference frames, the neighboring intensity frames are discarded, and solely the depth and intensity at reference frames are kept as a compact representation of the scene. The motivation for compactness stems from the desire to download only that information necessary for telepresence applications. This representation is then used to reconstruct arbitrary views located on or off the scanning trajectories.

The outline of the paper is as follows. In Section II, we discuss an adaptive approach to dense depth estimation. Section III describes the reconstruction algorithm used to generate the desired view from the representation. Results from real-world scenes are presented in Section IV. The paper concludes with a discussion in Section V.

II. Compact Representation

Our overall approach in deriving the depth information at reference locations is to establish correspondence between the reference frame[1] and each of its neighboring frames. The resulting disparity maps at the reference frames are normalized and combined in order to form a depth map for the reference frame. Once completed, the neighboring frames are discarded in the reconstruction process; therefore, their use affects only the quality of the representation and not its compactness. In the remainder of this section, each step will be discussed in detail.

A. Depth Estimation

In the first step of the representation process, local dense depth maps are generated by matching the reference frame and each neighboring frame. There are many approaches to accomplish this task. Some approaches fall under the classification of optical flow, e.g., see [19], [23], and [2]. The results provide a dense flow field and are generally acceptable. However, many of these algorithms work for only small motions and do not perform well across discontinuities without assuming local similarity.

A second class of approaches consist of stereo matching algorithms. With stereo algorithms [13], it is generally assumed that either camera positions or camera motion is known *a priori*. Typically, some additional information is furnished to aid in matching, such as uniqueness and disparity constraints for random dot stereograms [28], a third view [20] or even more views [32], shading information [15], or different filtered outputs [21].

Other approaches are classified as solving the structure-from-motion (SFM) problem [29], [36], [41], [37]. For these algorithms, a set of features, e.g., edges in [37] and corners in [41], are identified and tracked. The motion of the camera and the structure of these features are then computed simultaneously. Despite the complexity of solving this nonlinear optimization problem under perspective projection, the SFM algorithms perform reasonably well given two or more arbitrary views. However, many times they are practical in a computational sense for only a small number of points in the scene. Moreover, many of these algorithms require point or feature correspondences in advance.

Our approach is similar to the above approaches for estimating depth whereby the l_2 norm of intensity error is minimized over possible depth values. However, unlike the approaches that produce depth for a sparse set of points, we recover dense depth information as required by the problem of AVG. Since we have confined the motion to be planar, the depth estimation problem reduces to a one-dimensional (1-D) correspondence matching problem [17]. In this case, the epipolar lines of the two images are parallel and may be found using the algorithm described in [42]. For every pair of matches, (i,j) and (i_2,j_2), the depth z of the corresponding scene point is related to disparity[2] $d(i,j)$ by

$$d(i,j)=\sqrt{(i-i_2)^2+(j-j_2)^2}=\frac{f}{z}b \tag{1}$$

where f is the focal length and b is the baseline distance between the two images' coordinate systems. Hence, the depth may be estimated as the inverse of disparity $d(i,j)=\sqrt{m^2+n^2}$, obtained by

$$\min_{d(i,j)\in L}\left\{\sum_{(x,y)\in}\sum_{B(i,j)}|I_1(x,y)-I_2(x+m,y+n)|^2\right\} \tag{2}$$

where $I_1(\cdot,\cdot)$ and $I_2(\cdot,\cdot)$ are the two images, respectively, m and n represent the motion vector, L is the appropriate epipolar line, and $B(i,j)$ is the $B_1\times B_2$ region of intensities under consideration, not necessarily centered at (i,j).

[1] We shall assume the reference frames have been previously selected. The problem of choosing reference frames from the video sequences is an important issue but is beyond the scope of this paper.

[2] The terms "disparity" and "baseline" are typically defined with respect to horizontal motion only. However, we shall use them to describe the norms in the direction of the epipolar lines and motion. Alternatively, we may consider rectifying the initial images so that the epipolar lines are parallel with the scan lines.

There are some artifacts inherent both in the algorithm and the problem itself that induce incorrect disparities for certain regions [6]. In what follows, we describe four of the most important artifacts and explain techniques of minimizing their effects.

1) Artifacts: To begin with, if the relative motion between two images is translational in the $x-y$plane, then an artifact known as aperture ambiguity occurs for edges oriented parallel to the epipolar lines. It arises because the block B used for matching is too small and does not include enough distinct features when matching. A similar artifact occurs in regions of nearly constant intensity. Note that in both cases, the matching equation (2) is a shallow function over all possible disparities; the disparities are almost all equally good. The minimization depends largely on the actual intensity values, which may be noisy due to the imaging process and different lighting conditions. Despite the lack of distinct features, the matching algorithm may still lead to the correct disparity for horizontal edges and low textured regions.

In contrast, there are other artifacts of intensity-based matching algorithms which almost always produce the wrong disparity—these occur in occluded regions and near depth discontinuities. An occluded region is an area that appears in one image but not in the other. For instance, a moving object in the scene generally occludes some points and uncovers other points from view. In such regions, the matching algorithm blindly attempts to find the best match but fails miserably because only one image has information about the region.

Incorrect disparity information is also generated near depth discontinuities. It is difficult to identify depth discontinuities of a scene beforehand, since the ultimate goal is to estimate depth. Intensity discontinuities are often considered instead because it is not uncommon for depth discontinuities in the scene to be related to intensity discontinuities in the image. For points near object boundaries but not part of the object, the search block B is large enough to include some features of the object. In minimizing the intensity error for such a point, the matching algorithm yields a motion vector similar to the motion of the object itself. The end result is poor localization of the object boundary in the disparity domain by $B/2$ pixels, i.e., the object seems to have expanded in all dimensions. Clearly, the localization of depth discontinuities depends on the size of the block used for matching—the smaller the block, the better the localization. However, it is widely known that using blocks that are too small produces many false matches, since intensity patterns will be less distinctive [17].

An example of all four artifacts is shown in Fig. 2. The two images shown are related by horizontal translational motion, i.e., the two optical axes are parallel to each other and perpendicular to the direction of motion and the epipolar lines are coincident with the scan lines. The object is a rectangle of constant grey while the background is entirely white. If points in Image 1 are matched with those in Image 2, the aforementioned problems will lead to incorrect disparity estimates. Mismatches at horizontal line segments identified as Fig. 2(a) are due to aperture ambiguity. Constant intensity ambiguity occurs in both the foreground and background as with the point indicated by Fig. 2(b). Little information may

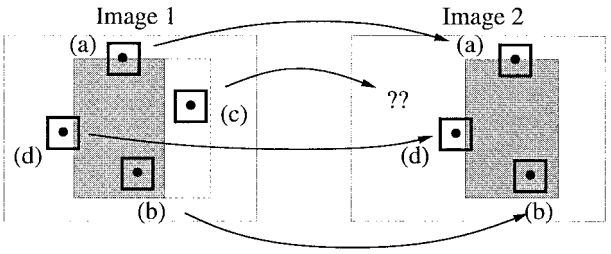

Fig. 2. Example of regions where matching fails due to (a) aperture ambiguity; (b) constant intensity ambiguity; (c) occlusion; and (d) depth discontinuity localization ambiguity.

be obtained in occluded regions like those in Fig. 2(c). As shown in Fig. 2(d), localizing depth discontinuities may also pose a problem.

It is straightforward to identify most of these artifacts and subsequently assign confidence levels to different regions in the scene. These confidence levels are important for locating the regions to ignore when combining multiple depth maps together. To detect aperture ambiguity (AP), a gradient-based edge detector [26] is used to locate the horizontal edges.[3] Points in the image near these edge pixels are marked as possibly spurious. To identify constant intensity regions (CONST), a small window is used to find regions where the intensity variance is below a prespecified threshold. A low variance suggests that the block consists of little texture and nearly constant intensity. Matching the images in both directions helps to identify occluded regions and inconsistent matches [15], [40]. Occluded regions (OCCL) are precisely the unmatched points in the images, whereas inconsistent matches (INCONS) may be found by validating matches in both directions. In the end, the scene will consist of low confidence regions marked according to the different artifacts: constant intensity, aperture ambiguity, occlusion, and inconsistencies in matching.

2) Adaptive Matching Scheme: Since many real world scenes consist largely of low textured regions, the above matching algorithm will produce a high percentage of low confidence regions due to constant intensity. To avoid too sparse a depth map, we attempt to improve estimates in these regions by proposing an adaptive matching approach [5]. The approach consists of essentially dividing the image into CONST and non-CONST regions and finding the best matches for both regions. To match images I_1 and I_2, all AP points in both images are located first using edge detection. We can avoid performing any matching for these points since they are likely to be wrong and we can incorporate information from other matches as described later. Once the AP points have been excluded, we identify CONST regions by low-variance thresholding (LVT) with a 3×3 block.[4] Once found, the

[3] Since the relative motion between the images is primarily horizontal translation, we need to worry about only horizontal edges in the scene. If the two images to be matched were related by a vertical translation, then we would require the edge detector to locate all vertical edges.

