Modeling from Reality: Representing and Integration

Heung-Yeung Shum

CMU-RI-TR-96-36

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Robotics

The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

July 1996

This research was sponsored in part by the Advanced Research Projects Agency under the Department of the Army, Army Research Office under grant number DAAH04-94-G-0006, in part by ONR under grant number N00014-95-1-0591, and in part by the National Science Foundation under Contract INI-9224521. Views and conclusions contained in this document are those of the author and should not be interpreted as necessarily representing official policies or endorsements, either expressed or implied, of the United States Government.

Carnegie Mellon

Robotics

Thesis

Modeling from Reality: Representation and Integration

Heung-Yeung Shum

Submitted in Partial Fulfillment of the Requirements
for the Degree of
Doctor of Philosophy
in the field of Robotics

ACCEPTED:

Raj Reddy	Thesis Committee Co-Chair	8/21/96 Date
Katsushi Ikeuchi	Thesis Committee Co- Chair	Aug 21, 1996 Date
Matthew T. Mason	Program Chair	August 21, 1996 Date
Raj Reddy	Dean	Aug. 21, 1996 Date

APPROVED:

Paul Christiano	Provost	21 August 1996 Date

Abstract

Traditional virtual reality systems rely on manual construction of the virtual environment, which is labor-intensive and usually not very realistic. This thesis describes a "*modeling-from-reality*" system which observes from multiple viewpoints, analyzes the geometrical shape of a scene or an object, and subsequently integrates the multiple views to build a complete scene or object model from the existing environment. Our system produces statistically optimal object models by adopting a new approach to modeling from reality -- *an integral approach to object modeling*. The integral approach consists of two parts: *how to integrate* and *what to integrate*.

Using a polyhedral object as an example, this thesis shows *how to integrate* multiple views in a statistically optimal fashion. It is illustrated that multiple view integration can be formulated as a problem of *principal component analysis with missing data* (PCAMD). In spite of the noisy input and missing data at each view, the PCAMD algorithm makes use of input redundancy among different views and guarantees that the recovered object model is statistically optimal. It is shown that the problem of PCAMD can be generalized as a weighted least-squares problem, which is solved using an efficient bilinear iterative algorithm.

Matching multiple views of a polyhedral object can be accomplished by tracking its planar surface patches. It is, however, difficult to match multiple views of a free-form (i.e., smooth) object. This thesis shows *what to be integrated* from multiple views of a free-form object by employing a novel global resampling technique. A semi-regularly tessellated spherical mesh is used to uniformly resample the free-form object. This mesh representation provides a one-to-one mapping between a resampled convex/concave mesh to its curvature distribution on a unit sphere. The correspondence between two meshes can then be established by minimizing the difference between two curvature distributions. The same mesh representation also enables one to compare 3D shapes and to synthesize new shapes.

In summary, this thesis shows that PCAMD can be used to integrate multiple views to obtain a statistically optimal object model provided that those views can be resampled and matched under appropriate representations. Based on the integral approach, our "modeling-from-reality" system has been successfully applied for modeling both polyhedral and free-form objects.

Acknowledgments

First and foremost, I am most indebted to my thesis advisors, Raj Reddy and Katsu Ikeuchi. Raj took care of me soon after I enrolled in the Robotics Institute in CMU, and has given me invaluable guidance ever since. He encouraged me to pursue my own research interests in visual perception, and later suggested to invite Katsu as my co-advisor.

I can not emphasize enough how much Katsu has helped me in my professional and personal development. At the earlier years of my study he gave me clear and direct instructions to keep me focused. More recently, he provided enough freedom to keep me interested. For many years, he stood in front of my computer almost every midnight checking out what I have done each day. I will always admire his integrity, knowledge, and dedication to work.

I am grateful to Martial Hebert, who challenged me with numerous ideas and different viewpoints, and helped me understand many mathematical issues in 3D computer vision. Chuck Thorpe is always available and helpful whenever I have questions. Demetri Terzopoulos helped me whenever I visited Toronto and encouraged me to combine vision with graphics. I am also indebted to Rick Szeliski for teaching me motion estimation when I was a summer intern at DEC CRL in 1994.

I would like to acknowledge Sing Bing Kang and Jie Yang for providing numerous timely and critical comments on my research. A number of people in CMU have helped me in one way or the other. I am grateful to Jim Rehg, Conrad Peolman, Mark Wheeler, Fred Solomon, Mark Maimone, Henry Rowley, Yan-Bin Jia, Kaz Higuchi, David Chan, and Marie Elm.

Along my road to a Ph.D in robotics, there are many people who taught me much about robotics and deserve my special thanks: Prof. Wei-Yi Huang of Nanjing Institute of Technology, Prof. Guo-Tai Wang of Academia Sinica of China, Prof. Shiu-Kit Tso of the University of Hong Kong, Prof. George Lee of Purdue University, and Dr. Yangsheng Xu of CMU.

Last but not least, I thank my beloved wife, Ka-Yan, for her love, understanding and support. Although we were hundreds of miles apart in the past five years, I am very proud that we encouraged each other to finish our Ph.Ds. From now on we will be together forever.

Table of Contents

Chapter 1

Chapter 2

Chapter 3

List of Figures

Chapter 1

Introduction

1.1 Motivation

Computer vision enables us to understand and explore the world surrounding us through what we can observe using cameras. Traditionally, applications of computer vision are mostly found in robotics [54][74]. If we want a robot to know what is in its environment, it must first be able to build models of the surrounding objects. More specifically, a robot must possess some perceptual ability to determine what kinds of objects are in the environment, and where they are.

Recently many multimedia and internet applications require more and more 3D virtual environment models [107]. Therefore, one of the most important and obvious applications for computer vision is modeling from reality, i.e., building models from existing objects or scenes.

The conventional way of creating a virtual environment is to manually model it using CAD tools. The process is labor-intensive and the result is not very realistic. It is therefore desirable to develop a system that can automatically build models of real objects and scenes that the system observes. Unlike manual modeling, modeling from reality can be highly automated. This method is displayed in Figure 1. If one wants to incorporate one's favorite scene into existing VR games, one can simply go to the actual location, take as many images as necessary, analyze the images, and build a complete model of the scene. What is needed is a reliable technique which can generate accurate 3D object models from actual observations of real objects by integrating multiple views. The use of such a technique can greatly reduce the effort and cost of model construction. We term modeling from a sequence of images of the existing environment as modeling from reality.

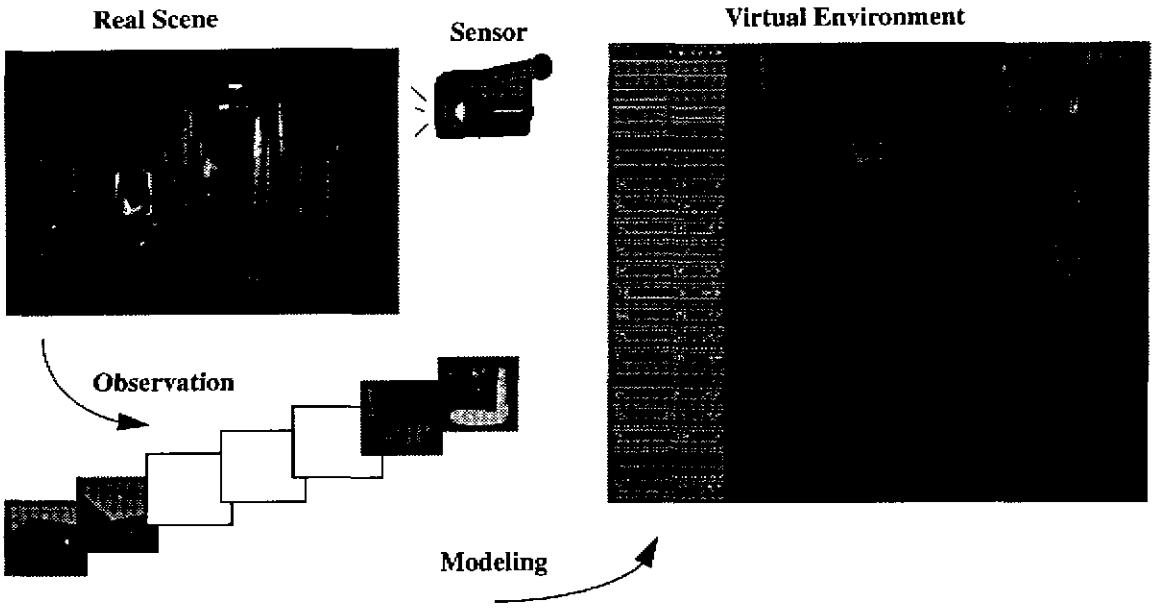


Figure 1 **Modeling from reality**

Another important application of modeling from reality is object recognition. To make a CAD-based recognition system, one needs to first build a library of object models [84][94]. Some object models may readily be available directly from existing CAD models, but many real world objects do not have pre-existing models. In the latter case, one may build individual models by taking range images of the objects of interest from multiple views. Once such models are built, they can be used to accomplish purposes such as reverse engineering. Other applications of modeling from reality include creating models for animation, reconstructing human body parts for surgical planning, and recovering machine parts for virtual factory simulation.

The objective of this thesis is not to attempt to solve all problems related to modeling from reality. Rather, it focuses on issues related to modeling from reality from a sequence of range images. Different solutions exist for various subproblems; each solution has its strengths and weaknesses. Part of the goal of this thesis is to explore these trade-offs, and to provide a framework by which one can better understand why those choices are necessary. In exploring this topic, a number of questions are considered. These issues include:

- How can integration of multiple views be statistically optimal?
- Which representations are to be used? Why?
- How can we make two views correspond?
- Why do we need to resample the raw range data?

Because modeling from reality is such an immense problem, it would not be possible to adequately cover all kinds of objects in this thesis. Therefore only two categories of objects, namely polyhedral objects and free-form objects, will be discussed.

1.2 Past Work

A great deal of work has been done on modeling existing environments using computer vision techniques. These efforts can best be summarized in three categories, namely, acquiring 3D information, modeling with range images, and modeling without range images.

1.2.1 Acquiring 3D Information

There are many ways to obtain 3D information. Some of the most popular ones currently used by computer vision researchers are the light-stripe range finder, the laser range finder, and stereo. Most of these sensors are surface-based; i.e., only points on object surfaces are measured. Other sensors, such as MRI and ultrasonic devices used in medical imaging, are volume-based. In addition, new ways of recovering 3D information continue to be developed. For example, the depth-from-defocus range finder has shown great promise even in real-time range sensing [88]. Other means include, for instance, depth-from-scattering.

1. Light stripe range finder

The light-stripe range finder is built on the principle of triangulation. The system we use in this thesis [103] consists of a projector and a CCD camera. The projector emits different light patterns, and the camera captures the intensity images under different lighting. Our light-stripe range finder has an accuracy of 0.5mm. The patterns are projected in either vertical or horizontal binary-coded stripes so that correspondence among different images is easily solved. To use the range finder, a calibration process is needed to determine the transformation between the camera coordinate and the world coordinate. Because structured patterns can not be projected over a long distance, the dynamic range of the light-stripe range sensor is limited to a few meters.

2. Laser range finder

Two different methods are used for finding range information using a laser, namely the time-of-flight method and the phase-based method [99]. Laser range finders are usually very accurate. For example, Higuchi et al. [48] reported an accuracy of 0.1mm with the Toyota ranger sensor. However, since most laser range finders are either point-based or line-based [25], they do not cover a large area [79].

3. Stereo

Stereo has been employed for many years to reconstruct 3D depth from a pair of intensity images. Like the light-stripe range finder, it uses the principle of triangulation. Unlike light-stripe range finders, however, stereo is usually passive, and no light patterns are projected. Area-based or line-based methods are commonly used for matching stereo. Stereo matching becomes difficult when little texture appears in the stereo pair. Recently there has been a trend toward using multiple images (more than 2) to eliminate some of the difficulties encountered in stereo matching [89]. Another improvement is projecting active light patterns [65]. Stereo machines with active patterns produce a degree of accuracy to that of light-stripe range finders.

4. Structure from motion

Structure from motion [36] is another way to obtain range information from a sequence of intensity images with calibrated or uncalibrated cameras. Similar to stereo, structure from motion does not work well when images contain large untextured regions.

1.2.2 Modeling from Range Images

A significant amount of work has been done on object modeling from a sequence of range images. Most work falling into this category assumes that the transformation between successive views is either known or can be recovered, so that all data can be transformed with respect to a fixed coordinate system.

- Known transformation

Bhanu [11] modeled objects by rotating them through known angles. Ahuja and Veenstra [1] constructed an octree object model from orthogonal views. By finding the correspondences from intensity patterns in all eight views, Vemuri and Aggarwal [128] derived the motion and transformed all eight range images with respect to the first frame.

- Feature-based range image registration

  To accurately recover the transformation between two views, different range image registration techniques using various features have been developed. Besl and Jain [9] proposed to segment range images using variable order polynomials. Ferrie and Levine [39] used correspondence points identified by correlation over the differential properties of the surface. Wada et al. [129] employed facets of an approximated convex hull of range images. Parvin and Medioni [91] proposed the construction of boundary representation (B-rep) models using segmented surface patches. The difficulty of feature-based registration is in realizing robustness, especially in the case of free-form objects.

- Featureless range image registration

  Many algorithms have been derived for featureless range data point matching. Besl and Kay [10] used the iterative point matching (ICP) method to project points from one surface to another during matching. A similar approach was proposed by Chen and Medioni [18]. Zhang [133] improved Besl and Kay's ICP algorithm by using robust statistics. Champleboux et al. [15] used the Levenberg-Marquart nonlinear minimization algorithm to minimize the sampled distance to surface using octree-splines. Voting based methods can also been used [22][23][61]. Several modified ICP algorithms have also been proposed lately [8][40][102]. Pennec and Thirion [92] presented an excellent discussion on validation of 3D point registration.

  These featureless registration algorithms are locally optimal: they work well for free-form objects only if a good transformation is initially given. Higuchi, Hebert and Ikeuchi [49] proposed a registration method which eliminates the problems associated with both feature-based and featureless methods. Their method builds discrete spherical meshes to represent the surfaces observed in each range image, and computes local curvature at each mesh node. Registration of different images is then achieved by comparing local curvature distribution of spherical meshes.

- Connectivity and merging

After transforming all range images to a world coordinate system using the registration result, an object model is usually obtained by running a connectivity algorithm (such as the Delaunay triangulation [96]) at the last step. Hoppe et al. [52] used graph traversal methods. Connectivity can also be modified and determined as more views are incorporated. Parvin and Medioni [91] used an adjacency graph to represent the connectivity of each segmented view. In their adjacency graph, nodes represent surface patches with attributes, and arcs represent adjacency between surfaces. Soucy and Laurendeau [112] made use of the structured information about where the images are taken; they proposed to triangulate each view and merge multiple views via a Venn diagram when the transformation is known. The parts in common in different views are then re-sampled. However, constructing such a Venn diagram is combinatorial in nature (only four-view merging is presented in their work). Turk and Levoy [124] proposed a similar approach but avoided the problem of Venn diagram construction by merging only two adjacent views at each step.

1.2.3 Modeling without Range Images

Inferring scene geometry and camera motion from a sequence of *intensity* images is also possible in principle. Historically efforts on structure from motion (SFM) can be characterized by the camera projection models with calibrated cameras. Recently research interest has been shifted to studies of SFM from weakly-calibrated or uncalibrated cameras.

- Structure from motion

  With a calibrated camera and under an orthographic projection model, Tomasi and Kanade [123] proposed a factorization method to simultaneously solve shape and motion. Poelman and Kanade [93] extended the factorization method to the case of paraperspective projection. Szeliski and Kang [117] proposed a nonlinear optimization method to solve shape and motion under perspective. Azarbayejani et al. [5] also used a full perspective model but proposed a Kalman-filter approach. However, in [93][123], the task is formulated as a least squares problem where missing data due to occlusion and mismatching is extrapolated from measured data and estimated motion. Although theoretically three views of four points are sufficient in determining structure and motion [126], it is difficult in practice to find a good sub-matrix to do "row-wise" and "column-wise" extrapolation. Szeliski and Kang [117] proposed to assign a weight to each measurement and incorporated an object-oriented perspective projection in a nonlinear least squares problem. The very nature of the nonlinear least squares formulation requires standard techniques in nonlinear optimization, e.g., Levenberg-Marquardt, in which convergence

to a local minimum may be a problem. In addition, most existing algorithms seem to be more useful for determining camera motion than for building 3D object models. This is because the recovered object shape is defined by a collection of 3D points whose connectivity is not explicitly known.

The factorization method [93][123][27], in essence, is the principal component analysis of a measurement matrix. The principal component analysis expresses the variance of the measurement matrix in a compact and robust way and has been extensively studied in computational statistics [34]. The singular value decomposition (SVD) method [42] is a straightforward solution when the measurement matrix is complete. When data is incomplete or missing, as often is the case in practice, obtaining principal components of a measurement matrix becomes much more complicated.

- Image-based rendering

  An alternative to structure from motion is image-based rendering, which bypasses 3D reconstruction. Chen and Williams [17] proposed view interpolation of two or three images using known correspondence. Szeliski [115] showed how multiple images are merged to form a big mosaiced image. Laveau and Faugeras [70] used the fundamental matrix to interpolate images. Chen [16] created the QTVR system which merges multiple images taken at the same camera nodal point to a cylindrical panorama. McMillan and Bishop [80] discussed how to interpolate between two panoramas. Debevic et al. [26] used a hybrid approach which combines geometry-based and image-based approaches.

  The advantage of image-based rendering is that it does not need to reconstruct a complete 3D model of the scene. For example, the model used can be a generic cylindrical or spherical image. However, the viewing points are severely restricted and so is the 3D sensation.

1.3 Our Approach to Modeling from Reality

Our approach to modeling from reality is based on several key observations: the widespread use of range images, data redundancy from multiple views, and the necessity of object-centered representation for integration. Our approach will be outlined after a brief comparison of different aspects of modeling is presented.

1.3.1 Range Images vs. Intensity Images

It is our belief that 3D information will be available and widely used in the very near future. As surveyed in the previous section, many ways of obtaining 3D information are already available. We should make use of range data for the task of modeling from reality.

Although this study is focused on modeling from a sequence of range images, this does not imply that the methods developed here are so restricted as to be useless for modeling from a sequence of intensity images. Indeed, the algorithms described later are of fundamental importance to structure from motion as well. Thus, while many questions remain to be answered, by exploring the redundancy from multiple observations, this thesis provides a framework for understanding problems associated with modeling from reality, using range images and/or intensity images.

1.3.2 Sequential vs. Integral

Modeling-from-reality systems usually work with a sequence of images of the object(s), where the sequence spans a smooth change in the positions of the sensor and/or object(s). Most previous systems have attempted to apply inter-frame motion estimates to successive pairs of views in a sequential manner [123]. Whenever a new view is introduced, it is matched with the previous view; the transformation between these two successive views has to be recovered before the object model is updated. This sequential method does not work well in practice because local motion estimates are subject to noise and missing data. Local mismatching errors accumulate and propagate along the sequence, yielding erroneous object models.

Rather than sequentially integrating successive pairs of views, we should instead search for a statistically optimal object model that is most consistent with all the views. Although each single view provides only partial information regarding the object, it is likely that any part of the object will be observed a number of times along the sequence. Object modeling from this sequence of views can be formulated as an over-determined minimization problem because significant redundancy exists among all the views. In this thesis, we will present an integral approach to modeling from a sequence of images. Bove [13] also proposed to integrate multiple views in a probabilistic sense, but failed to show how to take advantage of the redundancy of multiple views.

1.3.3 Polyhedral vs. Free-form

The real world around us is very complicated. It is unlikely that a single representation can be applicable for modeling everything in the existing 3D world. We should decide, depending on the task, which representation should be used. This observation is in accordance with the spirit of task-oriented vision proposed by Ikeuchi and Hebert [59]. Good representation makes it possible to combine multiple views in a statistically optimal manner. In this thesis, we will model a polyhedral object with planar boundary surface representation, and model a free-form object using a special spherical mesh representation.

In dealing with free-form objects, only objects with spherical topology (i.e., of genus zero) are considered in this thesis. A modeling system which deals with only spherical topology objects can be of considerable use even though it solves only a subset of the general modeling problem. Modeling free-form objects with arbitrary topology will be discussed in Chapter 7 under future work.

1.3.4 Local Resampling vs. Global Resampling

Because of discrete sampling of sensor measurements, we have to resample multiple views so that we may be able to establish the correspondence among them. Free-form objects, in particular, need special attention. To resample a free-form object, a special spherical mesh representation is used. For polyhedral objects, it is more appropriate to resample the range images using planar patches because they can be reliably segmented and they correspond well in different views.

Resampling is a mathematical transformation which maps one set of points to another. Surface resampling appears to be redundant. According to Dodgson [35], image resampling is used other than re-capturing or re-rendering when it is either the only available option or the preferred option. This statement is also true for surface resampling.

The resampling process does not come without cost. A practical problem with resampling is that the resampling process inevitably introduces errors that may affect the second step of merging. Initially, the resampling process can be viewed as a way of reconstructing a continuous surface from discontinuous measurements [121][12]. The range images we deal with are discrete. In fact, they are nicely laid out in rectangular grids in our light-stripe range finder. Another problem is that multi-pass resampling in free-form modeling may be required to make one-to-one correspondence; this requirement causes additional error.

1.3.5 Object-centered vs. Image-centered

The task of modeling from reality can be classified into two categories: scene modeling and focused object modeling. For example, to model an indoor scene, we can move the camera to various locations and take a sequence of images. This is called scene modeling. On the other hand, if we want to model a toy dog, we may put the object in front of a camera, and take a sequence of images by either rotating the object or rotating the camera around the object. We call this focused object modeling.

A similar decomposition of the task has also been used in the QTVR image-based rendering system [16] where both "inside-looking-out" (panorama image) and "outside-looking-in" (object movie) representations are used. In fact, the inside-looking-out approach can make use of viewer-centered or sensor-centered representations while the outside-looking-in approach has to use object-centered representation. This distinction is not as sharp in our case.

1.3.6 An Integral Approach

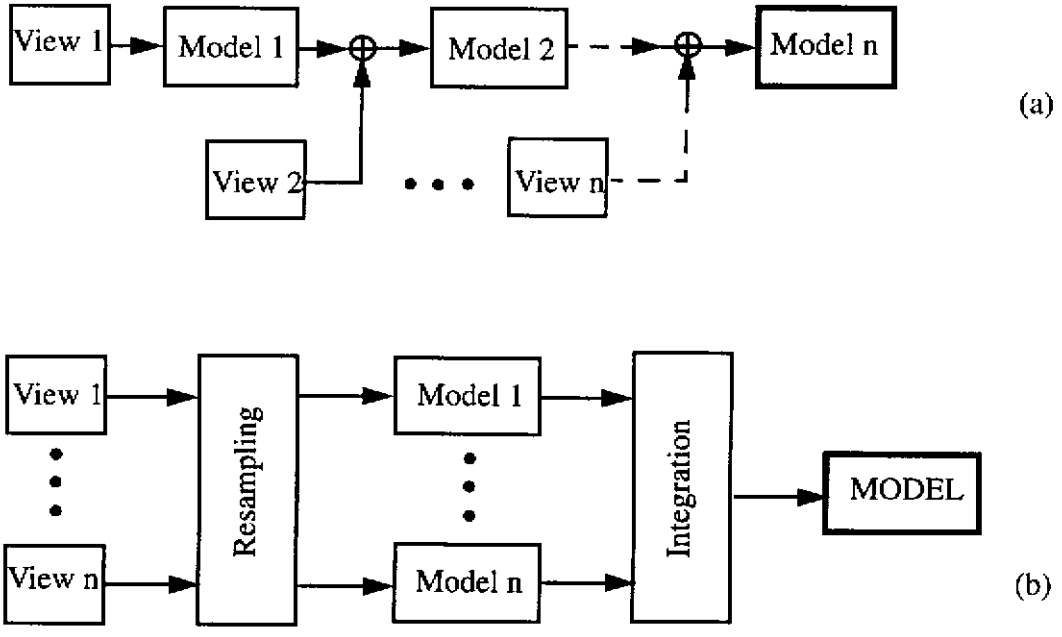


Figure 2 **Modeling-from-reality systems: (a) sequential modeling; (b) integral modeling**

The task of modeling from reality is essentially a problem of merging multiple views using an appropriate object-centered representation. Many previous modeling from reality techniques involve motion estimation between successive pairs of views in a sequential manner [49][91][93][67] as shown in Figure 2(a). Whenever a new view is introduced, it is matched with the previous view. The transformation between these two successive views is estimated before the object model is updated. This sequential method usually does not work well in practice because of errors in local motion estimation due to noise and missing data. These errors accumulate and are propagated along the sequence, yielding erroneous object models.

In this thesis, we present a new technique for modeling from reality. This technique, called the *integral approach,* is illustrated in Figure 2(b). Rather than sequentially integrating successive pairs of views, we propose to reconstruct a statistically optimal object model that is simultaneously most consistent with all views. Our method makes use of the significant redundancy existing among all the views, because it is likely that any part of the object will be observed a number of times along the sequence of images, although each single view provides only partial information. The key idea of integral object modeling is to enable a system to integrate a complete object model in terms of observable features. Because of the redundancy existing in the sequence of images, we can obtain a reliable solution from an overconstrained minimization problem even when data is missing. This thesis explains how this problem is formulated and presents an efficient solution to solve it.

1.4 Thesis Overview

We begin our study of modeling from reality in Chapter 2, where an integral approach is proposed for modeling from multiple images. The integral modeling approach consists of two parts: what to integrate and how to integrate. We explain how to integrate using a motivational example of modeling a 12-faced polyhedron from a sequence of views. We show that multiple view integration is formulated as a problem of principal component analysis with missing data (PCAMD). Then we outline Wiberg's formulation of PCAMD, and modify the formulation by proper indexing of the objective function. The modified formulation is then generalized as a weighted least squares (WLS) minimization problem. An efficient PCAMD algorithm is presented to solve this WLS problem.

In Chapter 3 we formulate the problem of modeling a polyhedral object and recovering transformations as a combination of two WLS problems. We compute the surface descrip-

tion and transformation by extracting the principal components of two highly rank-deficient measurement matrices with many missing elements; each matrix forms a WLS problem. The first WLS problem of recovering rotation matrices and surface normals is further simplified by using the quaternion representation of rotation. A two-step algorithm is presented to model the object from a sequence of segmented range images. A planar surface patch tracking system is also described. Different modules in the tracking system, such as range image segmentation, adjacency graph building, and two-view merging are presented. We also show that the problem of surface connectivity can be reduced to one of connectivity of supporting lines of a simple polygon. Since the problem of establishing connectivity of supporting lines can be regarded as both a modified convex hull-like problem and a cell decomposition problem, we propose a modified Jarvis' march algorithm which successfully reconstructs the simple polygon. In Chapter 4 we show how we apply our integral approach to a complicated indoor scene which is approximated by planar patches.

In Chapter 5 we extend our integral approach to modeling of free-form objects from multiple range images. We address the problem of "what to integrate" by presenting a novel global resampling scheme which can be used to determine correspondence among different views. In particular, we use a nearly uniform spherical mesh with fixed connectivity to represent free-form objects. Each range image is resampled using this global spherical representation. Some applications of our free-form object modeling are presented in Chapter 6. Specifically, we use the spherical mesh representation and the shape metric to compare and synthesize shapes.

Chapter 7 summarizes the word described in this thesis, and concludes with our major contributions and possible future work.

Chapter 2

An Integral Approach to Object Modeling

Several problems exist with modeling from reality. First, data acquisition [21] and image registration introduce significant errors [3]. Second, because of occlusion and self-occlusion [20], range images often have missing data points. More importantly, several important issues related to the realization of a practical modeling-from-reality system have not been resolved satisfactorily. These issues include:

- What kind of representation should be used to model the object?
- How can a sequence of images be integrated?
- Can model reconstruction be made statistically optimal?
- How can objects be sampled sufficiently and unambiguously?

These issues can be summarized as two essential problems: *what to integrate* and *how to integrate*. In order to solve these two problems, we propose a new approach called *integral object modeling*. The key idea of integral object modeling is to enable a system to resample a sequence of images of an object in terms of its observable features, and then integrate the images using principal component analysis with missing data (PCAMD). Integral object modeling works by integrating partial observations provided by different views such that a complete object model is created. Although object modeling from a sequence of images is nonlinear and possibly ill-conditioned, an integral approach makes use of the redundant data so that a statistically optimal reconstruction is feasible. The objective of this chapter is to illustrate the integral approach using the modeling of a polyhedral object as an example, and to explain how multiple views can be integrated in a statistically optimal fashion.

2.1 Integration of Multiple Views

Suppose that our task is to make a model for a dodecahedron from a sequence of segmented range images. A dodecahedron is a polyhedron with 12 faces. It is a simple Platonic solid. Assume that we have tracked *12* faces over *4* nonsingular views as shown in Figure 3. The segmented range images provide trajectories of plane coordinates $\{\mathbf{p}_p^{(f)} \mid f=1,...,4,\ p=1,...,12\}$, where $\mathbf{p} = (\mathbf{v}^T, d)^T$ represents a planar equation with surface normal $\mathbf{v}$ and normal distance to the origin d. With these observations, we may construct a 16×12 measurement matrix as follows:

$$W = \begin{bmatrix} \mathbf{p}_1^{(1)} & \mathbf{p}_2^{(1)} & \mathbf{p}_3^{(1)} & \mathbf{p}_4^{(1)} & \mathbf{p}_5^{(1)} & \mathbf{p}_6^{(1)} & * & * & * & * & * & * \\ \mathbf{p}_1^{(2)} & \mathbf{p}_2^{(2)} & \mathbf{p}_3^{(2)} & \mathbf{p}_4^{(2)} & * & * & \mathbf{p}_7^{(2)} & \mathbf{p}_8^{(2)} & * & * & * & * \\ \mathbf{p}_1^{(3)} & \mathbf{p}_2^{(3)} & * & * & * & \mathbf{p}_6^{(3)} & * & \mathbf{p}_8^{(3)} & \mathbf{p}_9^{(3)} & \mathbf{p}_{10}^{(3)} & * & * \\ * & * & * & * & * & * & \mathbf{p}_7^{(4)} & \mathbf{p}_8^{(4)} & \mathbf{p}_9^{(4)} & \mathbf{p}_{10}^{(4)} & \mathbf{p}_{11}^{(4)} & \mathbf{p}_{12}^{(4)} \end{bmatrix}.$$

Since there are only 6 visible faces from each nonsingular view, there are 6 unobservable faces from each view. Each of these unobservables is denoted by an *. Our modeling task is then to recover the poses of all the 12 faces in a fixed coordinate system.

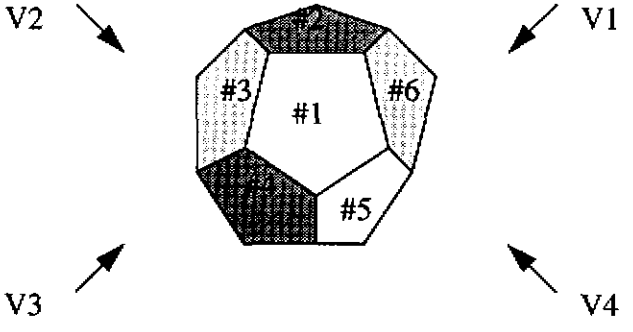


Figure 3 **Distinct views of a dodecahedron**

If the measurement matrix were complete, and assuming that data are noisy, our task would be to average all those 12 faces over 4 views. In the absence of noise, any set of 12 faces from one of the 4 views will do. The standard way to average is to apply singular value

decomposition (SVD) to this measurement matrix, the rank of which is at most 4 (see Section 3.1 for the argument). Such a measurement matrix can subsequently be factorized, with proper normalization, into a left matrix Q of transformation parameters and a right matrix P of plane coordinates,

$$W = Q\,P$$

where

$$Q = \begin{bmatrix} Q^{(1)} \\ Q^{(2)} \\ Q^{(3)} \\ Q^{(4)} \end{bmatrix}, P = \begin{bmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \cdots & \mathbf{p}_{12} \end{bmatrix}.$$

$Q^{(f)}$ is the transformation of fth view with respect to the fixed world coordinate system, and $\mathbf{p}_p$ is the pth plane equation in the same world coordinate system. SVD has also been successfully applied to shape and motion recovery from a sequence of intensity images [123].

Unfortunately, the measurement matrix is often incomplete in practice: it is not unusual for a large portion of the matrix to be unobservable. As we have seen in the above example, half of the measurement matrix is unknown. When the percentage of missing data is very small, it is possible to substitute the missing elements with the mean or an extreme value. This is a common strategy in multivariate statistics [63][100]. Such an approach is, however, no longer valid when a significant portion of the measurement matrix is unknown.

A common practice in modeling from a sequence of images with unobservables is to use extrapolation. For example, if there are at least three matched planar surfaces which are non-parallel [37], we can recover the transformation between view *1* and view *2*. We first extrapolate the invisible planar surfaces in view *1* from their corresponding but visible surfaces in view *2* using the transformation recovered. Then we apply the same extrapolation to the invisible surfaces in view *1*. By repeating this process, we can, in principle, extrapolate the locations of all invisible surfaces from those which are visible [91]. A final step can be added to fine-tune the result by factorizing the extrapolated measurement matrix using SVD. A similar extrapolation approach called the "propagation method" [123] is also used in motion and shape recovery from a sequence of intensity images.

One major problem with the extrapolation method, however, is that once the estimated transformation is inaccurate at any step, the extrapolated results will be erroneous. In sequential modeling, errors accumulate and propagate through the steps. The final fine-tuning process can not improve the result dramatically since the extrapolated measurement

matrix is inaccurate. To obviate this problem, we do not resort to error sensitive extrapolation. Rather, we make use of more rigorous mathematical tools developed in computational statistics that cater to missing data. We will demonstrate the formulation in Section 2.2. Applications of this formulation to multiple view merging of polyhedral object and free-form objects can be found in Chapters 3 and 5, respectively.

2.2 Principal Component Analysis with Missing Data

The problem of object modeling from a sequence of views, as shown in the previous section, can be formulated as a problem of principal component analysis with missing data (PCAMD). PCAMD has been extensively studied in computational statistics. Ruhe [101] proposed a minimization method to analyze one component model when observations are missing. One component model decomposes an $F \times P$ measurement matrix into an $F \times 1$ left matrix and a $1 \times P$ right matrix. Wiberg [131] extended Ruhe's method to the more general case of the arbitrary component model. In this section, we outline Wiberg's formulation of principal component analysis with missing data, before proposing a modified formulation by appropriate indexing, and generalizing the problem as a weighted least squares (WLS) problem.

2.2.1 Wiberg's Formulation

Suppose we have an $F \times P$ measurement matrix W, which consists of P individuals from an F-variate normal distribution with mean μ and covariance Σ. Let the rank of W be r. If the data is complete and the measurement matrix filled, the problem of principal component analysis is to determine $\tilde{U}$, $\tilde{S}$, and $\tilde{V}$, such that

$$\left\| W - \mathbf{e}\mu^T - \tilde{U}\tilde{S}\tilde{V}^T \right\|$$

is minimized. $\tilde{U}$ and $\tilde{V}$ are $F \times r$ and $P \times r$ matrices with orthogonal columns, $\tilde{S} = diag(\sigma_i)$ is an $r \times r$ diagonal matrix, μ is the maximum likelihood approximation of the mean vector, and $\mathbf{e}^T = (1, ..., 1)$ is an F-tuple vector with all ones. The solution to this problem is essentially the SVD of the centered (or registered) data matrix $W - \mathbf{e}\mu^T$.

If the data are incomplete, we have the following minimization problem:

$$min \quad \phi = \frac{1}{2}\sum_{I} (W_{f,p} - \mu_p - \mathbf{u}_{f.}^T \mathbf{v}_{p.})^2 \qquad \text{(EQ 2.1)}$$

$$I = \{ (f,p) : W_{f,p} \text{ is observed} \}$$

where $\mathbf{u}_{f.}$ and $\mathbf{v}_{p.}$ are column vector notations defined by

$$\begin{bmatrix} \mathbf{u}_{1.}^T \\ \dots \\ \mathbf{u}_{F.}^T \end{bmatrix} = \tilde{U}\tilde{S}^{\frac{1}{2}} \qquad \text{(EQ 2.2)}$$

and

$$\begin{bmatrix} \mathbf{v}_{1.}^T \\ \dots \\ \mathbf{v}_{P.}^T \end{bmatrix} = \tilde{V}\tilde{S}^{\frac{1}{2}}. \qquad \text{(EQ 2.3)}$$

Lemma 1

A necessary condition to uniquely solve this problem (EQ 2.1) is $m \geq r(F+P-r) + P$ where m is the number of observable elements in W.

Proof:

It is trivially true that there are at most $r(F+P-r)$ independent elements from LU decomposition of an $F \times P$ matrix of rank r. Hence, to uniquely solve (EQ 2.1), the number of equations (m) has to be no fewer than the number of unknowns ($r(F + P - r) + P$).

To sufficiently determine the problem (EQ 2.1), more constraints are needed to normalize either the left matrix $\tilde{U}$ or the right matrix $\tilde{V}$.

