Video Representation with Three-Dimensional Entities

Fernando C. M. Martins and José M. F. Moura, *Fellow, IEEE*

***Abstract*—Very low bit-rate coding requires new paradigms that go well beyond pixel- and frame-based video representations. We introduce a novel content-based video representation using tridimensional entities: textured object models and pose estimates. The multiproperty object models carry stochastic information about the shape and texture of each object present in the scene. The pose estimates define the position and orientation of the objects for each frame. This representation is compact. It provides alternative means for handling video by manipulating and compositing three-dimensional (3-D) entities. We call this representation *tridimensional video compositing*, or 3DVC for short. In this paper, we present the 3DVC framework and describe the methods used to construct incrementally the object models and the pose estimates from unregistered noisy depth and texture measurements. We also describe a method for video frame reconstruction based on 3-D scene assembly, and discuss potential applications of 3DVC to video coding and content-based handling. 3DVC assumes that the objects in the scene are rigid and segmented. By assuming segmentation, we do not address the difficult questions of nonrigid segmentation and multiple object segmentation. In our experiments, segmentation is obtained via depth thresholding. It is important to notice that 3DVC is independent of the segmentation technique adopted. Experimental results with synthetic and real video sequences where compression ratios in the range of 1:150–1:2700 are achieved demonstrate the applicability of the proposed representation to very low bit-rate coding.**

***Index Terms*—Content-based video handling, model-based video coding, range and image sequence processing, 3-D object modeling, video sequence representation.**

I. Introduction

A video representation is the intermediate form into which a coding system transforms video. The representation defines the internal structure of the codec, and determines the limitations and capabilities of video handling and content access methods.

Waveform-based representations describe video as a sequence of two-dimensional (2-D) signals in time. They lead to concepts like transform coding (DCT) and quantization. Temporal redundancy is reduced via differential pulse code modulation (DPCM) and block motion compensation. This representation is successfully incorporated in several standard video codec's such as the H.26x series, MPEG-1, and MPEG-2. Waveform-based representations have proven very useful, but have practically reached their limit in terms of coding efficiency. More importantly, being waveform-based, these representations are not tailored for content-based access to video.

Model-based representations consider video as a sequence of 2-D projections of a three-dimensional (3-D) scene. A set of models and parameters is extracted from the video sequence, such that the original video sequence is reconstructed using only this given set of constructs. Semantic coding [1], [2], object-oriented coding [3], [4], and layered representations [5]–[7] are examples of model-based representations.

Semantic coding assumes that detailed explicit parameterized 3-D object models are available. Typical examples are *head and shoulders* parametric models [8]. Analysis of the video sequence leads to adjustments to shape that are parametrically encoded. This technique provides high compression, but does not support free-formed objects.

Object-oriented coding constructs models from observations. These models can be 2-D or 3-D. Musmann *et al.* [3] propose object-oriented analysis–synthesis coding (OBASC), a generic model-based representation where objects in the scene are described by three parameter sets defining object motion, color, and shape. They construct 2-D models assuming that the scene is composed of planar rigid objects moving in 3-D. Shape is not explicitly extracted, and the motion between frames is encoded as a set of affine mappings, one for each planar patch detected in the scene. An extension using 3-D models is implemented by Ostermann [4] to encode typical *head and shoulders* videoconferencing sequences. His technique assumes the cross section of the subject to be elliptical in order to construct 3-D surface models from 2-D silhouettes.

In a visionary position paper, Bove [9] presents the general idea of object-oriented television—"a computational framework for a television receiver that can handle digital video in [several] forms from traditional motion-compensated transform coders to sets of three-dimensional objects."

The 3-D structure of objects is recovered for the purpose of data compression by [10] and [11]. Stereo pairs are used to construct triangular mesh descriptions of object shape. The proposed methods do not deal with uncertainties in the sensorial process. Our approach handles these uncertainties through the use of volumetric models that convey shape and texture, as explained in Section II.

Surveys about prior work in model-based representations for very low bit-rate (VLBR) coding and videotelephony applications are found in [12] and [13].

A current trend in model-based representations uses layered 2-D models. In layered models, each video frame is a

Manuscript received September 1, 1996; revised March 1, 1997. The work of F. C. M. Martins was supported in part by CNPq-Brazil.

The authors are with the Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh PA 15213-3890 USA.

Publisher Item Identifier S 0733-8716(98)00301-1.

superposition of a set of 2-D mosaic models constructed by cut-and-paste operations [5]–[7], [14]. Layered representations generate highly constrained 2-D models that mix motion and structure information. An object performing a simple rotation around an axis not parallel to the camera principal axis is not consistently represented by a layered 2-D model without major model updates from frame to frame.

Several researchers address the issues of object modeling from observations outside the specific realm of video representation. These techniques usually fail to attend the demanding requirements of video processing as incremental modeling, automatic integration without user interaction, and robustness.

Early work in object modeling from a sequence of range images assumes measurement registration, i.e., that the relative position and orientation of the object are known for all frames [15]–[18].

The factorization method of Tomasi and Kanade [19] computes shape and motion from a sequence of intensity images. The geometric object model obtained is a sparse set of unorganized points in space. Measurements are taken under orthography, features are extracted, tracked over time, and organized in a measurement matrix. The singular value decomposition (SVD) of the measurement matrix provides motion and shape factorization. For video processing, the number of observed views can be as large as the number of video frames. Complex objects may also require a large number of features to be tracked for a proper shape description. Under these circumstances, the computation of principal components from the measurement matrix via SVD becomes a computationally hard problem. This technique provides no support for incremental model construction, i.e., the computation of principal components must be performed whenever a new measurement is to be incorporated.

Azarbayejani [20] constructs polygonal surface models of outdoor sequences of buildings from intensity images without prior assumptions about shape or pose. Feature extraction and correspondence must be performed by hand, requiring a considerable amount of user interaction. Becker and Bove [21] propose a technique to extract polygonal models of scenes containing orthogonal planes and parallel lines, which are typical to man-made environments. Camera calibration is not assumed, but a great deal of human interaction is required to cluster features and establish cluster correspondences between frames.

Koch [22], [23] presents a system to build 3-D parametric surface models without user interface from a sequence of stereo images. The sequences depict buildings and other man-made objects. Surface model construction requires merging sequences of incomplete parametric models, a task that usually generates shape artifacts. Merging surfaces is a computationally hard problem, as discussed by Turk and Levoy [24].

Curless and Levoy [25] recently showed that surface merging techniques may fail if the object has high curvature or sharp edges and presented an alternative integration algorithm based on volumetric models and implicit functions. Their algorithm assumes accurate alignment of the data sets, does not handle surface texture, and does not provide support for handling sensor uncertainty.

To enable tridimensional handling and access to video content, and to also yield compact storage, we propose a novel system to represent video based on 3-D entities: textured object models and pose estimates.

To explain our approach, we start by considering the dual problem of generation of realistic synthetic images. First, 3-D geometric models with defined surface properties are created for each of the objects to appear in the scene. Then we assemble a scene by placing object models and light sources in space according to animation motion scripts. Finally, we position a virtual camera in the synthetic scene, and render an image by computing the light captured by each of the camera sensors after reflecting on the scene components. The fundamental entities for the representation of a synthetic video sequence are then the set of 3-D object models and respective motion scripts.

The generation of a representation for video based on 3-D entities is the *inverse* problem of synthetic image generation. Given a video sequence, the problem is to compute automatically for each of the objects present in the original scene a set of 3-D pose estimates and a 3-D object model representing shape and texture.[1]

We adopt a stochastic voxel-based structure for the object models that represents free-formed objects, allows incremental model updates, and considers sensor uncertainties. The original video sequence is reconstructed by rendering projections of 3-D scenes assembled using the 3-D constructs. We call this method of compositing video with 3-D entities *tridimensional video compositing*, or 3DVC for short.

In 3DVC, motion, shape, and texture are explicitly accounted for. Each individual element of the set of component 3-D entities can then be manipulated independently. For example, the shape of an object can be altered independently of the object motion script, or motion scripts modified to generate synthetic motion patterns. 3DVC allows the insertion of additional frames by rendering frames with interpolated pose estimates. This leads to video with higher frame rate and smoother perceived motion than the original sequence.

Video is represented in 3DVC as a composite of perceptually meaningful entities. Each object model can be visualized, and the motion scripts bear physical significance. Object curvature, texture moments, and acceleration can be extracted from the 3-D entities. By accessing the underlying structure of video, 3DVC creates innovative and flexible ways for editing and accessing the video content. 3DVC synthesizes distinct video sequences by altering the motion scripts and/or object models previously generated. Explorable features of 3DVC include missing frame insertion, variable focus of attention, cast selection, content-based search, and insertion/deletion of virtual and real entities.

The paper is organized as follows. Section II presents the 3DVC representation framework, and the stochastic structure for the object models and measurements. Section III is dedicated to video analysis. This section addresses the methods

[1] In this paper, "texture" refers to the light captured by a camera after reflecting on the object surface. Also known as object radiance, "texture" depends on object surface photometric properties and environment illumination conditions.

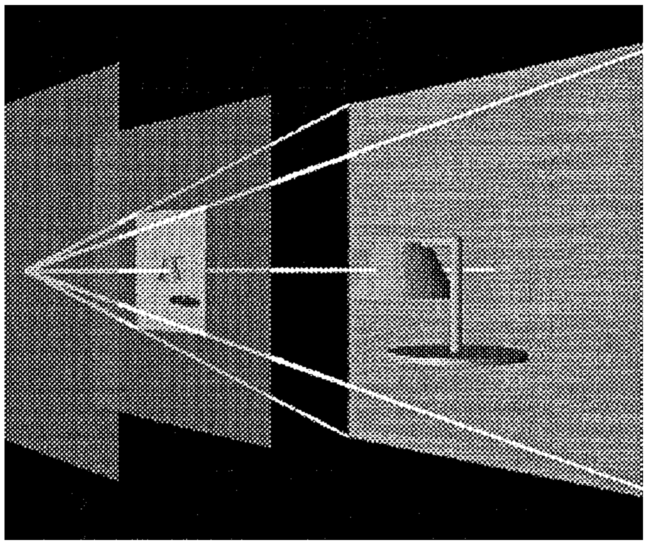


Fig. 1. In 3DVC, a video frame F_k is given by the 3-D object model Γ_{jk} and the object pose $\vec{q}_{jk}$.

for incremental object model construction and pose estimation. Section IV describes the synthesis of the original video from the constructed 3-D entities, i.e., scene assembly and frame reconstruction. Experimental results are in Section V. Applications of 3DVC to video coding and content-based handling are in Section VI. Section VII concludes the paper.