[4] Generally, a small block size is preferred, since textured regions near or along intensity discontinuities will be better localized. This tends to improve the localization of depth discontinuities since many times intensity edges are related to depth ones. We note that depth discontinuities can be much better localized by human interaction as done in [22].

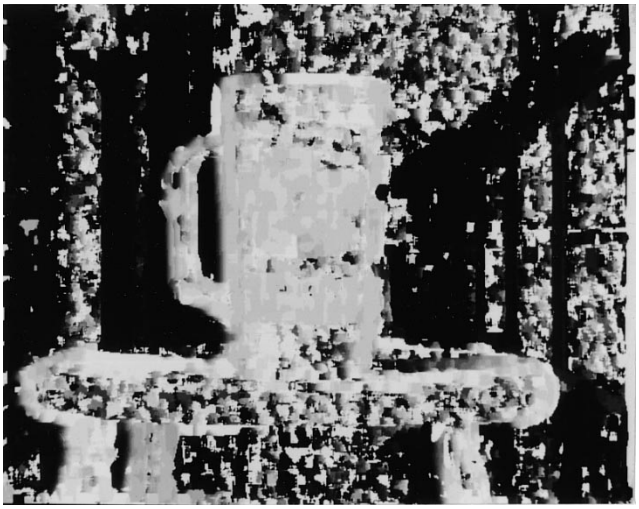

Fig. 3. Example of disparity map using fixed block size 9 × 9.

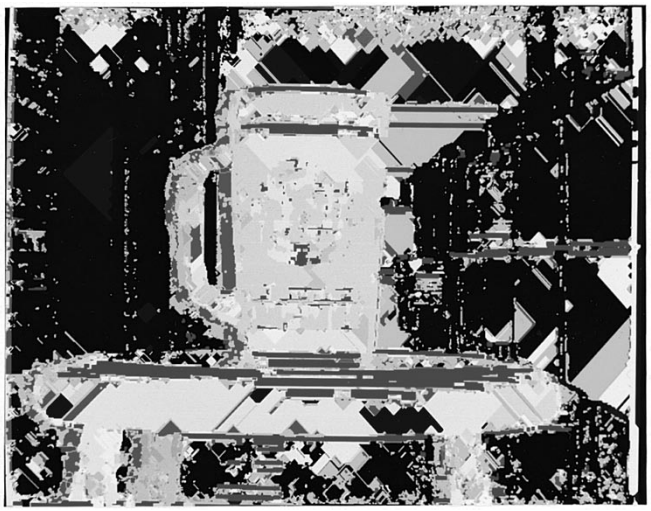

Fig. 4. Example of disparity map using adaptive block size. Legend: blue, CONST; red, AP; yellow, INCONS; green, OCCL.

non-CONST points are then matched using (2) for I_2 with respect to I_1 and also I_1 with respect to I_2.

With the non-CONST points matched, the next step is to find the best match for each CONST point (i,j). Since the main ambiguity stems from using a block that is too small, we consider instead using the largest rectangular block containing the point (i,j) that consists entirely of CONST points. Note that the block does not have to be centered at (i,j). One way to find such a block is by growing a 3 × 3 block around (i,j) and then extending each side evenly until the extension encounters at least one non-CONST point or until some prespecified dimension maximum, i.e., block size limit, has been reached. In this way, the algorithm utilizes the shape and relative size of the CONST region without including too many features that may mislead the algorithm.

Once every point has been classified, the algorithm determines the occluded points (OCCL) and inconsistent ones (INCONS) as described above. Notice that both the CONST and non-CONST points alike could be reassigned to OCCL or INCONS, depending on the outcome of matching.

Because the block size is not fixed and actually adapts to the confidence region, this adaptive scheme overcomes the well-known trade-off between good boundary localization with a small window and improved matching in low textured regions with a large window. The final result consists of fairly dense and reasonably accurate disparities. Consider an example of matching between two images, frames 37 and 34 of the Mug2 sequence described in Section IV; frame 37 is shown in Fig. 19. Fig. 3 shows the resulting disparity map using a fixed 9 × 9 block size. While the mug and stool are somewhat discernible, there are a large number of artifacts throughout the scene due primarily to the many regions of constant intensity. In contrast, Fig. 4 shows an improved disparity map obtained using an adaptive block size with various low confidence regions marked accordingly. These low confidence regions will be dealt with in the upcoming sections.

B. Normalization of Initial Estimates

The depth maps from the previous stage need to be normalized so that they are all related by the same scaling factor. For this task, we propose to estimate the translation parameter between maps and scale by the reciprocal. A point (u_i, v_i) in one image and $(u1_i, v1_i)$ in a translated second image are related by the disparity equation

$$d_{1,i} \triangleq \sqrt{(u1_i - u_i)^2 + (v1_i - v_i)^2} = \frac{f}{z_i} b_1 \tag{3}$$

where b_1 is the translation parameter relating the two images. If a third image is introduced, one yields a similar equation

$$d_{m,i} \triangleq \sqrt{(um_i - u_i)^2 + (vm_i - v_i)^2} = \frac{f}{z_i} b_m \tag{4}$$

with b_m the translation parameter linking the first and third images; Fig. 5 shows this relationship. Note that the depth z_i is the same in both cases, since all three image points correspond to the same physical point. Combining (3) and (4) leads to the following relation:

$$d_{1,i} \frac{b_m}{b_1} = d_{m,i}. \tag{5}$$

Suppose now we consider K high confidence disparity points common to the two depth maps. For each point i, (5) holds, thus leading to the matrix equation

$$\underbrace{\begin{bmatrix} d_{1,1} \\ d_{1,2} \\ \vdots \\ d_{1,K} \end{bmatrix}}_{A} \frac{b_m}{b_1} = \underbrace{\begin{bmatrix} d_{m,1} \\ d_{m,2} \\ \vdots \\ d_{m,K} \end{bmatrix}}_{y}. \tag{6}$$

By linear least squares, we may solve (6) for the ratio b_m/b_1 to get

$$\frac{b_m}{b_1} = (A^T A)^{-1} A^T y = \frac{\sum_{i=1}^{K} (d_{1,i})(d_{m,i})}{\sum_{i=1}^{K} (d_{1,i})^2}. \tag{7}$$

If b_1 is assumed to be one, then b_m is precisely the scaling factor by which we need to adjust the mth depth map. In this way, each of the depth maps can be normalized with respect to the same scale factor.

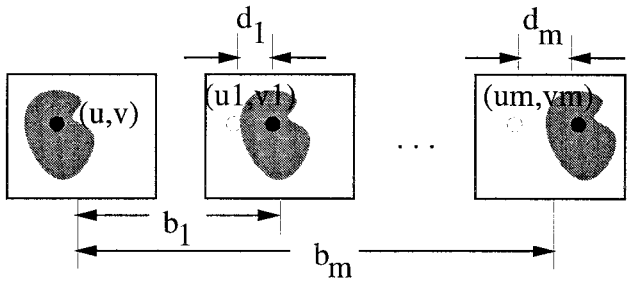

Fig. 5. Exploiting geometry of camera set up to normalize depth maps.

Since we are not confident about every disparity estimate, an iterative process may be used to improve the estimate by reducing the error $||Ab_m - y||_2$ to some desired amount. During every pass, outlier points greater than a given error percentage are disregarded when computing b_m. The procedure converges when the number of points does not change between iterations. This modification helps to further improve the accuracy of the scaling factor. In our experiments, we use a generous error of 30% since the vector y consists of possibly erroneous data. The algorithm typically converges in only three iterations.

C. Combination of Multiple Depth Maps

Once all the depth maps have been normalized to a common scaling factor, they are combined to form a single depth map for a particular reference frame. Since each local depth map may consist of low confidence areas and incorrect depth data, the combination process should retain only the information which seems consistent; otherwise, it should regard the information as invalid.

Let $D_i(\cdot,\cdot)$ for $i = 1,2,\ldots N$ denote the N normalized depth maps and let $D(\cdot,\cdot)$ represent the combined result. For every point (x,y), we may regard the problem as an estimation problem, i.e., given n votes for $D(\cdot,\cdot)$, determine the most accurate value. An iterative procedure is used to analyze the statistics of the given data, throw out outliers, and reduce the data set to a more consistent one.