If we write the measurement matrix W as an m-dimensional vector $\mathbf{w}$, the minimization problem can be written as

$$min \quad \phi = \frac{1}{2}\mathbf{f}^T\mathbf{f} \qquad \text{(EQ 2.4)}$$

where

$$\mathbf{f} = \mathbf{w} - \hat{\mu} - F\mathbf{u} = \mathbf{w} - G\tilde{\mathbf{v}} \qquad \text{(EQ 2.5)}$$

and

$$\mathbf{u} = \begin{bmatrix} \mathbf{u}_1 \\ \dots \\ \mathbf{u}_F \end{bmatrix}, \tilde{\mathbf{v}} = \begin{bmatrix} \tilde{\mathbf{v}}_1 \\ \dots \\ \tilde{\mathbf{v}}_P \end{bmatrix}, \tilde{\mathbf{v}}_p = \begin{bmatrix} \mathbf{v}_p & \mu_p \end{bmatrix}, \hat{\mu}_i = \mu_{p(i)}. \qquad \text{(EQ 2.6)}$$

F and G are of dimensions $m \times rF$ and $m \times (r+1)P$, respectively, and are computed by expanding every element f_i of $\mathbf{f}$

$$f_i = w_i - \mu_{p(i)} - u_{f(i)}^T v_{p(i)}. \qquad \text{(EQ 2.7)}$$

where the ith component of $\mathbf{w}$ is indexed to the *(f(i), p(i))*-th component of W, i.e., $w_i = W_{f(i),p(i)}$, and $W_{f,p} = w_{i(f,p)}$.

To solve the minimization problem stated in(EQ 2.4), the derivative of the objective function (with respect to $\mathbf{u}$ and $\tilde{\mathbf{v}}$) should be zero, i.e.,

$$\dot{\phi} = \begin{bmatrix} F^T F\mathbf{u} - F^T(\mathbf{w} - \hat{\mu}) \\ G^T G\tilde{\mathbf{v}} - G^T\mathbf{w} \end{bmatrix} = \mathbf{0}. \qquad \text{(EQ 2.8)}$$

Obviously (EQ 2.8) is nonlinear because F is a function of $\mathbf{v}$, and G is a function of $\mathbf{u}$. In theory, this equation can be solved by any appropriate nonlinear optimization method. In practice, however, the dimensionality is so high that we have to adapt the algorithm to make use of the special structure of the problem. It can be observed that:

(1) For fixed $\mathbf{u}$, we have a linear least-squares problem of $\mathbf{v}$; and for fixed $\mathbf{v}$, we have a linear least-squares problem of $\mathbf{u}$;

(2) Since (EQ 2.8) is also a bilinear problem of $\mathbf{u}$ and $\mathbf{v}$, we can successively improve their estimates by using the updating technique in the NIPALS algorithm [101], i.e., for a given $\mathbf{v}$, $\mathbf{u}$ is updated $\mathbf{u} = F^+(\mathbf{w} - \hat{\mu})$; and for a given $\mathbf{u}$, $\mathbf{v}$ is updated $\tilde{\mathbf{v}} = G^+\mathbf{w}$. F^+ and G^+ are the pseudo-inverses of F and G respectively.

2.2.2 An Example[1] of PCAMD

We present a simple example in this section to explain how principal component analysis with missing data is applied. Assume that we want to decompose the following 3×4 measurement matrix M with missing data into a 3×1 left matrix A, and a 1×4 right matrix B,

$$M = \begin{bmatrix} * & 7 & 5 & * \\ 0 & 2 & 1 & * \\ 3 & 6 & 2 & 5 \end{bmatrix} = A \times B,$$

where

$$A = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}, B = \begin{bmatrix} b_1 & b_2 & b_3 & b_4 \end{bmatrix}.$$

This decomposition is equivalent to analyzing the first principal component of the measurement matrix. Following the equations (EQ 2.4)-(EQ 2.8), assuming that $\hat{\mu} = 0$, we have

$$F^T = \begin{bmatrix} 0 & 0 & b_2 & 0 & 0 & b_3 & 0 & 0 & 0 \\ b_1 & 0 & 0 & b_2 & 0 & 0 & b_3 & 0 & 0 \\ 0 & b_1 & 0 & 0 & b_2 & 0 & 0 & b_3 & b_4 \end{bmatrix}, G^T = \begin{bmatrix} 0 & 0 & a_2 & 0 & 0 & a_3 & 0 & 0 & 0 \\ a_1 & 0 & 0 & a_2 & 0 & 0 & a_3 & 0 & 0 \\ 0 & a_1 & 0 & 0 & a_2 & 0 & 0 & a_3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & a_3 \end{bmatrix}$$

and

$$F^T F = \begin{bmatrix} b_2^2 + b_3^2 & 0 & 0 \\ 0 & b_1^2 + b_2^2 + b_3^2 & 0 \\ 0 & 0 & b_1^2 + b_2^2 + b_3^2 + b_4^2 \end{bmatrix},$$

---

1. Bot Holt at Bell Labs suggested that PCAMD should be compared with the EM algorithm, as is in the example illustrated on p.211 of Johnson and Witchern's book [62]. The same example is used here. The EM algorithm is a two-step iterative maximum likelihood estimation algorithm that predicts the contribution of missing observations and then computes a revised estimate of the parameters. The PCAMD method performs slightly worse than maximum likelihood estimation for simulated multivariate normal data [131]. However, the PCAMD algorithm works whether or not the measurement matrix is formed from a normal distribution [63].

$$G^T G = \begin{bmatrix} a_2^2 + a_3^2 & 0 & 0 & 0 \\ 0 & a_1^2 + a_2^2 + a_3^2 & 0 & 0 \\ 0 & 0 & a_1^2 + a_2^2 + a_3^2 & 0 \\ 0 & 0 & 0 & a_1^2 + a_2^2 + a_3^2 \end{bmatrix}.$$

It is straightforward that we can iteratively update our estimations by

$$\hat{a}_1 = \frac{7b_2 + 5b_3}{b_2^2 + b_3^2}, \hat{a}_2 = \frac{2b_2 + b_3}{b_1^2 + b_2^2 + b_3^2}, \hat{a}_3 = \frac{3b_1 + 6b_2 + 2b_3 + 5b_4}{b_1^2 + b_2^2 + b_3^2 + b_4^2},$$

and

$$\hat{b}_1 = \frac{3a_3}{a_2^2 + a_3^2}, \hat{b}_2 = \frac{7a_1 + 2a_2 + 6a_3}{a_1^2 + a_2^2 + a_3^2},$$

$$\hat{b}_3 = \frac{5a_1 + a_2 + 2a_3}{a_1^2 + a_2^2 + a_3^2}, \hat{b}_4 = \frac{5}{a_3}.$$

If we normalize the estimated parameter so that a_3 is the same as the initial sample mean $\mu = \begin{bmatrix} 6 & 1 & 4 \end{bmatrix}^T$, i.e., $a_3 = 4$, the PCAMD algorithm converges in about 10 steps from an arbitrary initial estimate to

$$A = \begin{bmatrix} 5.418 \\ 1.188 \\ 4 \end{bmatrix}, B = \begin{bmatrix} 0.689 & 1.375 & 0.776 & 1.25 \end{bmatrix}.$$

Notice that A is significantly different from the initial sample mean.

2.3 New Formulations of PCAMD

2.3.1 Modified Wiberg's Formulation

In practice F and G are usually sparse matrices with many zeros. We can appropriately index $\mathbf{w}$ as $\mathbf{w}_1$, such that

$$\mathbf{f}_1 = \mathbf{w}_1 - \hat{\boldsymbol{\mu}} - H\mathbf{u} \qquad \text{(EQ 2.9)}$$

or

$$\mathbf{f}_1 = \begin{bmatrix} W_{1,1} \\ W_{1,2} \\ \cdots \\ W_{1,P} \\ \cdots \\ W_{F,1} \\ \cdots \\ W_{F,P} \end{bmatrix} - \begin{bmatrix} \mu_1 \\ \mu_2 \\ \cdots \\ \mu_P \\ \cdots \\ \mu_1 \\ \cdots \\ \mu_P \end{bmatrix} - \begin{bmatrix} V_{1,1} & \cdots & V_{r,1} & & & & & \\ V_{1,2} & & & & & & & \\ & \ddots & & & & \mathbf{0}_{P\times r} & & \\ V_{1,P} & & V_{r,P} & & & & & \\ & & & \ddots & & & & \\ & & & & V_{1,1} & & V_{r,1} & \\ & \mathbf{0}_{P\times r} & & & & \ddots & & \\ & & & & V_{1,P} & & V_{r,P} & \end{bmatrix} \begin{bmatrix} U_{1,1} \\ U_{1,2} \\ \cdots \\ U_{1,r} \\ \cdots \\ U_{F,1} \\ \cdots \\ U_{F,P} \end{bmatrix}. \qquad \text{(EQ 2.10)}$$

Similarly, we can index $\mathbf{w}$ as $\mathbf{w}_2$, such that

$$\mathbf{f}_2 = \mathbf{w}_2 - K\tilde{\mathbf{v}} \qquad \text{(EQ 2.11)}$$

or

$$\mathbf{f}_2 = \begin{bmatrix} W_{1,1} \\ W_{2,1} \\ \cdots \\ W_{F,1} \\ \cdots \\ W_{1,P} \\ \cdots \\ W_{F,P} \end{bmatrix} - \begin{bmatrix} U_{1,1} & \cdots & U_{1,r} & 1 & & & & & \\ U_{2,1} & & U_{2,r} & 1 & & \mathbf{0}_{F\times(r+1)} & & & \\ & \ddots & & \cdots & & & & & \\ U_{F,1} & & U_{F,r} & 1 & & & & & \\ & & & & \ddots & & & & \\ & & & & & U_{1,1} & & U_{1,r} & 1 \\ & \mathbf{0}_{F\times(r+1)} & & & & & \ddots & & \cdots \\ & & & & & U_{F,1} & & U_{F,r} & 1 \end{bmatrix} \begin{bmatrix} V_{1,1} \\ V_{2,1} \\ \cdots \\ V_{r,1} \\ \mu_1 \\ \cdots \\ V_{1,P} \\ \cdots \\ V_{r,P} \\ \mu_P \end{bmatrix}. \qquad \text{(EQ 2.12)}$$

Note that H and K are block diagonal matrices.

Because $\mathbf{f}_1$ and $\mathbf{f}_2$ contain the same observables as $\mathbf{f}$,

$$\phi = \frac{\mathbf{f}^T\mathbf{f}}{2} = \frac{\mathbf{f}_1^T\mathbf{f}_1}{2} = \frac{\mathbf{f}_2^T\mathbf{f}_2}{2} \qquad \text{(EQ 2.13)}$$

and

$$\dot{\phi} = \begin{bmatrix} H^T H\mathbf{u} - H^T(\mathbf{w}_1 - \hat{\mu}) \\ K^T K\tilde{\mathbf{v}} - K^T\mathbf{w}_2 \end{bmatrix} \qquad \text{(EQ 2.14)}$$

since

$$H = -\frac{\partial \mathbf{f}_1}{\partial \mathbf{u}}, K = -\frac{\partial \mathbf{f}_2}{\partial \tilde{\mathbf{v}}}. \qquad \text{(EQ 2.15)}$$

If the data are complete, K is a block diagonal matrix of dimension $FP \times (r+1)P$, whose block elements are U matrices of dimension $F \times (r+1)$, replicated along the diagonal, i.e.,

$$K = \begin{bmatrix} U & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & U \end{bmatrix}, \qquad \text{(EQ 2.16)}$$

$$U = \begin{bmatrix} U_{1,1} & & U_{1,r}\ 1 \\ & \ddots & \\ U_{F,1} & & U_{F,r}\ 1 \end{bmatrix}. \qquad \text{(EQ 2.17)}$$

When the data are incomplete, the elements associated with the missing data are taken out, resulting in a matrix of dimension $m \times (r+1)P$,

$$K = \begin{bmatrix} U_{m_1 \times (r+1)} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & U_{m_P \times (r+1)} \end{bmatrix} = \begin{bmatrix} U_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & U_P \end{bmatrix} \qquad \text{(EQ 2.18)}$$

where

$$\sum_{p=1}^{P} m_p = m, \text{ and } m_p = \sum_{f=1}^{F} \gamma_{f,p}.$$

$\gamma_{f,p} = 1$ when $W_{f,p}$ is observed, otherwise $\gamma_{f,p} = 0$.

Similarly, when the data are incomplete, we have the following matrix of dimension $m \times rF$,

$$H = \begin{bmatrix} V_{n_1 \times r} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & V_{n_F \times r} \end{bmatrix} = \begin{bmatrix} V_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & V_F \end{bmatrix} \quad \text{(EQ 2.19)}$$

where

$\sum_{f=1}^{F} n_f = m$, and $n_f = \sum_{p=1}^{P} \gamma_{f,p}$.

The pseudo-inverse matrices of H and K can be easily computed because of their block diagonal structure,

$$K^+ = \begin{bmatrix} U_1^+ & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & U_p^+ \end{bmatrix} \quad \text{(EQ 2.20)}$$

$$H^+ = \begin{bmatrix} V_1^+ & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & V_F^+ \end{bmatrix}. \quad \text{(EQ 2.21)}$$

2.3.2 A Weighted Least-squares Formulation

The minimization problem (EQ 2.1) can in fact be generalized as a WLS problem,

$$min \quad \phi = \frac{1}{2}\sum_{f,p} (\gamma_{f,p} (W_{f,p} - \mu_p - \mathbf{u}_f^T \mathbf{v}_p))^2 \quad \text{(EQ 2.22)}$$

where $\gamma_{f,p}$ is the weighting factor for each measurement $W_{f,p}$.

In the previous discussion, we have assumed that all weights are one when data is observable, or zero when data is unobservable. However, in many cases, we may prefer to assign

weights other than ones or zeros to individual measurements. For example, in recovering the pose of a 3D plane, we can assign a confidence measurement to each recovered surface normal by its incidence angle with the viewing direction. Different sensor models can also be applied to obtain a weighting matrix if necessary. In the following, we formulate principal component analysis with missing data as a WLS problem.

We introduce two $FP \times FP$ diagonal weight matrices,

$$\Gamma = diag\,(\Gamma_1, \Gamma_2, \ldots, \Gamma_P) \qquad \text{(EQ 2.23)}$$

and

$$\Phi = diag\,(\Phi_1, \Phi_2, \ldots, \Phi_F) \qquad \text{(EQ 2.24)}$$

where

$$\Gamma_p = diag\,(\gamma_{p,1}, \gamma_{p,2}, \ldots, \gamma_{p,F})\ ,\ p = 1, \ldots, P\,,$$

and

$$\Phi_f = diag\,(\gamma_{1,f}, \gamma_{2,f}, \ldots, \gamma_{P,f})\ ,\ f = 1, \ldots, F\,.$$

The minimization problem becomes

$$min \quad \phi = \frac{\mathbf{f}_{\gamma 1}^T \mathbf{f}_{\gamma 1}}{2} = \frac{\mathbf{f}_{\gamma 2}^T \mathbf{f}_{\gamma 2}}{2} \qquad \text{(EQ 2.25)}$$

where

$$\mathbf{f}_{\gamma 1} = \Gamma\, \mathbf{f}_1 \quad \text{and} \quad \mathbf{f}_{\gamma 2} = \Phi\, \mathbf{f}_2\,.$$

The solution to the above problem is when the first derivative of the objective function becomes zero. The derivative of the objective function is

$$\dot{\phi} = \begin{bmatrix} H_\gamma^T H_\gamma \mathbf{u} - H_\gamma^T (\mathbf{w}_1 - \hat{\mu}) \\ K_\gamma^T K_\gamma \tilde{\mathbf{v}} - K_\gamma^T \mathbf{w}_2 \end{bmatrix} \qquad \text{(EQ 2.26)}$$

where

$$H_\gamma = \Gamma H = \begin{bmatrix} \Gamma_1 V & & \\ & \ddots & \\ & & \Gamma_P V \end{bmatrix} \quad \text{(EQ 2.27)}$$

and

$$K_\gamma = \Phi K = \begin{bmatrix} \Phi_1 U & & \\ & \ddots & \\ & & \Phi_F U \end{bmatrix}. \quad \text{(EQ 2.28)}$$

Therefore, after computing the pseudo inverses of H_γ and K_γ,

$$H_\gamma^+ = \begin{bmatrix} H_{\gamma 1}^+ & & \\ & \cdots & \\ & & H_{\gamma P}^+ \end{bmatrix} = \begin{bmatrix} \left(V^T\Gamma_1^T\Gamma_1 V\right)^{-1}\left(V^T\Gamma_1^T\right) & & \\ & \ddots & \\ & & \left(V^T\Gamma_P^T\Gamma_P V\right)^{-1}\left(V^T\Gamma_P^T\right) \end{bmatrix} \quad \text{(EQ 2.29)}$$

$$K_\gamma^+ = \begin{bmatrix} K_{\gamma 1}^+ & & \\ & \cdots & \\ & & K_{\gamma F}^+ \end{bmatrix} = \begin{bmatrix} \left(U^T\Phi_1^T\Phi_1 U\right)^{-1}\left(U^T\Phi_1^T\right) & & \\ & \ddots & \\ & & \left(U^T\Phi_F^T\Phi_F U\right)^{-1}\left(U^T\Phi_F^T\right) \end{bmatrix} \quad \text{(EQ 2.30)}$$

we can use the PCAMD algorithm to solve the WLS problem. Our formulation is essentially a modified NIPALS Ruhe-Wiberg algorithm. The algorithm is as follows:

Algorithm PCAMD

(1) initialize $\tilde{\mathbf{v}}$

(2) update

$$\mathbf{u} = H_\gamma^+ (\mathbf{w_1} - \hat{\mu})$$

(3) update

$$\tilde{\mathbf{v}} = K_\Gamma^+ \mathbf{w_2}$$

(4) stop if the algorithm converges, if not, go back to (2).

Remarks:

(1) Ruhe [101] also suggested using Newton and Gauss methods to speed up the convergence of the NIPALS method. In practice, the NIPALS method converges within the desired tolerance in several iterations in most experiments.

(2) Ruhe [101] and Wiberg [131] also showed that the more data are missing, the worse the result will be. It is hardly surprising because PCAMD is basically an interpolation of all observable elements. Statistically this corresponds to decreasing robustness of the estimates for the principal components, given the observations. Fortunately, in object modeling from multiple views, we can always increase the number of views to form a well constrained problem for our modeling purpose. Determining a minimally acceptable number of views can be regarded as a sensor planning problem.

(3) The missing data can also be extrapolated as long as we find some sub-blocks in the measurement matrix which satisfy *Lemma 1*. The issue of obtaining those blocks is non-trivial. Once the missing data have been augmented, a linear or nonlinear optimization method can be applied to solve the original problem. The method should work well if the data are noise-free, i.e., if only the first r singular values of the reconstructed measurement matrix are non-zero. However, this method is of questionable value when any result from the sub-block computation is inaccurate.

(4) The algorithm PCAMD runs essentially in batch mode. When a new measurement is introduced, i.e., when a new row is added, we have to run PCAMD again. Nevertheless, it has been shown by Morita and Kanade [81] that it is possible to replace this kind of batch-mode algorithm with an iterative one.

2.4 An Integral Approach to Object Modeling

Principal component analysis with missing data has been formulated as a WLS minimization problem in the previous section and a PCAMD algorithm was proposed to solve it. As illustrated by the motivational example of a dodecahedron in Section 2.1, it is clear that polyhedral object modeling from a sequence of views should be formulated as a WLS problem. In this section, we present an integral approach to object modeling from a sequence of range images; the approach utilizes the PCAMD algorithm to obtain a statistically optimal solution.

2.4.1 How to Integrate

We use the example of polyhedral object modeling shown in Section 2.1. Once each range image is segmented and all planar patches are tracked, we can form a measurement matrix, which consists of many missing elements. We have seen how, using the PCAMD algorithm, the measurement matrix can be decomposed, with proper normalization, into a left matrix Q of transformation parameters and a right matrix P of plane coordinates.

In other words, the measurement matrix can be decomposed into a shape matrix P and a transformation matrix Q; these matrices are estimated iteratively in two weighted-least-squares steps. Figure 4 shows the integration process with PCAMD.

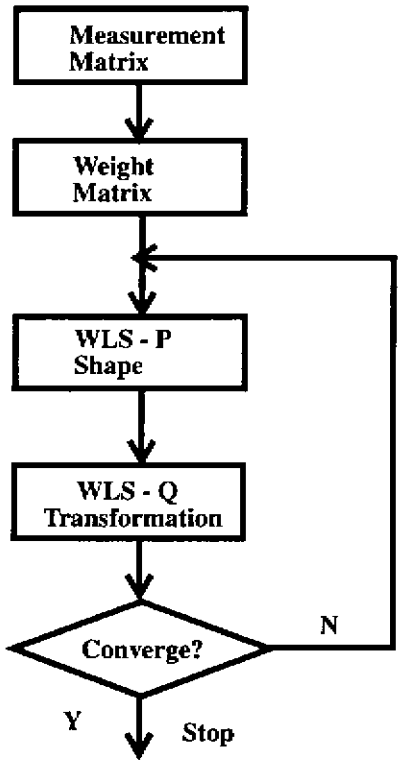


Figure 4 **Integration using PCAMD**

2.4.2 Make It Robust and Dynamic

The weight matrix in Figure 4 is pre-defined. However, the weights can be updated during the iterative process using some statistical methods, such as the metrically Winsorised residuals method [117]. This method is based on the assumption that each measurement is cor-

rupted by additive Gaussian noise. The metrically Winsorised residuals method adjusts the weight for each measurement depending on its residual error. We can downweight some elements which have suspiciously extreme values. This robust approach is illustrated in Figure 5.

In both Figure 4 and Figure 5 we have assumed that the measurement is given and can not be changed. However, the result from the PCAMD algorithm can be used to build a better correspondence. The update is tightly linked with the correspondence process. After we update the transformation matrix, we have to check whether the previous match is still valid. If we find a better correspondence, it can be used to form a new measurement matrix and a new weight matrix. In other words, we can even change the measurement matrix so that we have a more dynamic version of our integral approach.

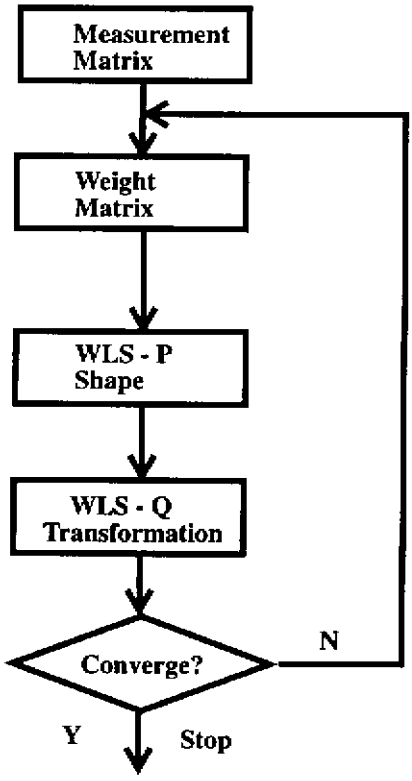


Figure 5 **Robust integration with updating weight matrix**

2.4.3 What to Integrate

As is evident by now, to integrate multiple views in a statistically optimal fashion, we first need to determine what is to be integrated from a sequence of range data. In polyhedral object modeling, we can successfully make use of the redundancy in multiple views because planar surface patches can be tracked over the sequence. This can be regarded as resampling polyhedral objects using an object-centered planar boundary surface representation. In Chapter 3, we show how this resampling is achieved, and how the integral approach is applied in polyhedral modeling. Due to the numerical instability of fitting high-order polynomial to noisy range data, we will not consider any higher-order polynomial segmentations.

Ideally, for free-form object modeling, we would prefer to register each data point from one view to another. However, the physical correspondences between data points in two different views are usually not known *a priori*. As a result, we have to search for some salient features which do have correspondence among different views. We show in Chapter 5 how free-form objects are resampled using a spherical mesh representation.

Chapter 3

What to Integrate: Polyhedral Objects

In this chapter, we show that multiple view merging of a polyhedral object can be formulated as a combination of two WLS problems. The first WLS problem involves the determination of rotation matrices and surface normals; the determination is independent of translation. It can be further simplified by representing the rotation matrix using the quaternion. Once the first problem is solved, the second WLS problem yields translation vectors and normal distances to the origin of the coordinate system. A straightforward two-step iterative algorithm can be devised to solve these two problems using the PCAMD algorithm explained in Chapter 2.

3.1 Two WLS Problems

Suppose that we have tracked P planar regions over F frames. We then have trajectories of plane coordinates $\{ (\mathbf{v}_{fp}, d_{fp}) \mid f = 1, \ldots, F, p = 1, \ldots, P \}$, where $\mathbf{v}_{fp}$ is the surface normal of the p-th patch in the f-th frame, and d_{fp} is the associated normal distance to the origin. To facilitate the decomposability of rotation and translation, instead of forming a $4F \times P$ measurement matrix as in Section 2.1, we form surface normals V_{fp} into a $3F \times P$ matrix $W^{(v)}$ and distances d_{fp} into an $F \times P$ matrix $W^{(d)}$. $W^{(v)}$ and $W^{(d)}$ are called the *normal measurement matrix* and *distance measurement matrix*, respectively.

$W^{(v)}$ and $W^{(d)}$ are highly rank-deficient. It can be easily shown that $W^{(v)}$ has at most rank 3 and $W^{(d)}$ has at most rank 4 when the data are noise-free. We decompose $W^{(v)}$ into

$$W^{(v)} = R\, V \qquad \text{(EQ 3.1)}$$

where

$$R = \begin{bmatrix} R^{(1)} \\ \cdots \\ R^{(F)} \end{bmatrix}$$

is the rotation matrix of each view with respect to the world coordinate system, and $V = [\mathbf{v}_1, \ldots, \mathbf{v}_P]$ is the surface normal matrix in the world coordinate system. Since R is an $3F \times 3$ matrix and V is an $3 \times P$ matrix, the rank of $W^{(v)}$ is at most 3.

Similarly, we can decompose $W^{(d)}$ into

$$W^{(d)} = T\,M \tag{EQ 3.2}$$

where

$$M = \begin{bmatrix} \begin{bmatrix} \mathbf{v}_1 \\ d_1 \end{bmatrix} & \cdots & \begin{bmatrix} \mathbf{v}_P \\ d_P \end{bmatrix} \end{bmatrix}, \text{ and } T = \begin{bmatrix} \begin{bmatrix} \mathbf{t}_1 R_1 & 1 \end{bmatrix} \\ \cdots \\ \begin{bmatrix} \mathbf{t}_F R_F & 1 \end{bmatrix} \end{bmatrix}$$

$\mathbf{t}_f$ and R_f are the translation vector and rotation matrix of view f with respect to a fixed world coordinate system.

Note that the decomposition of $W^{(d)}$ depends on the decomposition of $W^{(v)}$. Since M is $4 \times P$ and T is $F \times 4$, the rank of $W^{(d)}$ is at most 4.

We can also decompose $W^{(d)}$ into

$$W^{(d)} = \begin{bmatrix} \mathbf{t}_1 R_1 \\ \cdots \\ \mathbf{t}_F R_F \end{bmatrix} \begin{bmatrix} v_1 & \cdots & v_P \end{bmatrix} + \begin{bmatrix} 1 \\ \cdots \\ 1 \end{bmatrix} \begin{bmatrix} d_1 & \cdots & d_P \end{bmatrix} \tag{EQ 3.3}$$

When all elements in the two measurement matrices are known, we need to solve two least-squares problems. However, since only parts of the planar regions are visible in each view, we end up with two WLS problems instead. The first least squares problem, labeled as WLS-1, is

$$min \sum_{f=1,\ldots,F, p=1,\ldots,P} (\gamma_{f,p}^{(v)} (W_{f,p}^{(v)} - [RV]_{f,p}))^2 \tag{EQ 3.4}$$

and the second one, denoted as WLS-2, is

$$min \sum_{f=1,\ldots,F, p=1,\ldots,P} (\gamma_{f,p}^{(d)} (W_{f,p}^{(d)} - [TM]_{f,p}))^2 \tag{EQ 3.5}$$

where

$\gamma_{f,p} = 0$ if surface p is invisible in frame f, and $\gamma_{f,p} = 1$ otherwise. All weights can take on any value between zero and one, depending on the significance or confidence of each measurement.

3.1.1 Quaternion WLS-1

From the previous section, it appears that we can devise a simple two-step algorithm which solves WLS-1 and subsequently WLS-2 by applying the PCAMD algorithm to both problems. In order to solve WLS-1, we iterate $R^{(f)}$ as if it had nine independent parameters, while it is a nonlinear trigonometric function of three parameters. Although it is possible to normalize $R^{(f)}$ after every iteration, the naive algorithm may perform poorly in terms of robustness and efficiency.

In fact, several representations of rotation are often used in practice:

(1) An orthonormal rotation matrix R;

(2) A rotation axis **a** and a rotation angle θ;

(3) A rotation vector r [4].

(4) A unit quaternion q.

A quaternion is a 4-tuple (**w**,s) where **w** is a 3-vector and s is a scalar. The mapping between a unit quaternion and a rotation axis along with a rotation angle is given by $\mathbf{w} = \sin(\theta/2)\,\mathbf{a}$ and $s = \cos(\theta/2)$. The quaternion representation of the rotation matrix leads to a simple way of solving minimization problems of 3D point matching and surface normal matching, as demonstrated in [37].

The WLS-1 problem (EQ 3.4) can be decomposed into F minimization problems

$$min \quad \sum_{p=1}^{P} \gamma_f^{(v)} \left\| \mathbf{w}_f - R^{(f)} \mathbf{v}_p \right\| \qquad \text{(EQ 3.6)}$$

where

$f = 1, \ldots, F$, $\mathbf{w}_f = [W_{f,1}, \ldots, W_{f,P}]^T$, $\gamma_f^{(v)} = diag(\gamma_{f,1}, \ldots, \gamma_{f,P})$.

Therefore WLS-1 can be reformulated using quaternions as

$$min \quad \sum_{p=1}^{P} \gamma_f^{(v)} \left\| \mathbf{w}_f - q^{(f)} \mathbf{v}_p \tilde{q}^{(f)} \right\| \qquad \text{(EQ 3.7)}$$

$$subject\ to \qquad |q| = 1$$

where $\tilde{q}$ is the conjugate quaternion of q, and

$$\sum_{p=1}^{P} \gamma_f^{(v)} \left\| \mathbf{w}_f - q^{(f)} \mathbf{v}_p \tilde{q}^{(f)} \right\| = \sum_{p=1}^{P} \gamma_f^{(v)} \left\| \mathbf{w}_f q^{(f)} - q^{(f)} \mathbf{v}_p \right\| = \sum_{p=1}^{P} q^{(f)T} A_p^{(f)} q^{(f)} \qquad \text{(EQ 3.8)}$$

$A_p^{(f)}$ are symmetric matrices because $\mathbf{w}_f q^{(f)} - q^{(f)} \mathbf{v}_p$ is a linear function of $q^{(f)}$. Obviously

$$B^{(f)} = \sum_{p=1}^{P} A_p^{(f)} \qquad \text{(EQ 3.9)}$$

is also symmetric, and the minimization problem (EQ 3.7) becomes

$$min \quad q^{(f)T} B^{(f)} q^{(f)} \qquad \text{(EQ 3.10)}$$

The solution to the above minimization problem is the eigenvector $q_{min}^{(f)}$ corresponding to the minimum eigenvalue of the matrix $B^{(f)}$.

3.1.2 An Iterative Algorithm

We combine the quaternion-based rotation matrix updating to form a two-step algorithm to solve both the first and the second WLS problems. The algorithm is as follows:

Algorithm two-step WLS's

Step 0 Initialization

(0.1) read in measurement matrices $\mathbf{W}^{(\mathbf{v})}$, $\mathbf{W}^{(\mathbf{d})}$

(0.2) read in weight matrices $\gamma^{(\mathbf{v})}$, $\gamma^{(\mathbf{d})}$

(0.3) initialize R, vectorize R to v

Step 1 WLS-1

(1.1) vectorize $W^{(v)}$ to w_{v1} and w_{v2}

(1.2) update $H^{+}_{v\gamma}$

(1.3) update

$$u = H^{+}_{v\gamma} w_{v1}$$

(1.4) update

$$B^{(f)}$$

(1.5) update

$$q^{(f)}$$

and transform to R, vectorize to v

(1.6) go to (1.2) if not converged, otherwise advance to Step 2.

Step 2 WLS-2

(2.1) vectorize $W^{(d)}$ to w_{d1} and w_{d2}

(2.2) update $H^{+}_{d\gamma}$

(2.3) update

$$u = H^{+}_{d\gamma} w_{d1}$$

(2.4) update

$$K^{+}_{d\gamma}$$

(2.5) update

$$\tilde{v} = K^{+}_{d\gamma} w_{d2}$$

(2.6.) stop if converged, otherwise go to (2.2).

Until now, we have not yet explicitly discussed the normalization problem in our WLS approach. The problem of normalization occurs because the measurement matrix is rank-deficient. Hence, unless an additional constraint is imposed, there are infinitely many solutions to the minimization problem (EQ 3.1). This additional constraint is generally problem-dependent. For example, the 2-norm of the factorized left matrix is constrained to be unity [123]. Fortunately, we have implicitly constrained our rotation matrices in their quaternion representation. The remaining constraint in the first WLS is the normalization of surface normal vectors which are constrained to be of unit magnitudes.

3.2 Surface Patch Tracking

Prior to multiple view merging, we need to track surfaces so that a normal measurement matrix and a distance measurement matrix can be formed. In this section, we briefly overview each module of our surface patch tracking system: range image segmentation, adjacency graph building, and two-view matching.

3.2.1 Range Image Segmentation

There are many different techniques for range image segmentation. By and large, they can be divided into feature-based and primitive-based approaches, although a statistics-based approach has also been introduced recently. The feature-based approach yields precise segmentation but, in practice, it is sensitive to noise. For example, Gaussian and mean curvatures can be used to label different regions before region growing. However, this process is quite noise-sensitive because of the presence of the second-order derivative. The primitive-based approach, on the other hand, is more robust to noise, yet it is constrained by the number of primitives. The higher the degree of the surface polynomial, the more difficult and the less robust the segmentation is likely to be.

In this thesis, the planar surface region growing segmentation method of [37] is used. The regions are established via region growing from seed points that are chosen from those closest to their approximating planes. The regions are then merged with their neighbors until the best-fit errors become unacceptable.

3.2.2 Adjacency Graph

Once we have successfully segmented the range data for each view, the range image associated with view i can be represented as a set of planar regions $I_i = \{v_{ij}, d_{ij}, c_{ij}\}$, where v_{ij} and d_{ij} are the normal and distance of the j-th segment planar surface respectively, and c_{ij} is the centroid of the j-th segmented region.

From each view of the 3D object, we build an adjacency graph. Every node in the graph represents a visible planar region and each arc connects two adjacent nodes. The adjacency graph is updated whenever the view on which it is based is matched with another. After tracking all planar patches for the whole sequence, we have adjacency information among all visible planar regions. From the adjacency graph, all the object vertices can be located. Thus, a 3D object model is obtained. However, augmenting the adjacency graph is difficult

for concave objects because of occlusion. A better way of establishing spatial connectivity among all surfaces is discussed in Section 3.3.

We have implemented a planar surface patch tracking system which combines an adjacency graph with range data because there is significant change in range data across an occluding edge.

3.2.3 Matching Two Views

Given two adjacent segmented images I_1 and I_2, we would like to find a correspondence between different regions in the two views. We seek a mapping ϕ: $(I_1 \rightarrow I_2)$ such that a certain distance measurement $d(I_1, I_2)$ is minimized.

Two problems arise in matching two views of planar regions:

(1) how to establish correspondence between the two views; and

(2) how to recover the transformation between them.

Our solution to the first problem is to use the adjacency information between two segmented patches and between segmented surface normals. If the displacement between two views is relatively small, there should be only a linear shape change [58] within the same aspect. Corresponding segmented regions are of similar size (number of points), centroid, and surface normals. When a new aspect appears, indicating a nonlinear shape change, there would be significant change in these parameters. There may not always be solutions to the second problem because we need at least two corresponding non-parallel faces to determine rotation and three to determine translation. In practice, we can always make the assumption that we have two non-parallel corresponding faces in two adjacent views.

In fact, solving the second problem can be of help in solving the first one. This is because once we have an initial estimation of transformation, we can then make use of the hypothesize-and-test approach. We can iteratively select two pairs of non-parallel faces from the two images to be matched, estimate the corresponding rotation matrix, and then attempt to match the remaining faces. We always choose two adjacent faces from both images, and match them based on the surface normals, distances and centroids of the segment regions. The number of faces matched and the consistency in face adjacency are used in the distance measure between the two matches. The estimated transformation matrix is used only to help

in building the adjacency graph, while the precise transformation is robustly recovered using our WLS method.

Multiple view tracking is done by sequentially matching two adjacent views. Whenever a new view is added, the adjacency graph and the weight matrix are automatically modified. Because of the problems associated with updating the adjacency graph subsequent to surface patch tracking and multiple view merging, we use another algorithm in next section to establish the spatial connectivity among surfaces.

3.3 Spatial Connectivity

Once we have extracted the equations of planar surfaces of the object, we need to establish the spatial connectivity relationship among these surfaces. One approach is to build an adjacency graph from a sequence of views, as discussed in the previous section. However, augmenting the adjacency graph whenever a new view is introduced is quite ad-hoc. In this section, we present a new approach to recovering surface connectivity after all surface patches are recovered. We show that the problem of spatial connectivity between boundary surfaces can be reduced to one of connectivity between supporting lines of a simple polygon.