II. 3DVC Representation Framework

In 3DVC, a scene S_k is specified by a set of 3-D entities: the object models Γ_{jk} and the respective pose estimates $\vec{q}_{jk}$, as in (1)

$$S_k = \{(\Gamma_{jk}, \vec{q}_{jk})\}, \qquad \text{for } j = 1, 2, \cdots, N \tag{1}$$

where k is the discrete time index, j is the object index, and N is the number of objects in the scene.

Video is a sequence of frames in time. Each frame F_k is obtained via perspective projection of a dynamic 3-D scene S_k, as defined in (2) and (3)

$$F_k = \text{proj}(S_k) \tag{2}$$

$$\mathcal{V} = \{F_k\}, \qquad \text{for } k = 1, 2, \cdots, T \tag{3}$$

where T is the number of frames in the video sequence $\mathcal{V}$.

Factoring out frames and scenes, 3DVC provides a structured representation for video in terms of 3-D entities as in Fig. 1 and in (4)

$$\mathcal{V} = \{\text{proj}[\{(\Gamma_{jk}, \vec{q}_{jk}), \forall j\}], \forall k\}. \tag{4}$$

3DVC builds the 3-D object models Γ_{jk} incrementally without user interference. This is accomplished by analysis of the video sequence, via integration of multiple measurements in a Bayesian framework. The 3-D object models contain stochastic information about the shape and texture of the objects. As more video frames are processed, 3DVC reduces model entropy and eliminates redundant information. No prior information about the object shape and texture is assumed to be available.

For each frame F_k, the pose $\vec{q}_{jk}$ of object j is estimated by measurement/model registration. The pose is sequentially stored in a motion script for the given object. The methods for incremental object model construction and pose estimation are considered in Section III.

The video sequence $\mathcal{V}$ is reconstructed in two sequential steps: recreation of the scene S_k by positioning the 3-D object models Γ_{jk} in space according to the corresponding pose estimates $\vec{q}_{jk}$, and reconstruction of video frames F_k by a ray-casting volume renderer. Both steps are addressed in Section IV.

Object Model: We adopt a nonparametric volumetric structure for the object models. This model structure allows representation of free-formed objects, permits incremental model update, and provides efficient spatiotemporal support for data fusion and storage. This contrasts with the majority of previous work on 3-D model-based video coding that relies on parametric surface-based models [13]. Surface models are usually chosen because they allow the use of well-known surface renderers as image synthesis tools. For information fusion, however, surface representations present challenging surface-merging problems [24].

Our nonparametric object model is voxel-based. It is defined as a compact uniform tesselation of 3-D space $\Gamma = \{C_i\}$, where each cell C_i represents multiple properties and the index i spans the compact 3-D space. The object shape is represented by cell occupancy $O(C_i)$, and the object surface texture is represented by cell texture $T(C_i)$. Occupancy is a binary property: a cell C_i may either be *occupied* or *empty*. Texture may assume any valid value drawn from the color pallete $\mathcal{C}$.

A nonparametric voxel-based object description is suitable for video representation because free-formed objects are supported with selectable spatial resolution, frame rendering performance is independent of object and scene complexity, and the parallel nature of volume rendering algorithms can be explored [26].

The tesselation stores a set of conditional probabilities that is obtained through the integration of the multiple measurement grids $\{\Psi_0, \cdots, \Psi_k\}$ as in

$$P(O(\Gamma)) \triangleq \{P(O(C_i) \mid \{\Psi_0, \cdots, \Psi_k\}), \forall i\}. \tag{5}$$

Holding probability functions instead of current estimates is what makes Γ a useful representation for Bayesian integration of a sequence of measurements. Initial lack of knowledge is expressed by assigning equiprobable probability distributions.

The stochastic nature of the object models enables robust operation when dealing with inconsistent and noisy measurements or pose estimates. The stochastic representation supports both active exploration and passive integration of sensory data. Model entropy is used to guide exploration if active sensors are available. A stochastic framework for measurement integration has been considered for the unidimensional case by [27]. Deterministic integration does not cope with sensor uncertainties and imprecisions in pose estimation. This may cause insertion of persistent holes and fissures in the object surface.

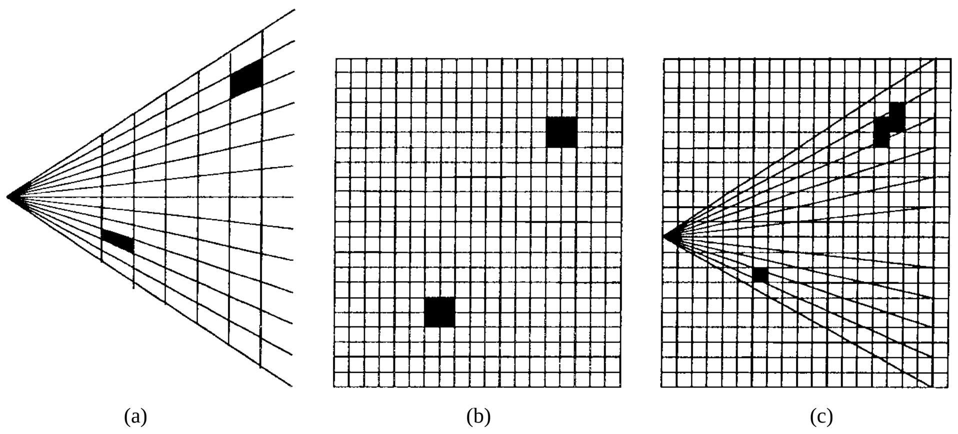


Fig. 2. Steps in the construction of a measurement grid. (a) Data are collected in a nonuniform tesselation established by the perspective pinhole camera. (b) The warped temporary grid Υ is constructed to avoid the perspective nonlinearities. (c) The uniform measurement grid Ψ_k is obtained by unwarping grid Υ.

Earlier work on sensor fusion for robot navigation and object modeling for robotic manipulation have successfully explored a similar stochastic model structure [28]. In visual communications, the probabilistic approach for measurement integration is seldom used. In [27], the outputs of multiple range-sensing methods are integrated using a probabilistic framework assuming that all observations are registered or taken along a single direction. 3DVC integrates multiple unregistered measurements taken from distinct points of view. Preliminary experiments with synthetic data applying the 3DVC model structure to video representations were reported in [29].

Measurements: We assume that a range sensor produces a depth map R and a coregistered image I. By coregistered, we mean that the pair of measurements is taken with respect to the same reference, through the same perspective camera, and that for every intensity image pixel, there is a corresponding depth measurement and vice versa.

The availability of the depth map simplifies the experimental setup. Alternatively, the depth map can be estimated directly from the video sequence using structure from motion [30]. If multiple cameras are available, depth is estimated in real time using stereo [23] or for added precision using multibaseline stereo [31]. Another alternative is to use depth from defocus techniques. These employ a single camera with controllable focus, and have been successfully implemented in real time [32]. 3DVC supports multiple heterogeneous depth sources as detailed in Section III-B.

A depth map R is defined as a set of points $\{\vec{r}\}$ in space. Similarly, an image I is an organized set of color or intensity measurements $\{\vec{t}\}$, where each measurement $\vec{t}$ assumes one of the colors listed in pallete $\mathcal{C}$. For color images, the pallete is a set of tridimensional vectors. For gray-level images, it is a set of scalars.

Intensity (or color) images provide information about the texture already mapped on the object surface. Texture information depends on the properties of the surface material, the viewing direction, and the environment illumination.

Sensor Model: A 3-D probabilistic sensor model characterizes the uncertainties in the data as measured by the range finder. It is given by the conditional probability function $p(\vec{r}\,|\,\vec{z})$, where $\vec{z}$ represents the ground truth and $\vec{r}$ the measurement. This sensor model is either provided by the sensor manufacturer or obtained experimentally.

Uniform Measurement Grid: The depth and intensity measurements corresponding to a video frame are organized into an auxiliary data structure that we call *uniform measurement grid*. This preprocessing step greatly simplifies the video analysis tasks by eliminating later concerns with camera geometry, sensor model footprint, and viewing direction.

As the sensor output is a set of discrete measurements, each individual measurement $(\vec{r}, \vec{t})$ corresponds to a volumetric cell in the compact 3-D space. Assuming that data are collected using a pinhole perspective camera, the dimensions of each cell in this 3-D lattice depend on the depth of the cell. Therefore, a pinhole perspective camera defines a nonuniform tesselation of 3-D space due to perspective nonlinearities; see Fig. 2 for a bidimensional example.

Dealing directly with these nonuniform tesselations is inconvenient because the tesselations depend on camera geometry and relative object-camera position. Also, model construction would require casting conic rays in distinct directions through a nonuniform voxel grid.

To avoid these difficulties, we collect the information available in each coregistered set of measurements (R_k, I_k) into a uniform measurement grid Ψ_k, taking into account the camera geometry and the sensor model.

We define Ψ_k, the uniform measurement grid associated with the kth set of measurements, as a compact uniform tesselation of the 3-D space. This tesselation is an organized set of cells $\Psi_k \triangleq \{M_i\}$ where the index i spans the compact 3-D space. Each measurement grid cell M_i represents multiple properties. Cell occupancy $O(M_i)$ defines the object shape, and cell texture $T(M_i)$ represents surface texture. Occupancy is a binary property: a cell may be either *occupied* or *empty*. Texture values are drawn from the color pallete $\mathcal{C}$. The structure of the measurement grid is similar to the structure of the object model described at the beginning of this section. The only difference regards the probabilities each cell carries. A measurement cell M_i carries the conditional probabilities of the current measurement given cell occupancy $p(\vec{r} \mid O(M_i) = \text{occupied})$ and cell emptiness $p(\vec{r} \mid O(M_i) =$

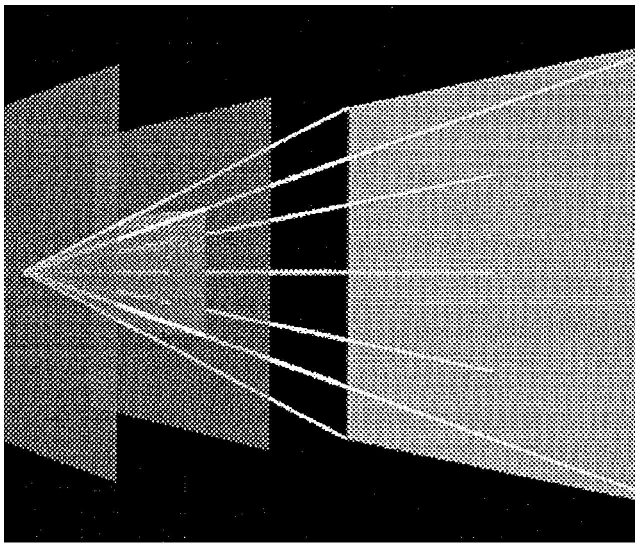


Fig. 3. Perspective camera model: viewing frustum in unwarped space.