Because of the predominantly bimodal distribution of the data, i.e., foreground and background points, we consider using the median instead of mean to throw out outliers [5]. Generally, the depth associated with the cluster consisting of the majority of points is reasonably correct. We found that when dealing with bimodal distributed data, outlier identification was significantly improved by using the median rather than using the mean. The effect is that one cluster of the bimodal distribution of depths is discarded; the underlying majority in depth wins. As an example, consider the set of depths $\{0.1, 0.2, 0.3, 1, 1.3\}$. The mean m is 0.58, the standard deviation σ is 0.4792, and the median v is 0.3. A general practice is to throw out outliers that lie outside the $m \pm \sigma$; in this case, the range is $[0.1008, 1.0592]$ and, hence, both 0.1 and 1.3 are discarded. If we instead consider $v \pm \sigma$, the range becomes $[-0.1792, 0.7792]$ and only the foreground points $\{0.1, 0.2, 0.3\}$ remain.

As discussed before, depth information from horizontal matches contains artifacts along horizontal edges due to horizontal aperture ambiguity. If only these depth maps are used in combination, then there will be considerable problems in AP regions. To circumvent the problems, we propose including information derived by matching a vertically related pair of images, that is, using corresponding images from two linear trajectories at different vertical elevations.[5] If the second image with respect to the reference frame is a perfect vertical translation, then solving correspondence leads directly to an estimate of depth. Observe that the depth map will have vertical aperture ambiguity and will contain occluded regions generally not coincident with those found in the horizontal matches. Hence, this information may be incorporated in the combining stage to improve the accuracy of the depth map in AP regions.

The algorithm may be further refined by introducing the notion of weights to the depth data. At every stage in the representation process, confidence levels are assigned based on the validity of the data. It is thus quite intuitive to weight points in the combination stage based on the confidence levels. For example, along horizontal edges, more weight is placed on the vertical information, since it is more reliable here than information from horizontal matches. Lower confidence AP, OCCL and INCONS points are not included during combination whereas CONST points are considered since they are seemingly correctable. The depth $D(x,y)$ is then given as the weighted average $\sum_k w_k(x,y) D_k(x,y)$ with

$$w_k(x,y) = \begin{cases} 1.0 & (x,y) \text{ high confidence} \\ 1.0 & (x,y) \text{ CONST}, k \text{ vertical match} \\ 0.3 & (x,y) \text{ CONST}, k \text{ horizontal match} \\ 0 & \text{otherwise.} \end{cases} \quad (8)$$

Fig. 6 provides an example of combining several disparity maps together as described. The map has been quantized to 256 levels, where brighter intensity level represents a larger disparity. The disparity map is then converted to a depth map by inverting each disparity pointwise. The depth map is a more accurate estimate of the given scene as compared with the disparity map in Fig. 4. The regions in the combined depth map which may be inaccurate are marked in yellow to indicate low confidence.

D. Cubic B-Spline Approximation

The depth map after the combination stage is fairly accurate in many regions. There are however a considerable number of low confidence regions. To fill in these regions and to make the map much denser while not sacrificing too much accuracy, nonuniform cubic B-splines are used [12], [3], [5]. Every depth point in low confidence regions is interpolated by its neighboring high confidence depth vertices along the same row or column, depending on the variance of these vertices. The depth surface is treated as a tensor product, i.e., the product of 1-D functions, so the data may be processed first along one direction and then along the other, which helps to simplify computations. We may apply this spline technique to Fig. 6 to obtain the final depth map shown in Fig. 20.

Once the depth map for each reference frame has undergone spline approximation, we are left with depth estimates at

[5] Note that other images may be considered as well, including those obtained by arbitrary translational motion in the x-y plane.

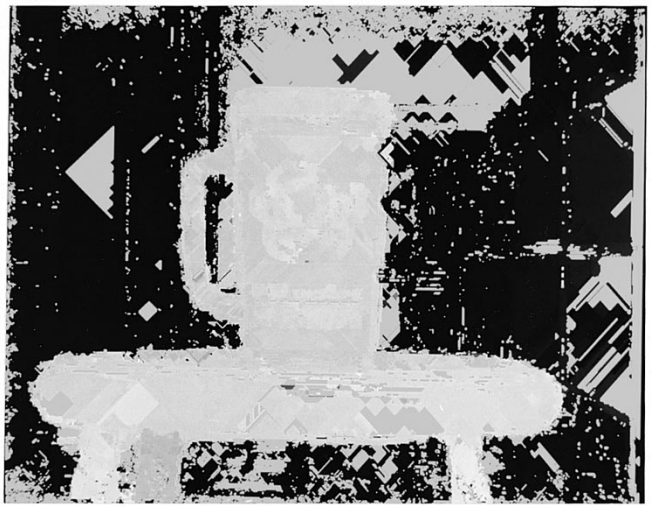

Fig. 6. Example of combined depth map. Legend: yellow, low confidence.

different locations around the scene. The final step in the representation process is to estimate the relative camera motion between reference frames using an approach like [36]. Once the relative motion between all reference frames is known, a geometric relationship may be constructed among the different reference frames. This enables us to select the reference frames needed to use in the reconstruction stage.

In the end, the representation of the scene consists of the intensity-depth pair at each reference location along with the relative motion among reference frames. Once these data have been derived, they may be stored in a database for later reconstruction.

III. Reconstruction of Views

Once we have generated the representation for a particular 3-D scene, we may choose to reconstruct the view of the scene at some specified viewpoint. Assume that the center of one reference frame coincides with the origin of the coordinate system and that the desired viewpoint is known with respect to this origin. The reconstruction algorithm consists of the following: First, the appropriate reference frames are chosen. Then initial estimates of the desired view are constructed by applying motion parameters to each reference frame. Finally, the estimates are combined into a single image, interpolating when necessary.

A. Selection of Appropriate Reference Frame(s)

Given the relative position and orientation of the desired view, it should be a straightforward task to determine which reference frames to use. One way of deciding is to include those frames with the smallest motion in norm relative to the view. This measurement may be used to determine the amount a particular reference frame contributes to the view estimate. Intuitively, the reference frame corresponding to the smallest motion in norm should be weighted the most, and vice versa. For example, suppose the representation consists of three reference frames lying in a plane, as shown in Fig. 7, and the desired view is at the location marked with an "×," a distance

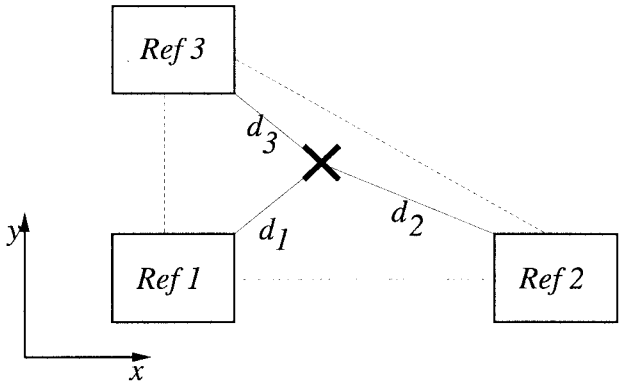

Fig. 7. Example of computing weights for three reference frames.

d_i away from the ith reference frame. Then, the first reference frame should contribute only $1/d_1$ out of $1/d_1+1/d_2+1/d_3$, or $\frac{d_2 d_3}{d_1 d_2+d_2 d_3+d_3 d_1}$ and likewise for the other reference frames. In general for N reference frames, the weight γ_i assigned to the ith reference is given by

$$\gamma_i = \frac{\prod_{j\neq i} d_j}{\sum_{k=1}^{N}(\prod_{j\neq k} d_j)}. \tag{9}$$

If $\hat{I}^{(k)}$ represents the view estimate from reference frame k, then the desired view $\hat{I}$ can be represented as the weighted average of view estimates, namely $\hat{I} = \sum_{k=1}^{N} \gamma_k \hat{I}^{(k)}$. This equation may be applied to most points; however, more detail will be seen in Section III-C.

Another consideration is the number of reference frames. If the specified view is very close to one of the reference frames, then we may choose to use only that single frame. However, at least two reference frames are needed to properly reconstruct the desired view to reduce noise and to recover occluded regions in the scene. Additional reference frames help to reduce noise further at the cost of requiring more precise registration among the frames.

B. Generation of View Estimates

In this section, we will describe our approach to generating a view estimate from one intensity-depth reference pair. In Section III-C, we will describe how estimates from multiple reference frames are combined.

A possible approach to view generation is to regard the points (u, v) in the reference frame as discrete independent points, since neither the image nor the depth map is a continuous surface. However, if we consider transforming only this set of points to generate the view estimate, the resulting image may exhibit inconsistencies in the ordering of foreground and background points [5], [6].