3.3.1 Half-space Intersection and Union

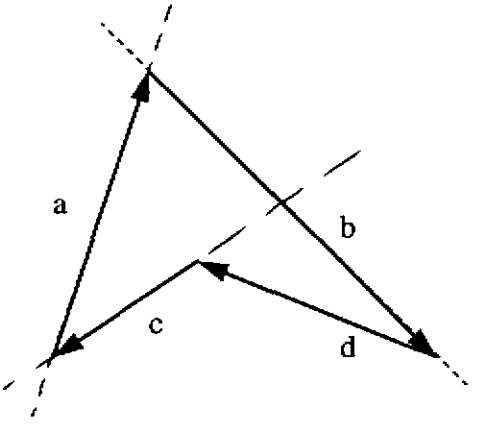


Figure 6 **A simple polygon and its supporting lines (stippled and solid lines)**

We assume that every planar patch P of the object model is a simple polygon. A simple polygon does not self-intersect. Every (infinite) plane divides the space into two parts, inside and outside, with surface normal pointing toward the outside of the object. Given an unbounded planar surface, if we intersect all other planar surfaces on it, we obtain supporting lines as illustrated in Figure 6. Each supporting line is directed so that the interior of P lies locally to its right. The right half-plane created by such a directed supporting line c is called the supporting half-plane, and is characterized as supporting the polygon [32]. However, a concave P might not all lie in the right half-plane as indicated in Figure 6.

For each point x on the plane, if we know on which side of each supporting line x lies, then we know if x is inside P. Therefore, the polygon P (and its interior) can be represented as a boolean formula whose atoms are those supporting lines. In other words, a simple polygon can be represented by the intersection and the union of its supporting lines. For example, a boolean formula for the polygon in Figure 6 can be $\overline{c\bar{a} \oplus a\bar{b} \oplus b\bar{d} \oplus \bar{d}c}$. This Guibas style [32] formula is obtained by complementing the second supporting line at a convex angle, and the first supporting line at a concave angle when we go around the polygon. Once spatial connectivity is established, the Guibas style formula is straightforward. Other boolean formulae such as the Peterson style are also possible [32].

3.3.2 Modified Jarvis' March

The problem of establishing spatial connectivity of supporting lines can be formulated as a modified convex hull-like problem which involves only vertices. This problem can also be regarded as one of cell decomposition which involves data points. We propose a modified Jarvis' march algorithm to reconstruct simple polygons from supporting lines and valid data points. Once transformations among different views are recovered, all valid range data points can be transformed to a single world coordinate system. The algorithm to recover spatial connectivity among 3D surfaces is discussed in Section 3.3.3.

Definition 1

A point is defined as valid in a simple polygon if there exist sufficient valid data points around its neighborhood.

Lemma 2

The intersection point P of two supporting lines is valid in a simple polygon if and only if the intersection of two corresponding half-planes is valid locally at P.

Proof:

When the intersection of two half-planes is valid locally at P, the intersection point of these two supporting lines is valid by definition.

Assume that the intersection point of two supporting lines is valid. Given that two lines divide the plane into four regions, at least one such region out of four around the intersection point must be a valid cell of the simple polygon. Therefore, the intersection of two half-planes is valid locally at P.

Lemma 2 leads to a modified Jarvis' march algorithm of reconstructing a simple polygon from supporting lines and valid data points.

To construct a simple polygon from all supporting lines and valid data points, we first precompute all intersection points which can be candidates of the vertices of the simple polygon. If we march successive vertices with the least turning angle, we obtain their convex hull. This is referred to as the Jarvis' march algorithm [96]. The kernel of the simple polygon, if it exists, can also be found by intersecting all half-spaces. *Lemma 2*, therefore, enables us to find the correct simple polygon by marching all points whose local neighborhood is valid. We call this algorithm the "modified Jarvis' march".

Assume that we start with the lowest left point p_1 of the set of vertex candidates. p_1 is certainly a convex hull vertex, but it is not necessarily a vertex for our simple polygon (unless it is valid locally). For example, in Figure 7, p_1 is not a simple vertex because $p_5p_1p_2$ is not a valid triangle cell (valid cells with valid data points are shaded). Since $p_6p_2p_3$ is a valid triangle cell, we start our algorithm from p_2.

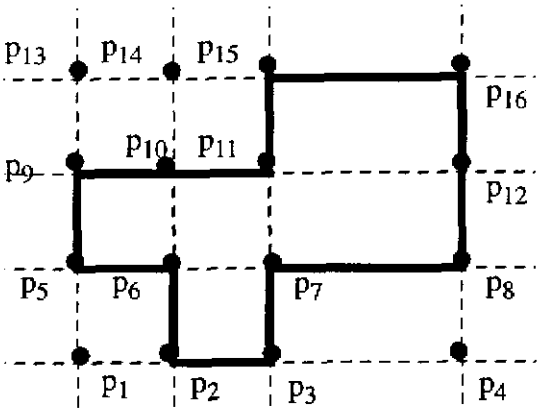


Figure 7 **An example of modified Jarvis' march and cell decomposition. Shaded area represents valid data points**

A data structure is defined for each intersection point *P* as follows:

```
typedef {
   intersect-point left, right, up, down;
   intersect-point previous, next;
} intersect-point  P.
```

Figure 8 shows the relationship among the members of the data structure. Assume that an intersection point is intersected by only two supporting lines.

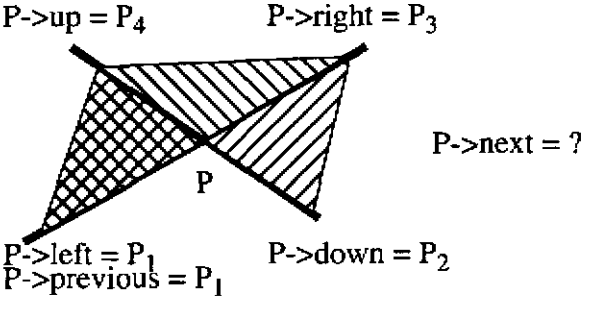


Figure 8 **An illustration of the data structure of intersection point**

After the starting vertex is found, we march to the next vertex as illustrated in Figure 8. If there are sufficient data points in cell PP_2P_3, the next valid vertex is P_2; if P_2 is not valid, we check if P_3 is valid; if P_3 is also invalid, P_4 must be valid, or an error will occur. The march ends when the next vertex is the starting vertex. The modified Jarvis' march (MJM) algorithm is given as follows:

Algorithm MJM

Step 1. initialize starting vertex

START->previous = **NULL**,
P = **START**->next,
P->previous = **START**;

Step 2. march

P->left = $\mathbf{P_1}$; **P**->down = $\mathbf{P_2}$; P->right = $\mathbf{P_3}$; **P**->up = $\mathbf{P_4}$;
if cell $\mathbf{PP_2P_3}$ **valid,** **P**->next = $\mathbf{P_2}$ (case 1)
else if cell $\mathbf{PP_3P_4}$ **valid,** **P**->next = $\mathbf{P_3}$ (case 2)
else if cell $\mathbf{PP_4P_1}$ **valid,** **P**->next = $\mathbf{P_4}$ (case 3)
else error occurs;

Step 3. terminate

if P->next = START.

A post-processing step may be necessary to remove points which belong to case 2 in step 2 of the march algorithm. Each of these points is on the same line with its previous point and its next point. For example, in Figure 7, p_{12} can be removed because p_8 and p_{16} make it redundant.

As can be seen from the above algorithm, shown in Figure 7, and further illustrated in Figure 8, the problem of single polygon reconstruction from supporting lines and valid data points is one of cell decomposition. As we march around all supporting lines, the Guibas style boolean formula of the simple polygon can be readily formulated.

3.3.3 3D Spatial Connectivity

So far we have discussed the problem of recovering the connectivity of supporting lines of a simple polygon. The approach uses information at both the signal level (real data points) and the algebraic level (line equations).

The same hybrid approach can also be applied to the problem of spatial connectivity of planar surfaces in 3D. Indeed, the problem of connectivity of planar surfaces in 3D can be reduced to a set of problems of connectivity in 2D. Assume that we have recovered a set of N face equations and transformation among different views (e.g., from *PCAMD*). All valid data points from multiple views can be merged in the same world coordinate system. For each face F_i, if we intersect all other *N-1* faces F_j ($j = 1, \ldots, N-1$, $j \neq i$) with F_i and project all these lines onto F_i, we get *M (=N-1)* supporting lines on face F_i. We also project nearby 3D points onto this face F_i. Without loss of generality, we assume that no two supporting lines are parallel (or a normal threshold d can be set such that $v_i v_j \geq d$). For any of the M supporting lines, if we intersect it with the remaining *M-1* lines, we get all possible candidates for vertices of the valid simple polygon which is the model of face F_i. The modified Jarvis' march algorithm can then be applied to each of the N faces accordingly. By connecting all polygons recovered, we get the entire 3D object model boundary. Accordingly, a simple algorithm can then be constructed to establish 3D spatial connectivity.

Figure 9 illustrates the reconstruction of such connectivity. It shows one face of a toy house model. Figure 9a shows the intersections of supporting lines and nearby data points projected on this face, while Figure 9b superimposes a reconstructed simple polygon model of this face on Figure 9a. The complete house model is reconstructed and presented in Section 3.4.

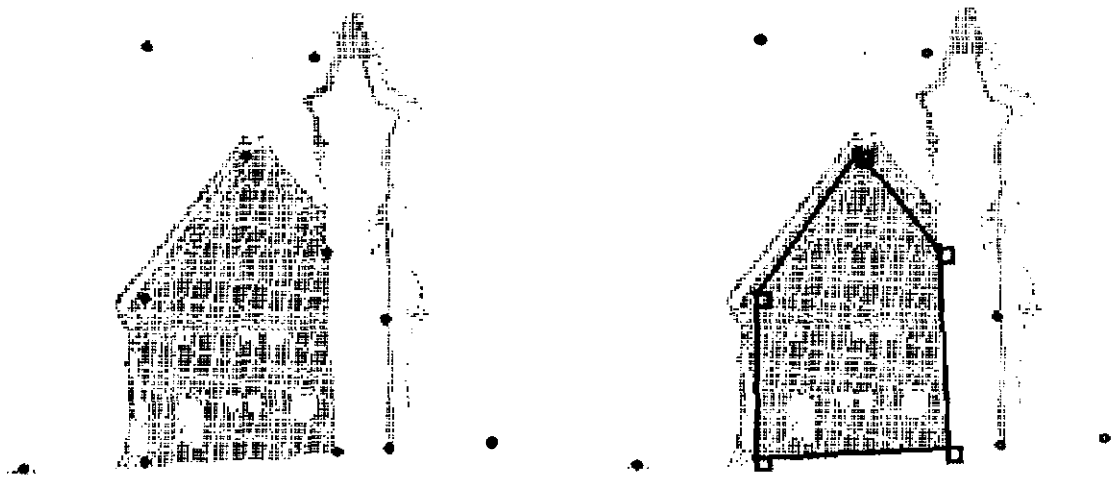


Figure 9 **Reconstruction of connectivity. The tiny dots represent projected nearby data points. Intersections of supporting lines are represented by black circles. Vertices of reconstructed simple polygon are represented by small squares**

3.4 Experiments

In this section, we present the results of applying our algorithm on synthetic data and on real range image sequence of objects. We demonstrate the applicability and the robustness of our approach using synthetic data, and present the recovered model from real range images.

Our synthetic data set consists of a set of 12 planes as in the case of the dodecahedron in Chapter 2. A dodecahedron with 4 different views is shown in Fig. 3 in Chapter 2.

3.4.1 Applicability

In this section we study the applicability of the proposed approach. In order to recover the shape of a dodecahedron, given that the correspondence is known, how many views are needed?

In this case, we pick 4 distinct views from the viewing sphere so that there is no singularity. A singularity occurs when fewer than 6 faces are visible. We can then formulate two measurement matrices for surface normals and planar distances as follows:

$$W^{(v)} = \begin{bmatrix} \mathbf{v}_1^{(1)} & \mathbf{v}_2^{(1)} & \mathbf{v}_3^{(1)} & \mathbf{v}_4^{(1)} & \mathbf{v}_5^{(1)} & \mathbf{v}_6^{(1)} & * & * & * & * & * & * \\ \mathbf{v}_1^{(2)} & \mathbf{v}_2^{(2)} & \mathbf{v}_3^{(2)} & \mathbf{v}_4^{(2)} & * & * & \mathbf{v}_7^{(2)} & \mathbf{v}_8^{(2)} & * & * & * & * \\ \mathbf{v}_1^{(3)} & \mathbf{v}_2^{(3)} & * & * & * & \mathbf{v}_6^{(3)} & * & \mathbf{v}_8^{(3)} & \mathbf{v}_9^{(3)} & \mathbf{v}_{10}^{(3)} & * & * \\ * & * & * & * & * & * & \mathbf{v}_7^{(4)} & \mathbf{v}_8^{(4)} & \mathbf{v}_9^{(4)} & \mathbf{v}_{10}^{(4)} & \mathbf{v}_{11}^{(4)} & \mathbf{v}_{12}^{(4)} \end{bmatrix} \quad \text{(EQ 3.11)}$$

$$W^{(d)} = \begin{bmatrix} d_1^{(1)} & d_2^{(1)} & d_3^{(1)} & d_4^{(1)} & d_5^{(1)} & d_6^{(1)} & * & * & * & * & * & * \\ d_1^{(2)} & d_2^{(2)} & d_3^{(2)} & d_4^{(2)} & * & * & d_7^{(2)} & d_8^{(2)} & * & * & * & * \\ d_1^{(3)} & d_2^{(3)} & * & * & * & d_6^{(3)} & * & d_8^{(3)} & d_9^{(3)} & d_{10}^{(3)} & * & * \\ * & * & * & * & * & d_6^{(4)} & * & d_8^{(4)} & d_9^{(4)} & d_{10}^{(4)} & d_{11}^{(4)} & d_{12}^{(4)} \end{bmatrix} \quad \text{(EQ 3.12)}$$

In order to solve WLS-1 uniquely for F frames, we need

$$18F \geq 3F + 3P,$$

since we have *18F* equations and *3F* unknowns for rotation matrices and *3P* unknowns for surface normals. For WLS-2, we have *6F* equations, but there are only F unknown translation vectors and P unknown plane distances after using results from WLS-1. Therefore, the necessary condition to uniquely solve WLS-2 is

$$6F \geq 3F + P.$$

Since P is *12*, $F \geq 4$ is the unique solution for both problems. Again, we are not concerned with the normalization problem at this point.

3.4.2 Robustness

We study the effectiveness of our approach when data is corrupted by noise, and mismatching occurs. Our synthetic data consists of a set of 12 surface patches randomly distributed around all faces of a dodecahedron. Correspondence is assumed to be known. Only the first WLS problem is studied because of the similarity between those two WLS problems. The minimization of weighted squares distance between the reconstructed and the given measurement matrices leads to the recovery of surface equations and transformations.

To study the error sensitivity on the reconstruction of our algorithm, we take four nonsingular views of the dodecahedron. Each component in every surface normal of these four views is corrupted by a Gaussian noise of zero-mean and variable standard deviation. As we have

shown in the previous section, at least 4 views are required to recover the dodecahedron model. Figure 10 shows that our algorithm converges in a few steps. The cases with standard deviation σ of 0.05, 0.1, 0.2 and 0.5 are studied. Notice that the case with standard deviation of 0.5 yields very noisy original data. As we take more views, the sum of the weighted squares error is reduced. Figure 11 plots the normalized weighted least squares error for 4, 8, 12, and 16 views respectively, while Gaussian noise with 0.5 standard deviation is present. The sums of squares errors are normalized because the number of observations increases as more views are introduced.

A more interesting case is when mismatching occurs. Obviously if a face appears only once in the whole sequence, then its reconstruction depends on the amount of noise. When this face appears in more and more views, its reconstruction using our WLS method is averaged over these views. Figure 12 gives the reconstructed errors of a face which appears 12 times in 16 views. When only two views are matched, the reconstructed surface normal deviates from its normal by 18.2 and 38.9 degrees when the respective standard deviation σ is 0.1 and 0.2. When more views are added, the angle between the reconstructed surface normal and its normal decreases to around 10 and 20 degrees, respectively.

When an observed surface normal is inaccurate in one particular view, the conventional sequential reconstruction method results in an erroneous recovered surface normal and transformation. Errors propagate as new views are introduced, regardless of the number of views in which this surface is visible. However, our WLS approach gives appreciably smaller reconstruction error on this observed surface normal by distributing the errors over all views. In any case, in general, our approach is better than the sequential approach. Figure 13 compares the reconstructed errors of the sequential method and WLS. There are 12 observations of this surface normal from 16 views and its first observation is off by an angle between 0^o and 40^o. The reconstructed models, for the case of a 40^o angle deviation of one surface normal in the first view, using the sequential and the WLS methods are shown in Figure 14 along with the original model. Figure 14a shows a badly-skewed model, which is the worst case from the sequential method, since error was introduced in the very first frame. Figure 14b shows the reconstructed model by the WLS method, while the original dodecahedron model is presented in Figure 14c.

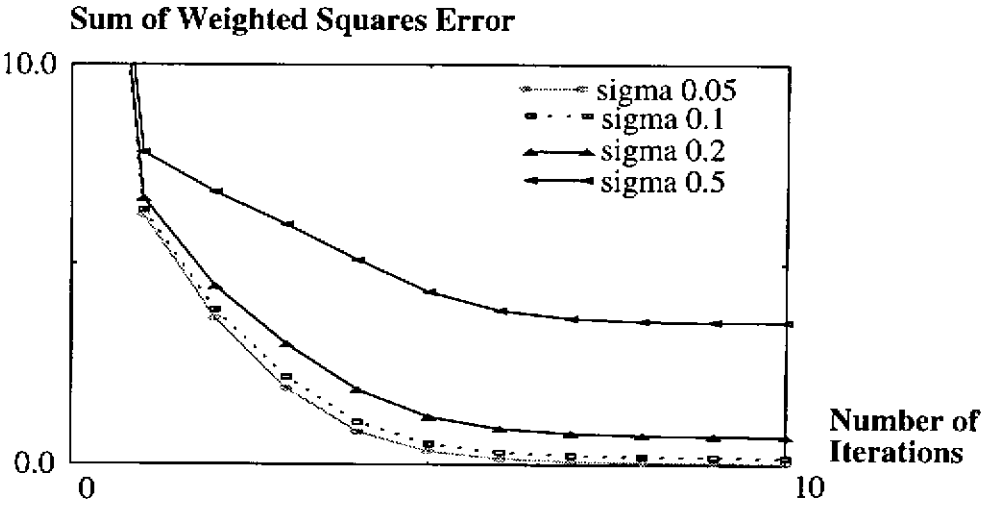


Figure 10 **Effect of noise**

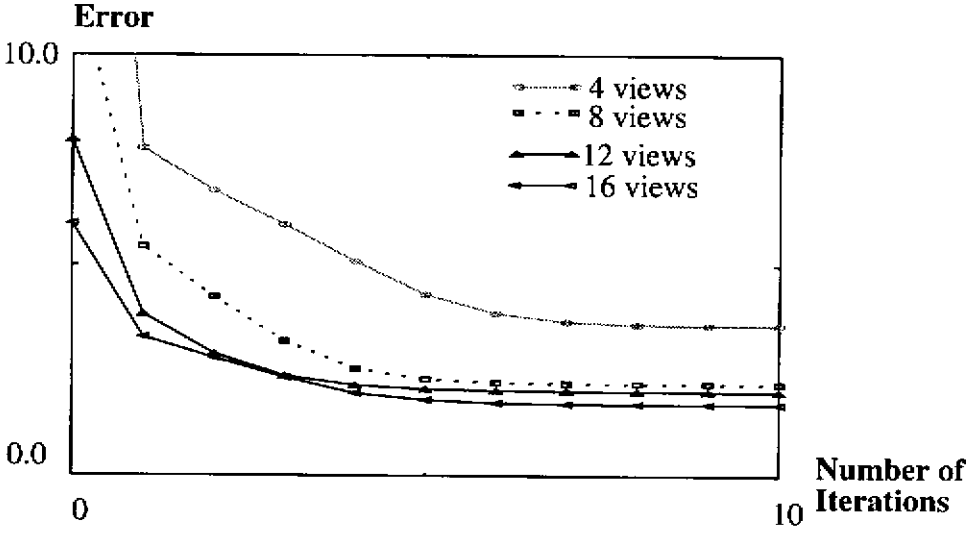


Figure 11 **Effect of number of views**

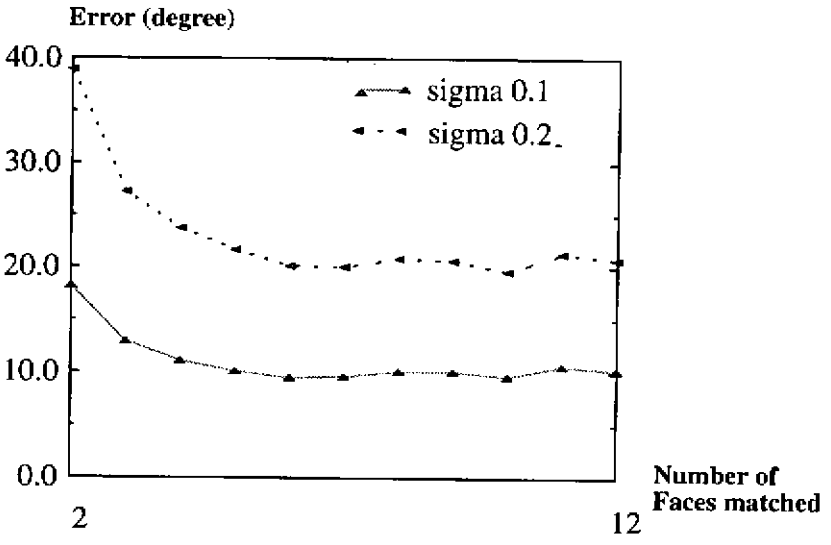


Figure 12 **Reconstructed error vs. number of matched faces**

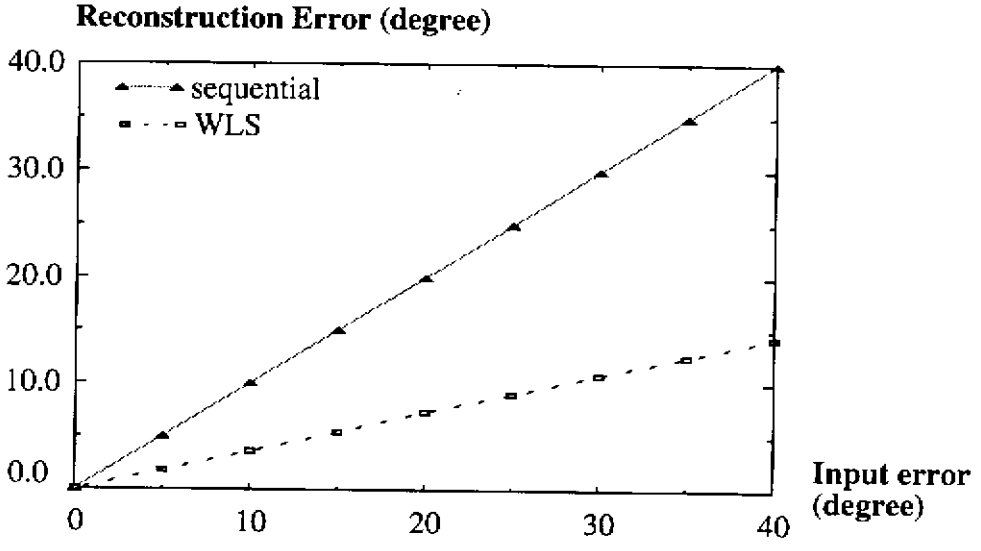
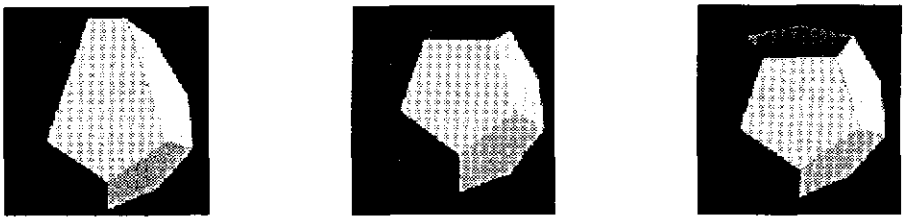


Figure 13 **A comparison between the sequential and the WLS methods**

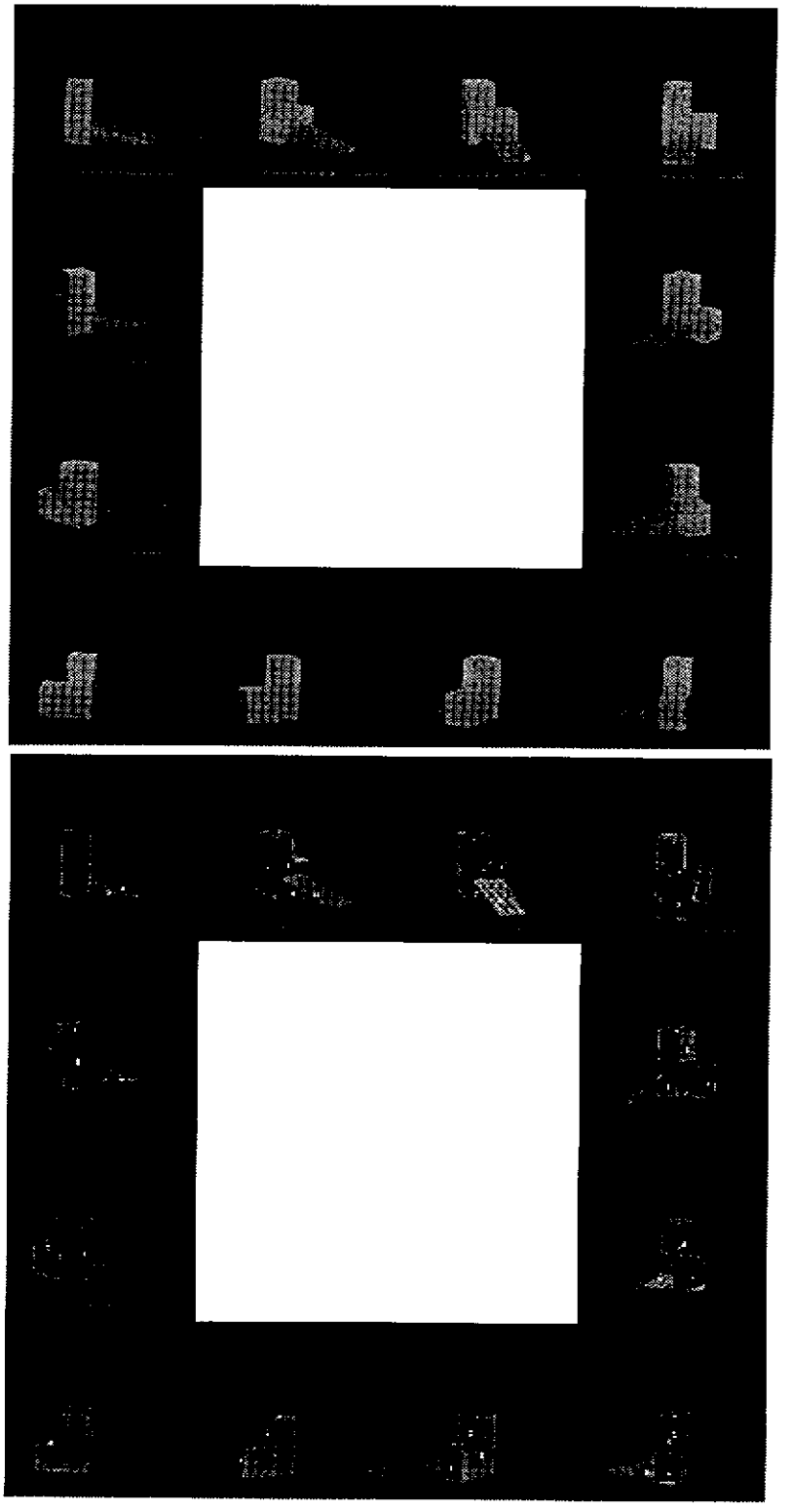


(a) (b) (c)

Figure 14 **Recovered and original dodecahedron models: (a) the worst case of the sequential method; (b) our WLS method; and (c) the original model**

3.4.3 Real Range Image Sequence

We have applied our algorithm to a sequence of range images of a polyhedral object, using the planar region tracker described in Section 3.2. Figure 15a and Figure 15b show the entire sequence with 12 views and their corresponding segmentation results. Segmentation is not perfect in several views. Figure 16 shows the result of our system, with two shaded views of a recovered object model. Figure 17 and Figure 18 show the example of a toy house.



(a)

(b)

Figure 15 **A sequence of images of a polyhedral object: (a) original images; and (b) after segmentation**

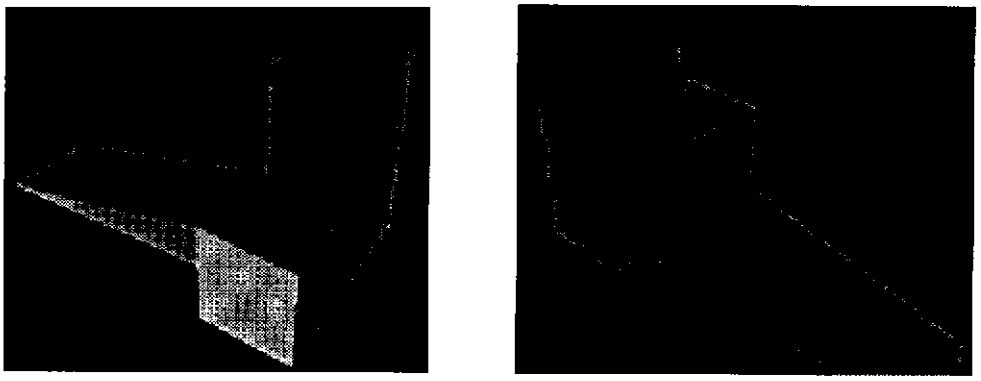
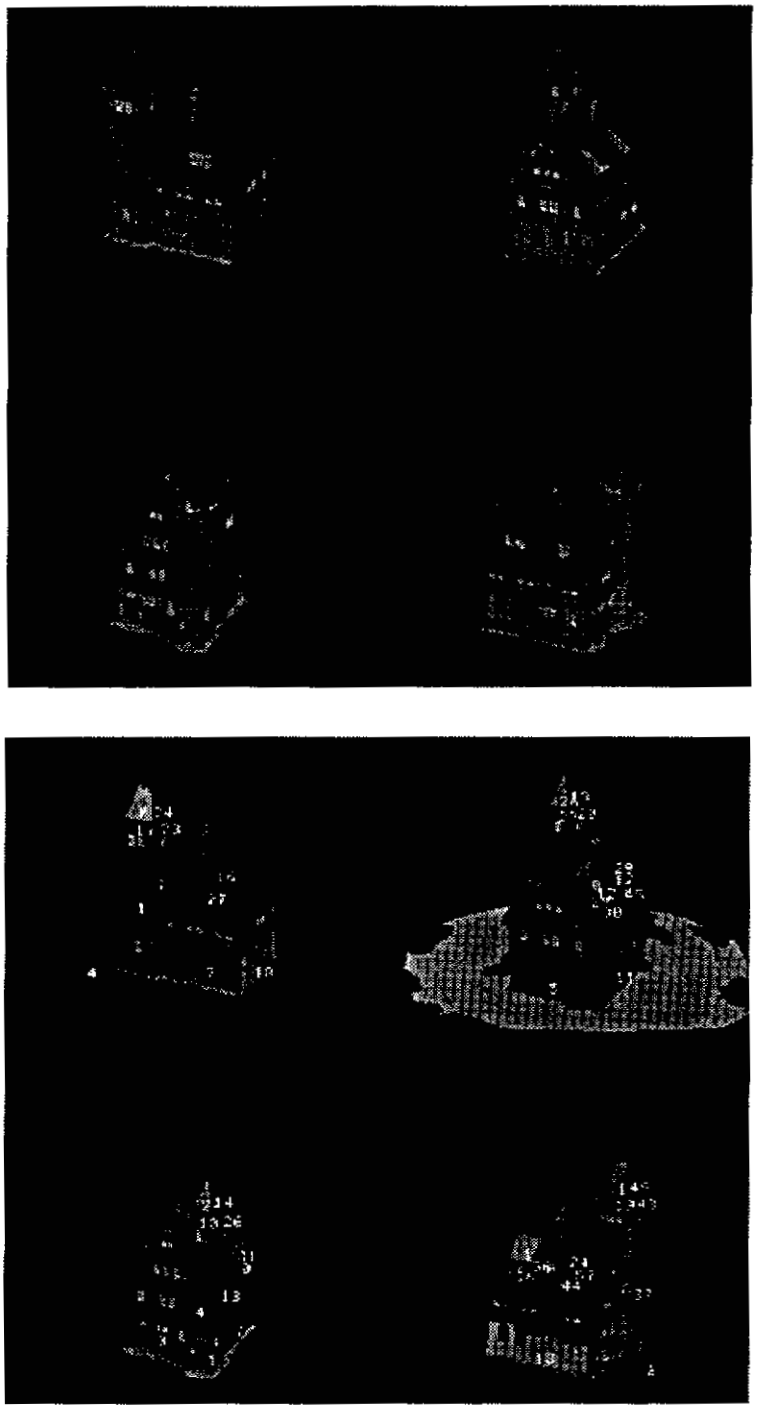


Figure 16 **Two views of shaded display of a recovered model**



(a)

(b)

Figure 17 **A sequence of images of a toy house: (a) original images; and (b) after segmentation**

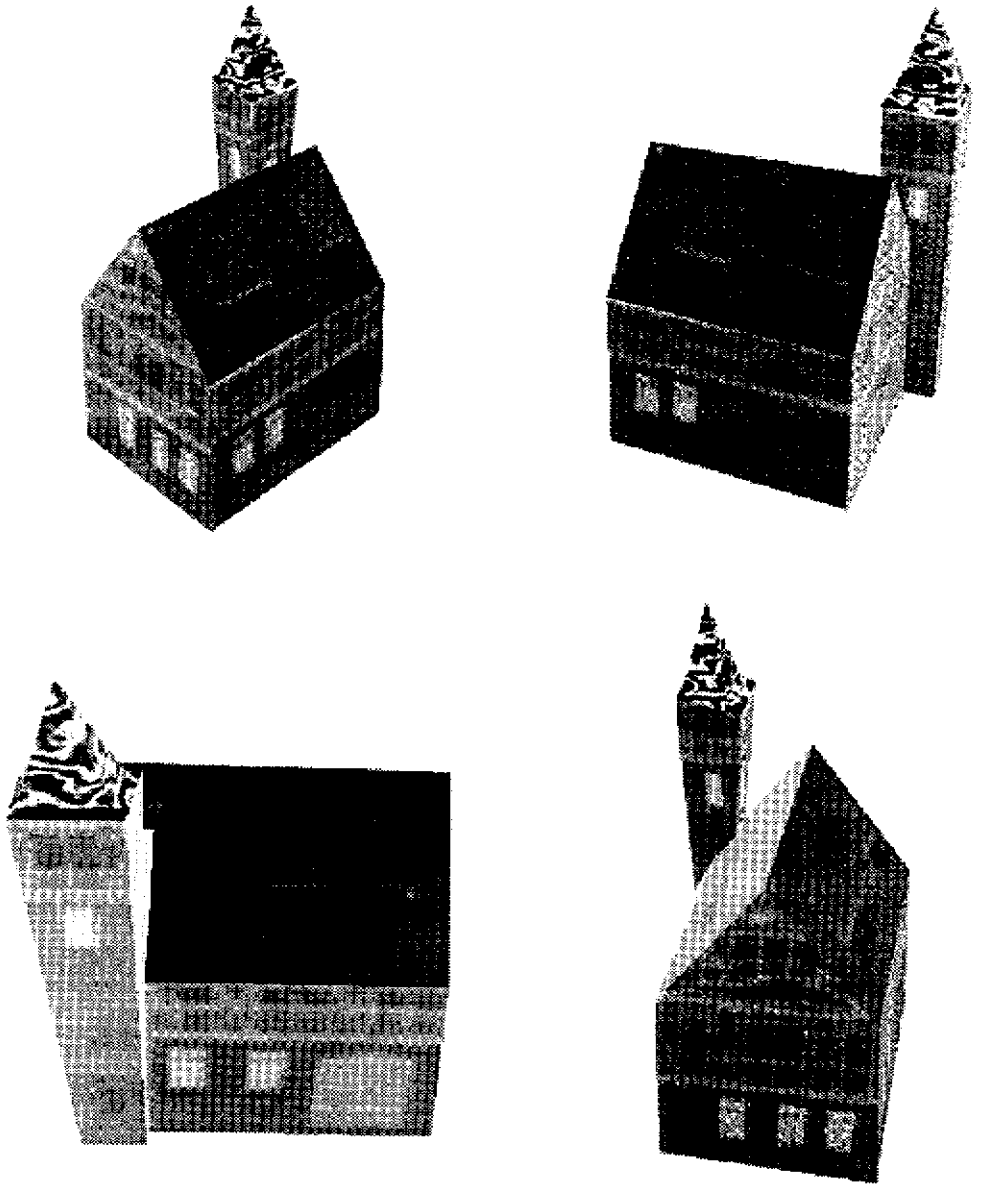


Figure 18 **Four views of texture-mapped display of a reconstructed house model**

3.5 Discussion

In this chapter, we have applied our integral approach to polyhedral object modeling. The planar boundary surface model is recovered and integrated from different views. An inherent problem in multiple view integration is that the information observed from each view is incomplete and noisy. One significant contribution of our work is the application of PCAMD to obtain a statistically optimal object model from a sequence of views.