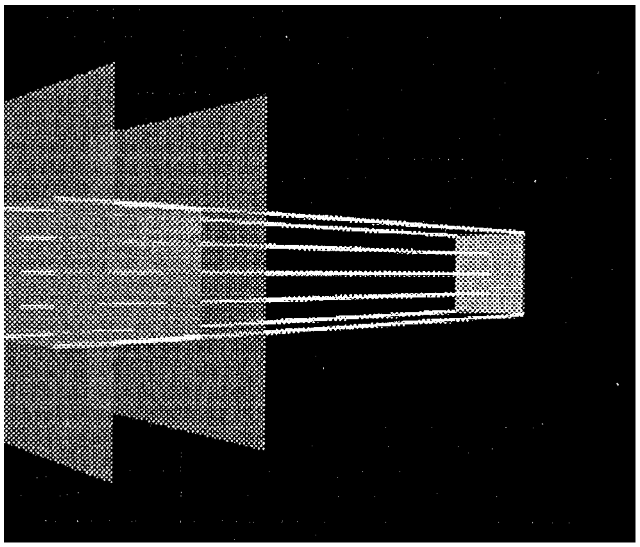


Fig. 4. Perspective camera model: viewing frustum in warped space.

empty). These probabilities are derived from the sensor model, and are fundamental to measurement integration in object model construction. Further discussion about the computation and role of these probabilities in measurement integration is carried out in Section III-B.

The measurement grid is aligned with the sensor principal axis. Given the depth map R and the image I, and with camera calibration and sensor model known, we compute the texture and conditional probabilities for each cell M_i of the measurement grid Ψ_k as follows.

The creation of measurement grid Ψ_k is realized in two steps. In the first step, the measurement data are used to create a warped temporary grid Υ. The warping is defined to cancel the perspective nonlinearities. In the second step, the warped grid Υ is unwarped via resampling to generate the uniform tesselation Ψ_k. Fig. 2 shows the two steps in the construction of a bidimensional measurement grid. Fig. 2(a) presents the nonuniform grid defined by the perspective pinhole camera associated with the input data. Fig. 2(b) displays the auxiliary warped grid Υ constructed by placing the individual measurements in a warped space where all cells have the same dimension. Fig. 2(c) presents the resulting uniform measurement grid Ψ_k obtained by unwarping grid Υ. Observe that both grid Ψ_k and grid Υ are uniform grids. The construction of Ψ_k could be implemented directly from the data, but the auxiliary grid Υ simplifies the computation of the probability distributions associated with the sensor model that are stored in the measurement grid.

In the perspective pinhole camera model, all visual rays reaching pixels in the retinal plane pass through the camera center. For a set of rectangular pixels, the set of visual rays forms a pyramid. The section of this pyramid delimited by the focal and retinal planes, shown in Fig. 3, is known as the viewing frustum.[2]

The warping and unwarping transformations used in measurement preprocessing are given by the camera geometry. We define the warping transformation such that the shape of the viewing frustum is mapped from a pyramid in the original unwarped space to a rectangular orthogonal prism in the warped space; see Figs. 3 and 4. Through this mapping, the viewing rays become parallel in the warped grid Υ and reach the retinal plane perpendicularly. This warping cancels the nonlinear effects of perspective, simplifying the construction of the measurement grid and avoiding generation of artifacts in the final object model Γ.

To construct the auxiliary grid Υ, we cast a ray parallel to the principal axis for each individual measurement $(\vec{r}, \vec{t})$ transversing the warped space through the point $\vec{r}$. We update texture and probabilities of occupancy for all cells hit by the transversing ray.

As all viewing rays are parallel and aligned with the grid Υ, ray casting becomes much simpler in the warped space. The set of cells from Υ that gets updated with nontrivial probabilities due to a given pointwise measurement $(\vec{r}, \vec{t})$ depends on the sensor model footprint. In the warped space, the direction of the sensor model is always parallel to the sensor principal axis, simplifying the identification of the cells affected by each individual measurement.

After Υ is constructed, it is unwarped to provide a properly resampled grid Ψ_k.

This method for 3-D model generation from measurements taken under perspective using an intermediate warped space is motivated by the shear-warp volume rendering algorithm [33]. In a volume renderer, 2-D views are generated from existing 3-D object models. The measurement grid creation method presented here is an *inverse volume renderer,* where 3-D point models are created based on the available 2-D measurements and knowledge about the sensor model.

We create a distinct measurement grid Ψ_k for each new measurement before integrating the new measurement into the object model.

The video analysis methods proposed for object model construction given a set of unregistered measurement grids are detailed in the following section.

[2]Frustum: a part of a solid, such as a cylinder or pyramid, between two parallel planes cutting the solid.

III. Video Sequence Analysis

For each segmented object, the process of model construction requires two steps for each new measurement: registration and integration.

Measurement registration, or pose estimation, estimates the current position and orientation of the object in 3-D space with respect to an arbitrary model-centered coordinate system, given the current measurement and a 3-D geometric model.

Integration of the registered measurement updates the current model by incorporating the new measurements. The first task in this step is to map the registered measurement to a canonical model-centered referential using the pose estimate computed in the registration step. The actual integration method is heavily dependent on the model structure. Distinct deterministic techniques have been proposed for the integration of range measurements into surface-based models [24], [34]–[36]. Stochastic integration techniques, as the one we propose in this work, consider a probabilistic sensor model, and are based on Bayesian updating or Kalman filtering [28], [29]. In these techniques, redundant measurements help to reduce overall model entropy, while conflicting or ambiguous measurements are handled gracefully.

A. Measurement Registration: Iterative Pose Estimation Method

Registration is dual to the problem of computing point correspondences between overlapping regions of a model and a measurement grid. Given a set of correspondences, there is a closed-form solution to the pose problem [37], [38]. With precise correspondences, it is possible to compute a mapping to juxtapose the new measurement and the existing model. This duality highlights the importance of accurate registration for successful model generation.

Registration is usually accomplished by minimizing a cost function based on discrepancy metrics between potential model-measurement correspondences. Traditional registration algorithms use only shape and adopt Euclidean distances as measure of discrepancy.

Each new measurement must have some overlapping area with the model in order to allow for the establishment of a large set of reliable correspondences. This assumption is satisfied if motion is small between successive measurements.

The iterative closest point algorithm (ICP), introduced simultaneously by several groups [39]–[42], solves the minimization problem by hypothesizing correspondences to be the closest points iteratively. An important issue not addressed by most ICP implementations is the handling of incomplete models, where measurement-model correspondences may not exist for all individual measurements. This is frequently the case in applications where registration is performed when models are still under construction. The approach we present below handles this problem.

Some extensions of ICP reduce the cost of closest point search by operating only on selected features, or by creating search index trees. A fast implementation of the ICP algorithm capable of pose estimation at 10 Hz has been reported [43]. Recently, ICP has been extended to employ a generalized distance function that considers surface normals as additional cues for increased reliability in pose estimation [44].

Other registration and motion estimation methods are based on features and on factorization [19]. These methods require precise feature extraction and the solution of large-dimensional singular value decompositions, which are computationally expensive tasks. For surveys on registration methods, refer to [39], [45], and [46].

We propose a method for model-measurement registration that extends the ICP algorithm. Our method uses texture and shape for pose estimation, and deals with incomplete or inconsistent models.

To consider texture and shape information, we define an intercell discrepancy metric $d(C_i, C_j)$ as

$$d(C_i, C_j) = (1-\lambda)\|\vec{r}_i - \vec{r}_j\|^2 + \lambda\|\vec{t}_i - \vec{t}_j\|^2 \tag{6}$$

where $\vec{r}_i$ is the position of cell C_i in 3-D with respect to a model-centered coordinate system, and $\vec{t}_i$ is the color of cell C_i. The parameter λ controls the relative importance of shape and texture in the discrepancy criterion. Notice that both positions $\vec{r}_i$ and $\vec{r}_j$ must be taken with respect to the same model centered reference.

Given an arbitrary pose $\vec{q}$, the generalized distance $D(\Gamma, \Psi_k, \vec{q})$ between a model Γ and the kth measurement Ψ_k is defined as

$$D(\Gamma, \Psi_k, \vec{q}) = \sum_{\{(i,j)|(C_i, M_j)\in\mathcal{S}\}} d(C_i, \mathcal{M}_{\vec{q}}(M_j)) \tag{7}$$

where C_i is a cell from model Γ, M_j is a cell from measurement Ψ_k, $\mathcal{S}$ is the set of correspondences (C_i, M_j), and $\mathcal{M}_{\vec{q}}$ is the mapping derived from the arbitrary pose $\vec{q}$. It is important to notice that the set of correspondences $\mathcal{S}$ also depends on the pose $\vec{q}$.

To handle models that are potentially incomplete, we modify the definition of generalized distance presented in (7) to consider only correspondences whose distances lie within a plausible range $d_{\max}$ as in

$$D(\Gamma, \Psi_k, \vec{q}) = \sum_{\{(i,j)|(C_i, M_j)\in\mathcal{S}, d<d_{max}\}} d(C_i, \mathcal{M}_{\vec{q}}(M_j)). \tag{8}$$

This modification removes false correspondences and outliers from the computation of the generalized distance. As motion is assumed small, correspondences based on shape cannot be too far apart. Texture variations due to differences in illumination are usually very small. Larger differences in texture, as specular reflection highlights, are localized spots that are considered outliers according to this improved criterion. Zhang [42] successfully applied a similar outlier removal technique to pose estimation based only on shape information. In his proposal, the threshold $d_{\max}$ is determined by identification of the clusters of outliers and matching points in the distances histogram for every frame. In our experiments, we adopted a simpler technique by normalizing the distances to lie between [0,1] and fixing the threshold.

The pose estimate is the argument $\vec{q}_k$ that minimizes the generalized distance

$$\vec{q}_k = \arg\min_{\vec{q}}\{D(\Gamma, \Psi_k, \vec{q})\}. \tag{9}$$

Pose Estimation Algorithm: Before the first measurement Ψ_0, the model Γ holds no information, so the object position and orientation are assumed to be in an arbitrary canonical position $\vec{q}_0$.

For each subsequent measurement Ψ_k, we perform the minimization in (9). As ICP performs a local minimization, a good initial estimate is important to ensure reliable results. When processing a sequence of measurements under the assumption of small motion between measurements, a good initial value for pose is the pose estimate obtained from the previous measurement $\vec{q}_{k-1}$.

Binary index trees are useful structures to speed up read transactions. To reduce the computational cost of searching for the closest cell, we construct an auxiliary four-dimensional binary tree for the model data.

Due to space constraints, the algorithm details are not reported here. For additional information and experimental results regarding the proposed pose estimation method, refer to [47].