A better approach is to consider the points of the reference frame arrays as vertices of a deformable 2-D wire mesh. Neighboring points in the reference frame are viewed as connected to one another to form a meshlike structure consisting of quadrilateral patches. Specifically, every set of four vertices $\{p_1, p_2, p_3, p_4\} = \{(u, v), (u+1, v), (u+1, v+1), (u, v+1)\}$, with the corresponding depth and intensity information, constitute the corners of a single patch in space. Notice that the order

of the four points is important for orientation; we consider the patch to have clockwise orientation starting with the upper left corner. We also consider the patch to have two sides, an outer one whose intensities can seen by the camera and an inner one whose intensity information is unknown. An alternative view is that the upper side of the patch has a surface normal given by the left-hand rule. To transform the 2-D mesh into 3-D, we consider the following mapping: A point (u, v) with depth z is mapped to the 3-D point $(X, Y, Z)' = (\frac{(u-x_c)z}{f}, \frac{(v-y_c)z}{f}, \zeta z)'$, where (x_c, y_c) is the center of the image plane, f is the focal length, and ζ is a scaling factor to adjust for the field of view. A view estimate is then generated by transforming every patch in the reference frames into 3-D as described above, applying the appropriate motion parameters to the mesh, and finally reprojecting the mesh to construct its 2-D image through rendering; these steps are described in detail below.

Once all the points of the reference frame have been mapped into 3-D accordingly, they are then transformed according to the appropriate motion parameters. The notion of applying motion parameters to a frame has been addressed in conventional computer vision and robotics literature [18], [30]. Let $(X_1, Y_1, Z_1)'$ be a point in the scene and suppose the frame of reference undergoes a rigid transformation (R, T) given by $R = [r_{i,j}] \in \mathcal{R}^{3\times3}$ and $T = (\Delta x, \Delta y, \Delta z)'$ where both rotation R and translation T are in terms of the world coordinates. Then, in matrix form, the new scene coordinates $(X_2, Y_2, Z_2)'$ are given by

$$\begin{bmatrix} X_2 \\ Y_2 \\ Z_2 \end{bmatrix} = \begin{bmatrix} r_{1,1} & r_{1,2} & r_{1,3} \\ r_{2,1} & r_{2,2} & r_{2,3} \\ r_{3,1} & r_{3,2} & r_{3,3} \end{bmatrix} \begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \end{bmatrix} + \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix}. \quad (10)$$

The subsequent image coordinates (u_2, v_2) are then given by

$$u_2 = f\frac{X_2}{Z_2} = f\frac{r_{1,1}X_1 + r_{1,2}Y_1 + r_{1,3}Z_1 + \Delta x}{r_{3,1}X_1 + r_{3,2}Y_1 + r_{3,3}Z_1 + \Delta z} \quad (11)$$

$$v_2 = f\frac{Y_2}{Z_2} = f\frac{r_{2,1}X_1 + r_{2,2}Y_1 + r_{2,3}Z_1 + \Delta y}{r_{3,1}X_1 + r_{3,2}Y_1 + r_{3,3}Z_1 + \Delta z} \quad (12)$$

with f as the focal length.

At this point, the reference frame has been viewed as a single deformable mesh consisting of connected patches. However, if regions of the mesh are not grouped into foreground or background categories, transforming every patch in the mesh will lead to a potentially incorrect view estimate. As an example, consider Fig. 8. The rectangle is an object in the foreground with small depth that moves to the right in front of a uniform background of far away depth. Notice if we consider rendering the square patch drawn, whereby its two left points A and B have far away depths while its two right points C and D are near to the camera, the result will interpolate the depths and thereby consist of streaks in the view estimate.

The need to segment the image by depth is apparent for obtaining accurate results. One simple approach is to identify the depth discontinuities in the reference frame. Patches which fall along depth discontinuities should be discarded and not even be transformed since connecting regions of different depths may lead to an inaccurate image. To detect patches along depth discontinuities, we estimate the local variance

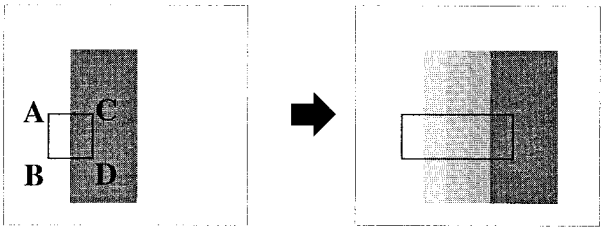

Fig. 8. Examples of invalid patches due to depth discontinuities.

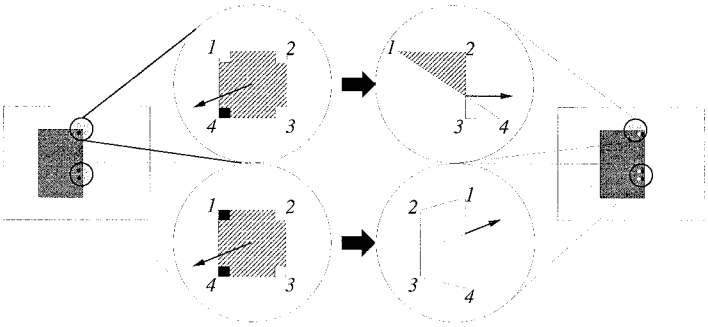

Fig. 9. Examples of invalid patches due to incorrect orientation.

with a 5×5 window on the depth maps and mark points whose variance is above a certain threshold [9]. This technique of searching for large depth variations is similar in nature to a crude intensity-based edge detection algorithm. Patches associated with a depth discontinuity are not rendered to avoid streaking. We note that the resulting edge map produces a rudimentary description of how to segment the given scene into foreground and background components.

After the patches in the reference frame have been transformed, it seems straightforward to render the new patches to generate an estimate of the view. However, not all patches need to be or ought to be rendered. More specifically, transformed patches that do not preserve orientation should not be rendered since they usually result from occlusion and cannot be seen. Consider an example of a rectangular object with small depth moving to the right in front of a uniform background with large depth. Fig. 9 shows two types of transformed patches whose orientation is not clockwise, namely twisted and flipped patches. The top patch consists of three background depths for Points 1, 2, and 3, and only one foreground depth for Point 4. Once transformed, the first three points remain in roughly the same relative position while Point 4 occurs to the right of Point 3, yielding a twisted patch. Similarly, the lower patch consists of two foreground depths at Points 1 and 4 and two background depths at Points 2 and 3. For an apparent motion to the right, Points 2 and 3 remain stationary while Points 1 and 4 move past them creating a flipped patch with counterclockwise orientation. Notice that in both cases, the transformed patches have a surface normal that is directed away from the camera, and hence their outer sides are barely, if at all, visible.

To determine whether the orientation of the candidate patch has been preserved, we consider the following. Assuming clockwise orientation as shown in Fig. 10, Point 1 (u_1, v_1) is

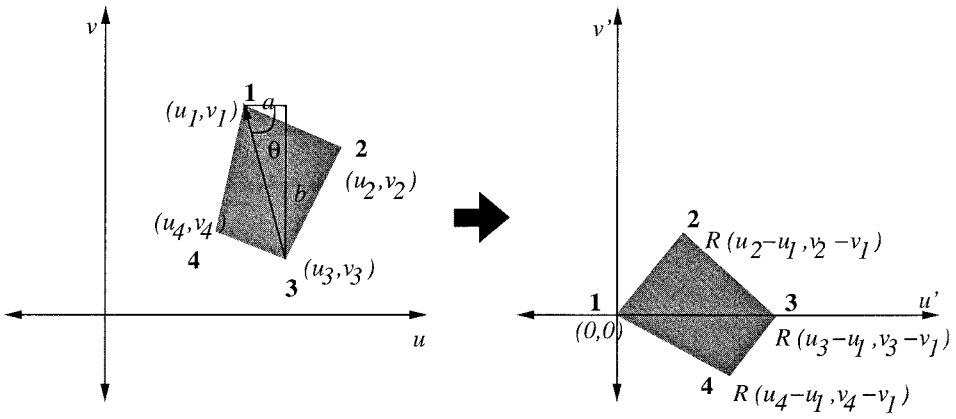

Fig. 10. Testing for valid patches.

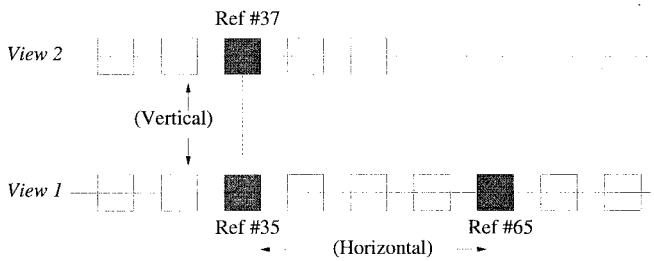

Fig. 11. Geometrical relationship among the reference frames for experiment.

translated to become the origin and the patch is rotated so that Point 3 (u_3, v_3) lies along the positive u-axis. A patch is said to preserve clockwise orientation if i) Point 2 has a positive v coordinate, and ii) Point 4 has a negative v coordinate. In particular, let $R \in \mathcal{R}^{2\times 2}$ be the rotation matrix to rotate Point 3 to the positive u-axis after Point 1 has been translated to the origin

$$R = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \tag{13}$$

where θ is the angle formed by the line connecting Point 1 and Point 3 with the u-axis. It can be easily shown that R is given by

$$R = \frac{1}{\sqrt{(u_3-u_1)^2+(v_3-v_1)^2}} \begin{bmatrix} u_3-u_1 & v_3-v_1 \\ -v_3+v_1 & u_3-u_1 \end{bmatrix}. \tag{14}$$

A point $p = (u, v)$ has the new coordinates $p' = (u', v') = R(p - p_1)$. To check for orientation, Points 2 and 4 are substituted into this expression. Since only the sign of the transformed v coordinate is important and since the root of any positive number is nonnegative, we may simplify the v coordinate to be

$$v' \approx -(v_3-v_1)(u-u_1)+(u_3-u_1)(v-v_1) \tag{15}$$

to check whether the two conditions are satisfied.