With zero weights assigned to the unobservable data, merging different views can be formulated as a combination of two WLS minimization problems. By applying the PCAMD algorithm to both problems, we get a straightforward two-step algorithm in which the first step computes surface normals and rotation matrices by employing the quaternion representation of the rotation matrix; the subsequent step recovers translation vectors and normal distances to the origin. Experiments on synthetic data and real range images indicate that our approach converges quickly and produces good models even in the presence of noise and mismatching. An accurate polyhedral object model reconstructed from a sequence of real range images is presented. A complex toy house model is also reconstructed. More complicated scene modeling is presented in the next chapter.

When the motion between two views is relatively small, we can track different segmented surface patches by making use of surface normals, distances, centroids, and adjacency information. An adjacency graph is built for each view and is modified as the viewing direction changes. A significant advantage of surface patch tracking, as opposed to other methods such as point matching and line segment tracking, is that surface patches can be more reliably extracted and tracked.

A hybrid approach has been used to establish spatial connectivity of boundary surfaces. The spatial connectivity of surfaces, and in particular, the supporting lines of a simple polygon, can be obtained by combining algebraic equations of surfaces and data points merged from multiple views once the transformation is recovered.

Chapter 4

Scene modeling

One of the most important applications of modeling from reality is modeling a virtual environment from existing scenes and objects [38]. One problem we face with modeling a virtual environment is the problem of levels of detail. For example, we can model indoor or outdoor scenes with a polyhedral representation, but we may have to resort to free-form representation for simple objects. Thus, we can decompose modeling from reality into two problems: scene modeling and focused object modeling. This chapter focuses on scene modeling -- a case of polyhedral object modeling from a sequence of images.

4.1 The System

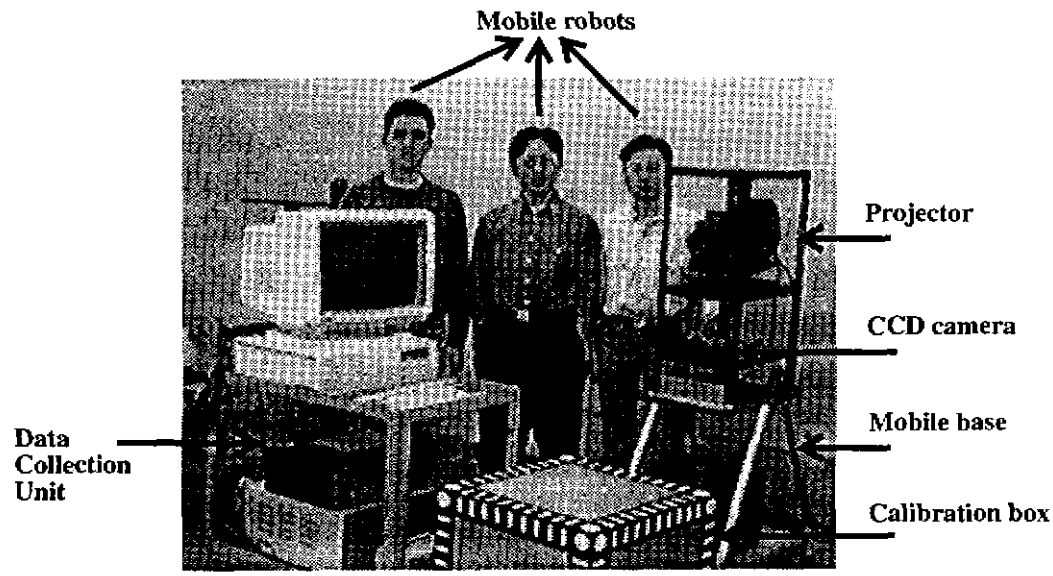


Figure 19 **The system used for modeling the virtual Wean Hall at CMU**

As shown in Figure 19, our system consists of a light-stripe range finder on a mobile base, a data collection unit and a calibration unit. The range finder is composed of a projector and a CCD camera. In order to calibrate the range finder, we make a cube of $60 \times 60 \times 60$ cm^3 which has equidistant marks on each face. The range finder has a dynamic range of about $2m$, which is a restriction of our data collection apparatus.

We have taken a sequence of images of the first floor Wean Hall corridor at CMU. Four representative views and their correspondent segmented views from the sequence are shown in Figure 20. After the sequence is tracked using segmented patches, the PCAMD algorithm is applied to obtain the planar surface equations.

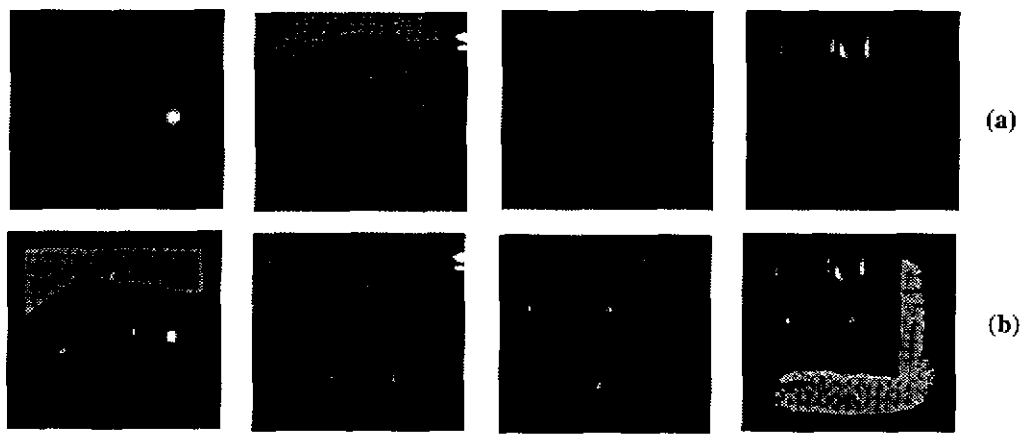
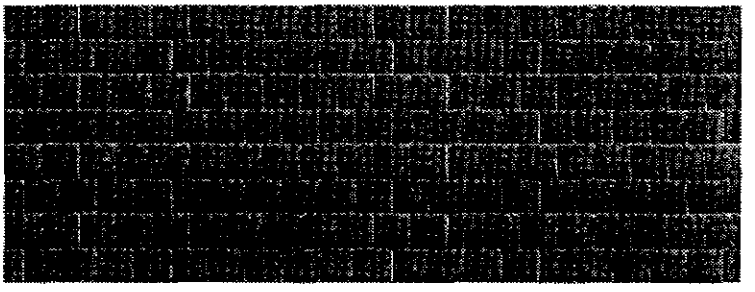


Figure 20 **Four views from the sequence of images of Wean Hall first floor: (a) intensity images; and (b) segmented images**

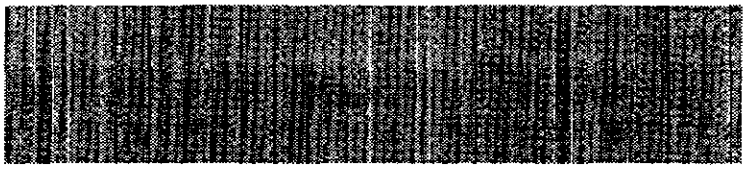
4.2 The Result

First, the geometry of the Wean Hall model is reconstructed using the tracking, integration and connectivity techniques presented in the previous chapter. We also carefully extract texture maps from a sequence of color images of the scene. The process is manual and quite tedious. To ensure good results, we use only images which are taken perpendicular to the camera viewing direction. Three samples of the texture maps are shown in Figure 21. Because our OGIS range finder has only a black and white CCD camera, we take additional color images to extract textures.

Several views of the reconstructed Wean Hall model are shown in Figure 22. To run visual simulation with texture mapping, we code the geometry of the model using the FLT format (i.e., OpenFlight Format Specification by MultiGen Inc. [90]). All views in Figure 22 are screen-dumped while running "perfly" (a demo program based on the IRIS Performer Graphics Library) on a Silicon Graphics SGI reality engine.



(a)



(b)

(c)

Figure 21 **Samples of texture maps used in Wean Hall model: (a) Brick (b) Concrete and (c) Wood**

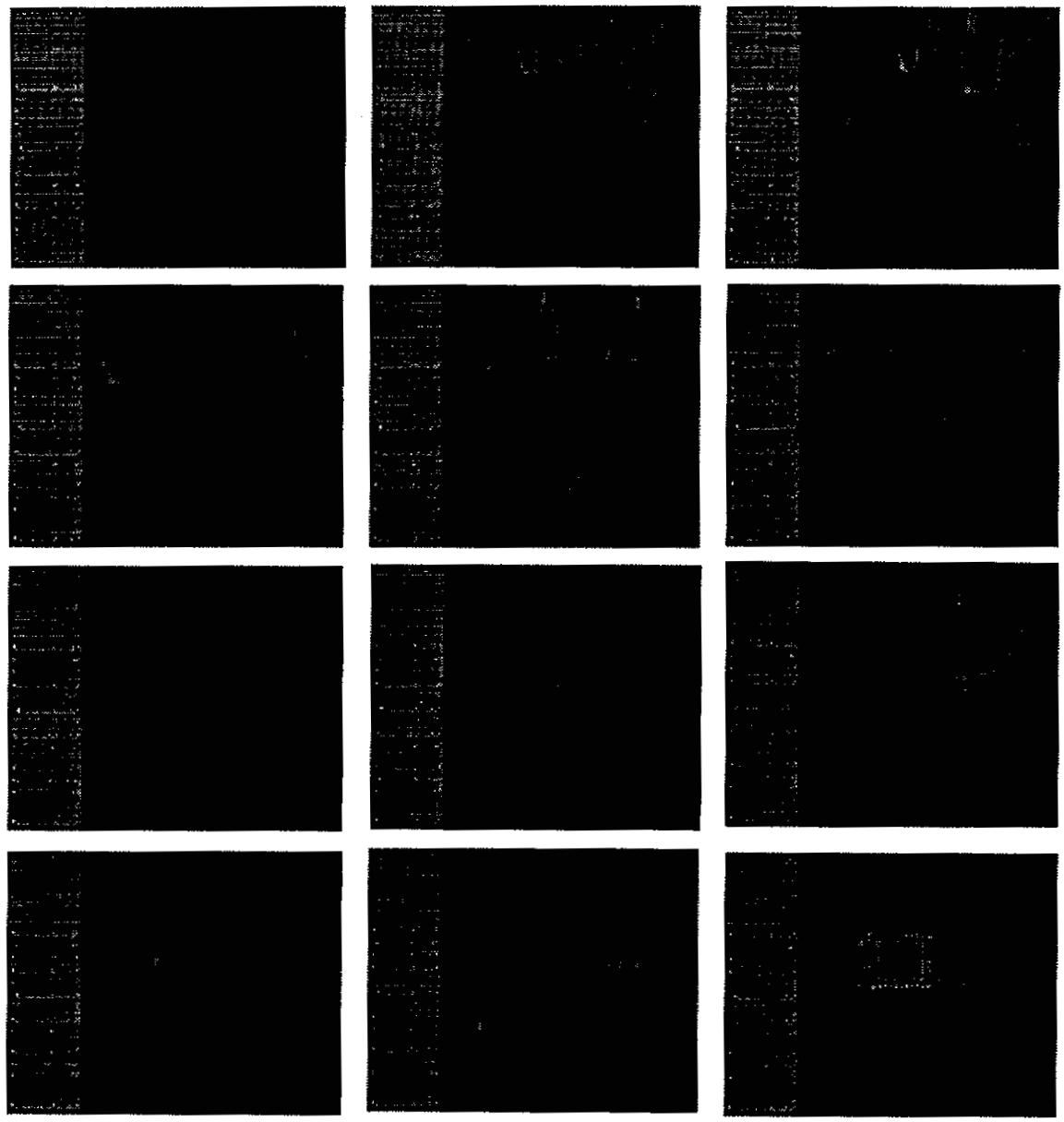


Figure 22 **Several views from the visual simulation using IRIS Performer**

4.3 Automatic Extraction of Texture Maps

One problem in modeling the virtual Wean Hall corridor is that it is difficult to extract realistic texture maps. Even though recovering the correct 3D geometry is the key, 3D geometry alone is not enough for virtual reality applications.

Fortunately, we can mosaic multiple images into a bigger image which can be used as a texture map of a large planar patch in our indoor scene model. Because we have recovered the 3D geometry, the back-projection is straightforward. To register multiple images of a planar surface, an 8-parameter projective motion model is suitable. From several images shown in Figure 23, we can obtain a large mosaiced image as in Figure 24 using image mosaicing techniques [115][118].

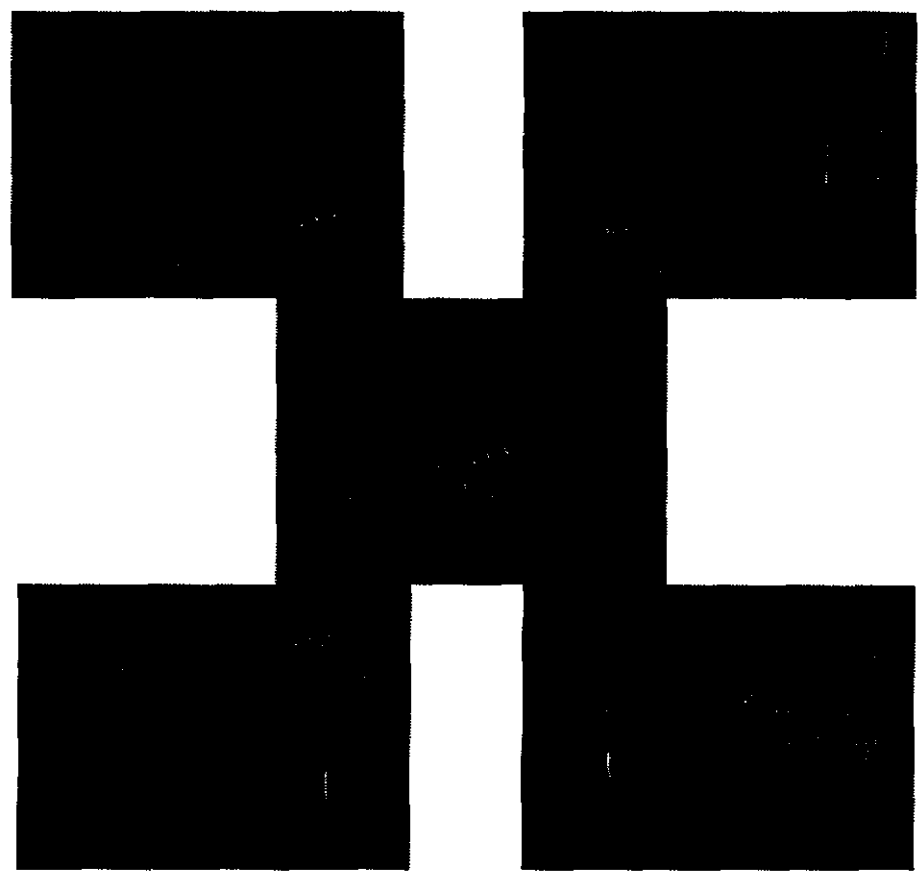


Figure 23 **A sequence of images**

Figure 24 **A mosaiced image**

4.4 Discussion

Our scene modeling approach is an orthodox way of modeling a 3D environment. First we extract the 3D geometry, then we extract textures. The advantage of our approach is that view synthesis from 3D geometry is simple and straightforward. Recently, much work has been done on image-based rendering for modeling virtual environments without reconstructing real 3D environments. The advantage of image-based rendering is that it simplifies texture mapping by going from picture to picture. Because no 3D environment is reconstructed, the viewing range in image-based rendering systems is usually restricted. For example, in QuickTime VR[16], a cylindrical panorama is used to represent the scene at a viewpoint by deliberately avoiding 3D motion parallax. To represent a large 3D scene, one has to manually specify a number of nodes, or a discretization of 3D space, from which panoramas have to be taken. To move from one panorama node to another, jumps or motion discontinuities are inevitable. One way to improve the jumps is to use view interpolation based on the dominant projective motion between two panoramas [111].

Chapter 5

What to Integrate: Free-Form Objects

Unlike with polyhedral objects whose planar patches can be well segmented and tracked over a sequence of images, the correspondence between different views of a free-form object can not be easily established. Due to discrete sampling in viewer space, sampled data points of free-form objects do not correspond to each other in different range images. It is difficult to segment noisy range images of free-form objects into a number of patches of high-order polynomials [41]. In this chapter, we introduce the concept of global resampling of free-form objects. For each range image, we first build a discrete mesh that approximates the object's surface with nearly uniform mesh nodes. At each mesh we encode its local curvature. Mesh matching can then be made based on the curvature distribution over the mesh. Using the object-centered spherical mesh coordinate system, global resampling helps to establish one-to-one correspondence among mesh nodes from multiple views.

5.1 Why is Free-form Difficult?

Two major problems regarding the correspondence between different views of a free-form object exist due to discrete and noisy sampling in practice. The problems are:

(1) the lack of physical correspondence among sampling points in different views; and

(2) the lack of global sampling for all sampling points.

To appreciate the difficulties in free-form object modeling, we examine in Figure 25 a simple example of a free-form curve observed from multiple views. For example, the measurements of view 1, m_{11},..., m_{16}, do not necessarily correspond to those in other views. In addition, because of the non-existence of global sampling, information regarding the ordering among all sampling points is absent. Thus, in order to recover an accurate model for the curve, we have to somehow figure out the proper linkage between the points in different

views (e.g., m_{13}, m_{21}, m_{22}, m_{23}, m_{14},...). The situation in the three dimensional world is even more complicated because transforming all range data from multiple views into a single coordinate system results in a "cloud" of noisy 3D points. It is difficult to make a surface model from this cloud of noisy 3D points.

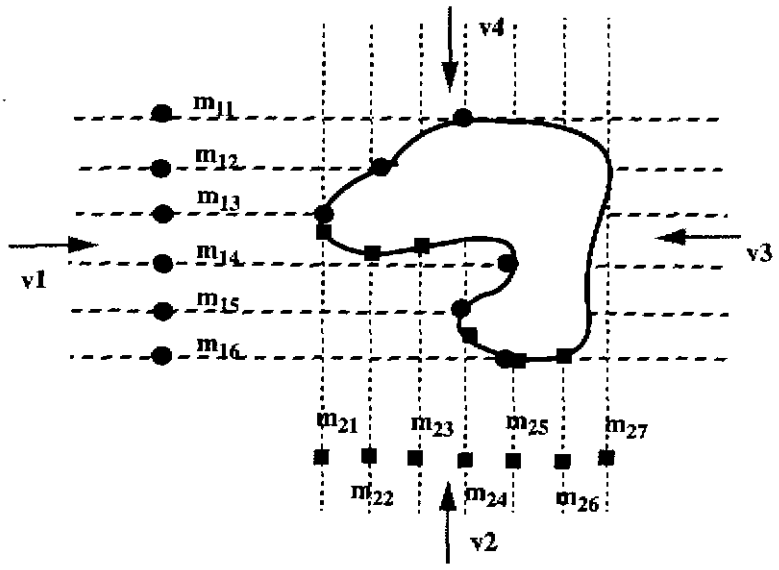


Figure 25 **The problem of resampling (black dots and grey squares are measurements in the sensor coordinates obtained from range sensor readings in different views).**

Since curvature is invariant under Euclidean group transformations, it is desirable to obtain correspondence among distinguished high curvature points in different views. In practice, however, only a finite number of sampling points of a surface from any particular viewing direction is available. These points are measured in the sensor coordinate system. The normals (or the tangent planes), and the curvatures, are not explicitly or directly known. Although surface normals can be approximated by fitting local planar surface patches of points, the process of computing curvature at each point using discrete range data is known to be highly noise sensitive and may be numerically unstable.

What we want is to sample the object in an object-centered coordinate system, instead of a viewer-centered coordinate system. All range images are viewer-based. This is why we need to resample all images so that we can find "features" which appear on an object. For free-form objects, it is essential to resample the measured discrete data points. For polyhedral objects, it is desirable to do so because big planar patches can be extracted reliably.

In this chapter, we propose a global resampling scheme, which consists of spherical surface representation, deformable mesh generation and one-to-one mesh node correspondence. Our method resamples the object by combining top-down knowledge of topological information (spherical representation) and bottom-up geometrical information (range data and local curvature).

5.2 Representation of a Free-form Surface

5.2.1 Discrete Representation of a 2D Curve

Before resampling a 3D free-form surface, it is helpful to understand how one can represent and resample a 2D curve. A standard way of representing a simple polygon is to describe its boundary by a circular list of vertices with known coordinates. To represent a simple closed 2D curve which is not self-intersecting, one can parameterize the curve by a number of points. For example, one can approximate the curve by line segments of equal lengths.

The curvature of a discrete curve at each node of the polygonal approximation can be approximated by the turning angles between adjacent line segments. The turning angle can be viewed as a discrete average measure of local curvature at the vertex. Like curvature, the turning angle is independent of rigid transformation and scaling. Figure 26 shows an example of turning angles defined on a polygon. Another way of visualizing turning angles is to map them to a unit circle. An important property of turning angles is that they sum up to 2π. It is true for either a concave or a convex polygon because topologically a non-self-intersecting polygon is equivalent to a circle. To avoid possible unstable representation under certain kinds of noise, dense equal length line segments have been adopted in [104]. Figure 27 shows an example of representing a 2D curve with equal-length segments. The degree of similarity between two curves can be measured by comparing the distributions of curvature measurements at the vertices of the approximating polygons. For noise-free polygons with few vertices, Arkin et al. [2] showed a very efficient algorithm which directly compares turning angles on vertices.

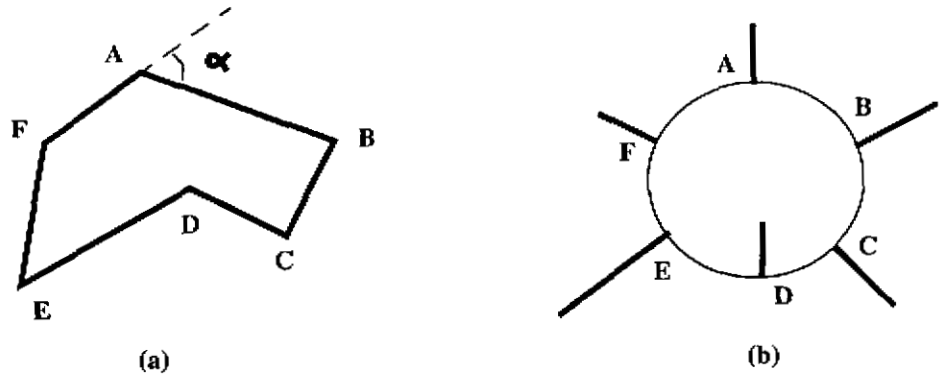


Figure 26 **(a) Definition of a turning angle at a polygon vertex. The turning angle at D is negative because it is concave. (b) Unit circle representation of a polygon using its turning angles**

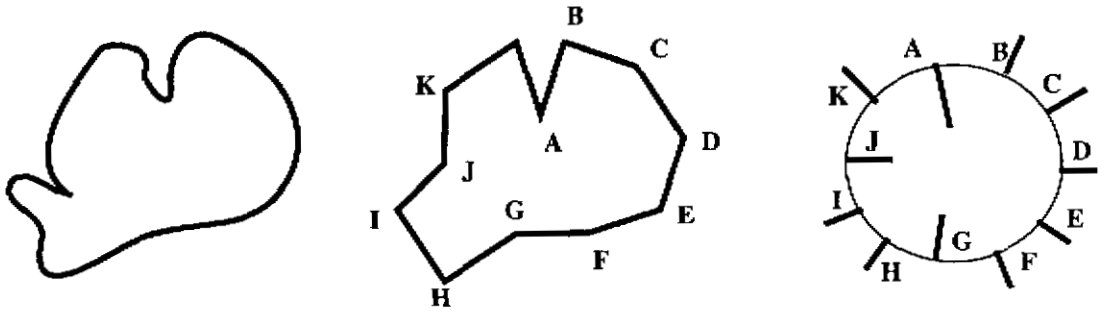


Figure 27 **Representation of 2D curve using equal-length line segments**

5.2.2 Spherical Representation of a 3D Surface

Essentially equal-segment approximation of a 2D curve can be regarded as a global resampling of the curve based on its arc-length. Therefore, 3D resampling can be considered as a straightforward extension of 2D resampling using uniform surface area resampling. For instance, particle-based surface resampling schemes [116][132] have been proposed to resample arbitrary surfaces.

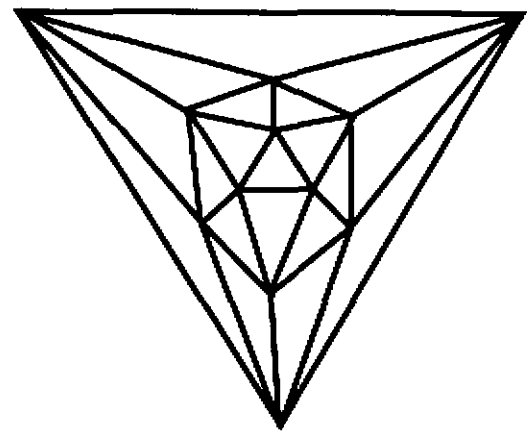


Figure 28 **A graph of an Icosahedron**

A natural discrete representation of a surface is a graph of nodes where each node is connected to its nearest neighbors by an arc of the graph. Figure 28 shows a graph of an icosahedron [51]. We use a special mesh, each node of which has exactly three neighbors. Such a mesh can be constructed as the dual of a triangulation of the surface [30]. To tessellate a unit sphere, we use the standard semi-regular triangulation of a unit sphere which is constructed by subdividing each triangular face of a 20-face icosahedron into N^2 smaller triangles. All vertices of the triangulation touch six triangle cells, with the exception of the twelve vertices at which five cells meet. Figure 29 shows how each triangle can be subdivided into N^2 smaller triangles. The final tessellation is built by taking the dual of the $20N^2$-face triangulation, yielding a tessellation with $10N^2+2$ meshes.

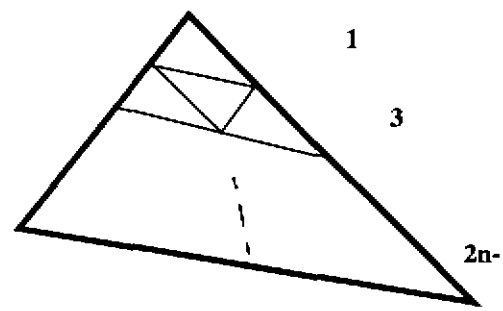


Figure 29 **Frequency n subdivision of a triangle. $1+3+...+(2n-1) = n^2$.**

In order to obtain a mesh representation for an arbitrary surface, we deform a tessellated sphere until it is as close as possible to the object surface. This algorithm drives the spherical mesh to converge to the correct object shape by combining forces of two kinds: a smoothness force which keeps the mesh from falling apart, and a data force which drives the mesh as close to the range data as possible. Our algorithm originates from the idea of a 2D snake [66] and 3D deformable surfaces [76][122]. An advantage of the deformed surface is that it can accurately represent both concave and convex surfaces. The deformation process is robust even in the presence of data noise and moderate change of parameters such as initial sphere center and radius [108]. We will explain in the next section how the deformable surface is obtained.

5.2.3 Deformable Mesh Surface

Many deformable models, such as irregular meshes [127], finite element models [24][76] and balloon models [19] have been proposed for 3D shape reconstruction. The reader is referred to [78] for a detailed reference of deformable surfaces in medical imaging application. Our basic surface representation is a discrete connected mesh that is homeomorphic to a sphere. The free-form objects are restricted to a topology of genus zero (i.e., no holes). A distinctive feature of our surface representation is its global structure, i.e., the connectivity of the mesh is such that each node has exactly three neighbors. The total number of mesh nodes depends on the resolution of the mesh.

Given a set of data points from a range image, a mesh representation is constructed by first placing a spherical or ellipsoidal mesh at approximately the center of the object. Then the shape of the mesh is deformed iteratively in response to "forces" generated by the data points, the image features, and their internal smoothness constraints. The deformation can be formulated as an optimization problem which minimizes the distance between the mesh nodes and range data points. The motion of each vertex is modeled as a second-order dynamics equation,

$$m\frac{d^2P_i}{dt^2} + k\frac{dP_i}{dt} = F_{data} + F_{smooth} \qquad \text{(EQ 5.1)}$$

where m is the mass unit of a node and k is the damping coefficient, and F_{data} is the force which attracts the mesh node to its closest data point, F_{smooth} is the force which keeps the tetrahedron formed by the mesh node and its three neighbors regular (i.e. the regularity con-

straint). The above equation is integrated over time using finite differences. Starting from an initial shape, e.g., a semi-tessellated sphere, we can update each mesh node of unit mass using Euler's method

$$P_i^{(t)} = P_i^{(t-1)} + (1-k)\left(P_i^{(t-1)} - P_i^{(t-2)}\right) + F_{data} + F_{smooth}. \qquad \text{(EQ 5.2)}$$

Notice that the data force and smoothness force have to be updated at each iteration. To speed up the iterative process, some additional feature forces can be introduced at the beginning. Feature forces can afford to have global influence because we use only a few features such as long edges on the surface. On the other hand, data forces are effective only within a small neighborhood of range data points.

An example showing the reconstruction of a surface model from a partial view is given in Figure 30. The mesh is always a closed surface. Notice that we have shown hexagon meshes in the display in Figure 30. However, since only part of the object's surface is visible from any given viewpoint, some of the mesh nodes do not correspond to visible surface data. These nodes, corresponding to the occluded part of the object, are flagged as interpolated nodes so that they can be treated differently when used. Because of the smoothness constrain, there is a bulging part for the occluded region such as in Figure 30d. Figure 31 shows the same deformable surface with and without the interpolated portion.

The key idea of our spherical representation of the surface is to produce meshes in which the density of nodes on the object's surface is nearly uniform[1]. Although it is impossible to achieve a perfect uniform distribution, a simple local regularity constraint can enforce a very high degree of uniformity across the mesh. With a semi-regularly tessellated sphere, the local regularity constraint is implemented in the deformable surface algorithm such that each mesh is similar to the others in area [49].

This local regularity constraint is a generalization to three dimensions of the regularity condition on two dimensional discrete curves where all line segments are of equal lengths. The difference between 2D and 3D cases is that it is always possible to create a uniform discrete curve in 2-D, while only approximately uniform discrete surfaces can be generated in 3-D. Our experiments show that the variation of mesh nodes on the surface is on the order of 2%.

---

1. Koenderink warned that one has to be very careful about any method that uses surface area of a polyhedral model (p.597 of [68]). Surface area depends on the way by which triangulations are done. In our previous work, we have shown how areas of different shapes are adjusted before comparison, in particular for partial views [49].

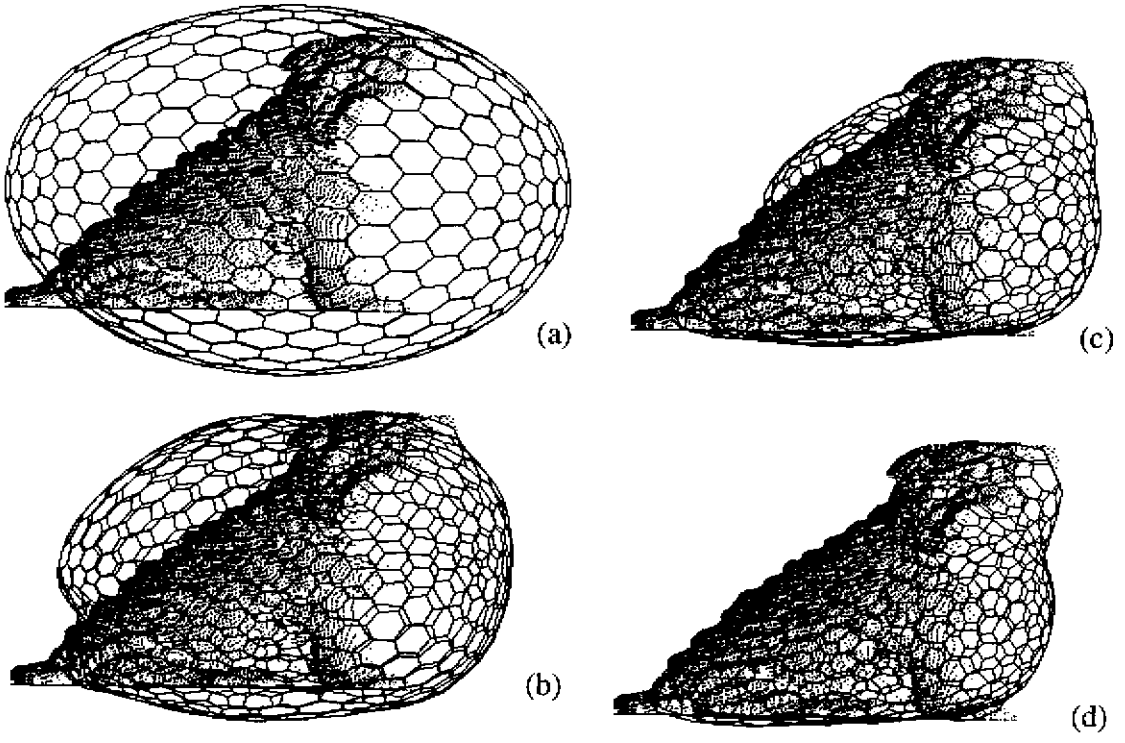


Figure 30 **Deformable surface reconstruction at different iteration steps (dots represent range data, solid lines are mesh models): (a) n=0 (start of deformation); (b) n=20; (c) n=50; and (d) n=100 (end of deformation)**

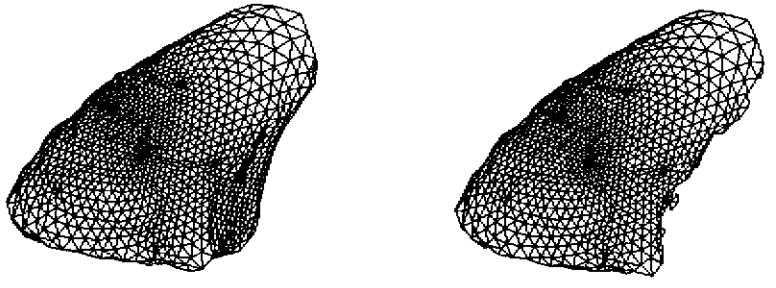


Figure 31 **Deformable surface: (a) with interpolated part; and (b) without interpolated part**

5.3 From Shape to Curvature Distribution: Forward Mapping

After we obtain a nearly uniform surface mesh representation, we need to define a measure of curvature that can be computed from the surface representation. Conventional ways of estimating surface curvature, either by locally fitting a surface or by estimating first and second derivatives [9], are very sensitive to noise. This sensitivity is mainly due to discrete sampling and, possibly, to noisy data.

5.3.1 Measures of Discrete Curvature

It is well known that curvature vanishes everywhere on a polyhedral object except at corners or on edges. In fact, there are only mean curvatures on edges (dihedral angles) and only Gaussian curvature (spherical deficit) at corners. Figure 32 shows how a dihedral angle and a spherical deficit are defined. Detailed discussion on dihedral angle and spherical deficit can be found in differential geometry textbooks such as [68].

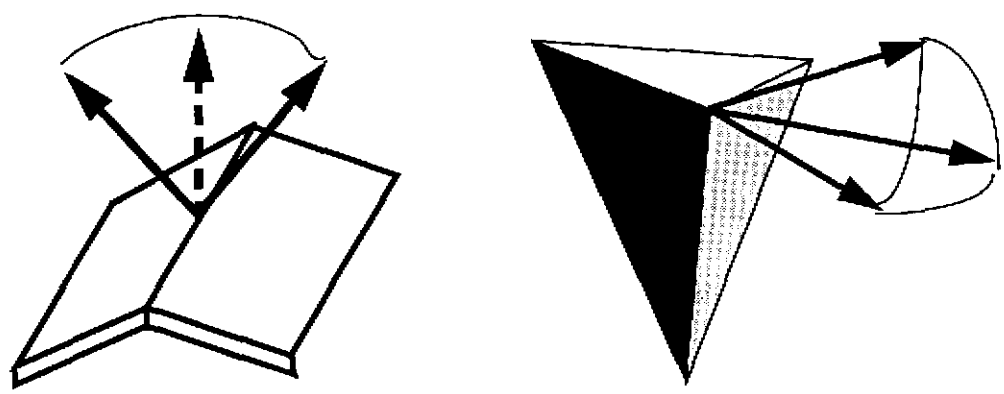


Figure 32 **Definition of discrete curvatures**

A robust measure of curvature computed at every node P from the relative positions of its three neighbors P_1, P_2 and P_3 is introduced in [29]. This measure of curvature is called the *simplex angle* and is shown in Figure 33. This measurement is robust because the deformable surface process serves as a smoothing operation over the possibly noisy original data. As a result, all the nodes are at relatively stable positions after the deformation process.