B. Measurement Integration: Incremental Object Model Construction

We integrate the set of registered measurement grids $\{(\Psi_0, \vec{q}_0), \cdots, (\Psi_k, \vec{q}_k)\}$ into the object model Γ_k using a Bayesian framework. We construct the model Γ_{k+1} incrementally using the previous model version Γ_k and the new registered measurement $(\Psi_{k+1}, \vec{q}_{k+1})$ as it becomes available.

We consider initially a single measurement. With a uniform cost function, the optimal Bayes' estimator for variable $\vec{z}$ given the measurement $\vec{r}$ is the MAP estimator $\hat{z}$ [48]

$$\hat{z} = \arg\max_{\vec{z}} p(\vec{z} \mid \vec{r}) \tag{10}$$

$$p(\vec{z} \mid \vec{r}) = \frac{p(\vec{r} \mid \vec{z})p(\vec{z})}{p(\vec{r})}. \tag{11}$$

The conditional probability distribution $p(\vec{r} \mid \vec{z})$ is the sensor model previously described. The prior knowledge about the parameter $\vec{z}$ is given by the prior probability $p(\vec{z})$. The probability $p(\vec{r})$ is a normalization factor.

If multiple measurements $\vec{r}_0, \cdots, \vec{r}_k$ are available, the MAP estimate is computed as follows. We introduce the notation

$$p(\vec{z})_k \triangleq p(\vec{z} \mid \{\vec{r}_0, \cdots, \vec{r}_k\}) \tag{12}$$

$$\hat{z}_k = \arg\max_{\vec{z}} p(\vec{z})_k. \tag{13}$$

Assuming independent measurements and applying Bayes' theorem, the incremental update rule follows:

$$p(\vec{z})_{k+1} = \frac{p(\vec{r}_{k+1} \mid \vec{z})}{p(\vec{r}_{k+1})} p(\vec{z})_k. \tag{14}$$

The estimate $\hat{z}_k$ is computed by (13), and is recursively updated in time through (14).

We now consider the estimation of the whole object shape $\hat{O}(\Gamma)$ from a single measurement $\vec{r}$. We introduce the notation $\hat{O}(\Gamma) \triangleq \{\hat{O}(C_i), \forall i\}$ to represent the occupancy state of the whole tesselation Γ, given by (17). Let $P(O(\Gamma) \mid \vec{r})$ be the probability distribution of the set of discrete variables given by (15)

$$P(O(\Gamma) \mid \vec{r}) = \frac{p(\vec{r} \mid O(\Gamma))P(O(\Gamma))}{p(\vec{r})} \tag{15}$$

$$p(\vec{r}) = \sum_{\forall \Gamma_j \in \Gamma} p(\vec{r} \mid O(\Gamma_j))P(O(\Gamma_j)) \tag{16}$$

$$\hat{O}(\Gamma) = \arg\max_{O(\Gamma)} P(O(\Gamma) \mid \vec{r}). \tag{17}$$

Unfortunately, the direct estimation of the occupancy state for the whole tesselation is a problem doomed by the curse of dimensionality. This can be noticed in (16), where the normalization factor $p(\vec{r})$ is a sum over all possible tesselation configurations Γ. As occupancy is a binary property, the number of possible occupancy configurations for a given tesselation is 2^N, where N is the number of cells in the tesselation. Typical tesselation sizes may be on the order of millions of voxels. To make this problem computationally tractable, we factor $P(O(\Gamma) \mid \vec{r})$ in (15), approximating the estimation of the occupancy status of each individual cell $\hat{O}(C_i)$ independently as follows:

$$\hat{O}(C_i) = \arg\max_{O(C_i)} P(O(C_i) \mid \vec{r}) \tag{18}$$

$$P(O(C_i) \mid \vec{r}) = \frac{p(\vec{r} \mid O(C_i))P(O(C_i))}{p(\vec{r})}. \tag{19}$$

To compute the probability $p(\vec{r} \mid O(C_i))$ required in (19), we resample the measurement grid $\Psi_k = \{M_i\}$ using the mapping $\mathcal{M}_{\vec{q}_k}$ derived from the pose estimate $\vec{q}_k$

$$p(\vec{r} \mid O(C_i)) = \mathcal{M}_{\vec{q}_k}[p(\vec{r} \mid O(M_i))]. \tag{20}$$

Recalling the structure for the measurement grid Ψ_k proposed in Section II, each cell carries probabilities of measurement conditioned to occupancy $p(\vec{r} \mid O(M_i) = \text{occupied})$ and emptiness $p(\vec{r} \mid O(M_i) = \text{empty})$. These probabilities are computed using the sensor model according to

$$p(\vec{r} \mid O(M_i)) = \sum_{G \in \{G(M_i)\}} p(\vec{r} \mid O(M_i), G)P(G) \tag{21}$$

$$p(\vec{r} \mid G) = p(\vec{r} \mid \vec{z}_{\min}) \tag{22}$$

where, for a given configuration G, $\vec{z}_{\min}$ is the depth of the nearest occupied cell.

Due to sensor uncertainties, captured by the sensor model, a single measurement may affect several cells in the object model during integration. Which cells are affected depends on the sensor model footprint, on the sensor orientation during data acquisition, and on the measurement value. The measurement grid is constructed such that the sensing direction is always parallel to one of the tesselation axes, which eliminates the concerns regarding sensor model orientation.

When multiple measurements are available in the form of a single measurement grid Ψ, we extend (19) assuming independent measurements and compute $P(O(C_i) \mid \Psi)$.

The integration of a new measurement grid Ψ_{k+1} is realized incrementally. The probability of occupancy $P(O(C_i))_{k+1}$ is computed for each cell C_i of the model Γ_{k+1}, according to the incremental update rule

$$P(O(C_i))_{k+1} = \frac{P(\Psi_{k+1} \mid O(C_i))}{P(\Psi_{k+1})} P(O(C_i))_k \tag{23}$$

where the conditional probability $P(\Psi_{k+1} \mid O(C_i))$ is computed from the measurement grid Ψ_{k+1} via resampling. The prior probability of occupancy $P(O(C_i))_k$ is available at the current model Γ_k, and the probability $P(\Psi_{k+1})$ is a normalization factor.

When multiple distinct sensors are available, we generalize (11) and (22) to consider the corresponding sensor models $p_\alpha(\vec{r} \mid \vec{z})$.

The only cells visible in the process of scene assembly are the occupied cells. Therefore, only cells that are occupied after the occupancy update need to have their texture updated. For all occupied cells in Γ_{k+1}, we obtain a new texture value by resampling the measurement grid Ψ_{k+1} as in

$$T(C_i)_{k+1} = \mathcal{M}_{\vec{q}_k}[T(M_i)_{k+1}]. \tag{24}$$

This texture update policy keeps models consistent with the most recent measurement. This may generate some model updating overhead for moving specular surfaces or when illumination changes over time. Alternative Bayesian texture update policies can be applied to minimize model updates, thus penalizing temporal model consistency.

IV. Video Sequence Synthesis

To reconstruct the original video sequence, two major tasks are performed for every video frame. First, a 3-D synthetic scene is assembled using information stored in the 3DVC entities for the current frame. Then, the 2-D video frame is rendered as a perspective projection of the 3-D synthetic scene.

Scene Assembly: A scene is a uniform tesselation of 3-D space with structure identical to an object model, as described in Section III.

A synthetic 3-D scene is assembled by positioning 3-D object models in space according to the pose estimates for the current frame. Due to potential model-scene grid misalignment, all models are resampled using the scene as sampling grid. For every cell of the scene, occupancy and texture are computed as an MAP estimate.

In the process of sequential scene assembly, only models that change, either by moving in space or by changing its shape or texture, are deleted from and reinserted in the synthetic scene. Taking into account the visibility of objects in the scene with respect to current camera position can further reduce the cost of scene assembly.

Frame Reconstruction: Given an assembled scene S_k, the corresponding frame F_k is reconstructed by computing the scene perspective projection. This is accomplished by a simplified first-opacity volume renderer operating in warped space.

In a conventional first opacity volume renderer, for every pixel in the projective plane, a ray is cast from the camera center toward the scene passing through the given pixel. When the ray hits the first occupied cell in the scene, the color of the pixel in the projective plane is computed by merging contributions from all of the light sources in the scene. The result is known as the rendering equation solution [49].

As we have texture information stored in each occupied cell of the scene, whenever a ray reaches an occupied cell, there is no need to solve the rendering equation. The solution captured by the video camera was recorded in the object model during the video analysis phase.

The available camera calibration provides information about the geometry of the camera frustum. This information is used to warp the assembled scene. This warp is dual to the one described in Section II, and reduces the overall complexity of the rendering process.

In summary, to retrieve pixel color for the current 2-D frame from the 3-D scene model, we prewarp the scene grid, and then trace parallel rays through every pixel of the 2-D frame. When the ray hits the first occupied cell of the warped scene model, we read the texture value for this cell.

V. Experiments

We implemented the 3DVC analysis and synthesis modules in "*C*" on a DEC AlphaStation 200 4/233. To demonstrate the potential of 3DVC for video representation, we use synthetic and real video sequences. In both data sets, the scene is composed of a single rigid object performing 3-D motion in front of a static background.

Synthetic Video Sequence: To create the synthetic sequence, we use constructive solid geometry (CSG) primitives, as cylinders, cones, and disks to define a 3-D object with the shape of a lamp. We create an animation script where the object performs a rotation around the horizontal axis. This motion pattern is chosen because it is especially difficult to be treated by waveform-based and layered representations, and provides abundant self-occlusion episodes.

The synthetic video and range sequence was rendered using *Rayshade* [50], a raytracer that uses the Z-buffer algorithm [49] to generate realistic synthetic images. The Z-buffer values are available, providing ground truth depth information. We use this sequence to evaluate the performance of 3DVC without the uncertainties that arise in actual range sensing.

The synthetic coregistered video and range sequence is composed of 36 frames, each of dimension 100 $\times$ 100 pixels, 8 bits/pixel. Six sample video frames extracted from the complete sequence are shown in Fig. 5, and the corresponding depth maps are shown in Fig. 6.

We specify the geometry of the camera-viewing frustum through a set of raytracer parameters. Knowledge of the intrinsic camera parameters is required to define the warp mapping required by 3DVC analysis and synthesis methods.