With the patches transformed, we are now in a position to generate the view estimate. The set of valid patches are projected onto the image plane and then rendered using a scan-line algorithm [14]. For each patch, the four edges are stored in memory and sorted in decreasing v coordinate. Starting at the smallest v coordinate, every scan line is filled in according to an intensity-based interpolation scheme. In addition, we employ a software z-buffering technique [16], [31] to determine the ordering of patches with respect to the

Fig. 12. Reference frame 35 (intensity) of Mug1.

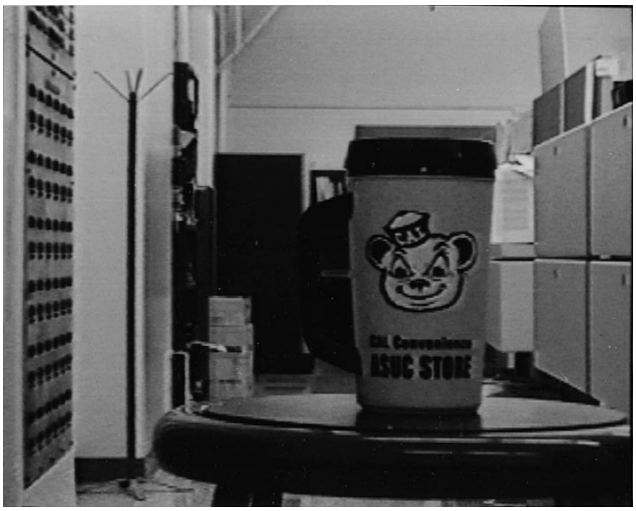

Fig. 13. Reference frame 65 (intensity) of Mug1.

Fig. 14. Reference frame 35 (depth) of Mug1 filled in by splines.

Fig. 15. Reference frame 65 (depth) of Mug1 filled in by splines.

smallest depth. Thus, a given pixel in the view estimate is assigned an intensity corresponding to the patch closest to the camera. This approach allows us to handle occlusions quite nicely when the foreground object obscures portions of the background scene. It is worth noting that a view estimate can be generated much faster on a platform with a specialized graphics library such as the HP Starbase Library.

The majority of pixels in the view estimate will have an associated intensity from this technique. However, it is possible for some pixels not to be assigned any intensity, leaving "holes" in the estimated image. Examples can be seen in Figs. 16 and 17, where the holes are marked in red. These holes arise because of two primary reasons. The first involves the accuracy of the depth maps. Since we estimate depth for every point in the reference frame and do not attempt to fit a surface through these data, it is quite likely that the depths are not completely consistent or smooth, and as a result, "cracks" in the view estimate may appear. In Figs. 16 and 17, cracks can be seen in the mug face and in front of the stool

In addition, holes also occur in areas of the view estimate that were previously unseen and become uncovered. Identifying depth discontinuities, and hence segmenting the image into foreground and background regions, induces these uncovered areas to form. Certainly, if the segmentation step was ignored

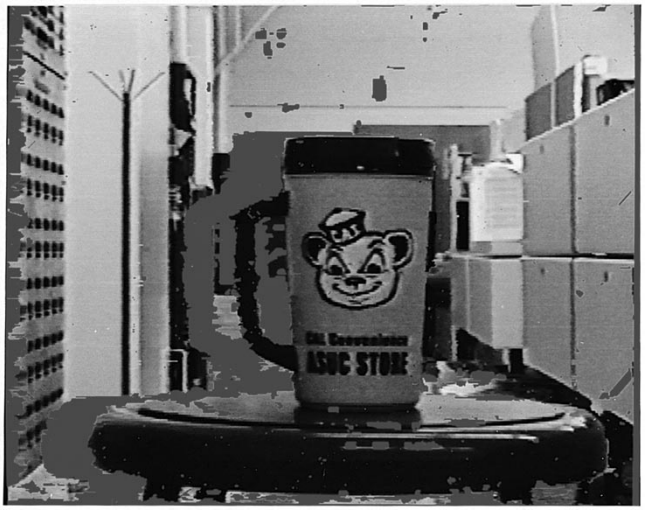

Fig. 16. Horizontal view estimate with respect to frame 35.

Fig. 17. Horizontal view estimate with respect to frame 65.

and patches that transcend depth boundaries were rendered, very few uncovered regions would appear, replaced by the smearing artifact mentioned previously. The red regions to the left of the mug and stool in Fig. 16 and to the right in Fig. 17 are precisely the uncovered regions in the scene.

In both cases, the holes are left unmarked in the estimated image, since a single reference frame has no information about these points. As we will see in the next section, some of these holes will be eliminated in a combination stage where other reference frames having information at these locations may fill in the holes. We also propose other techniques in Section III-C to deal with covering holes.

C. Combination of Reconstructed Data

Once we compute the estimates of the desired view with respect to each of the chosen reference frames, we must combine these data to generate the appropriate reconstruction and deal with the remaining holes. Suppose there are N reference frames for reconstruction with corresponding view estimates $\hat{I}^{(k)}$ for $k = 1, 2, \cdots N$. To find the intensity for

Fig. 18. Reconstructed view along horizontal trajectory.

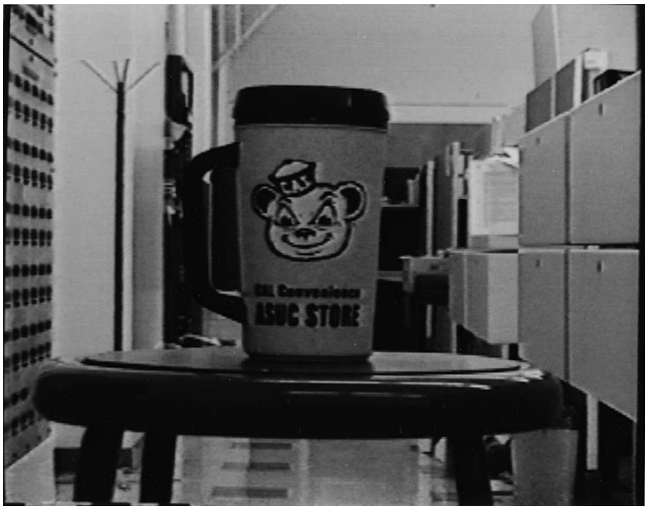

Fig. 19. Reference frame 37 (intensity) of Mug2.

pixel (i,j), we examine points from the N view estimates in a $S \times S$ pixel region $A_{i,j}$ centered around pixel (i,j).[6] Akin to the z-buffering technique from the previous section, only points in $A_{i,j}$ with the smallest depths, i.e., closest to the camera, are considered; all other points in $A_{i,j}$ are thrown out. When an object moves in a frame, we would like the pixels of the foreground to precede those from the background occupying the same region. The intensities of the remaining points are examined for consistency and outliers are discarded in a manner similar to the approach described in Section II-C. Holes are automatically filled in as long as at least one reference frame has information about the region in question. Once the number of points in $A_{i,j}$ has been reduced, the intensity $\hat{I}(i,j)$ is simply the weighted average of the remaining points, i.e.,

$$\hat{I}(i,j) = \sum_{k=1}^{M} \alpha_k \hat{I}^{(k)}(i,j) \tag{16}$$

where α_k is defined as the weight γ_k from (9) if none of the N reference frames has a hole at (i,j) (i.e., $M = N$) and as the weighted average of γ_k's for the M contributing reference frames for $M < N$.

It should be clear that this combination technique allows the uncovered points from one view estimate to be filled in by the other view estimates, thus minimizing the number of holes in the scene. However, it is still possible that there exists no points in a given $S \times S$ region from any of the view estimates; these are the holes that lie in the intersection of the holes of all the reference frames. As mentioned before, these holes are either due to regions that become uncovered or due to a rough depth map. In the former case, we cannot fundamentally do anything for these holes since there is no information about them. However, holes resulting from a rough depth map may be interpolated to fill them in.[7] One approach is to grow the area $A_{i,j}$ out to a $m \times m$ region, where $m > S$ is the smallest value for which a point falls within the area $A_{i,j}$, i.e., the region is no longer a hole. Once we find such an area, we then use the above technique to find the intensity value at the grid point (i,j). Examples of this will be shown in Section IV.