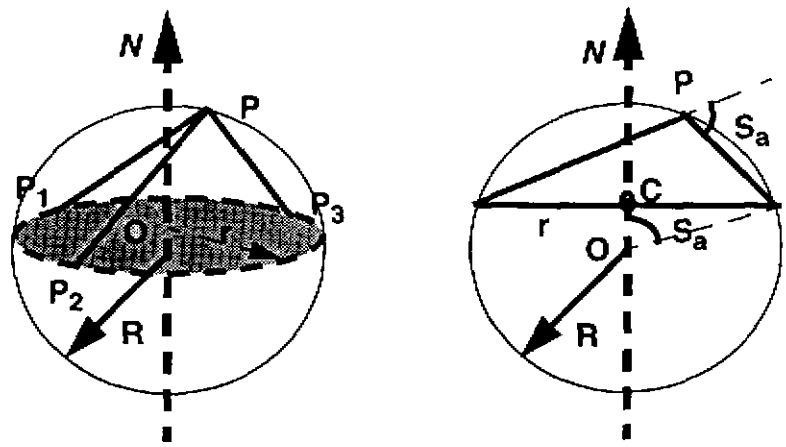


Figure 33 **Definition of simplex angle**

The simplex angle varies between -π and π. It is negative if the surface is locally concave, and positive if it is convex[1]. Given a configuration of four points, the angle is invariant to rotation, translation, and scaling because it depends only on the relative positions of the points, not on their absolute positions. The simplex angle is defined by

$$\cos(S_a) = \frac{\|OC\|}{R} sign(\overrightarrow{OC} \cdot \hat{N}) \qquad \text{(EQ 5.3)}$$

and

$$\sin(S_a) = \frac{r}{R} sign\left(\overrightarrow{PP_1} \cdot \hat{N}\right) \qquad \text{(EQ 5.4)}$$

where

$$S_a \in [-\pi, \pi] .$$

Another robust measure of local curvature from a polyhedral surface is to use the tensor of curvature [120], possibly after proper smoothing [119].

5.3.2 3D Intrinsic Representation

The spherical representation can approximate not only free-form objects, but is also useful in approximating polyhedral objects. Figure 34 shows an example of a spherical polyhedral

1. It is interesting to note that the simplex angle is related to the mean curvature at the mesh node [30].

approximation of an octahedron with one concave face. Because of the regularity constraint, corners and edges are not perfectly represented. All plane surfaces, however, are well approximated even though local regularity is enforced on all meshes. Similar to the EGI [57][53] representation, we exclude pathological cases such as very long and narrow cylindrical shapes from our study. In contrast to EGI, however, our representation can be used for both convex and concave objects.

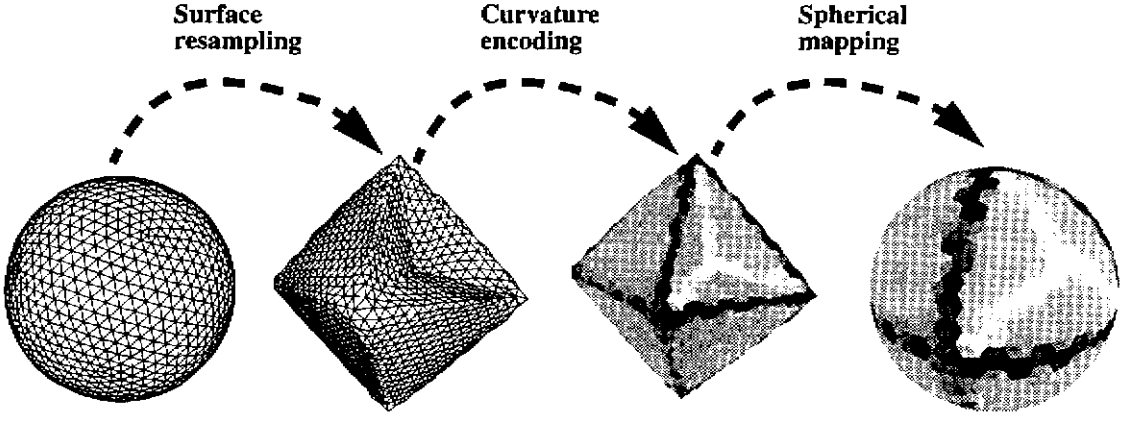


Figure 34 **(a) A spherical tessellation; (b) Deformable surface of an octahedron with a concave dent; (c) Local curvature on each mesh node; and (d) Curvature distribution on spherical representation (The curvature on (c) and (d) is negative if it is light, positive if dark, zero if grey)**

5.4 From Curvature Distribution to Shape: Inverse Mapping

Now we know how to map the object shape to its intrinsic representation using a curvature distribution on a special spherical coordinate system. But does inverse mapping exist? In other words, given a known intrinsic representation, can we reconstruct the right shape?

5.4.1 Shape Reconstruction

We formulate this reconstruction as an optimization problem which minimizes the curvature difference between the known and reconstructed curvature distributions. The initial shape can be, for example, a sphere where constant curvature exists at every mesh node. This minimization problem is, however, complicated because of the nonlinear and coupled nature of the local curvature computation. To solve this minimization problem, we devise an iterative method, similar to what has been used in deformable surface extraction. The motion of each vertex is modeled as a second-order dynamics equation,

$$m\frac{d^2P_i}{dt^2} + k\frac{dP_i}{dt} = F_{int} \qquad \text{(EQ 5.5)}$$

where m is the mass unit of a node and k is the damping factor. F_{int} is the force which deforms the surface to its assigned curvature distribution without breaking the mesh connectivity. Given its curvature distribution, a toy sharpei is reconstructed from a sphere.

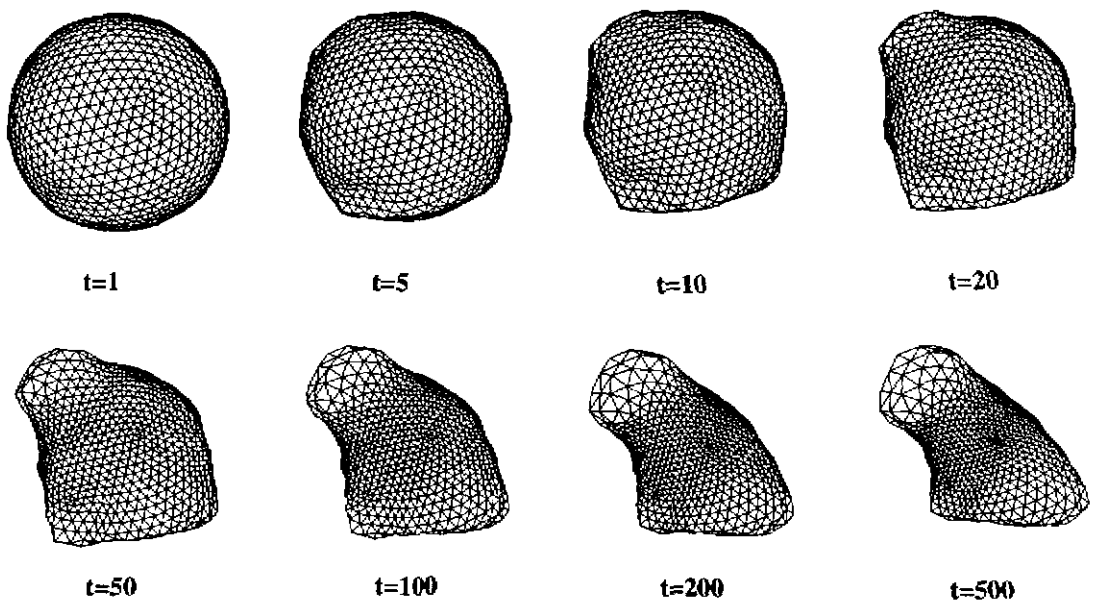


Figure 35 **A sequence of shapes at different steps of deformation from a sphere to a sharpei**

5.4.2 Regularization for the Underconstrained Minimization

The above minimization is, unfortunately, underconstrained. There are $3n$ unknowns (three-dimensional coordinates at each mesh node of the unknown spherical mesh), but only n equations are defined by the local curvature at each mesh node. The object shape can not be determined without additional constraints. It is well-known that Gaussian curvature and mean curvature are not sufficient to determine the shape.

5.4.3 Delingette's Constraint: Metric Parameters

One way to add more constraints is to record the relative positioning between each node P and its three neighbors P_1, P_2, and P_3. Delingette [30] defined a so-called metric parameter set $\{\varepsilon_1, \varepsilon_2, \varepsilon_3\}$, such that

$$Q = \varepsilon_1 P_1 + \varepsilon_2 P_2 + \varepsilon_3 P_3, \qquad \text{(EQ 5.6)}$$

$$\varepsilon_1 + \varepsilon_2 + \varepsilon_3 = 1, \qquad \text{(EQ 5.7)}$$

where Q is the projection of P on the plane $P_1P_2P_3$ as shown in Figure 36.

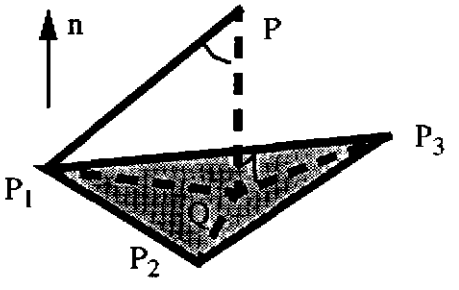


Figure 36 **Metric parameters relating a mesh node with its three neighbors**

If the shape is known, i.e., if each mesh node P and its three neighbors P_1, P_2, and P_3 are known, the surface normal of the plane $P_1P_2P_3$ can be computed as

$$n = \frac{\overrightarrow{P_1P_2} \times \overrightarrow{P_1P_3}}{\left\| \overrightarrow{P_1P_2} \times \overrightarrow{P_1P_3} \right\|}. \qquad \text{(EQ 5.8)}$$

From

$$\overrightarrow{QP} = \left\| \overrightarrow{QP} \right\| \vec{n} = \left(\overrightarrow{P_1P} \bullet \vec{n} \right) \vec{n} \text{ and } \overrightarrow{P_1Q} = \overrightarrow{P_1P} - \overrightarrow{QP} = \varepsilon_2 \overrightarrow{P_1P_2} + \varepsilon_3 \overrightarrow{P_1P_3},$$

we can easily compute the metric parameters from the following 2x2 linear equation

$$\begin{bmatrix} \overrightarrow{P_1P_2} \bullet \overrightarrow{P_1Q} \\ \overrightarrow{P_1P_3} \bullet \overrightarrow{P_1Q} \end{bmatrix} = \begin{bmatrix} \left\| \overrightarrow{P_1P_2} \right\|^2 & \overrightarrow{P_1P_2} \bullet \overrightarrow{P_1P_3} \\ \overrightarrow{P_1P_2} \bullet \overrightarrow{P_1P_3} & \left\| \overrightarrow{P_1P_3} \right\|^2 \end{bmatrix} \times \begin{bmatrix} \varepsilon_2 \\ \varepsilon_3 \end{bmatrix}. \qquad \text{(EQ 5.9)}$$

The above equation provides us with two equations at each mesh node. Thus we have *3n* sparse nonlinear equations for *3n* unknowns if we augment our intrinsic representation from $\{S_{a_i}\}$ to $\{S_{a_i}, \varepsilon_{2i}, \varepsilon_{3i}\}$, $i = 0 \ldots n$.

5.4.4 Regularity Constraint: a Means of Regularization

The above metric parameter augmentation to our intrinsic shape representation may not be necessary. In fact, we have introduced a regularity constraint in shape reconstruction which forces each mesh node to project onto the center of its three neighbors. This regularity constraint implicitly gives a complete intrinsic description $\{S_{a_i}, \frac{1}{3}, \frac{1}{3}\}$, $i = 0 \ldots n$. It implies that our intrinsic representation is sufficient to guarantee the inverse mapping. This regularity constraint is, in spirit, similar to imposing a regularization term for ill-posed problems [121]. Given known local curvature, the expected location of a mesh node can be regularized through its three neighbors. As shown in Figure 37, we can then define F_{int} as the scaled distance between the current mesh node P and its regularized position P^*,

$$F_{int} = \alpha \cdot \overrightarrow{PP^*}, \alpha = [0, 0.5]. \qquad \text{(EQ 5.10)}$$

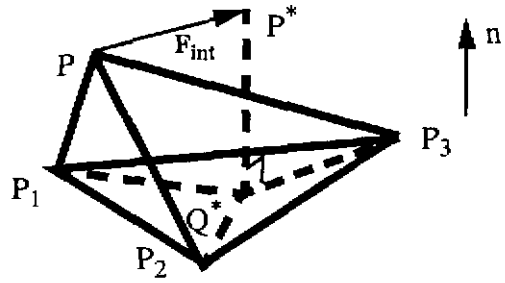


Figure 37 **Regularity and internal force. $Q^* = (P_1 + P_2 + P_3)/3$**

How does this regularity constraint ensure that we reconstruct the correct original shape? The reason is that this same regularity constraint has been enforced in the model extraction process when we construct an object model from range data [29][49]. At the end of model extraction, the metric parameters converge to the expected value. Figure 38 shows the distribution of metric parameters of the sharpei model and their histograms. It clearly shows that the metric parameters are well regularized around their nominal values of $\frac{1}{3}$. It is also equiv-

alent to the constraint that each triangular patch has the same area. Obviously, if we keep the intrinsic representation but pick an arbitrary set of metric parameters, we will reconstruct another shape which may be close to, but different from, the original shape.

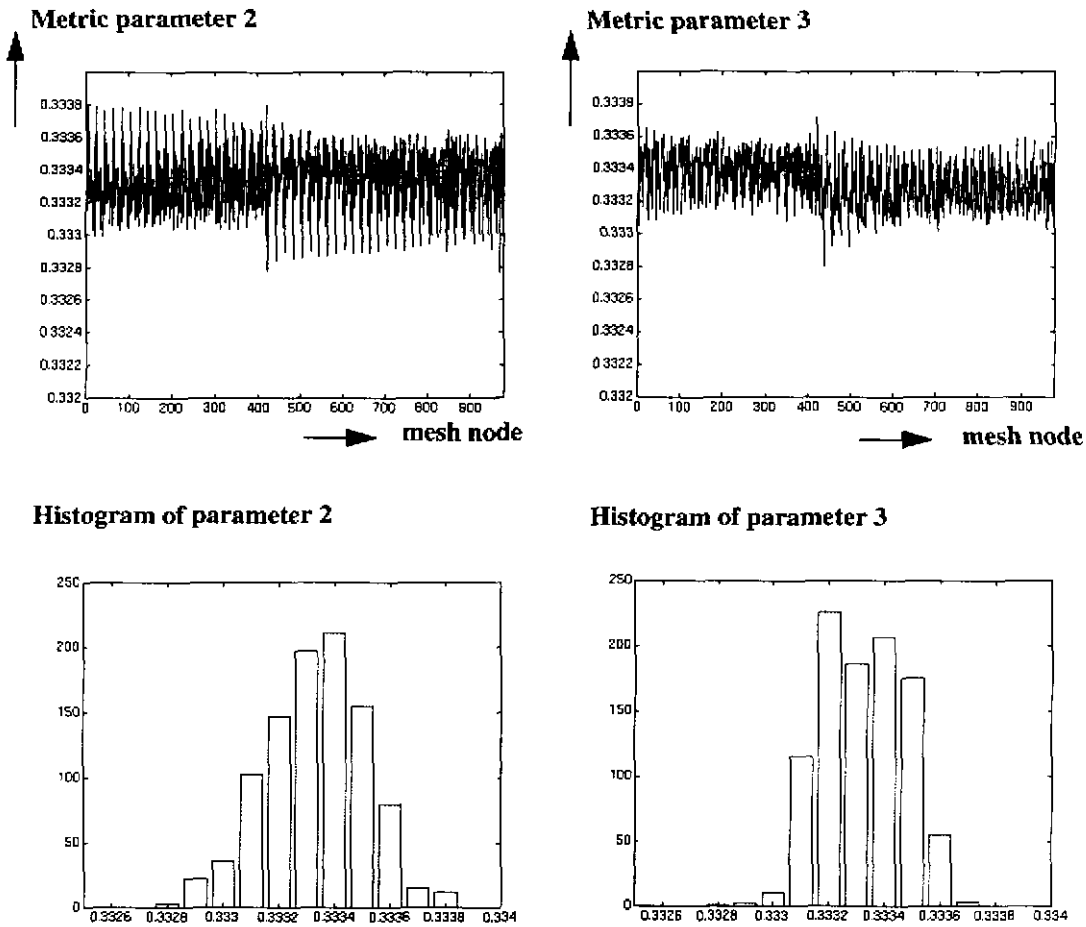


Figure 38 **Metric parameter distributions of a sharpei with 980 mesh nodes**

5.4.5 Examples

The regularity constraint does not eliminate the local minimum problem associated with our iterative process. Therefore, the initial shape plays an important role in whether a correct shape is converged. In all our experiments, we start our reconstruction from a semi-tessellated sphere. We show three examples of shape reconstruction: a polyhedral object in Figure 39, and two free-form objects in Figure 40 and Figure 41.

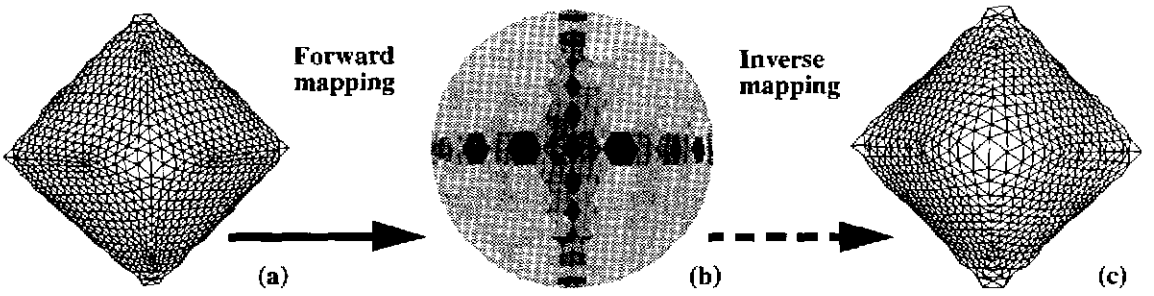


Figure 39 An example of polyhedral object shape reconstruction (the solid arrow shows forward mapping from shape to curvature: the dashed arrow shows inverse mapping from curvature to shape): (a) Deformable surface of an octahedron; (b) Intrinsic representation or its curvature distribution (The curvature is negative if it is light, positive if dark, zero if grey); and (c) The reconstructed shape from the curvature distribution (b)

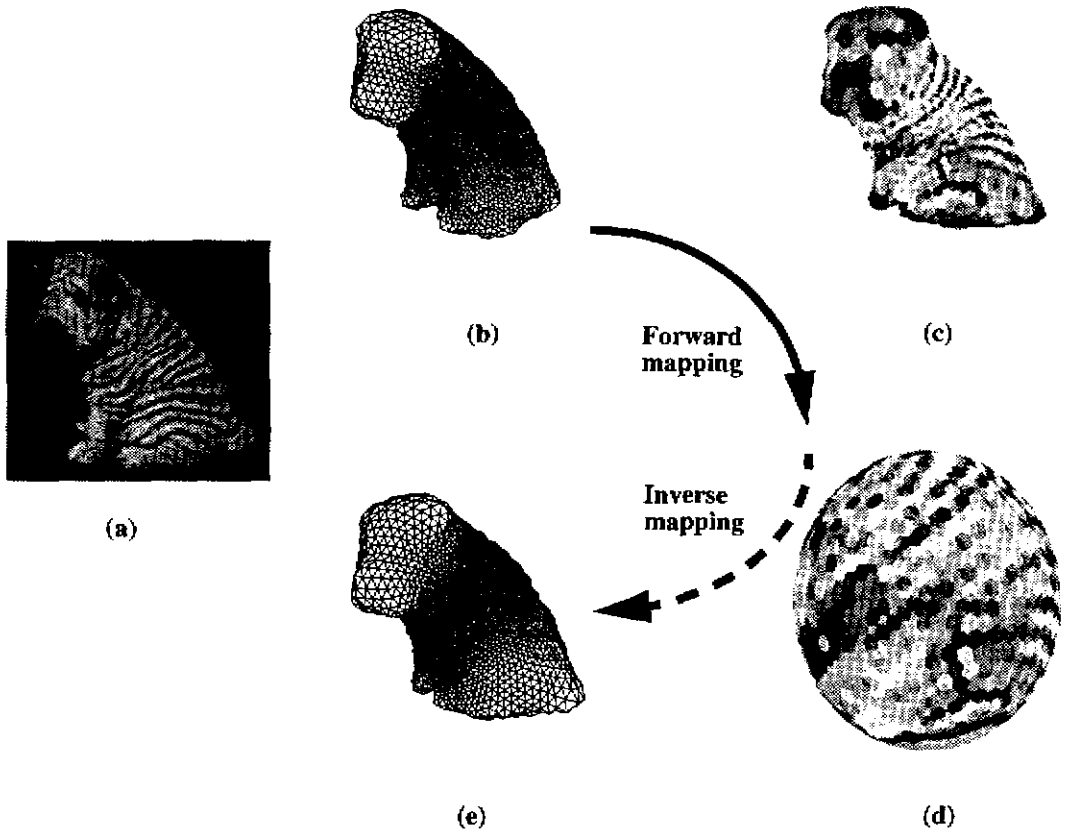


Figure 40 An example of free-form object shape reconstruction (the solid arrow shows forward mapping from shape to curvature: the dashed arrow shows inverse mapping from curvature to shape): (a) An image of a sharpei; (b) A sharpei model constructed from range data; (c) Local curvature distribution at each mesh node of a sharpei; (d) Intrinsic representation of a sharpei; (e) Reconstructed sharpei model from its intrinsic representation

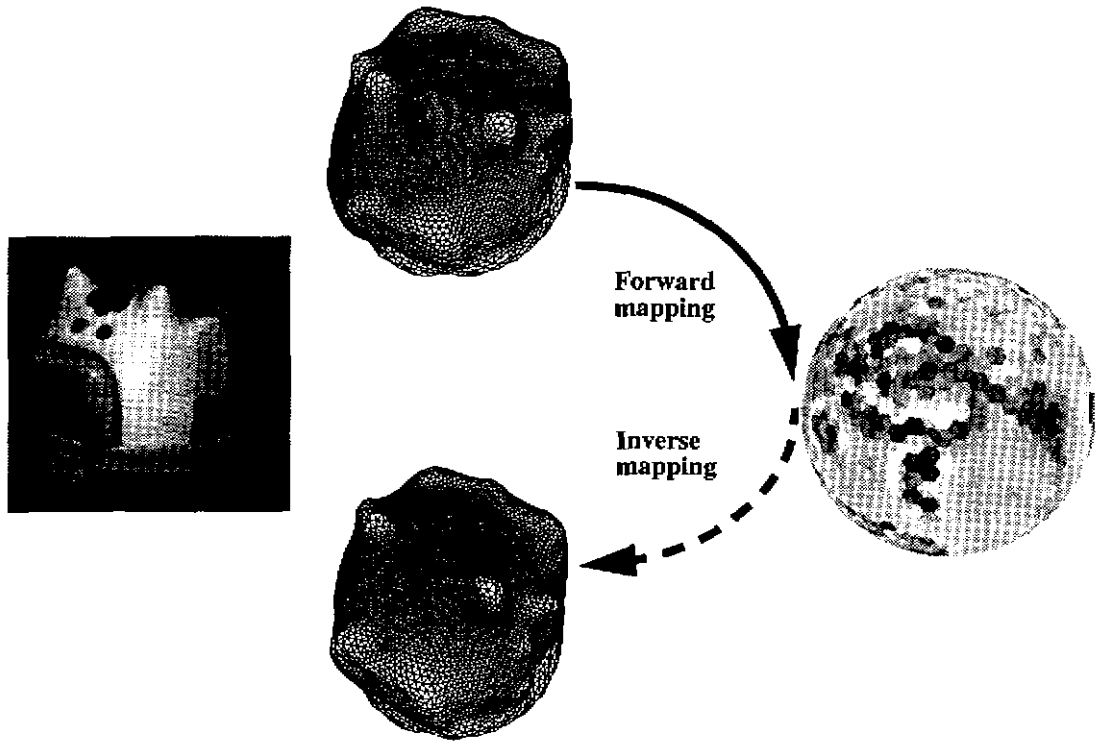


Figure 41 **Another example of inverse mapping of a free-form object representation.**

All examples show very good reconstruction. The reconstructed shapes retain a curvature distribution very similar to that of the original ones. The relative error in the simplex angle between the original and the reconstructed models is less than 1% for each example. However, we do observe an apparent discrepancy between the reconstructed and the original shapes mainly because of the discretization effect. Because the shape reconstruction is only up to an unknown rigid transformation (rotation and translation) and an unknown scale factor, constant area or constant volume [71] should be enforced during the reconstruction process.

5.5 Mesh Matching with a 3D Shape Similarity Metric

In Sections 5.2 and 5.3 we have explained how we obtain mesh representation and curvature distribution of a 3D surface. In order to match two different views, we need to find the correspondence between the two sets of mesh nodes. Such a correspondence may not exist,

however, because the distributions of nodes on the two surfaces may be completely different. Nevertheless, the regularity constraint enforces a nearly uniform mesh node distribution on the surface. When this constraint is enforced, meshes representing the same surface viewed from different viewpoints have the property that their nodes do correspond. This constraint can be evaluated locally at each node which has exactly three neighbors. Given meshes from two different views, matching proceeds by comparing the simplex angles at the nodes of the two meshes.

Different attributes may be associated with each node of the mesh. The most prominent attribute for the purpose of model building is curvature. For example, an approximation of mean curvature, called the simplex angle [29], has been proposed for matching two different views of an object. A simplex angle is computed at each node using its position and the positions of its three neighbors. Other attributes such as colors [50] can also be used.

Let S_A and S_B be the mesh representations of shape A and shape B, and $k_R(S_A)$ and $k_R(S_B)$ be the curvature distribution functions under a spherical rotation R. Specifically, $k(S_A)$ represents all simplex angles on all mesh nodes of the nearly uniform shape A. We then formally define the distance function between two 3D surfaces A and B as the L_p distance between their local curvature functions $k_I(S_A)$ and $k_R(S_B)$, minimized with respect to the rotation matrix R over the sphere. The function $k_I(S_A)$ denotes the curvature distribution of S_A under no rotation where I is the identity matrix. Other alternative distance measures include Hausdorff distance [56] and geometrical distance [114].

5.5.1 A Distance Function and a Shape Metric

We define the L_p distance $d_p(S_A, S_B, R)$ between A and B at a certain spherical rotation R as

$$d_p(S_A, S_B, R) = \left(\int \left| k_I(S_A) - k_R(S_B) \right|^p dS \right)^{\frac{1}{p}} \qquad \text{(EQ 5.11)}$$

which is the sum of all the curvature differences over the sphere. Then the distance function between A and B, $D_p(A, B)$, becomes

$$D_p(A, B) = min_R \; d_p(S_A, S_B, R) \qquad \text{(EQ 5.12)}$$

which is minimized d_p over all possible rotations R.

Property 1: $D_p(A, B)$ **is a metric for all** $p>0$.

Proof: See Appendix A.

5.5.2 A Global Search over Rotational Space

Property 1 shows that we can search over the spherical rotation space to compute the distance between two curvature distributions. A naive algorithm can then be easily constructed. Because this is an exhaustive search, the global minimum is always found, provided that the search step is small enough (that is, the number of searches is sufficiently large). This leads to the following property:

Property 2: **The distance between two shapes** A **and** B, $D_2(A, B)$, **can be computed in time** $O(m^3)$ **where** m **is the number of searches in each rotational space.**

5.5.3 A Global Search over Mesh Node Space

The time bound in Property 2 can be improved by employing a property of the semi-regularly tessellated sphere: the property that each node has exactly three neighbors. We have observed [29] that the only rotations for which $d(S_A, S_B)$ should be evaluated are the ones that correspond to a valid list of correspondences $\{P_i, P_j\}$ between the nodes P_i of S_A and the nodes P_j' of S_B. In Figure 42, node P_1 of S_A corresponds to P_1' of S_B, and the two neighbors of P1 (P_2 and P_3) are put in correspondence with two of three neighbors (P_2', P_3' and P_4') of P_1'. Figure 42 shows only 3 valid neighborhood matchings, since each node has exactly three neighbors and the connectivity among them is always preserved. Given correspondences of three nodes, a spherical rotation can be calculated. This rotation defines a unique assignment for the other nodes. In other words, there is a unique correspondence between a node P_j' of S_B and a node P_i of S_A, given the initial correspondences between $\{P_1, P_2, P_3\}$ and $\{P_1', P_2', P_3'\}$. Moreover, the number of such correspondences is $3n$ where n is the number of nodes of the spherical tessellation [29]. Figure 42 shows that there are three possible matchings at each node. Equivalently, there are $3n$ distinct valid rotations of the unit sphere. This analysis leads us to an efficient algorithm for comparing two shapes.

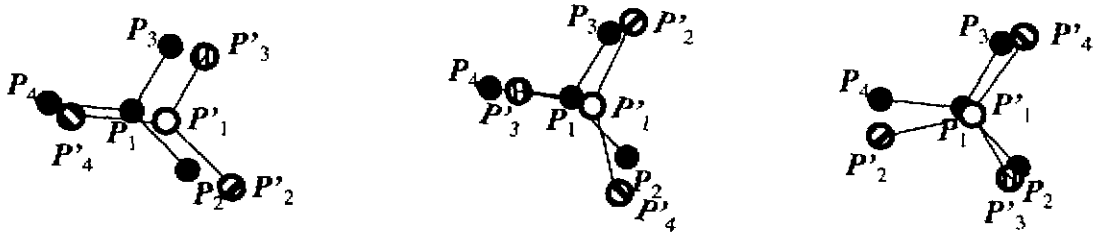


Figure 42 **Matching of neighbors from (P_2, P_3, P_4) to: (a) (P_2', P_3', P_4'); (b) (P_3', P_4', P_2'); (c) (P_4', P_2', P_3') when P_1 of shape *SA* is matched to P_1' of shape *SB***

Property 3: The distance between two shapes *A* and *B*, $D_2(A, B)$, can be computed in time $O(n^2)$ where n is the number of nodes, with preprocessing storage $O(n^2)$.

Proof:

Because the total match for each node is $3n$, we can find the global minimum in $3n^2$. To speed up the search for correspondence, we can make the lookup table for each node on S_A matching each node on S_B. Since each match gives 3n correspondences, this lookup table requires $3n^2$ for storage.

5.6 Integration of Multiple Views of a Free-form Object

5.6.1 Matching from Curvature Space to Mesh Node Space

If $\{P_i\}$ and $\{Q_j\}$ are the nodes of the two meshes defined in the sensor coordinates, the matching problem can be defined as finding the set of correspondences $C = \{(P_i, Q_j)\}$ such that the value of the curvature at any node P_i is as close as possible to the value of the corresponding node Q_j in the other mesh. Once the best set of correspondences C is obtained, we compute an estimate of the transformation (R, T) between the two views by minimizing the distance:

$$\sum_{C} \| P_i - (RQ_j + T) \|^2. \qquad \text{(EQ 5.13)}$$

The resulting transformation (R, T) is only an initial estimate because the semi-regular tessellation of the mesh prevents an exact correspondence between nodes P_i and Q_j. In object

modeling, however, the estimate is accurate enough to be used as a starting point for the ICP algorithms [10][133], for example. The estimate of *(R, T)* and the node correspondence set *C* are then used as inputs to the PCAMD algorithm to integrate multiple views.

5.6.2 One-to-One Correspondence

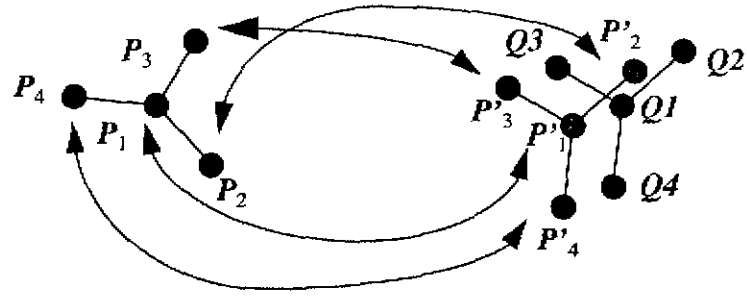
In order to apply PCAMD to take advantage of the redundancy of multiple views, we need a one-to-one mapping among different views. The matching procedure above, however, does not guarantee one-to-one correspondence between two sets of mesh nodes generated from two views because of discrete sampling and the requirement of local regularity.

Therefore, we establish one-to-one correspondence by resampling each deformable surface using the global spherical coordinate. This is similar to multiple-pass image resampling in image processing [35] and texture mapping [46]. Once a set of spherical mesh nodes (along with its curvatures) is obtained, it is possible to interpolate any point on the spherical coordinate from this set. For example, in Figure 43, although there exist many-to-one mappings between P and Q (both P_1 and P_2 are matched to Q_1), mapping between P' and P is one-to-one because P' results from the rotation of P. The new mesh node at P'_1 and its SAI value can be interpolated from its nearest point Q_1 on set Q and three neighbors of Q_1. Let $g(P')$ and $g(Q_1)$ be the values of the simplex angles at node P' and its nearest node Q_1, respectively; then we have the following local interpolation:

$$g(P') = \sum_{i=1}^{4} w_i Q_i \qquad \text{(EQ 5.14)}$$

where Q_2, Q_3, and Q_4 are three neighbors of Q_1, and w_i is the weight depending on the distance between P' and Q_i $(i=1,2,3,4)$. The coordinates of the mesh node at P' can be interpolated in the same way. Because of one-to-one mapping between any two views, one-to-one mapping among multiple views can be established. When the measurement matrix is ready, we can apply the PCAMD algorithm to compute a complete set of mesh nodes and transformations among different views. The object model is then obtained based on the known connectivity among all mesh nodes.

(a)



(b)

node number

1	2	3	4	- - -	K
P_1	P_2	P_3	P_4	- - -	P_K
P'_1	P'_2	P'_3	P'_4	- - -	P'_K
Q_1	Q_1	Q_3	Q_4	- - -	Q_K

Figure 43 : **One-to-one matching: (a) Valid correspondence between nodes; (b) Table of correspondences**

5.7 Experiments

Once we establish mesh node correspondence, we can apply the PCAMD algorithm to integrate multiple views. In this section, we present the results of applying our algorithm to synthetic data and real range image sequences of objects. We demonstrate the robustness of our approach using synthetic data, and present the reconstructed models from real range images.

5.7.1 Synthetic Data

Our synthetic data set consists of a set of 20 mesh nodes of an icosahedron whose connectivity is assumed to be known. The object size is approximately the same as a unit sphere. We study the effectiveness of our approach when the input data are corrupted by noise. Correspondence is assumed to be known. The minimization of the weighted squares distance between the reconstructed and the given measurement matrices leads to the recovery of mesh node coordinates and transformations.

To study the error sensitivity on reconstruction using our algorithm, four nonsingular views of the object are taken. Each measurement is corrupted by a Gaussian noise of zero-mean and variable standard deviation. Figure 44 shows that our algorithm converges in a few

steps. Two separate steps in Figure 44 are counted as one iteration in our algorithm. The cases with standard deviations σ of 0.0, 0.1, and 0.2 are studied.

If a point appears only once in the whole sequence, then its reconstruction depends on the amount of noise. When this point appears in more than one view, its reconstruction using our integral method is averaged over all views. Figure 45 gives the reconstructed errors of a point which appears 12 times in 16 views. When only two views are matched, the reconstructed point is deviated from its original position by 0.58 and 0.69 when standard deviation σ is 0.1 and 0.2, respectively. When 12 views are matched, the error decreases to 0.19 and 0.27.

When the observed points are corrupted by noise, the sequential method results in an erroneously recovered shape and transformation. Errors propagate as new views are introduced and vary with different matching orders. However, our integral approach gives an appreciably smaller reconstruction error by distributing the errors over all views, regardless of the order of matching. In Figure 46, we use a different starting view in a sequence of 12 images for different matching orders shown in horizontal axis.