In this experiment, the depth sensor is considered approximately ideal, and segmentation is obtained by thresholding depth information. The ideal sensor model is not adopted to avoid the persistence of spurious artifacts that may arise due to imprecisions in pose estimation. The parameter λ is set to 0.1, giving more importance to shape information during registration because the texture surface of the object in

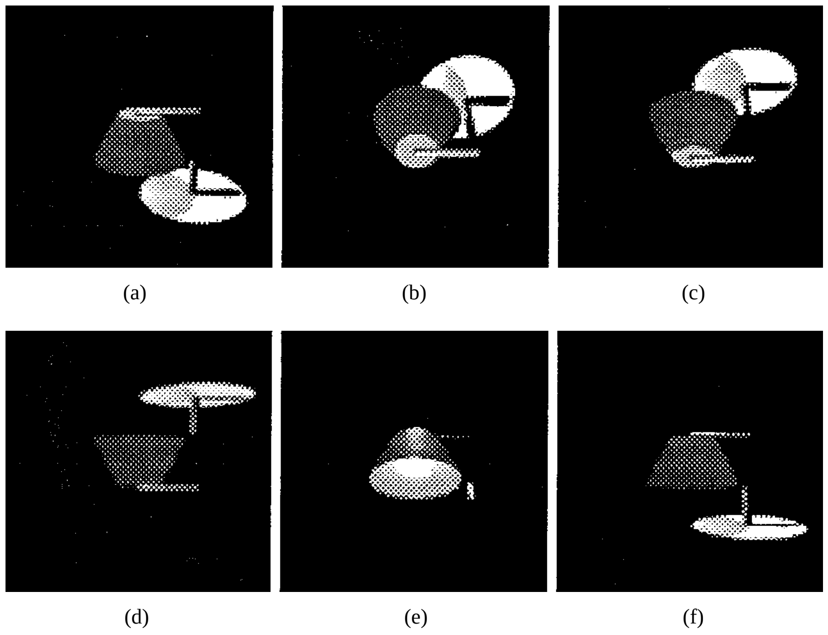


Fig. 5. Selected intensity frames from synthetic video sequence.

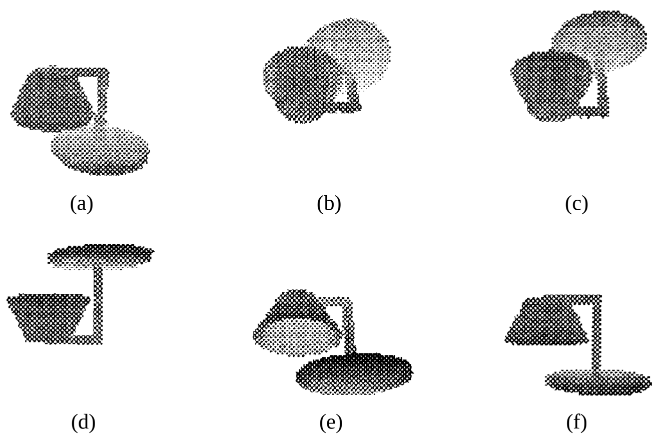


Fig. 6. Selected range frames from synthetic video sequence.

this particular sequence lacks distinctive features. Using these assumptions, the stochastic model Γ_k is incrementally built according to the method described in Section III.

In order to visualize the constructed 3-D probabilistic object models, we create a visual representation using CSG primitives. For each cell C_i of the model Γ_k, we create a small sphere positioned in 3-D space according to the position of the cell, with radius proportional to the occupancy probability. To avoid excessive image cluttering, we subsample the grid by 2 in every direction. Fig. 7 presents a pair of views of the model after the first two frames are registered and merged into the model. We notice that even though model Γ_2 is clearly incomplete, it carries sufficient information for the reconstruction of the two frames already processed. In Fig. 8, the model is observed after 30 frames have been processed. We notice a higher level of model completeness in Γ_{30} when compared to Γ_2.

The original video sequence is reconstructed using the method proposed in Section IV. Samples of the reconstructed sequence are presented in Fig. 9. A comparison with the original sequence reveals that some artifacts are introduced, especially near the object edges.

These artifacts are due to two major factors: aliasing and resampling. *Aliasing* artifacts are introduced by the discrete nature of the model. To eliminate them, we must choose voxel dimensions small enough to meet the Nyquist criterion for shape and texture frequencies. This establishes a tradeoff between model size and spatial resolution. The object models generated by this experiment are composed of approximately 4.5×10^6 voxels, of which 4879 voxels were occupied in model Γ_2 and 13 986 were occupied in model Γ_{30}. *Resampling* is required because the motion script may require the model grid to be placed in a nonaligned position with respect to the scene grid. Proper resampling requires reconstruction of the field for all new sampling points using the currently available set of samples. Each reconstruction entails taking into account contributions from all samples available, i.e., one must compute the sum of the values of a set of sync functions, each centered at one of the voxels that compose the model. This is a costly procedure, given that the models constructed usually have large voxel counts. This procedure is simplified by local weighted averaging or trilinear interpolation. These simplifications reduce significantly the computational cost, but introduce artifacts near the object edges.

Real Video Sequence: In this experiment with real data, we use a video and coregistered range sequence obtained by using a light stripe range finder with a liquid crystal shutter and a color CCD video camera.

To have control of object pose during data acquisition, we mounted the object of interest at the end effector of a precise computer-controlled robotic arm. To create the sequence, we define a simple motion script where the object performs a complete 360° rotation around the vertical axis.

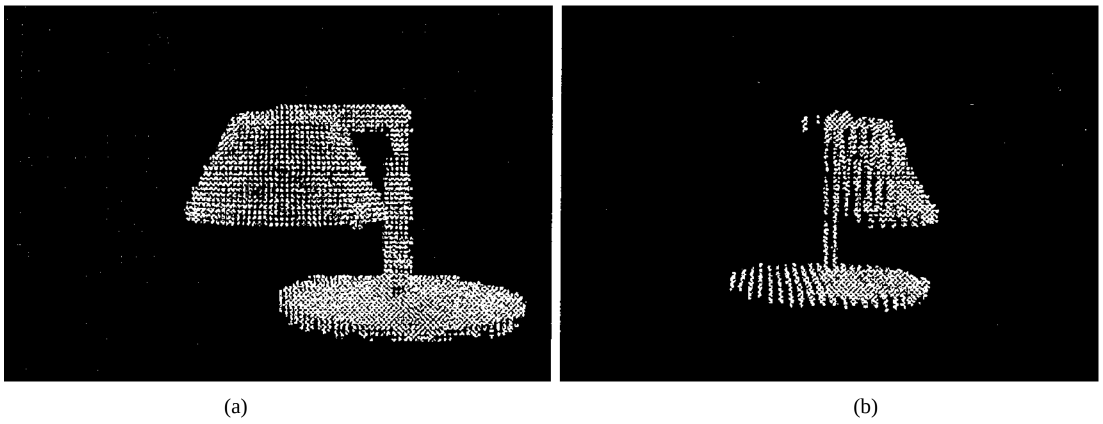


Fig. 7. Frontal and side views of object model Γ_2, obtained after integration of two frames.

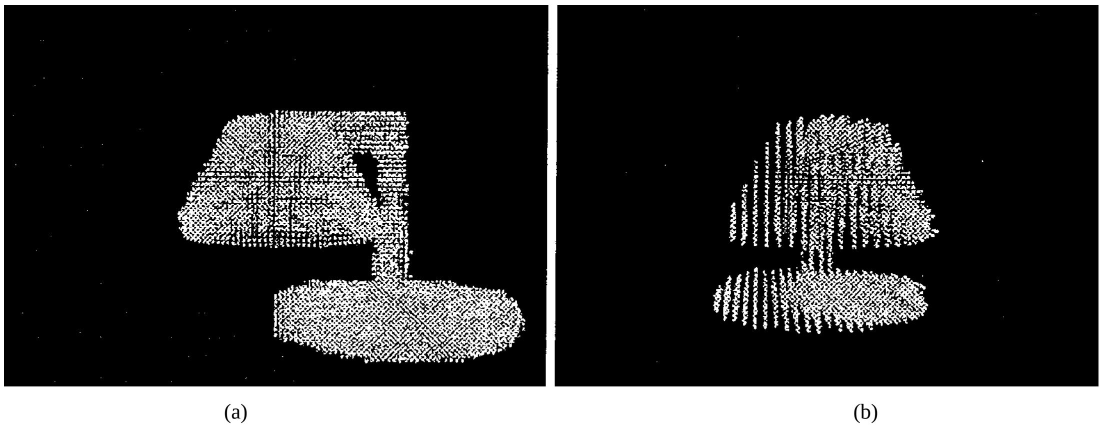


Fig. 8. Frontal and side views of object model Γ_{30}, obtained after integration of 30 frames.

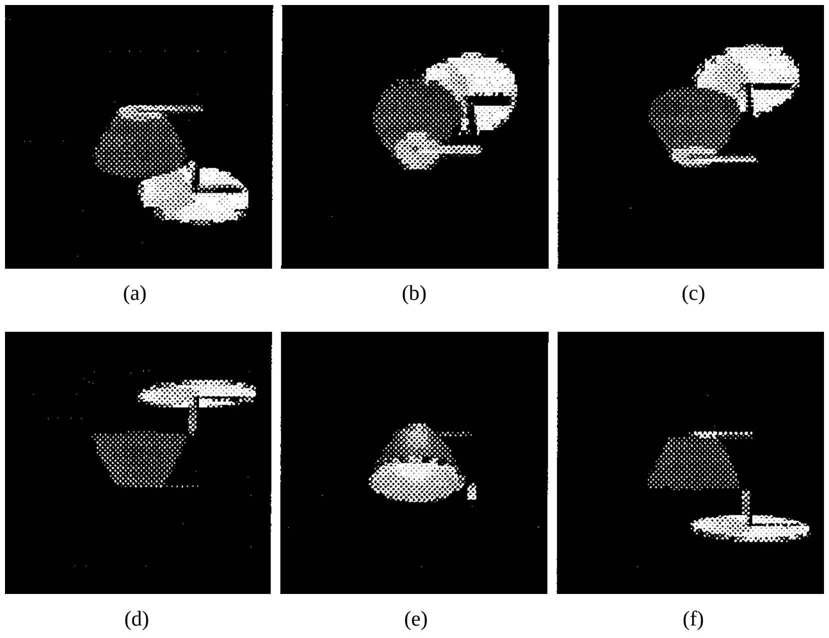


Fig. 9. Frames reconstructed using object models and pose estimates.

Camera calibration is obtained using a calibration box of known size and shape. The calibration procedure produces a projection matrix that represents the transformation between world and image coordinate systems. This information is required to determine the warping mapping used by 3DVC in measurement grid creation and frame rendering.