[6] We select S to be 1 because a larger S tends to blur the final image too much.

[7] Introducing even more reference frames and view estimates may help to further reduce the size of both types of holes.

Fig. 20. Reference frame 37 (depth) filled in by splines.

IV. Results

We shall now examine some results using the techniques described above. Since it is difficult to reconstruct an arbitrary view that is precisely coincident with any one frame from the original sequence, we have not computed reconstruction error for the following results.

A. Mug Scene

The first scene consists of a mug placed atop a stool. A CCD camcorder is moved horizontally by hand to follow trajectories at two different elevations to generate an image sequence for each trajectory, similar to the set up drawn in Fig. 1. The reference frames from both trajectories are shown in Fig. 11. Each frame is 640 $\times$ 480 pixels large and consists of intensity only. No special lighting was used to film the

scene; specularities of the stool and the lid of the mug are very apparent in the images.

For the first set of results, the desired view is roughly halfway between two reference frames along the same horizontal trajectory. Frames 35 and 65 from the first trajectory Mug1 sequence are selected as the reference frames; they are shown in Figs. 12 and 13. The chosen view is perhaps the one most prone to errors due to the large occluded regions. Note that there is roughly a maximum of a 120-pixel disparity between the two reference frames.

Figs. 14 and 15 show the corresponding depth maps obtained by using the proposed matching algorithm.[8] The mug and stool are estimated well and do not contain many spurious depths. There is a gradual change in depth as expected for a hallway scene. Artifacts are most prevalent in the handle of the mug. Problems arise here because intensity-based matching schemes perform poorly for background regions that can be seen through foreground regions.

The estimates of the desired view generated by the two reference frames are shown in Figs. 16 and 17. As described in Section III-B, the holes marked in red are the points in the scene that have become uncovered or that stem from bumpy depth maps. However, combining the information in both view estimates into a single one eliminates the uncovered points entirely; only cracks remain due to the rough depth maps. Interpolating these remaining cracks results in Fig. 18. The image quality is good for the most part. The horizontal edges, e.g., top of the door, top of the mug, specularities in front of the stool, and the drawers, have been reconstructed quite well. The proposed algorithms take care of problems in occluded regions; there are only a few errors to the right of the mug and near the mug handle. These artifacts arise because the depth edges were not localized perfectly.

To generate a view not originally scanned by the camcorder, two frames from different vertical elevations, frame 35 from Mug1 and frame 37 from Mug2 are chosen as reference frames. The intensity and depth for frame 37 are shown in Figs. 19 and 20, respectively. The desired view is roughly the midpoint on the vertical trajectory relating the two views given in Fig. 21. As before, the view estimates each possess uncovered regions about which a single reference frame has no information. However, the combined image turns out to be a reasonable estimate of the desired view where the cracks have been filled in appropriately. As before, the most troublesome region in the image lies inside the handle of the mug.

Using all three reference frames, we can reconstruct the view translated arbitrarily along the x-y plane. The image is shown in Fig. 22. For the most part, the image appears to be a good estimate. As expected, the artifacts appear at the occluded or uncovered portions of the image, namely underneath the mug handle and the stool legs. The regions marked in red indicate precisely those uncovered regions for which the three reference frames have no information. For instance, the cluster of points near the right stool leg is a portion of the background that is obscured in the three original reference frames. Notice that filling these holes with the algorithm described above will inevitably produce an inaccurate image since the holes are relatively large and since we have no information about them.

[8] As described in Section III-C, the depth maps have been quantized to 25-b levels. In addition, their histograms have been equalized to increase contrast for visualization.

Fig. 21. Reconstructed view along vertical trajectory.

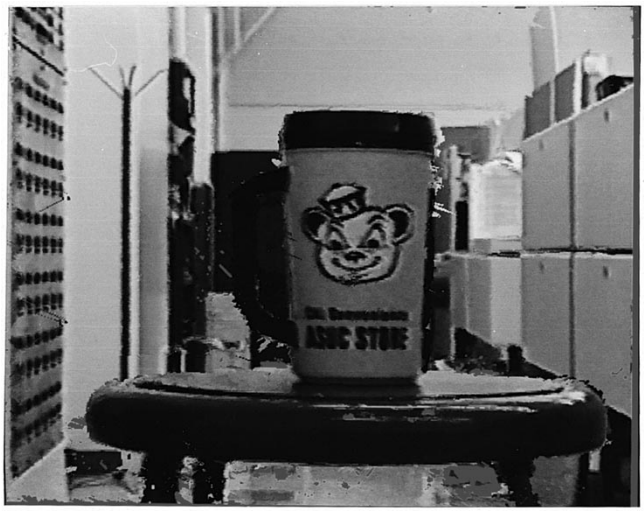

Fig. 22. Reconstructed Mug view from x-y plane.

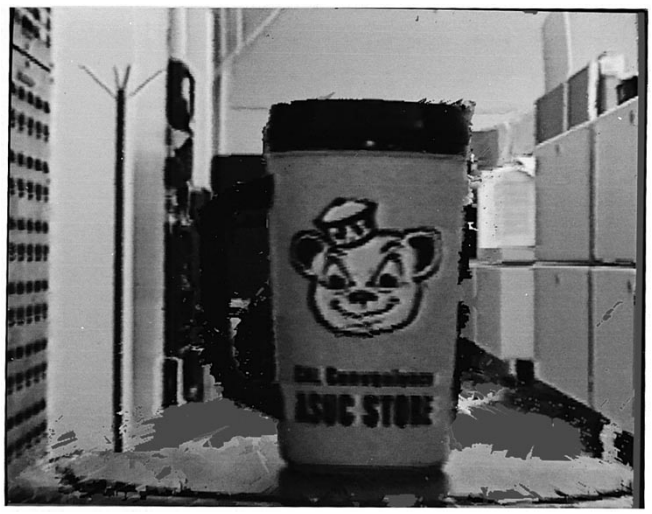

Fig. 23. Reconstructed Mug view from translation along $-z$ axis.

Fig. 24. Reconstructed Mug view from translation along $+z$ axis.

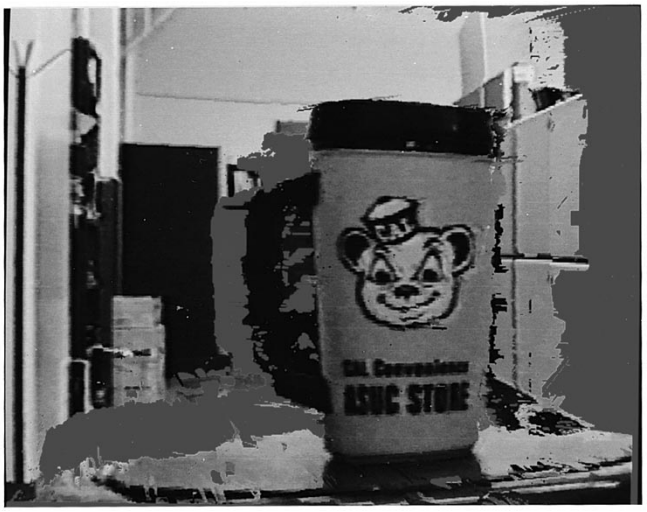

Fig. 25. Reconstructed Mug view from pan of 10° clockwise, translation of 12 units along the x-axis and 0.02 units along the z-axis.

More interesting views not necessarily confined to the x-y plane may be reconstructed with this representation. For instance, the viewpoint of a camera translated toward the scene can also be rendered quite easily; it is given in Fig. 23. Note that this view differs from a simple "zoom-in," since the latter requires only a larger focal length and it does not uncover occluded regions. The two regions above the stool are marked red because none of the reference frames has information about what lies behind the stool in the scene. Fig. 24 shows the view translated away from the scene with the uncovered regions marked accordingly. Finally, Fig. 25 shows an oblique view of the scene taken by rotating the camera 10° clockwise and translating along both the x- and z-axes. The quality of the reconstructed image is quite good given the amount of uncovered regions.

B. Chess Scene

A more complicated scene featuring a chess set is filmed by the camcorder. We have confined the motion so that the image plane is parallel to the direction of motion; the camcorder moves primarily in a horizontal motion with occasional motion in the vertical direction. Each frame is digitized to 320 $\times$ 240 pixels, and again consists of intensity only.

Fig. 26. Reference frame 10 (intensity) of Chess scene.

Fig. 27. Reference frame 23 (intensity) of Chess scene.