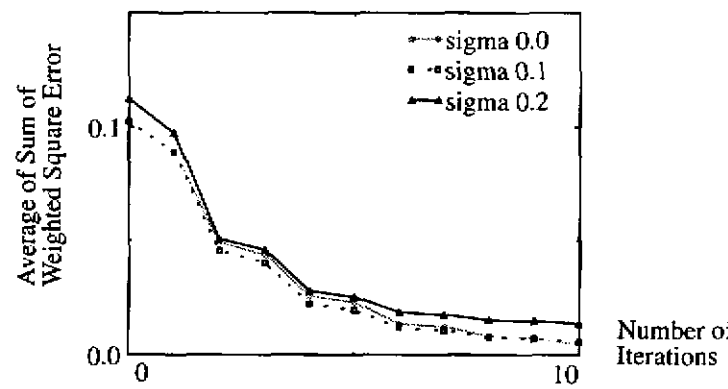


Figure 44 **Effect of noise on the convergence of PCAMD**

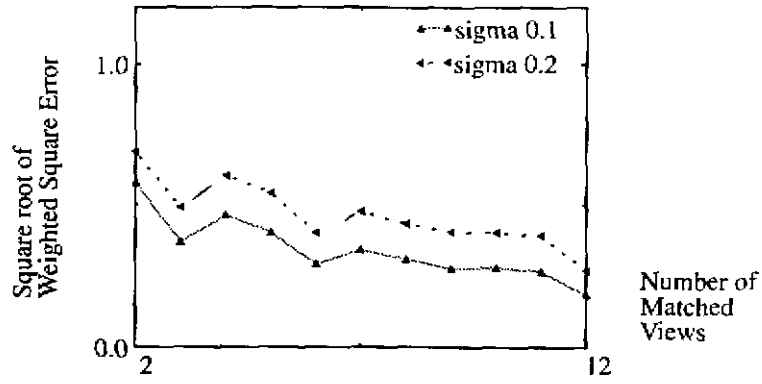


Figure 45 **Reconstructed error vs. the number of matched views for a point**

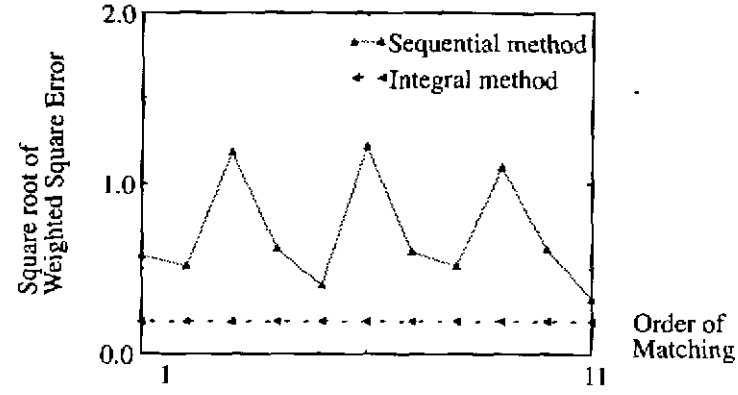


Figure 46 **Comparison between the sequential method and the integral method with different matching orders**

5.7.2 Real Range Image Sequence

Our integral approach has also been applied to real range images. All range images used in our experiments were taken using a light-stripe range finder which has a resolution of 0.2mm. The objects were placed on a rotary table about 1 meter in front of the range finder. To obtain the ground truth of the transformation, the rotation axis and the rotation center of the rotary table are calibrated using a known geometry calibration cube.

Figure 47 shows 9 views of a sequence of a peach. Figure 48 shows the result, of the use of two wireframe and two shaded views of a recovered model. The peach model does not depend, *a priori*, on motion estimation, nor on the ordering of matching. Figure 49 illus-

trates the improvement resulting from the application of PCAMD through cross-section segments of the reconstructed models and range data (merged from multiple views using the known transformation). It is clear that the model reconstructed with PCAMD shows better averaging, which results in a smaller margin of error.

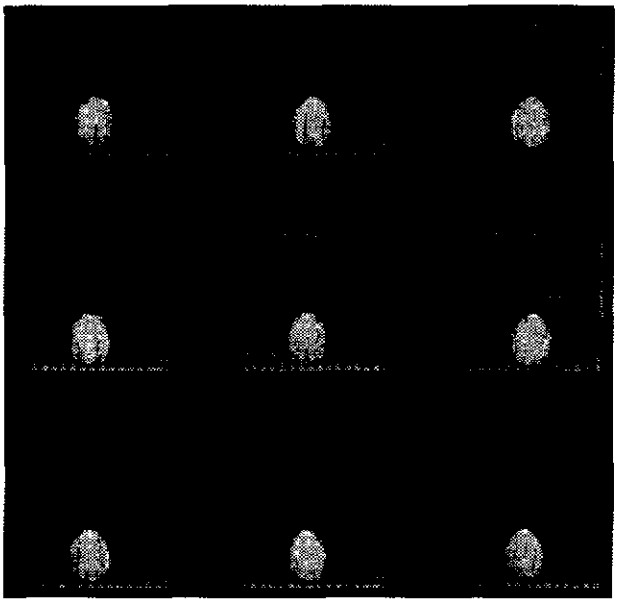


Figure 47 **A sequence of images of a free-form object (a peach)**

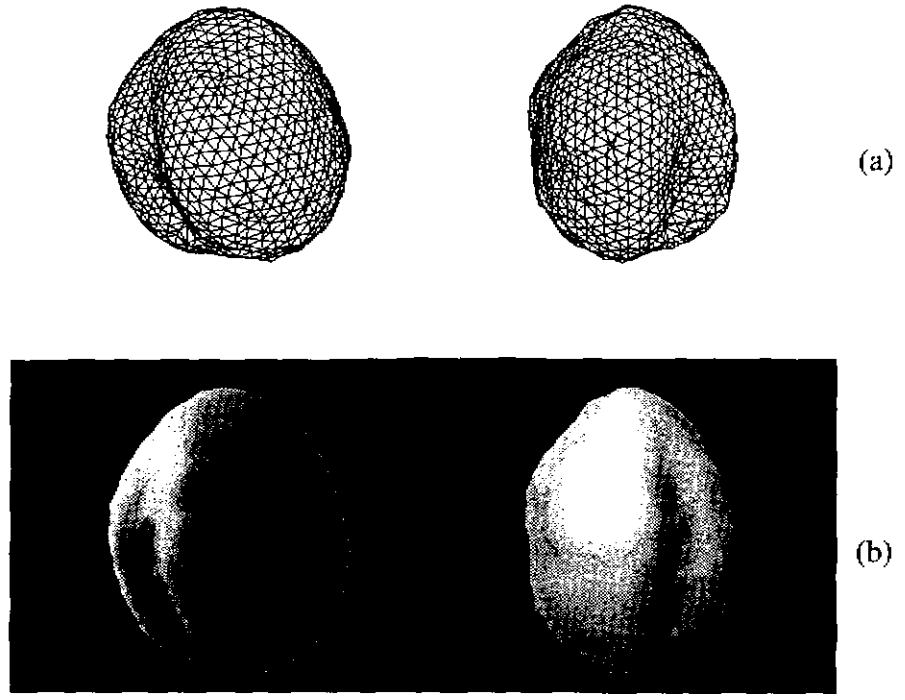


Figure 48 **Two views of a reconstructed peach model: (a) wireframe display; and (b) shaded display**

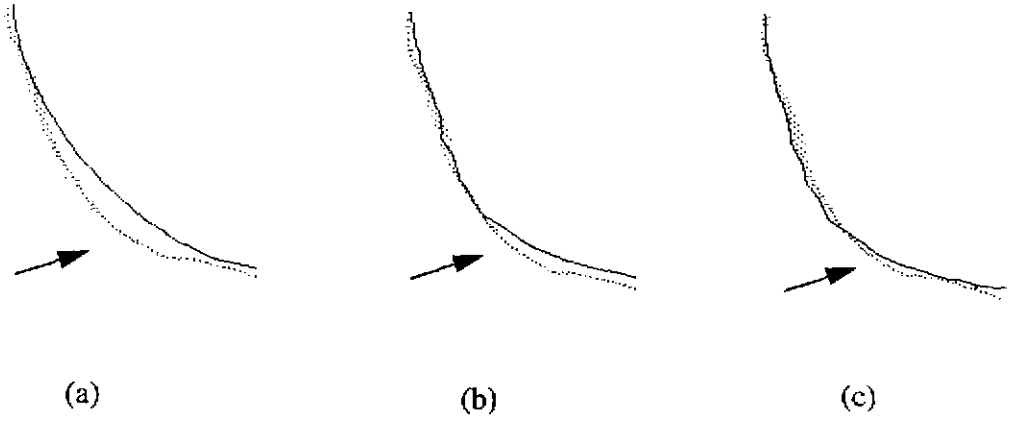


Figure 49 **Comparison using cross-section display of model (solid line) and range data (dots) (a) sequential method; (b) PCAMD after 5 steps; and (c) PCAMD after 10 steps (arrows indicate the places where improvement is the most significant)**

Another set of experiments is conducted to reconstruct a more complex object, a toy sharpei dog. Figure 50 and Figure 51 show a sequence of images of a sharpei, and the deformable surfaces from four different views. The occluded part of the object is interpolated in the deformable surface mesh of each view. The reconstructed object models in different resolutions are shown in Figure 52 and Figure 53. To compare the result of PCAMD with that of the sequential method, we show the cross-section contours of the reconstructed models in Figure 54, and the error at each mesh node in Figure 54. We compare the reconstructed models with the merged range data (using known transformation) in Figure 55. Again it can be observed that the integral method (Figure 54b) produces much better results than the conventional sequential method (Figure 54a), although it is not as perfect as that with known transformation (Figure 54c) due to the least-squares nature of the PCAMD algorithm. Figure 55 shows that the mean error and maximum error for the integral method are 2.7 and 9.2, as opposed to 3.7 and 16.8 for the sequential method.

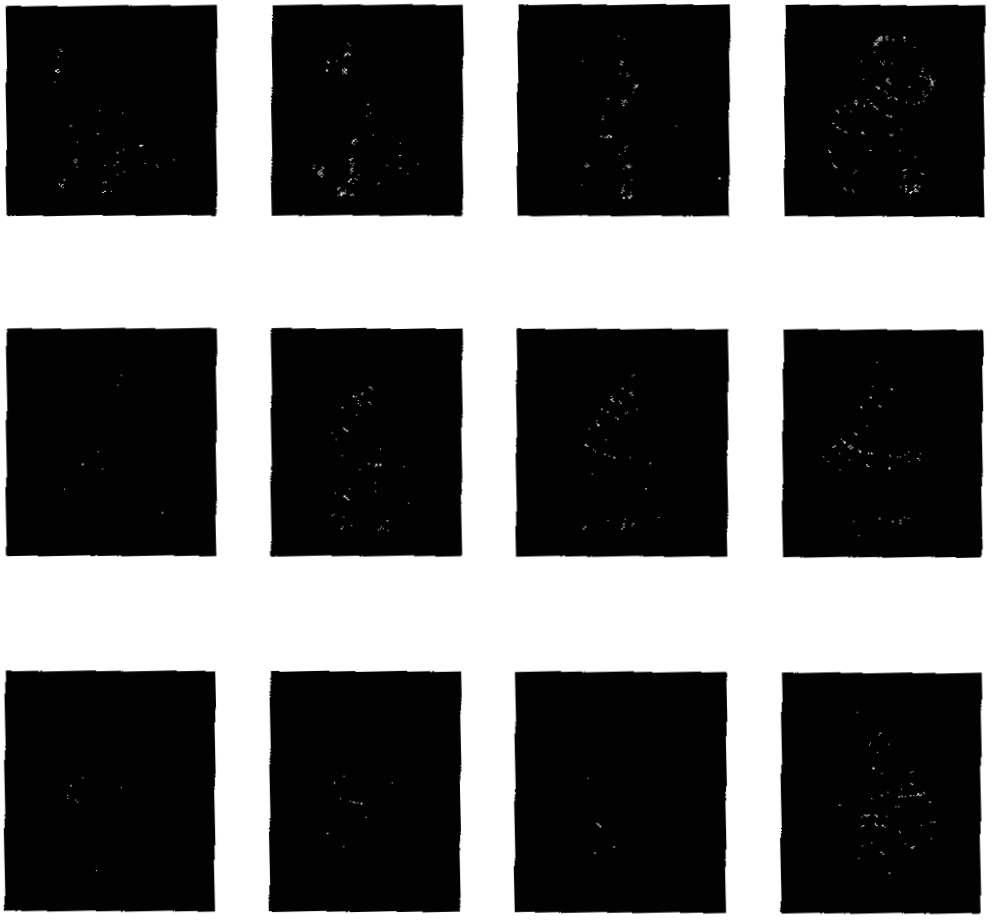


Figure 50 **A sequence of images of a free-form object (sharpei)**

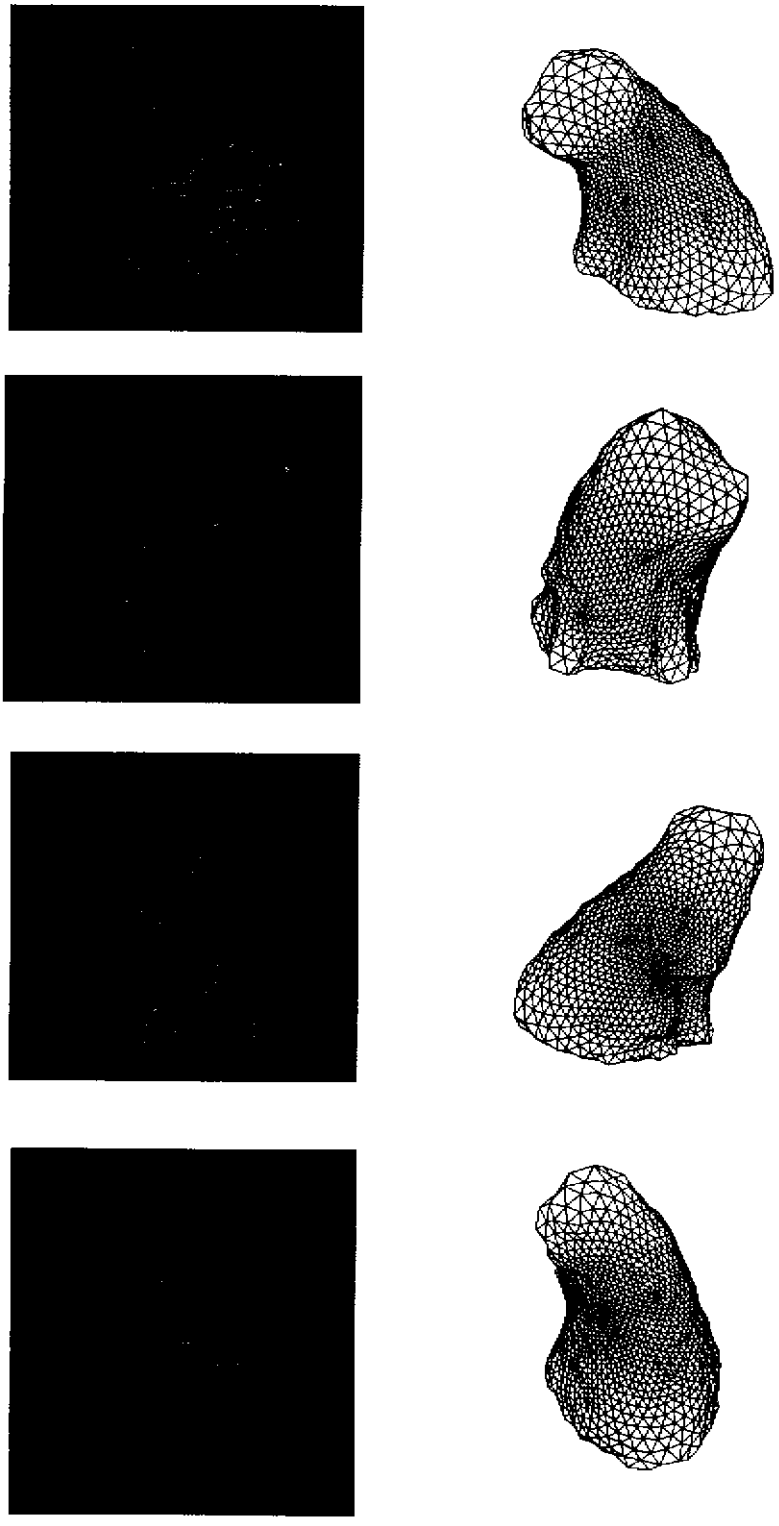


Figure 51 **Examples of deformable models from different views**

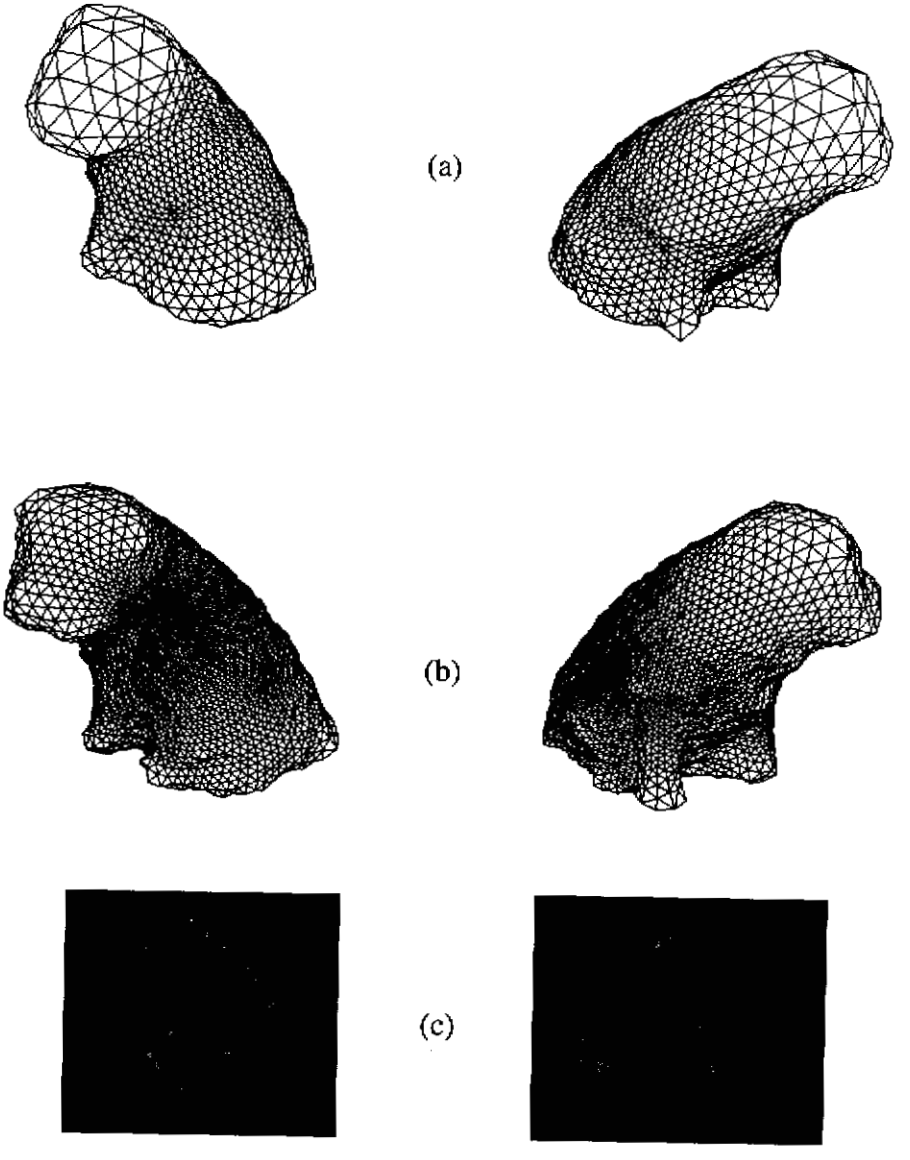


Figure 52 **Reconstructed models of sharpei: (a) coarse resolution: 980 points; (b) fine resolution: 3380 points; and (c) texture mapped display**

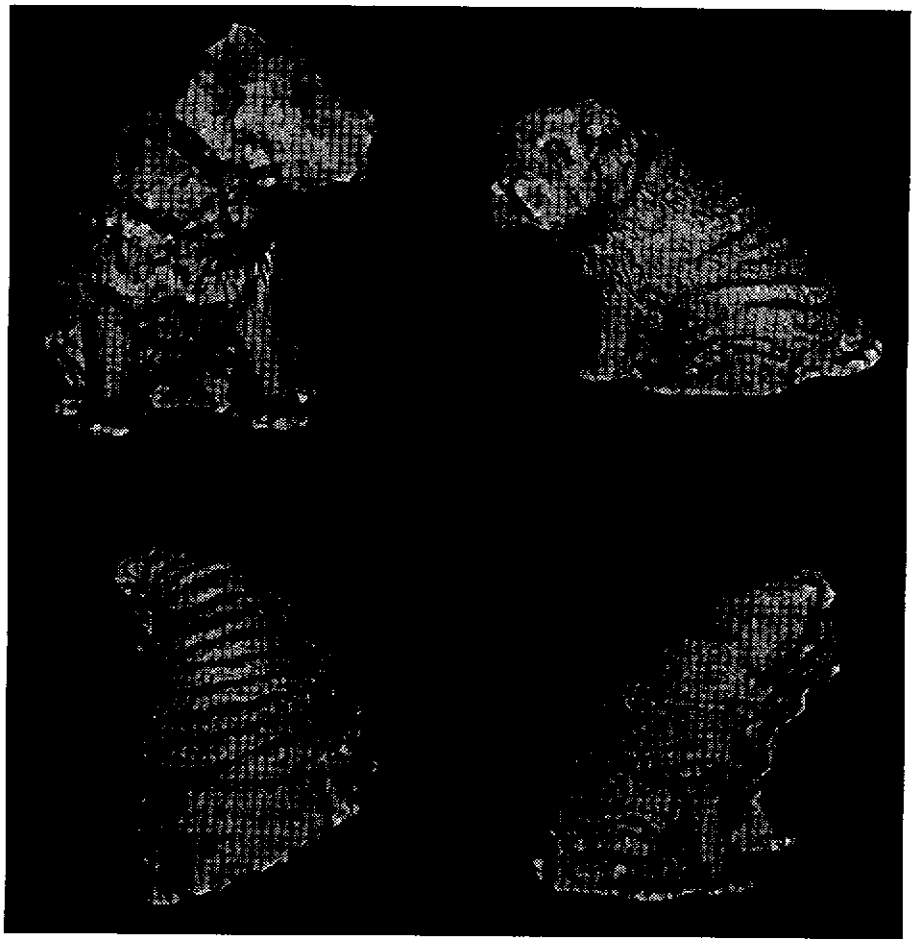


Figure 53 **A shaded model of reconstructed sharpei: high resolution with 8000 points**

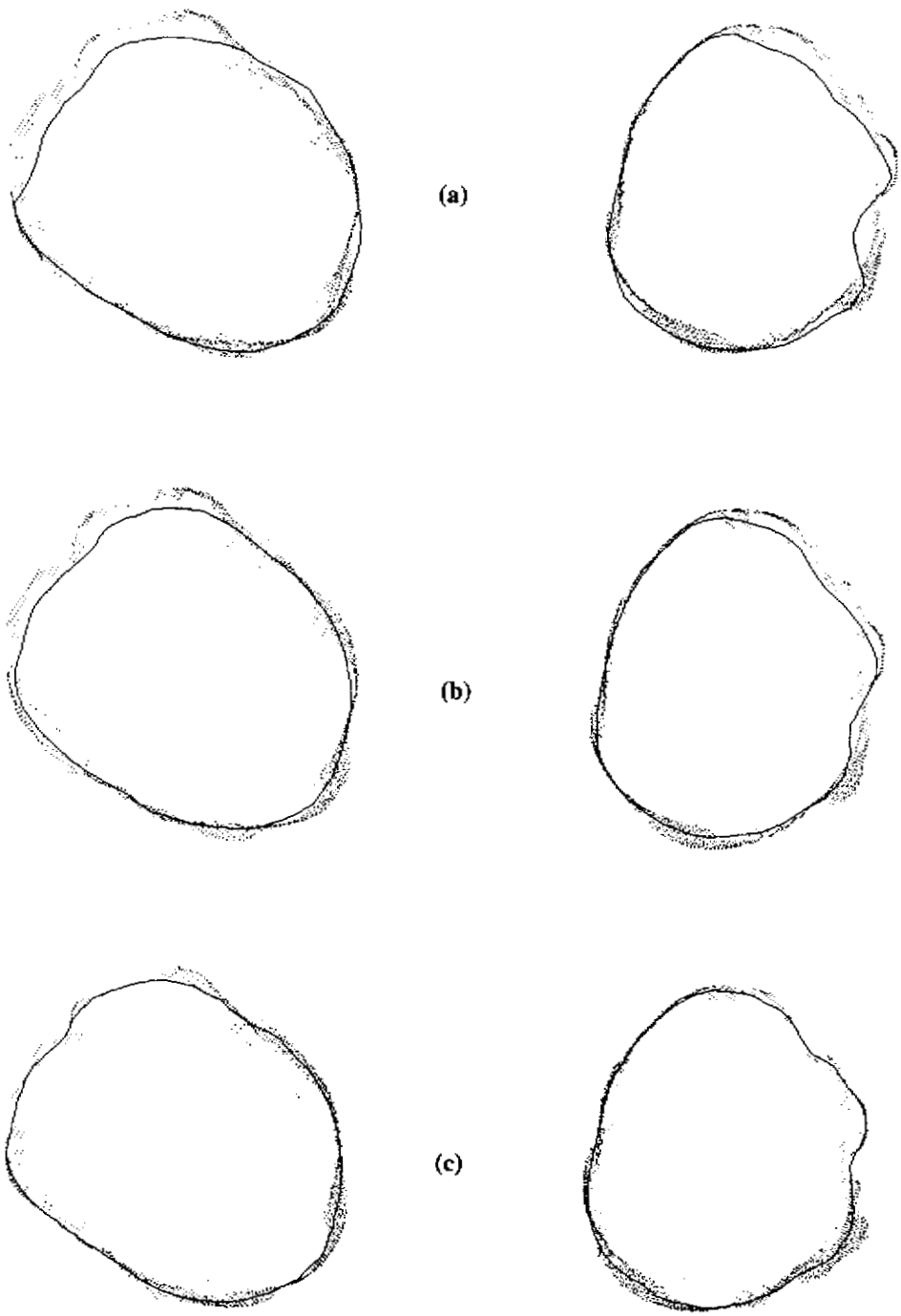


Figure 54 **Comparison between PCAMD and the sequential methods: two contours (small dots are range data, solid line is reconstructed model. (a) sequential method; (b) PCAMD (10 steps); and (c) known transformation**

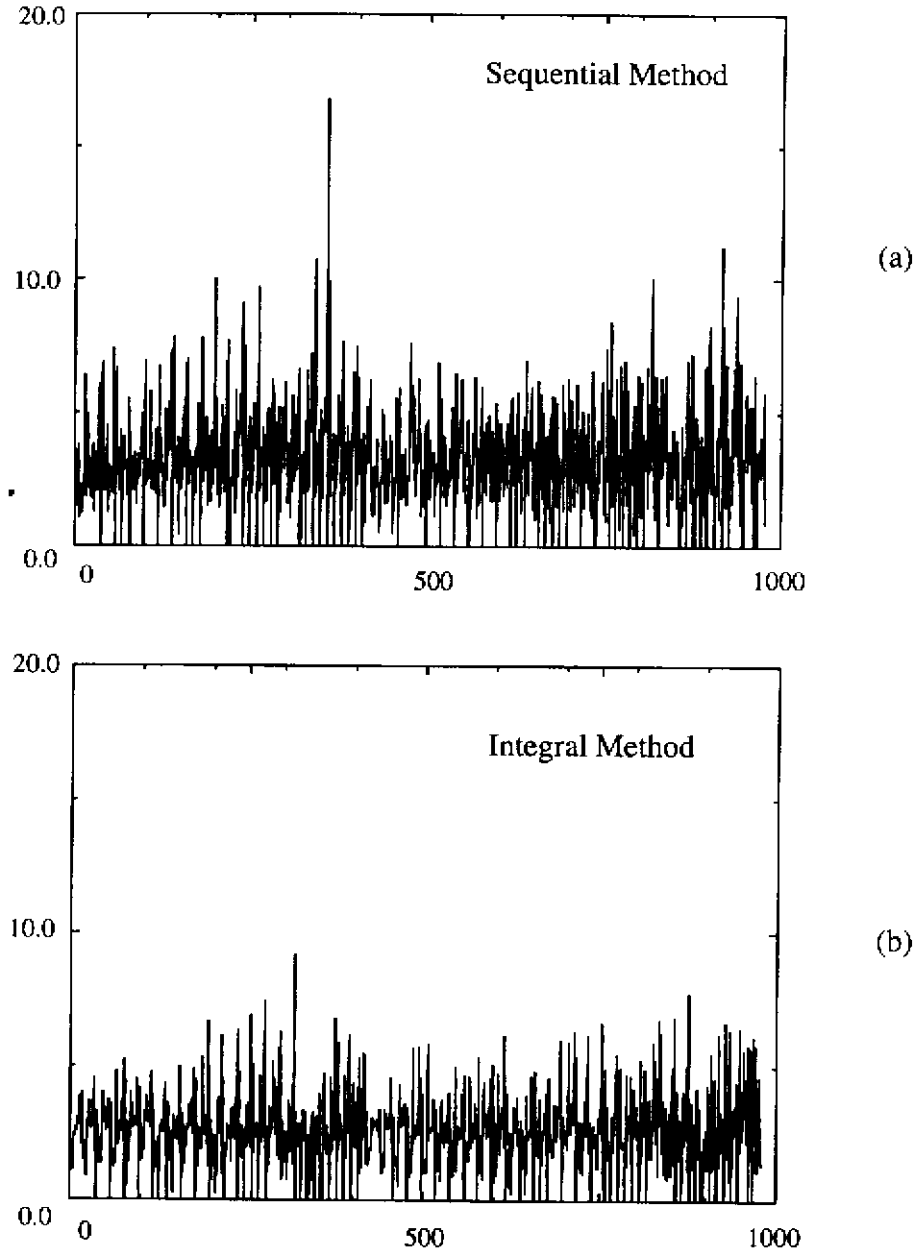


Figure 55 **Comparison between the integral and the sequential methods: error at each mesh node (total number is 980). (a) sequential method; and (b) PCAMD (10 steps)**

5.8 Discussion

An integral approach to free-form object modeling using multiple range images has been described in this chapter. To reconstruct free-form object models from often incomplete and noisy range images, we introduce a global resampling scheme which provides one-to-one correspondence between mesh nodes. Global resampling is achieved using salient feature points which reliably encode local curvature information by taking local connectivity into account. As a result of global resampling, spatial connectivity among different mesh nodes is easily established, leading to the simple construction of object models. Experiments on synthetic data and real range images indicate that our approach converges quickly and produces good models even in the presence of noise and mismatching. Accurate free-form object models reconstructed from sequences of real range images are presented.

One drawback of global resampling is its trade-off between accuracy and robustness. The error introduced in the first step (resampling from range data points to mesh nodes) will affect the second step merging of mesh nodes. To obtain more accurate object models, we can use a coarse-to-fine global resampling scheme similar to [47]. The object models at different levels of detail can then be represented. A simple coarse-to-fine pyramid can be implemented using tessellation at different levels. Another way of building more accurate models is to use adaptive surface subdivision on the coarse level model. Real range data can be filtered by coarse level models before the process of adaptive subdivision.

Chapter 6

Comparing and Synthesizing Shapes

Spherical mesh representation, shown in the previous chapter, is useful in modeling from multiple views. The same representation, along with the shape metric, can also be used for other tasks such as comparing and synthesizing shapes. If we can use the shape metric introduced in the previous chapter to compare different views of an object, can we use the same shape metric to compare different objects as well? Given that the shape metric evaluates how different two objects are, can we synthesize some intermediate shapes which are in between the two originals? These questions are addressed in this chapter.

6.1 Shape Comparison

6.1.1 2D Shape Comparison

Comparison of object shapes is common in many computer vision tasks such as object model categorization and hypothesis verification in model-based object recognition [20]. Previous work has focused on comparing 2D scene images with 2D object models. A gradual shape change of a 2D closed curve, such as what is shown in Figure 56, from a square to a concaved triangle, can be captured by existing shape similarity measures (e.g., [2][104]).

Figure 56 **An example of 2D shape similarity: how to measure the gradual shape change from left to right?**

However, recent developments in 3D sensors such as laser range finders and real-time stereo machines makes it necessary to address the problem of comparing 3D objects with 3D or 2D scene images. In this chapter, we attempt to answer the following question: To what extent is a 3D shape A similar (or dissimilar) to a 3D shape B?

The desirable properties of such a shape similarity measure are as follows. First, such a measure between two geometrical shapes should be a metric. In particular, the triangle inequality property should hold since it is desirable in pattern matching and object recognition applications. Second, the distance function between two shapes should be invariant under rigid transformation and scaling, easy to compute, and intuitive with human shape perception [2].

The problem of shape similarity has been studied extensively in both machine vision and biological vision. Readers interested in human perception of similarity, such as contextual and asymmetrical properties, are referred to Tversky [125] and Mumford [83]. It is true that features such as color or functional information are often used to compare objects in human perception. However, in this chapter, we focus on geometrical shape similarity because other features such as color and reflectance, etc. are based on geometrical representations. As an initial step toward 3D shape similarity with arbitrary topology, we confine ourselves to objects of genus zero (objects without holes). We want to compare polyhedral shapes as well as smooth surfaces.

To compare different shapes, one must first understand how they should be represented. Most prior work assumed object shapes to be two dimensional closed contours. Accordingly, many methods have been devised to evaluate the shape similarity among a set of 2D closed polygons as those shown in Figure 56. For instance, Schwartz and Sharir [104] proposed to approximate a closed 2D curve, after proper smoothing if necessary, by a simple polygon with equal-length edge segments. The polygon was then represented by the turning angle (a measure of local curvature) at each vertex (see Section 5.3). Similarly, Arkin et al. [2] represented local curvature at each vertex of a polygon using a turning angle. In addition, they proposed an efficient algorithm to directly compare polygons. Unfortunately because of the lack of a proper coordinate system, their approach can not be extended to 3D polyhedra. Mumford [83] suggested the use of moments as an alternative to curvature because moments are also invariant to rigid transformation and scaling. Other 2D shape similarity methods include 2D planar graph and graph matching by Kupeev and Wolfson [69], and shape deformation by Basri et al. [6].

6.1.2 3D Shape Comparison

While a closed 2D curve can be simply parameterized by its arc-length, it is much more difficult to find an appropriate "data structure" to store a 3D surface. How the curvature information on a 3D surface is computed and stored depends on the choice of the coordinate system. In addition, since the curvature of polyhedral shapes is zero everywhere except on vertices and edges, it is unclear how the shapes should be compared. For example, it is no trivial task to compare 3D shapes as simple as those shown in Figure 57. In practice, local curvature on each sample point of surface is difficult to estimate robustly from noisy range data. Because of surface discontinuity and occlusion, this problem is even more severe when only a single view depth map is available.

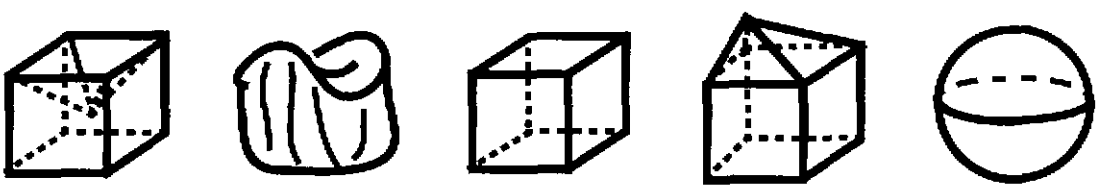


Figure 57 **An example of 3D shape similarity: how similar are these shapes?**

Because a closed 3D surface is topologically equivalent to a sphere, many spherical representations have been proposed to represent closed 3D surfaces. Ikeuchi [57] and Horn [53] proposed to represent objects with an extended Gaussian image (EGI) which uses a distribution of mass over the Gaussian sphere. A Gaussian sphere is a unit sphere on which the surface normal at each point is mapped. Little [72] showed that an EGI could be used for pose determination. It has been proven that two convex objects are congruent if they have the same EGIs. A Complex EGI was proposed by Kang and Ikeuchi [64] to store both surface area and distance information. This method can be very useful in recovering translation. Nalwa [87] augmented Gaussian images by some support function which was the signed distance of the oriented tangent plane from a predefined origin. Hebert, Ikeuchi and Delingette [45] proposed a simplex attribute image (SAI) to characterize convex/concave surfaces, both as a coordinate system and as a representation. Brechbuhler, Gerig and Kubler [14] also defined a one-to-one mapping from a simply-connected surface to a unit sphere, using extended 3D elliptical Fourier descriptors. For a summary of different spherical representations, the reader is referred to [60] by Ikeuchi and Hebert.

The lack of a proper coordinate system (or data structure) for geometrical entities has prompted many researchers to compare 3D shapes in domains other than geometrical space. For example, Sclaroff and Pentland [105] used many modes to represent shapes and to compare shapes based on the coefficients of the modes. Murase and Nayar [86] represented objects in eigenspace, and compared objects using the proximity of two eigenvalues to one another. Hancock and Thorpe used a similar eigenspace representation successfully for land vehicle navigation [43]. Unfortunately, these similarity measures do not provide us with geometrical intuition.

Even with an appropriate data structure, the choice of a good metric for comparing shapes can be difficult. For example, Arkin et al. [2] used L_2 norm to compare polygons. Huttenlocher and Kedem [56] used Hausdorff distance to compare the distance between two point sets under translation. Kupeev and Wolfson [69] used graph matching to compare 2D shapes. Basri et al. [6] emphasized that the distance function has to be continuous and should matter less as curvature becomes greater. Comparisons among different metrics can be found in [83].

In this chapter, we use a special spherical coordinate system to represent a closed curved or polyhedral 3D surface without holes. In the previous chapter, a semi-regularly tessellated sphere is deformed so that the meshes sit on the original data points while preserving the connectivity among the mesh nodes. At the end of the deformation process, we obtain a spherical representation with local curvature at each mesh node. The problem of comparing two shapes becomes that of comparing corresponding curvature distributions on spherical coordinates. This approach is illustrated in Figure 58. The local curvature at each node is calculated by its relative position to its neighbors. A shape metric between two objects is the minimum distance between two corresponding curvature distributions on spherical coordinates.