The coregistered range and video sequence has 256×256 pixels/frame, 8 bits/pixel. Six sample video frames extracted

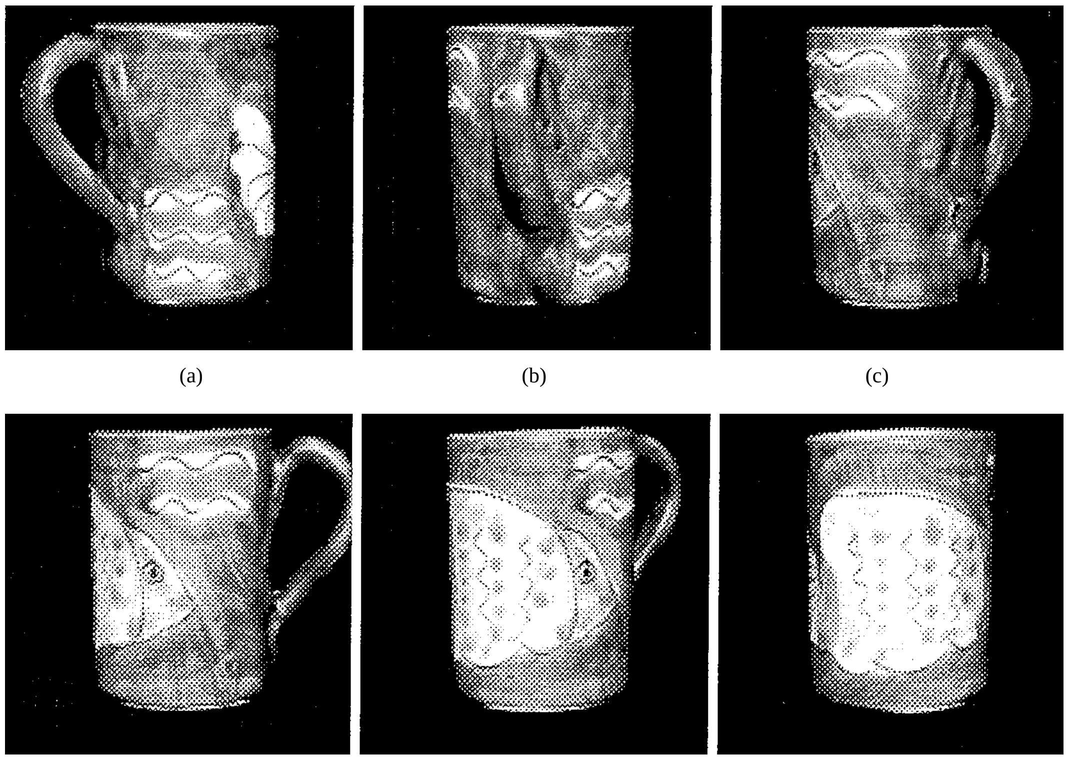

(a) (b) (c) (d) (e) (f)

Fig. 10. Selected intensity frames from real video sequence.

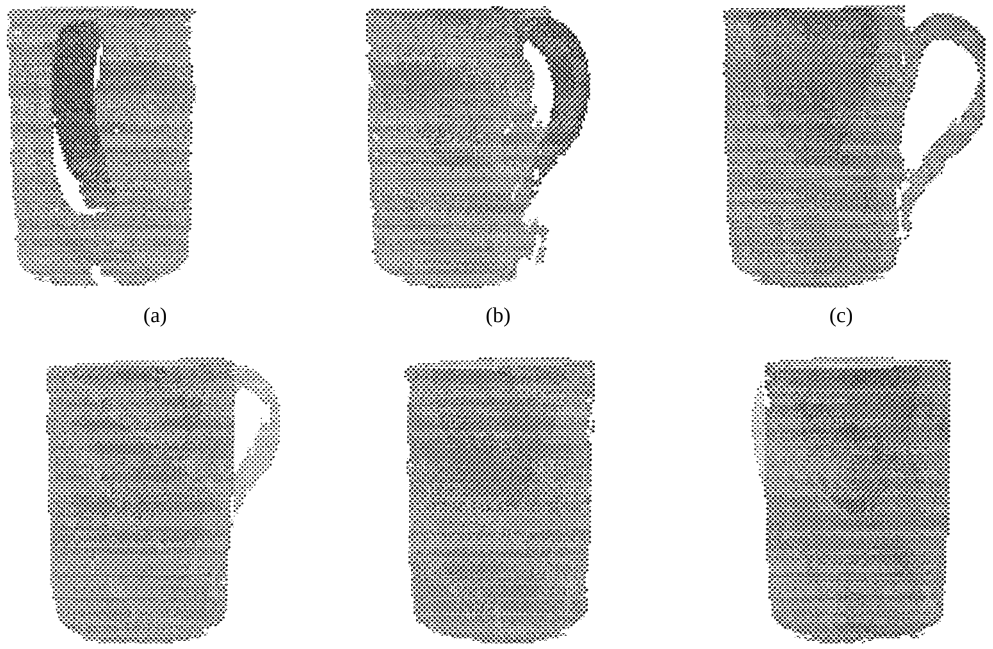

(a) (b) (c) (d) (e) (f)

Fig. 11. Selected range frames from real video sequence.

from the complete sequence are shown in Fig. 10, and the corresponding depth maps are shown in Fig. 11.

As before, the object is segmented by thresholding depth information. The parameter λ is set to 0.5, giving equal importance to shape and texture information during registration. We assume a zero-mean unit variance Gaussian sensor model, and apply the 3DVC analysis method to incrementally construct the stochastic model Γ_k.

Using the same technique for object model visualization described in the previous experiment, we generate a pair of views of the model after two and eight frames are processed; these views are presented in Figs. 12 and 13, respectively. We

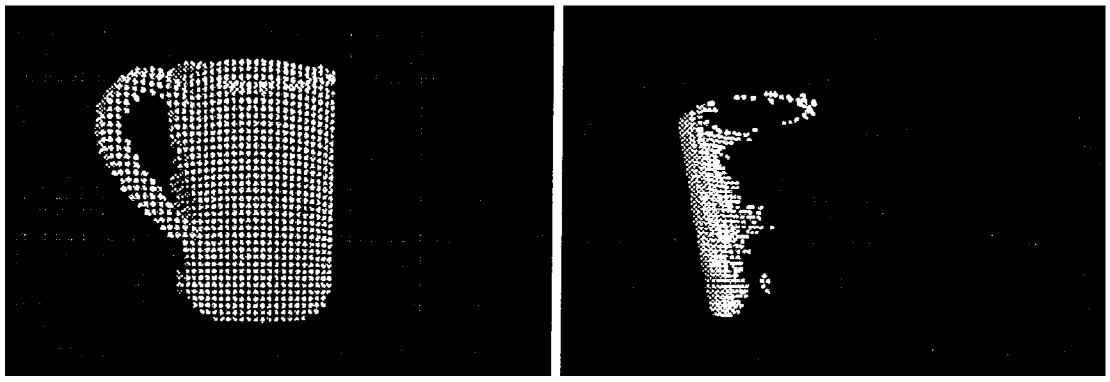

(a) (b)

Fig. 12. Frontal and side views of object model Γ_2 obtained after integration of the first two frames.

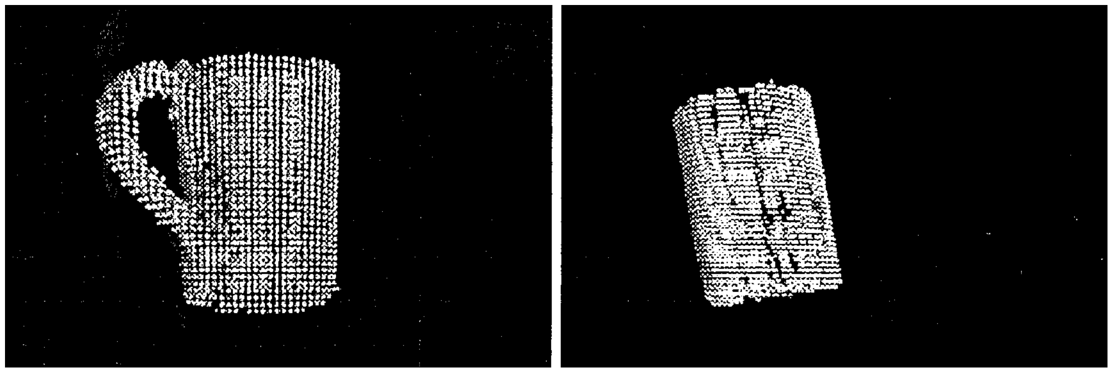

(a) (b)

Fig. 13. Frontal and side views of object model, obtained after integration of all frames.

notice that model Γ_k is sufficient to reconstruct the already processed frames.

Samples of the video sequence reconstructed by 3DVC are shown in Fig. 14. As with the synthetic data, artifacts are introduced near the object edges.

VI. 3DVC Applications

Video Coding: A video codec based on 3DVC implements the video analysis module at the transmitting end. The video synthesis module reconstructs the video sequence at the receiving end. We send through the channel only the required object model updates and pose estimates.

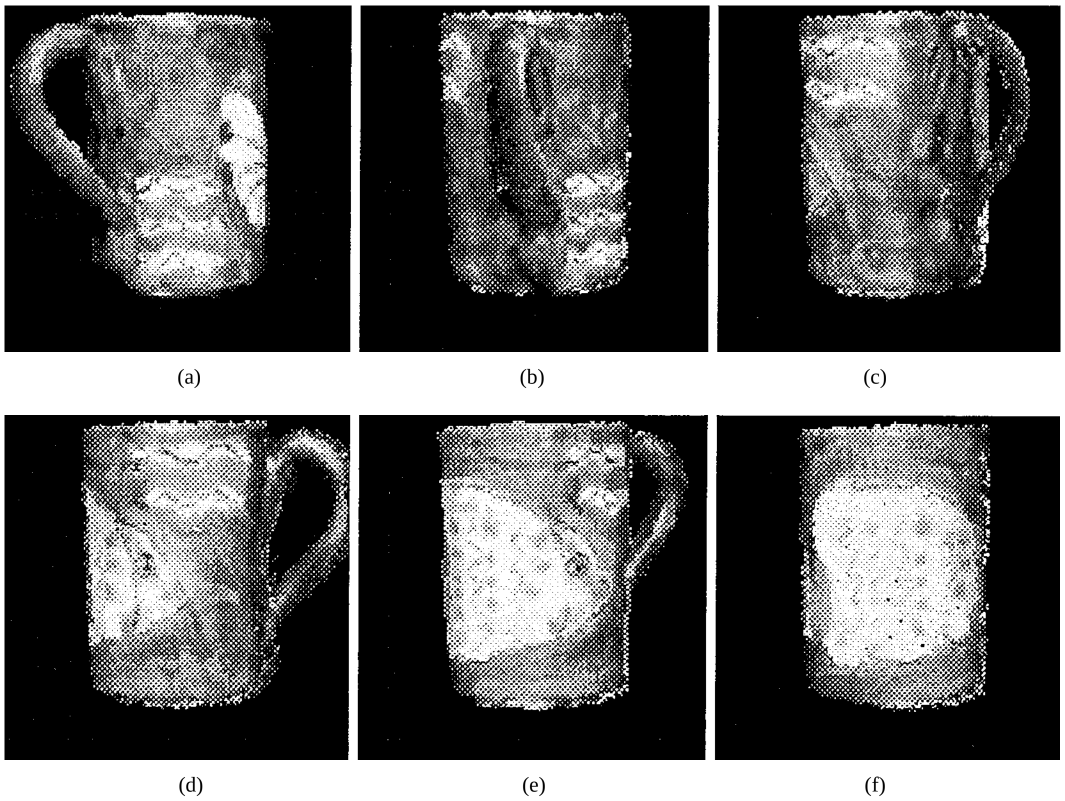
The codec achieves high intraframe compression because the size of the object model updates reduces as the model becomes complete, and the pose estimate specification requires a small amount of data that is independent of the frame size.