Figs. 26 and 27 show frames 10 and 23 from the sequence. We note that this scene exhibits a much larger motion than in the previous case; the largest disparity is 170 pixels for the bottommost white pawn, over 50% of the entire image. The described matching algorithm leads to the corresponding depth maps in Figs. 28 and 29. The different chess pieces have been recovered in the depth maps as indicated by the differing levels of grey.

The view that lies between the two reference frames is given in Fig. 30. The resulting image quality is quite good especially given the complexity of the scene. It is interesting to note that that the algorithm incorporates the bishop, seen in only the first image, and the rightmost pawn from the second image into the new view. If we consider translating the camera toward the scene, we obtain the view shown in Fig. 31. The holes in the scene are again drawn in red. Translating away from the scene leads to the image shown in Fig. 32. We observe that the result consists of the union of the points seen in the two reference images and is similar to a 3-D parallax corrected mosaic [24]. Finally, Fig. 33 demonstrates an oblique view obtained by tilting the camera 10° and translating along all three axes. The results for each of the views are quite reasonable.

Fig. 28. Reference frame 10 (depth) of Chess scene.

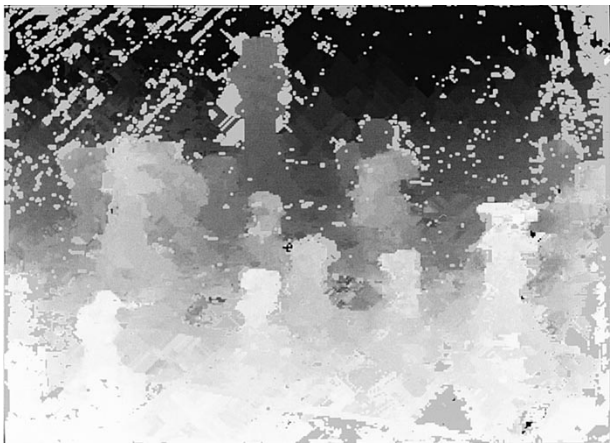

Fig. 29. Reference frame 23 (depth) of Chess scene.

Fig. 30. Reconstructed view along horizontal trajectory.

Fig. 31. Reconstructed Chess view from translation along $-z$ axis.

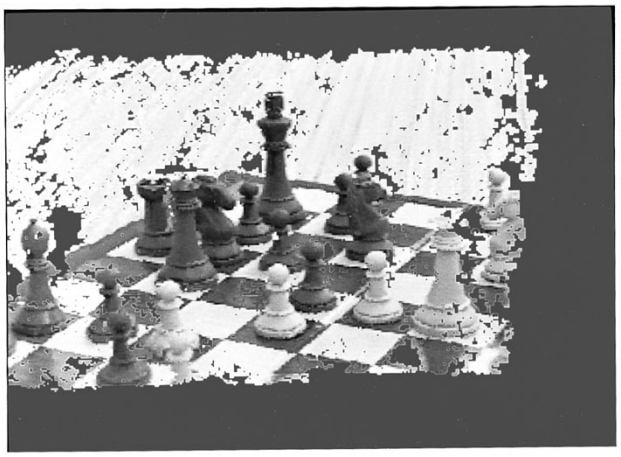

Fig. 32. Reconstructed Chess view from translation along $+z$ axis.

C. Comparison to Previous Work

We found that most of the methods described in Section I have different input data requirements from the proposed approach, making it difficult for direct comparison. For example, image mosaics [35], [27] restrict camera motion to be panning, not roughly translational. The approach of Skerjanc and Liu [34] requires a calibrated trinocular set-up, while that of Kanade *et al.* [22] uses clusters of fixed arrays of cameras.

In the end, we have chosen to compare our results with the approach of Laveau and Faugeras [25]. They require only a set of reference frames and a dense disparity map, both consistent with our approach. The algorithms also have similar storage requirements; their approach needs two images and one disparity map, whereas the proposed approach requires one additional depth map. To construct a new view, Laveau and Faugeras employ a ray-tracing-like algorithm whereby the intersection of the projections of certain optical rays is examined. This step is however computationally intensive for reconstruction. In contrast, much of the complex processing in our approach may be done offline, since our representation and reconstruction processes are distinct, thus leading to a faster reconstruction stage.

In our implementation of Laveau and Faugeras' algorithm, the reference frames in Figs. 26 and 27 are the input images, while the depth map in Fig. 29 serves as the correspondence map. Because of our reference pair configuration, their algorithm exhibits artifacts near the trifocal plane [25]. Also, we found that views like Fig. 30 lying along the baseline between

Fig. 33. Reconstructed Chess view from tilt of 10°, translation of four units along the x- axis, three units along the y-axis, and 0.04 units along the z-axis.

Fig. 34. Reconstructed Chess view with Laveau and Faugeras' algorithm for translation along $-z$ axis.

the two reference images are unattainable due to collinearity of the optical centers. Fig. 34 shows the camera translating toward the scene. As compared to Fig. 31, their algorithm has difficulty recovering the king and the rightmost knight. Also our approach performs better in uncovered regions, such as to the right of the rook and above the white pawn at the bottom of the image. Similarly, we may compare the view with the camera translating away as given in Fig. 35. It should be clear that only the points that are common to both images are drawn in their algorithm in contrast to ours in Fig. 32.

V. Discussion

We have proposed an approach for representing and reconstructing static 3-D scenes. For views along a horizontal trajectory, the algorithms produce reasonable reconstructed images where most of the error is concentrated near the occlusion boundaries. For views not scanned by the camcorder, the discussed approach leads to promising results. Our results are comparable to full 3-D modeling techniques yet not as complicated. Moreover, using depth surfaces to estimate scene structure results in recovering uncovered background points in the scene much better and leads to a faster rendering approach. Direct methods based on projective geometry, such

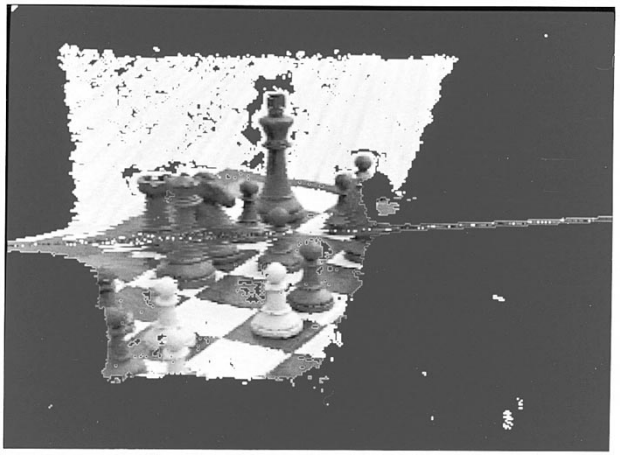

Fig. 35. Reconstructed Chess view with Laveau and Faugeras' algorithm for translation along $+z$ axis.

as that in [25], reconstruct only those points that lie in the intersection of the input views, whereas our approach can render points that lie in the union of the input views. In addition, there is no depth recovery or explicit representation stage with direct methods. As a result, an exhaustive search over every optical ray needs to be performed for every view generation. Consequently, no off-line processing can be executed beforehand and, hence, the approach becomes more computationally intensive. This is in contrast to our approach where the intensity-depth representation can be generated off-line, leading to straightforward and relatively fast rendering of new views.

Future work in this area includes examining the optimum choice and number of reference frames to fully capture a scene. The reference frames in the paper were chosen rather arbitrarily. With our current approach, a desired view chosen far away from the reference frames leads to very erroneous results. In addition, there is a noticeable decrease in performance as the baseline between reference images increases. The issue of automatically selecting reference frames and choosing an optimum set remains a difficult problem to solve. A more complete analysis must be performed in order to determine what is the scope of a single reference frame, or conversely, what is the optimum set and locations of reference frames to compactly represent a given scene.

In addition, the proposed representation is certainly not the most optimum. While multiple reference frames help to fill in uncovered regions of the scene, there is a significant amount of redundancy in both the intensity and depth maps. Since most of the frames of the original data have been discarded, we do not utilize all the information about the scene with our representation. We are currently examining layered representations [39] and multivalued intensity and depth maps to encapsulate the scene information better [8].

Finally, a real-time implementation of the reconstruction algorithm would expedite the development of a virtual environment. Using a stereoscopic display and head tracking device, we will be able to simulate such a system by reconstructing an arbitrary view of a scene in real time as the user moves his/her head.

ACKNOWLEDGMENT

The authors thank the anonymous reviewers of this paper for the insightful comments they provided. We would especially like to thank S. Laveau for answering questions about his approach.