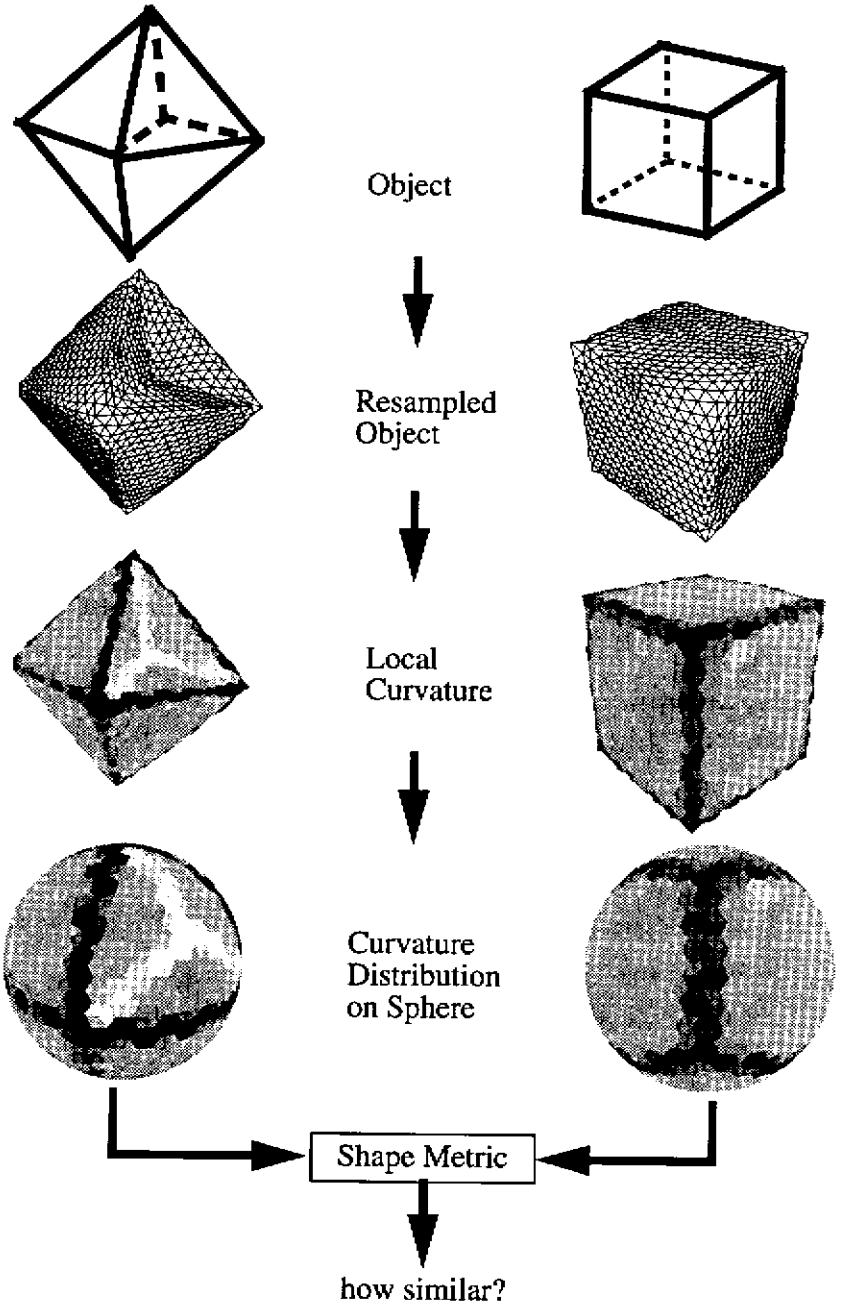


Figure 58 **Comparing shapes from curvature distribution: an example of a sphere and a hexahedron. The curvature has been color-coded so that the darker represents a bigger positive curvature and the lighter represents a bigger negative curvature**

6.2 Shape Similarity Experiments

This section presents the results of the application of our shape similarity metric to synthetic data and to real objects. For ease of computation, we use L_2 distance in the metric function defined in Section 5.5 for all the experiments in this section.

Our data set consists of several polyhedra including an icosahedron and a dodecahedron whose shapes are known in advance. To make deformable surfaces, we generate uniformly random-sampled data points over each object surface. We also use the free-form object model generated from real range images. Unless specified otherwise, the frequency of spherical tessellation is set to 7, which means that the total number of mesh nodes is 980.

Figure 59 shows the approximation of a sphere by a set of regular polyhedra: a tetrahedron, a hexahedron, a dodecahedron, and an icosahedron. Figure 60 shows the distance between these regular polyhedra (convex) and a sphere. Since curvature is constant everywhere on a sphere, computing the minimum distance between a polyhedron and a sphere can be greatly simplified. Figure 61 shows a sequence of concave objects which are generated by making concave dents on an octahedron. A comparison of the shape similarity between this sequence of concave objects and an octahedron is summarized in Figure 62. The distance between object (1) and the octahedron is big because object (1) is more concave and no longer star-shaped.

Figure 64 compares the degree of shape similarity among a set of free-form objects shown in Figure 63. The distance functions among all objects are plotted in Figure 64a, in which all the distances along the diagonal are zero. Figure 64b and Figure 64c show the distance between the object dog and others, and the distance between the object Sharpei and others, respectively. Figure 65 shows the shape change from others to dog. Figure 66 shows the shape change from others to Sharpei.

One possible drawback of our approach is that the quality of approximation of a polyhedral or free-form surface is highly dependent on the number of patches chosen. With a frequency 7 semi-regular spherical tessellation, we have 980 surface patches. We have 3380 patches when the frequency increases to 13. Obviously the more surface patches we use, the better the approximation. Figure 67 presents the curvature distribution of an approximated hexahedron when different tessellation frequencies are used. With a higher frequency, the higher curvature distribution is narrower because of better approximation. Figure 68 shows how the shape similarity measures differ when different tessellation frequencies are used. The

results demonstrate that the shape similarity measure is robust provided that a sufficiently fine tessellation is adopted.

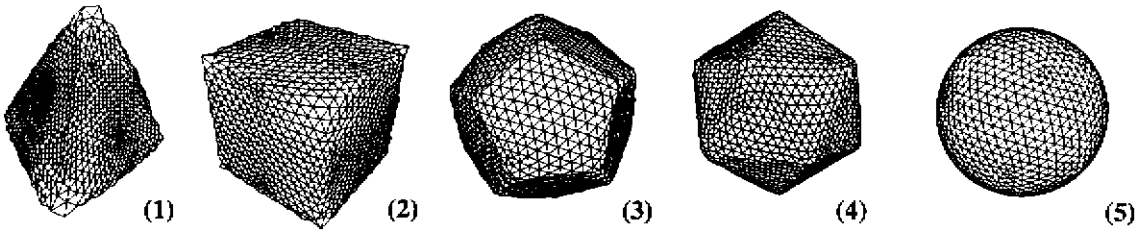


Figure 59 **Polyhedral approximation of a sphere: (1) Tetrahedron; (2) Hexahedron; (3) Dodecahedron; (4) Icosahedron; and (5) Sphere**

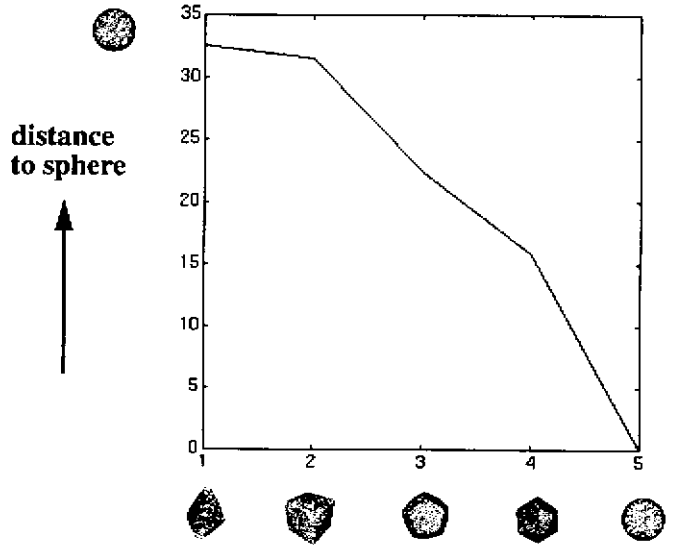


Figure 60 **Distance between a sphere and its polyhedral approximations**

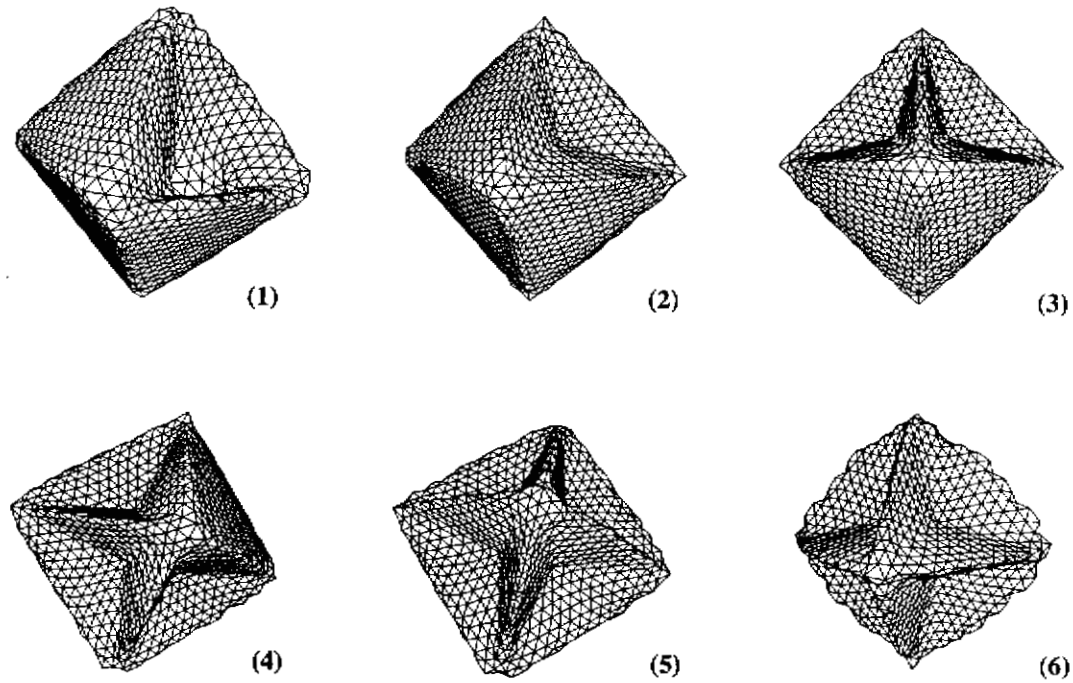


Figure 61 **Concaved octahedra: (1) with one deep concave dent; (2) with one concave dent; (3) with two dents; (4) with three dents; (5) with four dents; and (6) with eight dents**

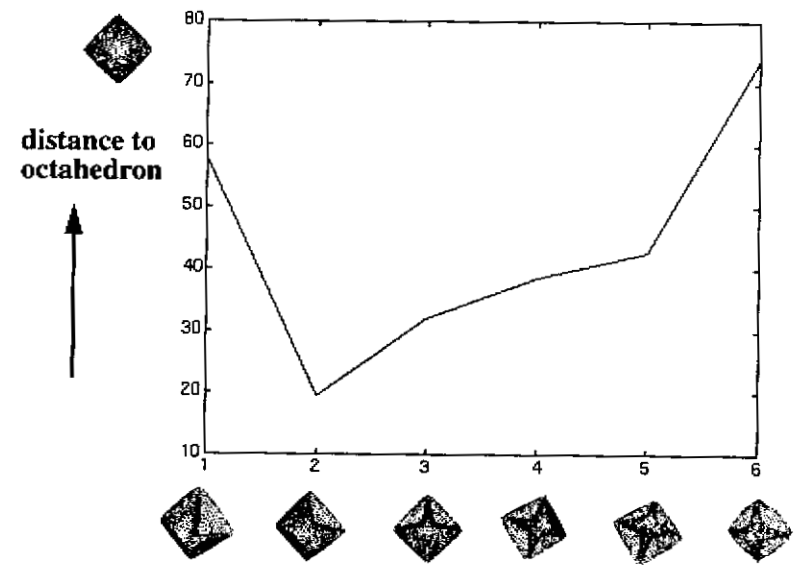


Figure 62 **Distance between an octahedron and several concaved octahedron objects**

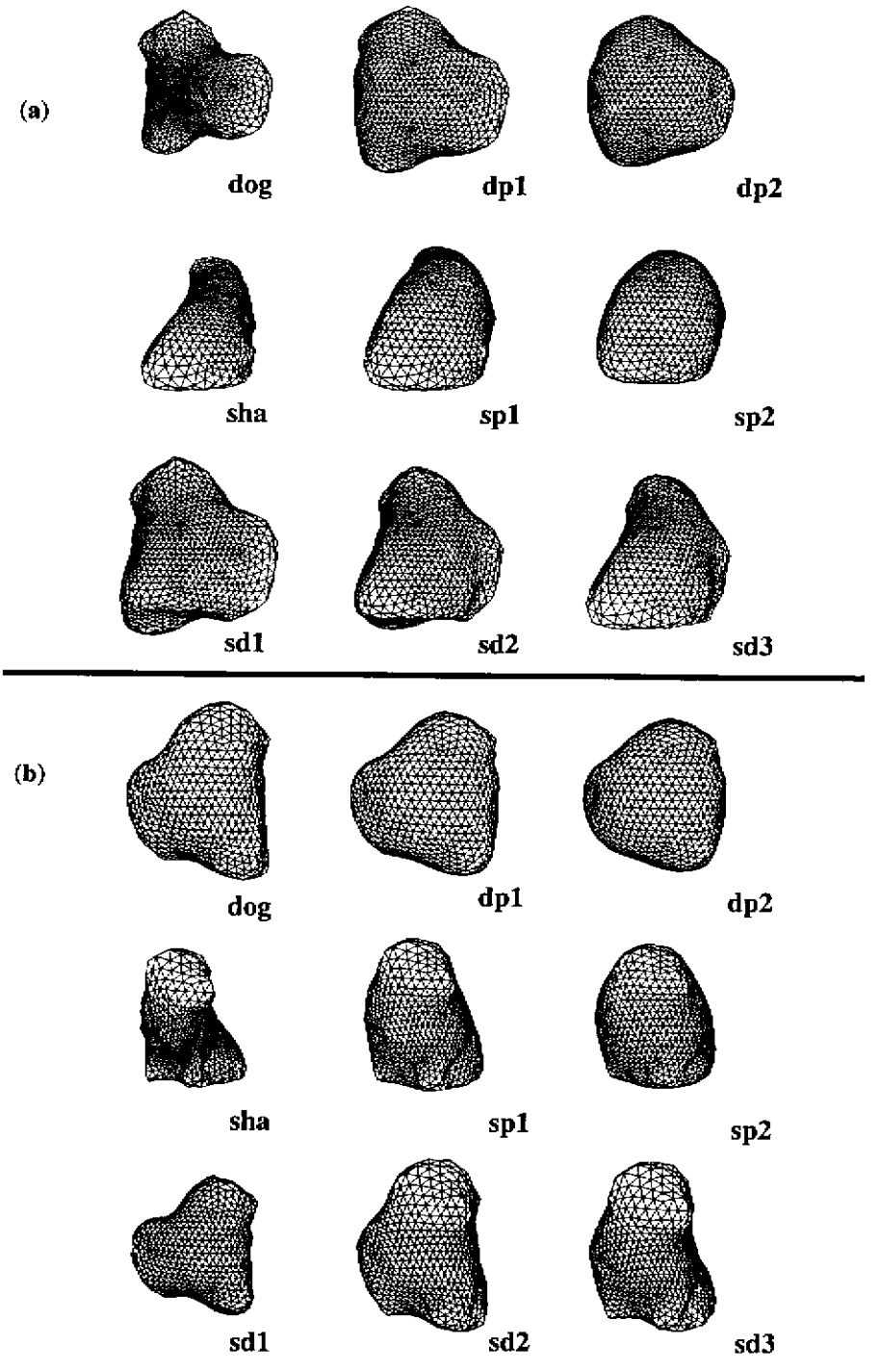


Figure 63 (a) Free-form objects: "dog" and "sharpei" are generated from real range data; dp1 and dp2 are two approximations of "dog"; sp1 an sp2 are two approximations of "sharpei"; sd1, sd3, and sd3 are three intermediate shapes between "sharpei" and "dog". (b) A different view of all 9 objects

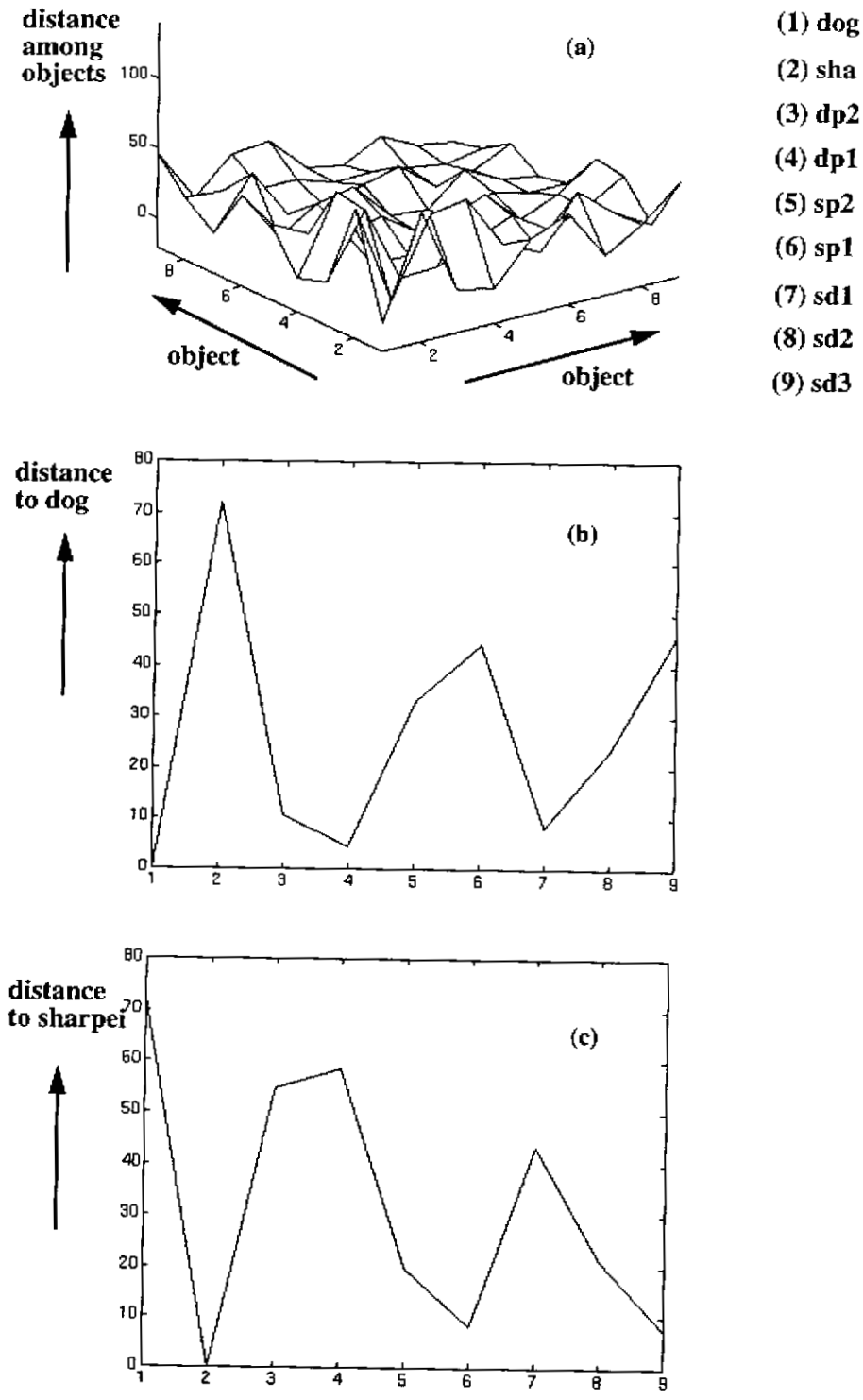


Figure 64 (a) Shape similarity among all free-form objects: distance of pair-wise comparison. (b) The distance between the object "dog" and others. (c) The distance between the object "sharpei" and others

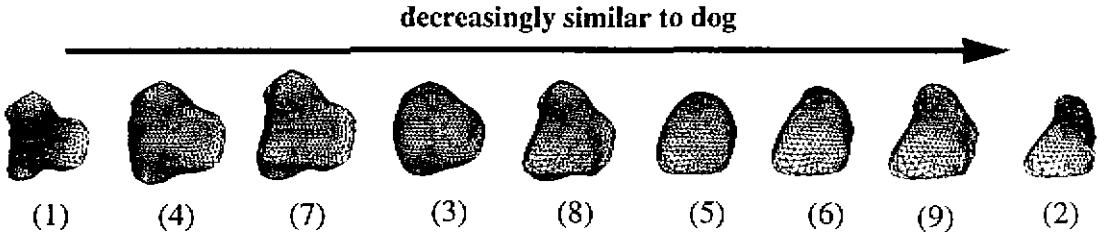


Figure 65 **Shape change from the object "dog" to others. From left to right: shapes are more and more dissimilar to "dog"**

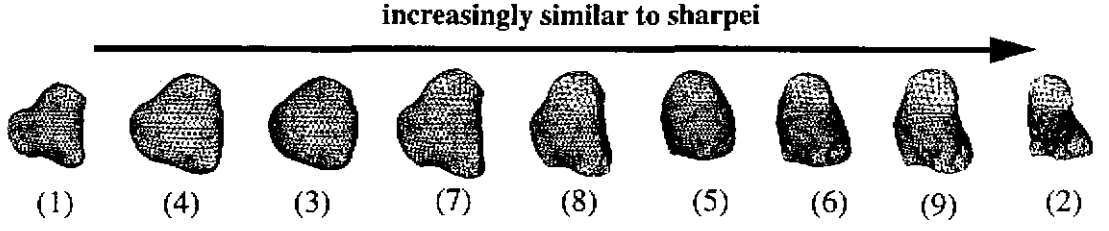


Figure 66 **Shape change from the object "sharpei" to others. From left to right: shapes are more and more similar to "sharpei"**

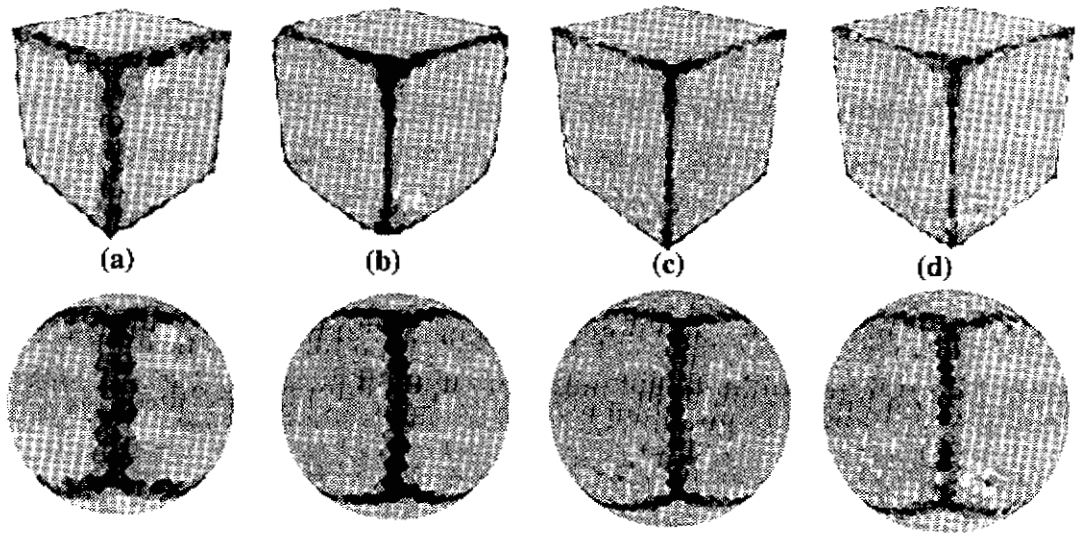


Figure 67 **An example (hexahedron) of curvature distribution of mesh representation at tessellation frequencies: (a) f=7; (b) f=9; (c) f=11; and (d) f=13**

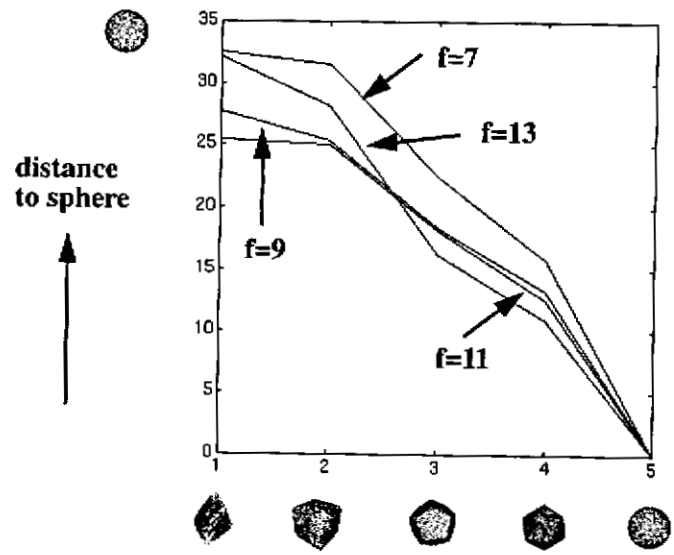


Figure 68 **Effect of tessellation frequency on shape similarity between regular polyhedra and a sphere**

6.3 Shape Synthesis with Curvature Interpolation

6.3.1 Shape Synthesis

Another interesting application of our spherical surface representation is 3D shape synthesis. The problem of 3D shape synthesis is defined as follows. Given two 3D shapes A and B, we synthesize a sequence of intermediate shapes (or morphs) which change smoothly from shape A to shape B. Similar to image morphing [7], shape morphing is achieved by first warping the two original shapes to a new shape, then interpolating both into an intermediate shape. It is difficult to design a good 3D morphing system for three reasons. First, it is unclear what should be chosen as features. Second, it is difficult to match the features automatically. Third, what is to be interpolated along the morphing sequence has to be carefully selected to guarantee the smoothness of the shape transition.

The above difficulties arise from the lack of a proper representation for arbitrary shaped 3D objects. Thus, we propose to represent shapes of genus zero using a semi-regularly tessellated spherical mesh. After an original shape is resampled by a nearly uniform spherical mesh, we store the local curvature at each node of the resulting mesh as its intrinsic representation. Using the fixed connectivity of all mesh nodes in the semi-regularly tessellated mesh, the correspondence between two shapes can be easily established. Instead of directly interpolating corresponding mesh node coordinates of two known objects, we consider the task of synthesizing a new object as one of interpolating the two known curvature distributions, and then mapping the interpolated curvature distribution back to a 3D shape. It has been shown in the previous chapter that such an inverse mapping from intrinsic representation to object shape can be achieved. From the intrinsic representation, the original shape can be reconstructed up to a scale factor and a rigid transformation, provided that a regularity constraint is satisfied, i.e., each mesh node is projected onto the center of its neighboring triangle. The application to surface morphing is again a demonstration of the inverse mapping property.

6.3.2 Basic Idea

Our approach to shape synthesis is summarized in Figure 69. For each original shape, we uniformly resample it with semi-regularly tessellated spherical mesh. Then we build its curvature distribution as its intrinsic representation. Instead of manually searching for individual features, the correspondence between two shapes can be automatically obtained by minimizing the difference between two curvature distributions. Additional correspondence specified by the user can also be incorporated. To synthesize a new shape from two known

shapes, we interpolate the new curvature distribution from two known curvature distributions. The new shape is then reconstructed from the interpolated curvature distribution. Delingette proposed a similar synthesis approach where deformation is used to interpolate intermediate morphs [31].

In addition to the automatic correspondence between two shapes, our shape synthesis approach using curvature distribution has another advantage in that the resulting shapes vary smoothly. This is because the morphs are interpolated according to the original curvature distributions. Volume morphing, on the other hand, can hardly quantitatively specify this kind of shape change. Instead, it has to rely on careful selection of features, and thus it provides only qualitative shape change at best. By making explicit use of the curvature information, our method also reduces the ambiguity of matching two shapes. This is especially important for smooth curved objects, where features are very difficult to identify manually.

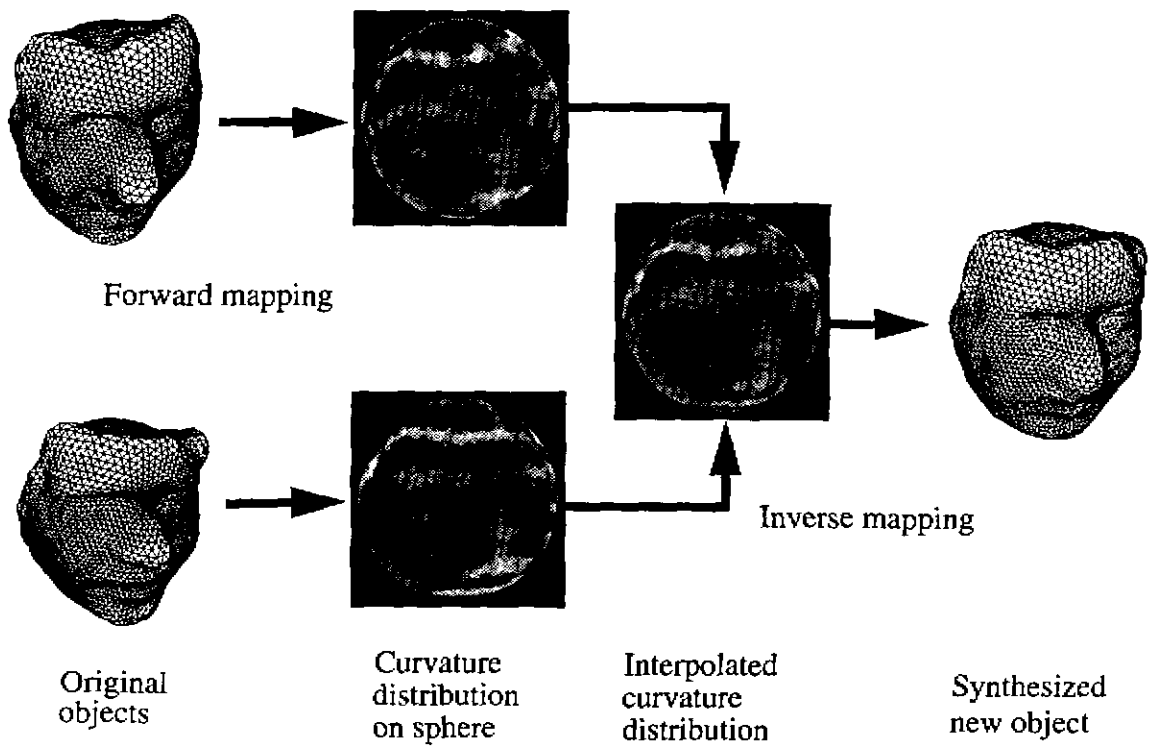


Figure 69 **Shape synthesis with local curvature interpolation**

6.4 Curvature Interpolation

It is unclear how to interpolate two shapes unless we know how to compare them. It is very difficult to compare two 3D shapes because of the unknown scale factor and rigid transformation. We have shown that it is possible to quantitatively measure the distance between two shapes using the intrinsic representation. Thus we can also interpolate these two shapes to obtain a new curvature distribution from the intrinsic representations of two original shapes and their correspondence. An advantage of this approach is that it shows quantitatively how much the morph is different from the originals. For example, at each mesh node of the new mesh C, its curvature can be computed by a linear interpolation of its counterparts in the original shapes A and B. More specifically,

$$\varphi_{C_i} = (1-t)\,\varphi_{A_i} + t\varphi_{B_i},\ t \in [0, 1],\ i = 1...n, \qquad \text{(EQ 6.1)}$$

where φ is the local curvature measure. Alternatively nonlinear cross-dissolve techniques can also be used. For instance, if we use the following interpolation function,

$$\varphi_{C_i} = \left(1-2t^2\right)\varphi_{A_i} + 2t^2\varphi_{B_i},\ t \in [0, 0.5],\ i = 1...n, \qquad \text{(EQ 6.2)}$$

$$\varphi_{C_i} = 2(1-t)^2\varphi_{A_i} + \left(1-2(1-t)^2\right)\varphi_{B_i},\ t \in [0.5, 1],\ i = 1...n, \qquad \text{(EQ 6.3)}$$

to blend curvatures, we can enforce desirable small shape change in the initial steps. Figure 70a shows the linear change of a shape similarity distance between each morph and an original object (a toy sharpei), while Figure 70b shows the distance between each morph and the other original object (a toy pig). Figure 71 shows the nonlinear change of a shape similarity distance. The distance between two shapes is defined as the L_2 norm of the distance between two curvature distributions, which has also been used as a measure of shape similarity in Section 6.1. It is clear that our synthesis approach is not only consistent with, but also can be controlled by, our metric measure of shape similarity.

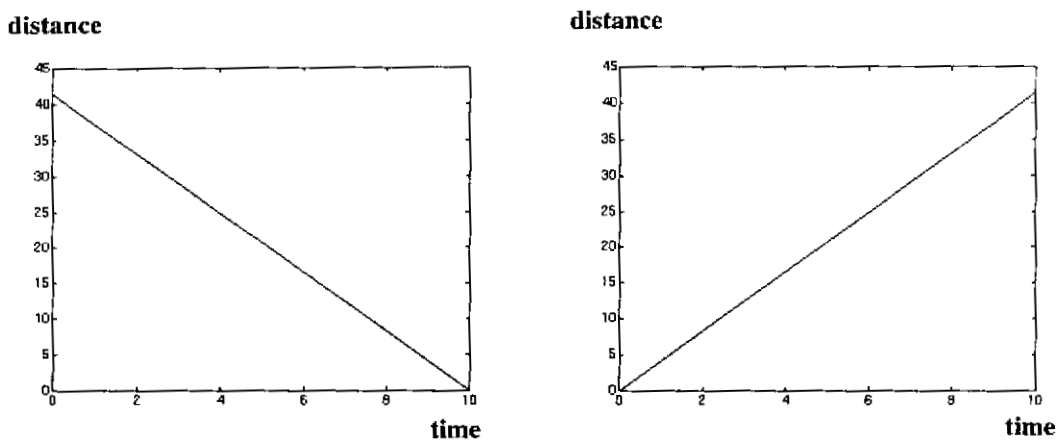


Figure 70 **Linear interpolation of shape distance between the morph sequence and two originals: (a) sharpei; and (b) pig**

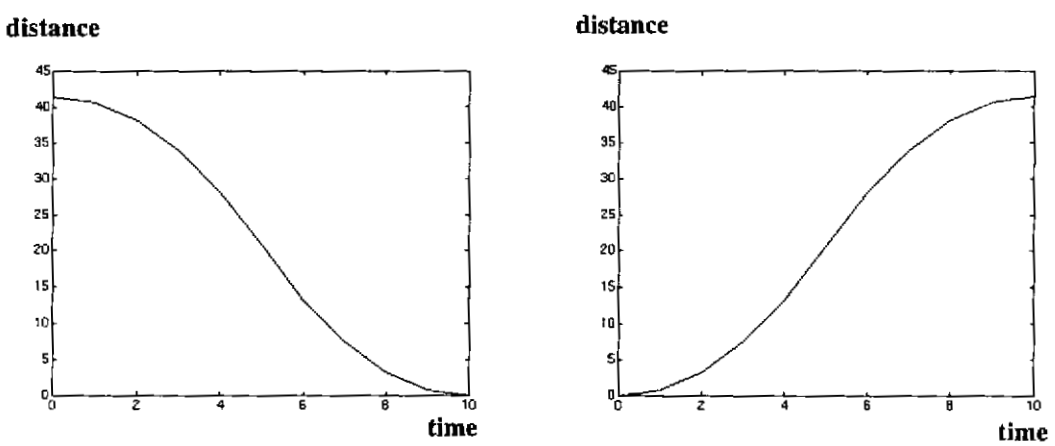


Figure 71 **Nonlinear interpolation of shape distance between the morph sequence and two originals: (a) sharpei; and (b) pig**

6.5 Shape Synthesis Results and Discussion

Figure 72 shows a sequence of morphs which are synthesized from a toy sharpei to a toy pig, and Figure 73 shows a different view of the same sequence. The models of these free-form objects are constructed from real range images using methods described in the previous chapter. The frequency of spherical tessellation is set to 13, which means that the total number of mesh nodes is 3380. After the models and their curvature distributions are

obtained, we use linear interpolation to generate the intermediate curvature distributions. These intermediate curvature distributions are then inversely mapped to the morphs. These morphs clearly show a gradual and smooth shape transition between the sharpei and the pig. Figure 74 shows a gradual shape morphing from one human head to another.

Compared with volume morphing [44][55], our method is fast and easy to implement. While volume morphing takes more than a day [71] to render a sequence of morphs, our method takes less than an hour to complete the shape synthesis. In our experiments, it takes 20 minutes to build deformable models and curvature distribution, and 20 minutes for cross-dissolve curvature interpolations and shape synthesis of a sequence of 10 morphs on a SUN Sparc 20. This does not take into account the time spent in taking images and rendering images for the final display. To speed up the iteration process, the morph from the previous step is used to initialize the synthesis of the next morphing step.