During model construction, the occupancy entropy decreases from frame to frame proportionally to the new information about the model in each frame. If the object performs a motion pattern that displays the same regions of the object in a repetitive fashion, the model entropy and voxel count reach a steady state after several frames are processed. In our experiments with a rotating mug, after the first 360° rotation, the model voxel count reaches steady state; refer to Fig. 15.

Ideally, after the model reaches steady state, only pose information is required for reconstruction. A six-degree-of-freedom pose estimate is represented by six floating-point variables, i.e., 24 bytes of information. Without further applicable motion prediction via Kalman filtering, or lossless compression, this leads to the compression ratio of $\mathcal{C}_{\text{ideal}} = (24F_m/S_{\text{raw}}F)$. Here, F_m is the number of frames with interframe motion, F is the total number of frames, and S_{raw} is the byte count for a given input video frame. In our real data experiment, each frame has $S_{\text{raw}} = 65\,536$ bytes and, as the object keeps moving continuously, all frames require transmission of the pose estimate. For this scenario, the codec requires $\mathcal{C}_{\text{ideal}} = 1 : 2731 = 0.000366$ bytes/pixel or a bandwidth of 2.812 kbps/s at 15 frames/s.

Changes in environment illumination and objects with specular reflection cause modifications to the surface texture from frame to frame. Imprecision in pose estimation and range sensing generates ambiguous information that leads to alterations in occupancy. Due to these factors, models usually do not reach the ideal steady state, where no update information is required, but reach a distinct steady state that requires a fraction of the voxels to be updated from frame to frame.

To avoid the overhead of transmitting model update information every frame, we implement an update policy based on model entropy after the model reaches this steady state. We keep updating the model at the transmitting end every frame. We only transmit information to update the model at the receiving end if the entropy difference between the models at the transmitting and receiving ends is larger than a threshold.



(a) (b) (c)

(d) (e) (f)

Fig. 14. Frames reconstructed using object models and pose estimates obtained from real data.

Frames where an update is necessary are called keyframes. To avoid reconstruction of nonkeyframes with incomplete models, we use a pipeline. The length of the pipeline determines the amount of look-ahead allowed for keyframe detection. Once a keyframe is detected, the model is immediately updated.

The number of voxels to update depends on the motion between updates. Larger motion implies larger uncertainty in motion estimates and larger illumination effects. For the real data sequence example, we observe that, for an interframe motion of 45°, approximately 5% of the occupied voxels are updated. This update policy leads to the compression ratio $\mathcal{C}_{\text{policy}}$ given by

$$\mathcal{C}_{\text{policy}} = \mathcal{C}_{\text{model}} + \mathcal{C}_{\text{ideal}} + \mathcal{C}_{\text{key}} = \frac{M + 24F + S_{\text{key}}F_{\text{key}}}{S_{\text{raw}}F} \tag{25}$$

where M is the the number of bytes required to transmit the steady-state model once, F is the number of frames, S_{key} is the average number of bytes per keyframe, and F_{key} is the number of keyframes.

In our experiments, after we detect that the model has reached steady state, we set the entropy threshold to 5000 voxels. Updating each voxel independently requires the transmission of 30 bits: 22 bits for addressing and 8 bits for texture. Addressing each voxel individually, the transmission of the steady model requires $M = 337\,500$ bytes. For the $F = 360$ frames encoded with the entropy policy, we get $F_{\text{key}} = 8$ key frames with an average size of $S_{\text{key}} = 18\,750$ bytes/key frame.

The compression ratio for this setup is $\mathcal{C}_{\text{policy}} = 1 : 47.55 = 0.02103$ bytes/pixel. We analyze each of the terms in (25): $\mathcal{C}_{\text{model}} \triangleq (M/S_{\text{raw}}F) = 0.014305$, $\mathcal{C}_{\text{key}} \triangleq (S_{\text{key}}F_{\text{key}}/S_{\text{raw}}F) = 0.006358$, and $\mathcal{C}_{\text{ideal}} \triangleq (M/S_{\text{raw}}F) = 0.000366$.

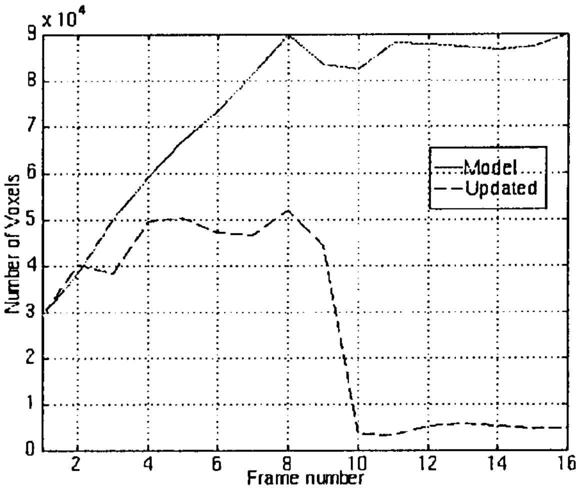


Fig. 15. Number of occupied and altered voxels in the model over time.

The $\mathcal{C}_{\text{model}}$ term can be substantially reduced. We used a naive addressing scheme to encode the model (22 bits/voxel address), which leads to a large M. Octree addressing and other entropy coding methods will reduce this overhead significantly. Further, as the number of frames F grows, $\mathcal{C}_{\text{model}}$ in (25) becomes negligible. Not accounting for this term, the remaining terms lead to a compression ratio $\mathcal{C}_{\text{policy}} = 1 : 157.29 = 0.006358$.

The $\mathcal{C}_{\text{key}}$ term is a function of the entropy threshold and scene properties. A higher entropy threshold leads to a smaller number of keyframes and to a reduced $\mathcal{C}_{\text{key}}$. Object surface and illumination characteristics define the average keyframe update size. If objects present specular surfaces, a reduction in the number of keyframes may lead to an increase in illumination artifacts in the reconstructed sequences. If these illumination artifacts are not tolerable, the $\mathcal{C}_{\text{model}}$ term can still be reduced by extending 3DVC models to carry surface reflectance properties that can be estimated as in [51]. The reduction of $\mathcal{C}_{\text{key}}$ leads to the compression ratio of $\mathcal{C}_{\text{ideal}} = 1 : 2731 = 0.000366$ bytes/pixel.

In summary, not accounting for the overhead introduced by the term M, this experiment shows that a codec based on 3DVC has a potential compression ratio in the range of 150–2700.

Video Editing and Handling: Using 3DVC, we access the 3-D structure of the scene. It is possible to insert objects not present in the scene, change the camera viewpoint, alter the motion patterns of objects, and even remove objects.

All of these alterations are constrained by model completeness. If the original sequence explored the regions occluded by an object, these regions will be visible after the given object is removed. If regions were last observed from a given viewpoint, it is likely that artifacts will be generated if the current viewpoint is completely different, especially in the presence of specular surfaces. These cases would require modeling the surface properties as in [51].

Due to space constraints, we explore and demonstrate these novel methods for video handling and editing enabled by 3DVC elsewhere.

VII. Conclusion

We introduce 3DVC, a novel system for compact video representation that provides a useful description of the 3-D structure of the scene. 3DVC decomposes the video sequence in a Bayesian framework into a set of perceptually meaningful 3-D entities: object models and motion scripts.

3DVC has interesting features: it is robust against sensor noise and estimate uncertainties, accepts explicit knowledge about sensor behavior, and allows incremental model construction. 3DVC eliminates interframe redundancy and enables content-based access to video.

We discuss the applicability of 3DVC to content-based video coding, handling, and editing. Analysis shows that the interframe compression ratio grows with video sequence length, and also depends on the complexity of scene dynamics and object persistence in the scene.

Experiments achieve high-quality results without blocking artifacts, meeting VLBR bandwidth requirements without using further applicable intraframe coding.

The proposed video analysis method is a generic tool for the efficient generation of 3-D textured models from video sequences. We currently investigate alternative applications beyond the scope of video coding, e.g., object recognition, special effects, generation of realistic models for telepresence, virtual manipulation, and 3-D facsimile.

References

[1] R. Forchheimer, O. Fahlander, and T. Kronander, "A semantic approach to the transmission of face images," in *Picture Coding Symp. (PCS'84)*, Cesson-Sevigne, France, July 1984.

[2] K. Aizawa, H. Harashima, and T. Saito, "Model-based analysis-synthesis image coding (MBASIC) system for a person's face," *Signal Processing: Image Commun.*, vol. 1, pp. 139–152, Oct. 1989.

[3] H. G. Musmann, M. Hotter, and J. Ostermann, "Object-oriented analysis-synthesis coding of moving images," *Signal Processing: Image Commun.*, vol. 1, pp. 112–138, Oct. 1989.

[4] J. Ostermann, "Object-based analysis-synthesis coding based on the source model of moving rigid 3D objects," *Signal Processing: Image Commun.*, vol. 6, no. 2, pp. 143–161, 1994.

[5] J. Y. A. Wang, E. H. Adelson, and U. Desai, "Applying mid-level vision techniques for video data compression and manipulation," Vision and Modeling Group, MIT Media Lab., Tech. Rep. 263, May 1994.

[6] R. S. Jasinschi, J. M. F. Moura, J. Cheng, and A. Asif, "Video compression via constructs," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, vol. 4, Detroit, MI, May 1995.

[7] J. M. F. Moura, R. S. Jasinschi, H. Shiojiri, and J.-C. Lin, "Video over wireless," *IEEE Personal Commun.*, vol. 6, pp. 6–16, Feb. 1996.

[8] M. Kaneko, A. Koike, and Y. Hatori, "Coding of a facial image sequence based on a 3D model of the head and motion detection," *J. Visual Commun. Image Representation*, vol. 2, Mar. 1991.

[9] V. M. Bove Jr., "Object-oriented television," *SMPTE J.*, Dec. 1995.

[10] D. V. Papadimitriou and T. J. Dennis, "Stereo in model-based image coding," in *Picture Coding Seminar*, 1994, pp. 296–299.

[11] T. Naemura, M. Kaneko, and H. Harashima, "3-D object-based coding of multi-view images," in *Picture Coding Seminar*, 1996, pp. 459–464.