REFERENCES

[1] E. Ardizzone, M. A. Palazzo, and F. Sorbello, "3-D scene reconstruction from multiple 2-D views," in *Proc. 5th Int. Conf. Image Analysis and Processing*, Positano, Italy, Sept. 20–22, 1989, pp. 394–398.
[2] J. L. Barron, D. J. Fleet, and S. S. Beauchemin, "Performance of optical flow techniques," *Int. J. Comput. Vis.*, vol. 12, pp. 43–77, Feb. 1994.
[3] R. H. Bartels, J. C. Beatty, and B. A. Barsky, *An Introduction to Splines for Use in Computer Graphics and Geometric Modeling*. Los Altos, CA: Morgan Kaufmann, 1987.
[4] C. Braccini, A. Grattarola, and S. Zappatore, "Volumetric and pictorial reconstruction of 3D objects from correspondences in moving 2D views," in *Recent Issues in Pattern Analysis and Recognition*, V. Cantoni, R. Creutzburg, S. Levialdi, and G. Wolf, Eds. New York: Springer-Verlag, 1989, pp. 249–258.
[5] N. L. Chang, "View reconstruction from uncalibrated cameras for three-dimensional scenes," Master's thesis, Dept. Elect. Eng. Comput. Sci., Univ. Calif., Berkeley, CA, 1994.
[6] N. L. Chang and A. Zakhor, "Arbitrary view generation for three-dimensional scenes from uncalibrated video cameras," in *Proc. ICASSP*, Detroit, MI, May 8–12, 1995, vol. 4, pp. 2455–2458.
[7] ——, "Intermediate view reconstruction for three-dimensional scenes," in *Proc. ICDSP*, Nicosia, Cyprus, July 14–16, 1993, vol. 2, pp. 636–641.
[8] ——, "Multivalued representations for image reconstruction and new view synthesis," in preparation.
[9] ——, "View generation for 3-D scenes from video sequences," in *Proceedings of IMDSP Workshop*, Belize City, Belize, Mar. 3–6, 1996, pp. 134–135.
[10] S. E. Chen and L. Williams, "View interpolation for image synthesis," in *Proc. SIGGRAPH*, New York, Aug. 1–6, 1993, pp. 279–288.
[11] C. H. Chien and J. K. Aggarwal, "Identification of 3D objects from multiple silhouettes using quadtrees/octrees," *Comput. Vis., Graph., Image Processing*, vol. 36, pp. 256–273, Nov.–Dec. 1986.
[12] C. de Boor, "On calculating with B-splines," *J. Approx. Theory*, vol. 6, pp. 50–62, July 1972.
[13] U. R. Dhond and J. K. Aggarwal, "Structure from stereo—A review," *IEEE Trans. Syst., Man, Cybern.*, vol. 19, pp. 1489–1509, 1989.
[14] J. D. Foley *et al.*, *Introduction to Computer Graphics*. Reading, MA: Addison-Wesley, 1994.
[15] P. Fua, "A parallel stereo algorithm that produces dense depth maps and preserves image features," *Machine Vis. Appl.*, vol. 6, pp. 35–49, Winter 1993.
[16] D. Hearn and M. P. Baker, *Computer Graphics*. Englewood Cliffs, NJ: Prentice-Hall, 1986.
[17] K. Higuchi, M. Hebert, and K. Ikeuchi, "Building 3-D models from unregistered range images," Carnegie Mellon Univ., Tech. Rep. CMU-CS-93-214, Nov. 1993.
[18] B. K. P. Horn, *Robot Vision*. Cambridge, MA: MIT Press, 1991.
[19] B. K. P. Horn and B. G. Schunck, "Determining optical flow," *Artif. Intell.*, vol. 17, pp. 185–203, Aug. 1981.
[20] M. Ito and A. Ishii, "Three-view stereo analysis," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-8, pp. 524–532, July 1986.
[21] D. G. Jones and J. Malik, "A computational framework for determining stereo correspondence from a set of linear spatial filters," in *Proc. ECCV*, Santa Margherita Ligure, Italy, May 18–23, 1992, pp. 395–410.
[22] T. Kanade, P. J. Narayanan, and P. W. Rander, "Virtualized reality: Concepts and early results," in *Proc. IEEE Workshop on Representation of Visual Scenes*, Cambridge, MA, June 24, 1995, pp. 69–76.
[23] J. J. Koenderink and A. J. van Doorn, "Facts on optic flow," *Biol. Cybern.*, vol. 56, pp. 247–254, 1987.
[24] R. Kumar, P. Anandan, and K. Hanna, "Shape recovery from multiple views: A parallax based approach," in *Proc. Image Understanding Workshop*, Monterey, CA, Nov. 13–16, 1994, vol. 2, pp.
[25] S. Laveau and O. Faugeras, "3-D scene representation as a collection of images and fundamental matrices," INRIA, Tech. Rep. 2205, Feb. 1994.
[26] J. S. Lim, *Two-Dimensional Signal and Image Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1990.
[27] S. Mann and R. W. Picard, "Video orbits of the projective group: A new perspective on image mosaicing," MIT Media Lab. Percept. Comput., Tech. Rep. 338, May 1995.
[28] D. Marr and T. Poggio, "Cooperative computation of stereo disparity," *Science*, vol. 194, pp. 283–287, Oct. 1976.
[29] R. Mohr, F. Veillon, and L. Quan, "Relative 3D reconstruction using multiple uncalibrated images," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, New York, NY, Jan. 15–18, 1993, pp. 543–548.
[30] R. M. Murray, Z. Li, and S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation*. Boca Raton, FL: CRC, 1994.
[31] W. M. Newman and R. F. Sproull, *Principles of Interactive Computer Graphics*. New York: McGraw-Hill, 1979.
[32] M. Okutomi and T. Kanade, "A multiple-baseline stereo," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 15, pp. 353–363, Apr. 1993.
[33] A. Shashua, "Projective structure from two uncalibrated images: Structure from motion and recognition," MIT AI Lab., Tech. Rep. 1363, Sept. 1992.
[34] R. Skerjanc and J. Liu, "A three camera approach for calculating disparity and synthesizing intermediate pictures," *Signal Processing: Image Commun.*, vol. 4, pp. 55–64, 1991.
[35] R. Szeliski, "Image mosaicing for tele-reality applications," Digital Equip. Corp. Cambridge Res. Lab., Tech. Rep. CRL 94/2, May 1994.
[36] R. Szeliski and S. B. Kang, "Recovering 3D shape and motion from image streams using nonlinear least squares," Digital Equip. Corp. Cambridge Res. Lab., Tech. Rep. CRL 93/3, Mar. 1993.
[37] C. J. Taylor and D. J. Kriegman, "Structure and motion from line segments in multiple images," Ctr. Syst. Sci., Yale Univ., Tech. Rep. 9402b, Jan. 1994.
[38] S. Ullman and R. Basri, "Recognition by linear combinations of models," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 13, pp. 992–1006, Oct. 1991.
[39] J. Y. A. Wang and E. H. Adelson, "Representing moving images with layers," *IEEE Trans. Image Processing*, vol. 3, pp. 625–638, Sept. 1994.
[40] J. Weng, N. Ahuja, and T. S. Huang, "Matching two perspective views," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 14, pp. 806–825, Aug. 1992.
[41] ——, "Optimal motion and structure estimation," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 15, pp. 864–884, Sept. 1993.
[42] Z. Zhang *et al.*, "A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry," INRIA, Tech. Rep. 2273, May 1994.

Nelson L. Chang (S'90) received the B.S.E. degree in electrical engineering from Princeton University, Princeton, NJ, in 1992. He attended graduate school at the University of California (U.C.), Berkeley, where he joined the Video and Image Processing Laboratory in January of 1993. He received the M.S. degree from U.C. Berkeley in 1994 for earlier work on view reconstruction from depth-based representations. Currently, he is pursuing the Ph.D. degree at U.C. Berkeley.

Avideh Zakhor (M'87) received the B.S. degree from California Institute of Technology (Caltech), Pasadena, and the S.M. and Ph.D. degrees from Massachusetts Institute of Technology, Cambridge, all in electrical engineering, in 1983, 1985, and 1987, respectively.

In 1988, she joined the faculty at the University of California, Berkeley, where she is currently Associate Professor in the Department of Electrical Engineering and Computer Sciences. Her research interests are in the general area of signal processing and its applications to images and video, and biomedical data. She has been a consultant to a number of industrial organizations, holds four U.S. patents, and is the co-author of the book, *Oversampled A/D Converters* (with S. Hein).

Ms. Zakhor was a General Motors scholar from 1982 to 1983, received the Henry Ford Engineering Award and Caltech Prize in 1983, and was a Hertz fellow from 1984 to 1988. She received the Presidential Young Investigators (PYI) award, IBM Junior Faculty Development Award, and Analog Devices Junior Faculty Development Award in 1990, and the Office of Naval Research Young Investigator Award in 1992. She is currently a member of the Technical Committee for Image and Multidimensional Digital Signal Processing.