[12] K. Aizawa, C. S. Choi, H. Harashima, and T. S. Huang, "Human facial motion analysis and synthesis with application to model-based coding," in *Motion Analysis and Image Sequence Processing*, M. I. Sezan and R. L. Lagendijk, Eds. Boston: Kluwer, 1993, ch. 11.

[13] H. Li, A. Lundmark, and R. Forchheimer, "Image sequence coding at very low bitrates: A review," *IEEE Trans. Image Processing*, vol. 3, pp. 589–609, Sept. 1994.

[14] P. Anandan, M. Irani, R. Kumar, and J. Bergen, "Video as an image data source: Efficient representations and applications," in *Proc. IEEE Int. Conf. Image Processing (ICIP'95)*, Washington, DC, Oct. 1995.

[15] M. Potmesil, thesis, R. P. I., Troy, NY; also, Tech. Rep. RPI IPL TR-033, 1982.

[16] B. C. Vemuri and J. K. Agarwal, "3D model construction from multiple views using range and intensity data," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognition*, IEEE Computer Society, 1986, pp. 435–437.

[17] B. Bhanu, "Representation and shape matching of 3-D objects," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 6, pp. 340–351, 1984.

[18] N. Ahuja and J. Veenstra, "Generating octrees from object silhouettes in orthographic views," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 11, no. 2, 1989.

[19] C. Tomasi and T. Kanade, "Shape and motion from image streams under orthography: A factorization method," *Int. J. Comput. Vision*, vol. 9, no. 2, pp. 137–154, 1992.

[20] A. J. Azerbayejani, T. Galyean, B. Horowitz, and A. Pentland, "Recursive estimation CAD model recovery," Perceptual Computing, MIT Media Lab., Tech. Rep. 242 Feb. 1994.

[21] S. Becker and M. Bove, "Semiautomatic 3-D model extraction from uncalibrated 2-D camera views," in *Proc. SPIE—Visual Data Exploration and Analysis II*, San Jose, CA, Feb. 1995.

[22] R. Koch, "Automatic reconstruction of buildings from stereoscopic image sequences," *EUROGRAPHICS*, vol. 12, no. 3, pp. 339–350, 1993.

[23] ——, "3-D surface reconstruction from stereoscopic image sequences," in *Proc. IEEE Int. Conf. Comput. Vision*, Cambridge, MA, June 1995.

[24] G. Turk and M. Levoy, "Zippered polygon meshes from range images," in *Proc. SIGGRAPH 21st Annu. Int. Conf. Comput. Graphics Interactive Techniques (SIGGRAPH'94)*, July 1994, pp. 311–318.

[25] B. Curless and M. Levoy, "A volumetric method for building complex models from range images," in *Proc. SIGGRAPH 23rd Annu. Int. Conf. Comput. Graphics Interactive Techniques (SIGGRAPH'96)*, ACM, Aug. 1996.

[26] A. Kaufman, *Volume Visualization*. Los Alamitos, CA: IEEE Computer Society Press, 1991.

[27] V. M. Bove, Jr., "Probabilistic method for integrating multiple sources of range data," *J. Opt. Soc. Amer. A*, vol. 7, Dec. 1990.

[28] A. Elfes, "Occupancy grids: A stochastic spatial representation for active robot perception," in *Proc. 6th Conf. Uncertainty and AI*, Cambridge, MA, AAAI, July 1990.

[29] F. C. M. Martins and J. M. F. Moura, "3-D video compositing: Toward a compact representation for video sequences," in *Proc. IEEE Int. Conf. Image Processing (ICIP'95)*, Washington, DC, Oct. 1995.
[30] J. Costeira and T. Kanade, "A multi-body factorization method for motion analysis," in *Proc. IEEE Int. Conf. Comput. Vision*, Cambridge, MA, June 1995.
[31] T. Kanade, H. Kano, S. Kimura, and K. Oda, "Development of a video-rate stereo machine," in *Proc. 1995 IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, Pittsburgh, PA, Aug. 1995.
[32] M. Watanabe, S. K. Nayar, and M. Noguchi, "Real-time computation of depth from defocus," in *Three-Dimensional and Unconventional Imaging for Industrial Inspection and Metrology*, Philadelphia, PA, SPIE, Oct. 1995.
[33] P. Lacroute and M. Levoy, "Fast volume rendering using a shear-warp factorization of the viweing transformation," in *Proc. SIGGRAPH 21st Annu. Int. Conf. Comput. Graphics Interactive Techniques (SIGGRAPH'94)*, July 1994, pp. 451–458.
[34] F. Solina and R. Bajcsy, "Recovery of parametric models from range images: The case for superquadrics with global deformations," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 12, pp. 131–147, Feb. 1990.
[35] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, "Surface reconstruction from unorganized points," in *Proc. SIGGRAPH 19th Annu. Int. Conf. Comput. Graphics Interactive Techniques (SIGGRAPH'92)*, 1992, pp. 71–77.
[36] M. Soucy and D. Laurendeau, "A general surface approach to the integration of a set of range views," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 17, pp. 344–358, Apr. 1995.
[37] B. K. P. Horn, "Closed form solution of absolute orientations using unit quaternions," *J. Opt. Soc. Amer.*, vol. 4, pp. 629–642, Apr. 1987.
[38] O. D. Faugeras and M. Hebert, "The representation, recognition, and locating of 3D objects," *Int. J. Robotic Res.*, vol. 5, pp. 27–52, Fall 1986.
[39] P. J. Besl and H. D. McKay, "A method for registration of 3-D shapes," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 14, pp. 239–56, Feb. 1992.
[40] Y. Chen and G. Medioni, "Object modeling by registration of multiple range images," *Image and Vision Computing (UK)*, vol. 10, pp. 145–55, Apr. 1992.
[41] G. Champleboux, S. Lavallee, R. Szeliski, and L. Brunie, "From accurate range imaging sensor callibration to accurate model-based 3D object localization," in *Proc. CVPR: Comput. Soc. Conf. Comput. Vision Pattern Recognition*, IEEE Computer Society, 1986, pp. 435–437.
[42] Z. Zhang, "Iterative point matching for registration of free-form curves and surfaces," *Int. J. Comput. Vision*, vol. 13, pp. 119–152, Oct. 1994.
[43] D. Simon, M. Hebert, and T. Kanade, "Real time 3D pose estimation using a high-speed range sensor," Robotics Inst., Carnegie Mellon Univ., Tech. Rep. CMU-RI-TR-93-24, Nov. 1993.
[44] J. Feldmar, N. Ayache, and F. Betting, "3D-2D projective registration of free-form curves and surfaces," in *Proc. IEEE Int. Conf. Comput. Vision*, Cambridge, MA, June 1995.
[45] B. Sabata and J. K. Aggarwal, "Estimation of motion from a pair of range images: A review," *CVGIP: Image Understanding*, vol. 54, pp. 309–324, Nov. 1991.
[46] L. G. Brown, "A survey of image registration techniques," *Comput. Surveys*, vol. 24, pp. 325–376, Dec. 1992.
[47] F. C. M. Martins, H. Shiojiri, and J. M. F. Moura, "3-D 3-D Registration of free formed objects using shape and texture," in *Conf. Visual Commun. Image Processing (VCIP'97)*, San Jose, CA, Feb. 1997, SPIE, 12 pp., to be published.
[48] H. Van Trees, *Detection, Estimation, and Modulation Theory—Vol. 1*. New York: Wiley, 1968.
[49] A. Watt and M. Watt, *Advanced Animation and Rendering Techniques*. Reading, MA: Addison-Wesley, 1992.
[50] C. E. Kolb, *Rayshade User's Guide and Reference Manual*, Princeton Univ., Jan. 1992.
[51] Y. Sato and K. Ikeuchi, "Reflectance analysis for 3D computer graphics model generation," Dept. Comput. Sci., Carnegie Mellon Univ., Tech. Rep. CS-95-146, June 1995.

Fernando C. M. Martins was born in São Paulo, Brazil, in 1965. He received the B.S. degree in electrical engineering with emphasis in digital systems, and the M.S. degree in electrical engineering with citations of distinction and honors in 1987 and 1992, respectively, from Escola Politécnica da Universidade de São Paulo, Brazil. He received the Ph.D. degree in electrical and computer Engineering from Carnegie Mellon University, Pittsburgh, PA, in 1997.

His research experience includes work in programming environments for the specification of sequential automata at Rockwell Allen-Bradley; research in neural networks, image processing, and low-level computer vision at Universidade de São Paulo, Brazil; investigation of rate control methods for very low bit-rate video coding at IBM T. J. Watson Research Center; and research and development of video coding algorithms at Intel Architecture Labs. His research interests are in image and video processing, computer graphics, computer vision, and their application to visual communications and networked multimedia. His recent research has been centered on the development of efficient representations for video sequences exploring the merger of sensor fusion, computer vision, and computer graphics techniques.

José M. F. Moura (S'71–M'75–SM'90–F'94) received the engenheiro electrotécnico degree from Instituto Superior Técnico (IST), Lisbon, Portugal, in 1969, and the M.Sc., E.E., and D.Sc. degrees in electrical engineering and computer science from the Massachusetts Institute of Technology (M.I.T.), Cambridge, in 1973 and 1975, respectively.

He is presently a Professor of Electrical and Computer Engineering at Carnegie Mellon University (CMU), Pittsburgh, PA, which he joined in 1986. Prior to this, he was on the faculty of IST where he was an Assistant Professor (1975), Professor Agregado (1978), and Professor Catedrático (1979). He has had visiting appointments at several institutions, including M.I.T. (Genrad Associate Professor of Electrical Engineering and Computer Science, 1984–1986) and the University of Southern California (Research Scholar, Department of Aerospace Engineering, Summers 1978–1981). His research interests include statistical signal processing (one and two dimensional), digital communications, image and video processing, radar and sonar, and the application of multiresolution and wavelet techniques. He has organized and codirected two international scientific meetings on signal processing theory and applications. He has over 170 published technical contributions, including invited ones in international journals and conference proceedings, and is coeditor of two books.

Dr. Moura is currently the Editor in Chief for the IEEE TRANSACTIONS ON SIGNAL PROCESSING. He was a member of the IEEE Press Board (1991–1995), a Technical Associate Editor for the IEEE SIGNAL PROCESSING LETTERS (1993–1995), and an Associate Editor for the IEEE TRANSACTIONS ON SIGNAL PROCESSING (1988–1992). He was a Program Committee member for the IEEE International Conference on Image Processing (ICIP'95) and for the IEEE International Symposium on Information Theory (ISIT'93). He is a corresponding member of the Academy of Sciences of Portugal (Section of Sciences). He is affiliated with several IEEE societies, Sigma Xi, AMS, IMS, and SIAM.