One possible drawback of our approach is that the quality of approximation of a polyhedral or free-form surface depends on the tessellation frequency or the number of mesh nodes specified. Obviously, the more mesh nodes we use, the better the approximation. Our shape synthesis approach generates a good morphing sequence, provided that a sufficiently fine tessellation is adopted.

For the purpose of shape synthesis, we may want to satisfy the user's specifications such as manual correspondences, etc. This user-specific requirement can be incorporated into a force F_{ext} such that (EQ 5.1) is modified as

$$m\frac{d^2P_i}{dt^2} + k\frac{dP_i}{dt} = F_{int} + F_{ext} \qquad \text{(EQ 6.4)}$$

which is similar to deformable surface model extraction where F_{ext} is dominated by data force.

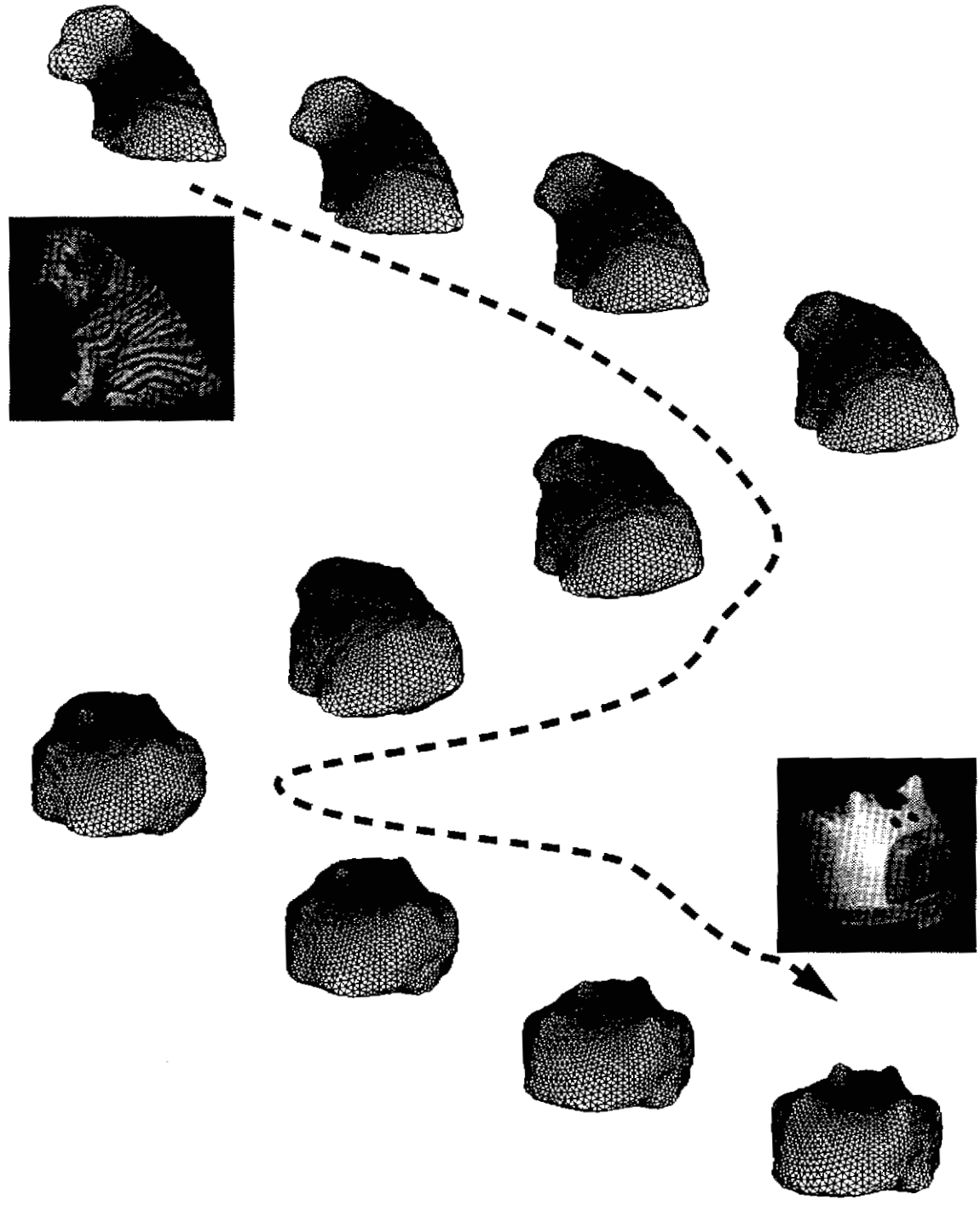


Figure 72 **A morphing sequence between a toy sharpei and a toy pig**

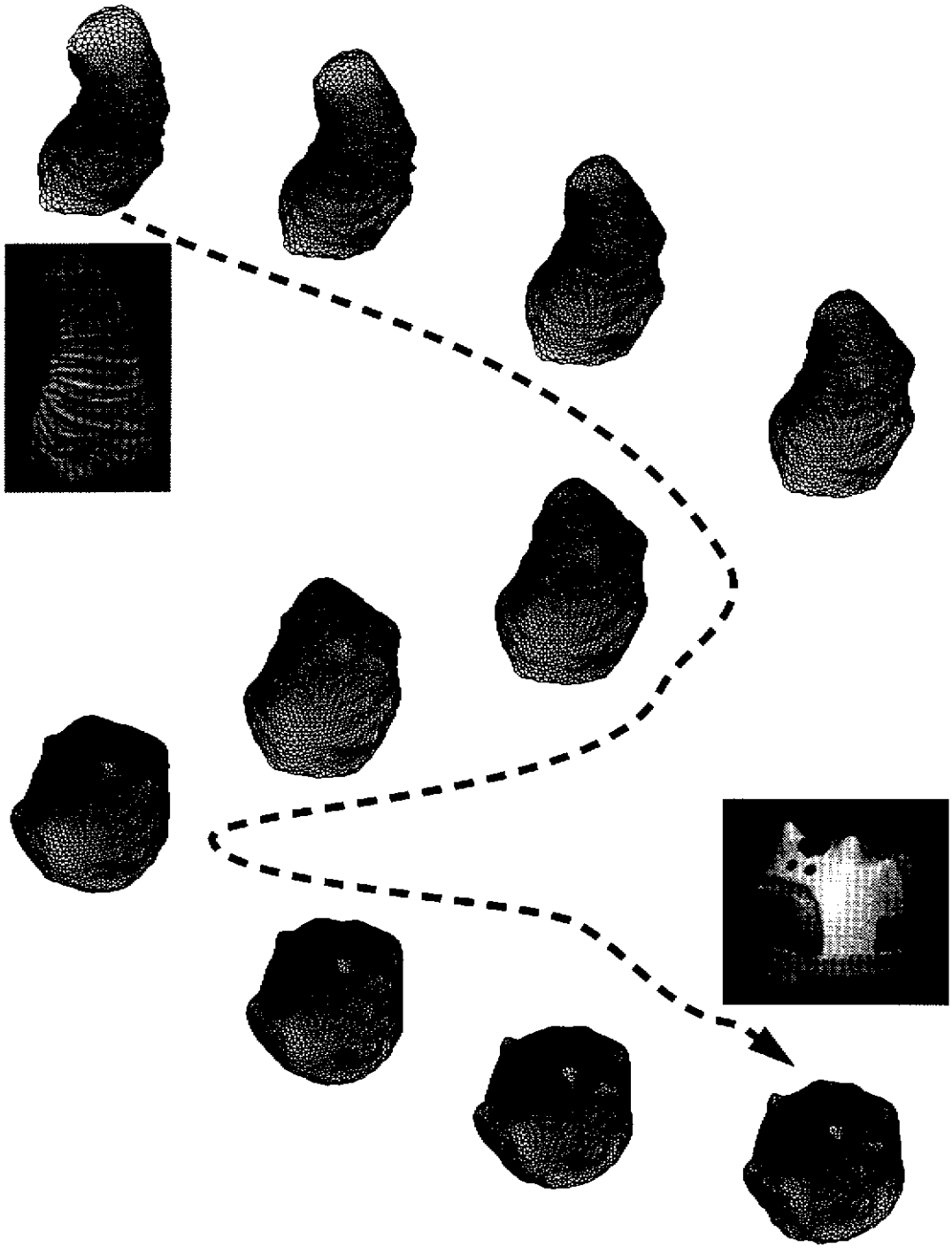


Figure 73 **Another view of the morphing sequence between a toy sharpei and a toy pig**

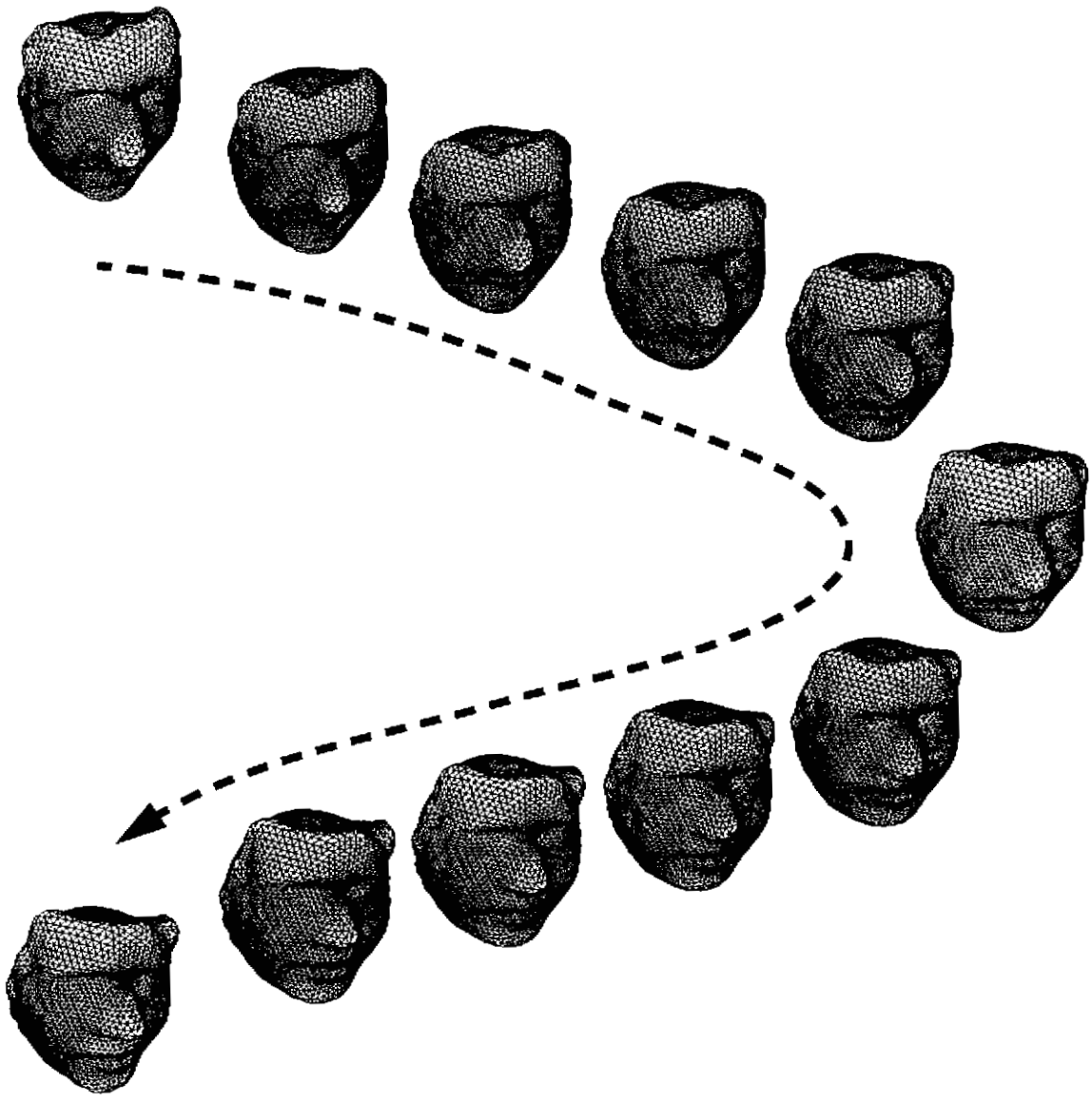


Figure 74 **A morphing sequence between Mark W. and Mark M.**

6.6 Global Properties of our Curvature Representation

When we interpolate two local curvature distributions using simplex angles, we can not prove that any global property exists for our simplex angle image. Even though we have had good results using the simplex angle which is an approximation of mean curvature at each mesh node, we unfortunately do not have a global property similar to the turning angles for 2D curves. Recall that if we sum all the turning angles for a 2D curve, we end up with a constant 2π. This is a very important property because we can then verify if a curvature distribution is sufficient for a shape.

However, a similar global property does exist if we replace the simplex angle by a spherical deficit angle at each mesh node. We have shown in the previous chapter that there are different ways of representing local curvature. For example, the spherical deficit angle can be an alternative to the simplex angle. In fact, the Global Gauss-Bonnet theorem [33] shows that the integration of total Gaussian curvatures over an orientable compact surface is always a constant, which is $2\pi\chi$ where χ is the Euler number. The Euler number of an object with spherical topology is 2. We present a simple proof of the discrete version of this theorem.

Discrete Gauss-Bonnet Theorem

$$\sum_i k_i = 2\pi\chi$$

Proof.

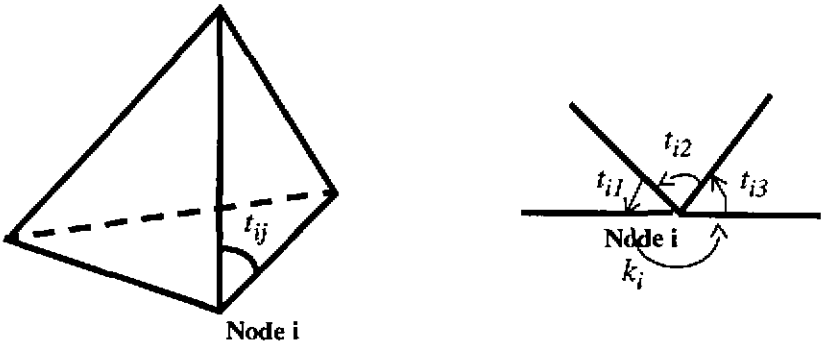


Figure 75 **Spherical deficit angle and exterior angles**

From the definition of spherical deficit, we have

$$k_i = 2\pi - \sum_j t_{ij}$$

where t_{ij} is the j-th exterior angle on node i, as shown on Figure 75.

If we sum all spherical deficit angles over all V vertices, we get

$$\sum_i k_i = 2\pi \times V - \pi \times F$$

where F is the number of triangle faces. For triangulated meshes,

$$3F = 2E$$

where E is the number of edges. Therefore, we get

$$\sum_i k_i = 2\pi (V - E + F) = 2\pi\chi .$$

A different proof can be found in [82]. Similar to simplex angle, we can also use the spherical deficit angle as the local curvature distribution for shape synthesis.

6.7 Summary

The spherical mesh representation, which has been introduced in the previous chapter, can be employed as a data structure for storing object shapes of genus zero, both free-form and polyhedral. Once the object shapes are resampled by our spherical mesh representation, and the local curvature at each mesh node is computed, the task of comparing two shapes is essentially one of comparing two curvature distributions generated from the resampled/ deformed meshes. An important observation is that the curvature distribution can be used to compare shapes efficiently and effectively.

The minimum distance function between curvature distributions can be used as a shape similarity metric. This shape metric is then used to analyze shape similarity between sets of concave and convex objects. Experiments show that our shape similarity metric is robust and invariant under rigid transformation and scaling, easy to compute, and intuitive with human perception on shape. Our approach is, in spirit, similar to the one used by Schwartz

and Sharir [104], in which they approximated a 2D curve from noisy data points by discretizing the turning function (a 2D curvature in some sense) of two polygons into many equally spaced points. We discretize the polyhedral and/or free-form surfaces into many approximately equally spaced patches. An advantage of our distance function is that it is stable under a certain amount of noise.

The task of synthesizing a morph from two original shapes is essentially one of interpolating two known curvature distributions of the deformed meshes generated from original shapes. Inverse mapping from the curvature distribution to shape has been shown to be possible. Our curvature-based interpolation yields smooth curvature migration along the morphing sequence. Experiments show that our shape synthesis creates realistic and intuitive shape morphs which show a gradual change between two originals.

Our study has been restricted to genus zero shape topology. Recent progress on geometrical heat equation and geometry diffusion shed some light on how to compare topological shape similarity as well as geometrical similarity. A better user interface can also be used to incorporate the user's specification for more realistic and specific shape synthesis. This kind of user-specified constraint can be easily implemented under our framework.

Chapter 7

Conclusions

This thesis studies modeling from reality and focuses, in particular, on two very important problems: representation and integration. These problems have been carefully examined by an integral approach to modeling from a sequence of range images. Using the integral approach we have reconstructed statistically optimal object models that are simultaneously most consistent with all the views. The integral approach has been applied to model both polyhedral objects and free-form objects.

7.1 An Integral Approach to Object Modeling -- A Summary

This thesis developed an integral approach to object modeling from a sequence of range images. The integral approach formulates modeling from multiple images as a problem of principal component analysis with missing data (PCAMD), and solves the problem with efficient algorithms. It has been argued that depending on the types of objects to be modeled, objects can be resampled with either planar surface patches, or a special spherical mesh representation. More specifically, the integral approach consists of two components: how to integrate and what to integrate.

7.1.1 How to integrate

How can we integrate multiple views of range images in a statistically optimal fashion? In other words, how can we build a model which is simultaneously most consistent with all the views? The answer is to formulate multiple view integration as a problem of principal component analysis with missing data, provided that correspondence among different images can be established. Based on Wiberg's formulation, the problem of principal component analysis with missing data has been generalized as a weighted-least-squares problem, and solved by an efficient bilinear iterative algorithm. The integral approach makes use of the significant redundancy existing among all the views in the formulation of PCAMD. Because

of the redundancy in the sequence of images, we can get a reliable solution from an over-constrained minimization problem even when data are missing.

7.1.2 What to Integrate

Then what correspondence can be established among multiple views? The prerequisite to applying PCAMD is that multiple views are resampled in such a way that they correspond to each other. The resampling process completely depends on the representations used for the objects of interest. For polyhedral objects, we resample them using planar surface patches. The polyhedral object is segmented and tracked over a sequence of images. Planar patch normals and distances to the origin are used for establishing the correspondence. For free-form objects, we present a novel global resampling scheme with a spherical mesh which can be used to determine correspondence among different views. In particular, we use a nearly-uniform spherical mesh with fixed connectivity to represent free-form objects. Local curvature at each mesh node is then used to build the correspondence.

7.1.3 Modeling System

The modeling system is made up of two parts: scene modeling and focused object modeling. For scene modeling, a boundary planar surface representation suffices. After a sequence of range images is taken, all images are segmented into planar patches. These patches are tracked over the sequence, and are then integrated into a consistent scene model using the PCAMD algorithm. Scene modeling can be used for the purpose of virtual reality. An example of modeling an indoor scene was presented.

Focused object modeling comprises both polyhedral modeling and free-form modeling. Polyhedral modeling is essentially the same as scene modeling. However, if the object is free-form, each range image is resampled with a special spherical mesh. Then all mesh nodes are tracked over the sequence of images before applying PCAMD to obtain a statistically optimal model. Both polyhedral modeling and free-form modeling have been implemented and examples have been given. Focused object modeling has been applied to model polyhedral objects such as a toy house, and free-form objects such as a toy dog.

The special spherical mesh representation for free-form objects, developed for resampling free-form surfaces, is also used to compare and synthesize shapes. The key observation is that the mesh representation acts as a proper coordinate system which allows shapes to be compared and interpolated by their curvature distributions.

7.2 Thesis Contributions

This thesis has described a modeling-from-reality system which reconstructs statistically optimal models from a sequence of range images. The integral approach presented in this thesis consists of a set of techniques for modeling both polyhedral and free-form objects.

The specific contributions of this thesis are:

1. Development of a framework for modeling objects and scenes from a sequence of images. An integral approach is introduced to combine multiple views so that a statistically optimal model is obtained. The integral approach consists of two key components: how to integrate and what to integrate.

2. Formulation of a weighted-least-squares approach to the problem of principal component analysis with missing data and its application to modeling from a sequence of images.

3. Introduction of a special spherical mesh representation used to resample free-form objects. The mesh representation enables one-to-one mesh node correspondence so that integration of multiple views becomes simple.

4. Implementation of the modeling-from-reality system which approximates the existing world as a combination of polyhedral and free-form objects. Applications of modeling from reality include virtual reality modeling, shape similarity and shape synthesis.

7.3 Future Work

This thesis has shown that a modeling from reality system is implementable. While significant progress has been made toward the development of an operational system for modeling reality with computer vision techniques, there remain some important issues to be explored to increase the applicability of our modeling system.

7.3.1 A Coarse-to-Fine Approach: Better Modeling for Fine Details

Due to the regularity constraints used in free-form object modeling, we have to make trade-offs in areas of high curvatures. This problem is particularly evident around the ears of the sharpei model. In the future, a coarse-to-fine approach could be applied to obviate this problem. One simple coarse-to-fine strategy is to use a ternary triangulation for each triangular

mesh. Unfortunately there is no simple relationship between any N frequency tessellation to N+1 tessellation.

7.3.2 Interactive Modeling and Synthesis

Most of the work described in this thesis focuses on the development of an automatic modeling system. However, as evident from the examples shown in this thesis, automatic systems are not perfectly functional in many real-life situations. Reddy [98] suggested that a good working system should be 90% automatic and 10% interactive with some assistance from humans. Most likely humans can do a much better job in finding the connectivity among planar patches.

Another place that interaction can play an interesting role is in the area of shape synthesis. Our shape synthesis framework can easily accommodate some interactive forces in the form of external forces. Under different circumstances, user-specified matches have higher priority than the implicit constraints imposed by mesh regularity. In this case, we must devise a way to resolve the conflicts.

7.3.3 Moving Beyond Geometry: Probe Topology

In this thesis, we have confined ourselves to considering only objects with spherical topology. Undoubtedly, in reality, many objects are not of genus zero. It is therefore necessary to investigate how multiple views of these objects can be integrated. One possibility is to start with a uniformly tessellated mesh of known genus such as a torus. An interesting observation is that an object of genus one can be regularly tessellated, in contrast to a spherical mesh which can only be semi-regularly tessellated if more than 20 mesh nodes are needed. Another possibility is to start with spherical topology and evolve adaptively [28][75][77][85].

7.3.4 Voxel-based approach

An alternative solution to the topology problem is to use volume resampling of the object. Volume sampling is obtained by assigning a small voxel size to each range data point. To use the voxel-based approach we need to start with a good initial transformation. For each voxel, we find its closest point on each surface (view). Then we form a measurement matrix of all surface normals on each voxel. After we have recovered the rotation matrices at every view, we can apply the rotation matrix to all points around their centroid, and create a second measurement matrix with the positions of all voxels. From the second measurement matrix we can recover the translation. A dynamic PCAMD algorithm can be applied to

update the voxel positions and normals. Finally, a marching cube algorithm [73] can be applied to obtain a surface model.

7.3.5 Structure from Motion

Many ideas presented in this thesis are applicable to modeling from a sequence of intensity images. Structure from motion, for instance, may use PCAMD to find a statistically optimal solution to merge multiple images. Another interesting topic is how to combine the photometric information with geometrical information to obtain more accurate and realistic models.

Appendix A

Proof of Shape Similarity Metric

This appendix shows why the minimum distance between two curvature distributions of two resampled free-form objects is a metric.

We define the L_p distance $d_p(S_A, S_B, R)$ between A and B at a certain spherical rotation R as

$$d_p(S_A, S_B, R) = \left(\int |k_I(S_A) - k_R(S_B)|^p dS \right)^{\frac{1}{p}}$$

which is the sum of curvature differences over the sphere. Then the distance function between A and B, $D_p(A, B)$ becomes

$$D_p(A, B) = min_R \; d_p(S_A, S_B, R)$$

which is minimized d_p over all possible rotations R.

Property 1: $D_p(A, B)$ **is a metric for all** $p>0$.

Proof:

Because d_p is a L_p norm, we have

- D_P is positive. $D_p(A, B) \geq 0$;
- D_P is identity. $D_p(A, A) = 0$;
- D_P is symmetric. $D_p(A, B) = D_p(B, A)$.

The only thing left to prove is the triangle inequality $D_p(A, B) + D_p(B, C) \geq D_p(A, C)$.

Let R_1 and R_2 be the rotation matrices which minimize the $D_p(A, B)$ and $D_p(B, C)$, respectively,

$$D_p(A,B) + D_p(B,C)$$

$$= min_{R_1}\ d_p(S_A, S_B) + min_{R_2}\ d_p(S_B, S_C)$$

$$= min_{R_1}\ \left(\int \left|k_I(S_A) - k_{R_1}(S_B)\right|^p dS\right)^{\frac{1}{p}} + min_{R_2}\ \left(\int \left|k_I(S_B) - k_{R_2}(S_C)\right|^p dS\right)^{\frac{1}{p}}$$

$$\geq min_{R_1, R2}\left(\left(\int \left|k_I(S_A) - k_{R_1}(S_B)\right|^p dS\right)^{\frac{1}{p}} + \left(\int \left|k_{R_1}(S_B) - k_{R_1^{-1}R_2}(S_C)\right|^p dS\right)^{\frac{1}{p}}\right)$$

$$= min_{R_1, R_2}\ \left(d_p(S_A, S_B, R_1) + d_p(S_B, S_C, R_2)\right)$$

$$\geq min_{R_3}\ d_p(S_A, S_C, R_3)$$

$$= D_p(S_A, S_C)$$

where $R_3 = R_1^{-1}R_2$.

Appendix B

Calibrating The Rotary Table

B.1 Calibration Setup

In order to gain the true transformation between two different views, we calibrate the rotary table using a known geometry cube with grid pattern as shown in Figure 76. Each range image can then be segmented into 3 planar patches. The rotary table is rotated by an angle θ around its rotation axis k which is located on q from the origin (Figure 77). A point x_1 in view 1 can be transformed to x_2 in view 2 by

$$x_2 = R(x_1 - q) + q, \qquad \text{(EQ B.1)}$$

or

$$x_2 = Rx_1 + t \qquad \text{(EQ B.2)}$$

where

$$t = q - Rq. \qquad \text{(EQ B.3)}$$

B.2 Calibration as a Minimization Problem

With the above representation (k, q, θ), we show that calibrating the rotary table can be formulated as a minimization problem. Range data of two views of the calibration box is obtained and their segmented planes are also matched. A plane is represented by a vector v and a scalar d with the equation $v^T x - d = 0$, or (v, d) in short. The transformed plane (v', d') of (v, d) is given by

$$v' = Rv, \; d' - d = t^T v' = q^T (v' - v). \qquad \text{(EQ B.4)}$$

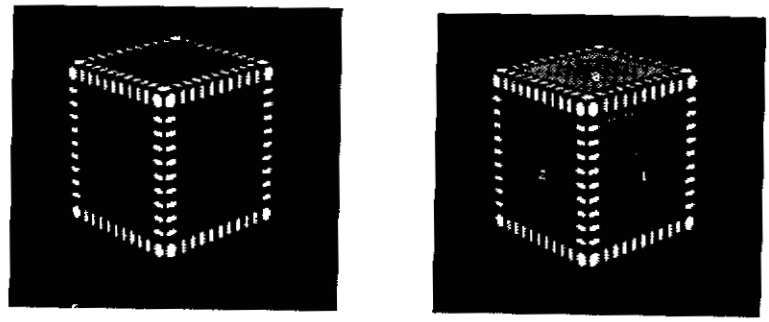


Figure 76 **Calibration cube and its segmented patches**

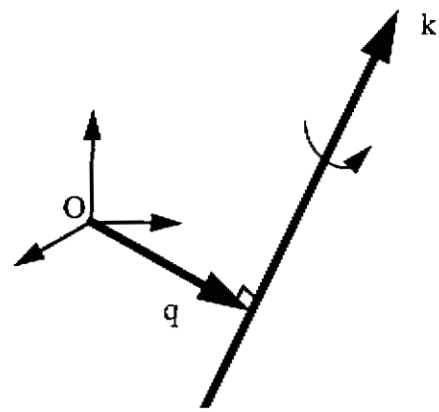


Figure 77 **Representation of a rotation: rotation axis k located at q**

The calibration problem becomes

$$\min \sum_{i=1}^{N} \left\| v_i' - R v_i \right\|^2, \qquad \text{(EQ B.5)}$$

and

$$\min \sum_{i=1}^{N} \left(d_i' - d_i - q^T (v_i' - v_i) \right)^2 \qquad \text{(EQ B.6)}$$

$$\text{subject to} \quad q^T k = 0, \qquad \text{(EQ B.7)}$$

where $N = 3$ for the calibration cube. The problem (EQ B.5) can be solved using quaternion [37]. We can then compute (k, θ) from known R. We can reformulate (EQ B.6) and (EQ B.7) as

$$\begin{bmatrix} v_1' - v_1 \\ v_2' - v_2 \\ v_3' - v_3 \\ k \end{bmatrix} q = \begin{bmatrix} d_1' - d_1 \\ d_2' - d_2 \\ d_3' - d_3 \\ 0 \end{bmatrix} \qquad \text{(EQ B.8)}$$

whose solution is straightforward.

B.3 Accurate, Fast and Automatic Calibration

To increase the accuracy of calibration, the above minimization problem can be made more over-constrained if we take a sequence of range images (more than two) of the calibration cube with equal angular increments (e.g. 2° or 5°). It is possible that we segment the range image of each view, then track three planes over the whole sequence automatically. However, the segmentation program itself is rather slow. To automate the rotary table calibration process, we have adopted 3DTM, a robust three-dimensional pose estimation technique, developed by Wheeler and Ikeuchi [130]. The cube is modelled as three planes and its pose can be accurately estimated at different views. A process of 3DTM pose estimation is shown in Figure 78.

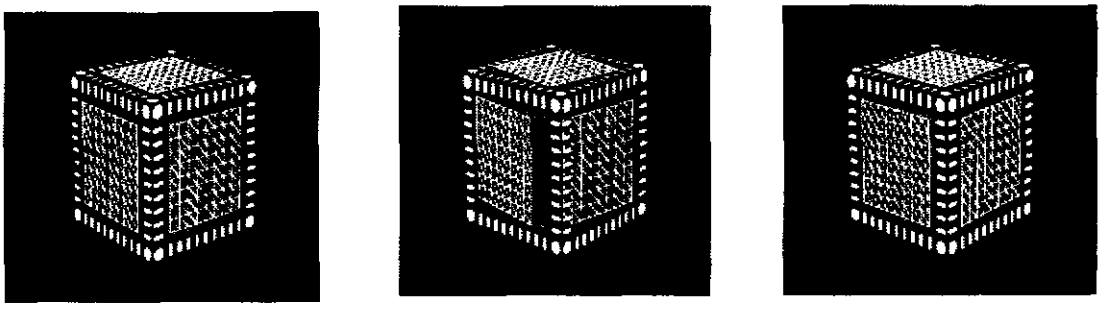


Figure 78 **(a) First image overlaid with cube model (solid lines); (b) Second image overlaid with model at initial pose; (c) Second image overlaid with model after 3DTM pose estimation**

Bibliography

[1] N. Ahuja and J. Veenstra. Generating Octrees from Object Silhouettes in Orthographic Views. *IEEE Trans. PAMI.* Vol. 11 No. 2, pp. 137-149, 1989.

[2] E. Arkin, L. Chew, D. Huttenlocher, K. Kedem and J. Mitchell. An Efficient Computable Metric for Comparing Polygonal Shapes. *IEEE Trans. PAMI.* Vol. 13, No. 3, pp. 209-215, 1991.

[3] F. Arman and J.K. Aggarwal. Model-based Object Recognition in Dense-Range Images - A Review. *ACM Computing Surveys.* Vol. 25, No. 1, pp. 5-43, 1993.

[4] N. Ayache. *Artificial Vision for Mobile Robots - Stereo-vision and Multisensor perception.* The MIT Press. 1981.

[5] A. Azarbayejani, B. Horowitz and A. Pentland. Recursive Estimation of Structure and Motion using Relative Orientation Constraints. *Proc. IEEE CVPR.* pp. 294-299, New York, June, 1993.

[6] R. Basri, L. Costa, D. Geiger and D. Jacobs. Determining the Similarity of Deformable Shapes. *Proc. of Workshop on Physics-based Modeling in Computer Vision.* pp. 135-143, Boston, June 18-19, 1995.

[7] T. Beier and S. Neely. Feature-based Image Metamorphosis. *Proc. SIGGRAPH.* pp. 35-42, July, 1992.

[8] R. Bergevin, D. Laurendeau and D. Poussart. Registering Range Views of Multipart Objects. *Computer Vision and Image Understanding.* Vol. 61, No. 1, pp. 1-16, January, 1995.

[9] P. Besl and R. Jain. Segmentation Through Variable-Order Surface Fitting. *IEEE Trans. PAMI.* Vol. 10, No. 2, pp. 239-256, 1988.

[10] P. Besl and N.D. Kay. A Method for Registration of 3-D Shapes. *IEEE Trans. PAMI.* Vol. 14, No. 2, pp. 239-256, 1992.

[11] B. Bhanu. Representation and Shape Matching of 3-D Objects. *IEEE Trans. PAMI.* Vol. 6, pp. 340-351, 1984.

[12] A. Blake and A. Zisserman. *Visual Reconstruction*. The MIT Press. 1987.

[13] V. M. Bove. Probabilistic Method for Integrating Multiple Sources of Range Data. *Journal of the Optical Society of America A*. Vol. 7, No. 12, December, 1990.

[14] C. Brechbuhler, G. Gerig, and O. Kubler. Parametrization of Closed Surfaces for 3D Shape Description. *Computer Vision and Image Understanding*. Vol. 61, No. 2, pp. 154-170, March, 1995.

[15] G. Champleboux, S. Lavallee, R. Szeliski and L. Brunie. From Accurate Range Imaging Sensor Calibration to Accurate Model-Based 3-D Object Localization. *Proc. IEEE CVPR*. pp. 83-89, June, 1992.

[16] S. Chen. QuickTime VR - An Image-Based Approach to Virtual Environment Navigation. *Proc. SIGGRAPH'95*. pp.29-38.

[17] S. Chen and L. Williams. View Interpolation for Image Synthesis. *Proc. SIGGRAPH'93*. pp. 279-288.

[18] Y. Chen and G. Medioni. Object Modeling by Registration of Multiple Range Images. *Proc. IEEE Int. Conf. R & A*. pp. 2724-2729, April, 1991.

[19] Y. Chen and G. Medioni. Surface Description of Complex Objects from Multiple Range Images. *Proc. IEEE CVPR*. pp. 153-158, June, 1994.

[20] R. Chin and C. Dyer. Model-based Recognition in Robot Vision. *ACM Computing Surveys*. Vol. 18, No. 1, 1986.

[21] W. Choi. *Computational Analysis of Three Dimensional Measurement DATA*. Ph.D Thesis, Department of Mechanical Engineering, Carnegie Mellon University, 1996.

[22] C. Chua and R. Javis. 3D Free-form Surface Registration and Object Recognition. To appear in *IJCV*.

[23] C. Chua and R. Javis. Point Signatures: A New Representation for 3D Object Recognition. To appear in *IJCV*.

[24] L. Cohen and I. Cohen. Finite Element Methods for Active Contour Models and Balloons from 2D to 3D. *IEEE Trans. PAMI*. Vol. 15, No. 11, pp. 1131-1147, November, 1993.

[25] Cyberware Laboratory Inc. *4020/RGB 3D Scanner with Color Digitizer*. Monterey, 1990.

[26] P. Debevec, C. Taylor and J. Malik. *Modeling and Rendering Architecture from Photographs*. Technical report UCB/CSD-96/893.

[27] C. Debrunner and N. Ahuja. Motion and Structure Factorization and Segmentation of Long Multiple Motion Image Sequences. *Proc. ECCV.* pp. 217-221, 1992.

[28] D. DeCarlo and D. Metaxas. Adaptive Shape Evolution using Blending. *Proc. ICCV.* pp. 834-839, 1995.

[29] H. Delingette, M. Hebert and K. Ikeuchi. A Spherical Representation for the Recognition of Curved Objects. *Proc. ICCV.* Berlin, 1993.

[30] H. Delingette. *Simplex Meshes: a General Representations for 3D Shape Reconstruction.* INRIA Report 2214, 1994.

[31] H. Delingette, H. Watanabe and Y. Suenaga. *Simplex Based Animation.* Tech report NTT Human Interface Lab. Japan, 1994.

[32] D. Dobkin, L. Guibas, J. Hershberger and J. Snoeyink. An Efficient Algorithm for Finding the CSG Representation of a Simple Polygon. *Algorithmica.* Vol 10, pp. 1-23, 1993.

[33] M. DoCarmo. *Differential Geometry of Curves and Surfaces.* Prentice-Hall. 1976.

[34] Y. Dodge. *Analysis of Experiments with Missing Data.* Wiley. 1985.

[35] N. A. Dodgson. *Image Resampling.* Ph.D. thesis, University of Cambridge, Computer Laboratory, 1992.

[36] O.D. Faugeras. *Three-Dimensional Computer Vision.* The MIT Press. 1992.

[37] O.D. Faugeras and M. Hebert. The Representation, Recognition, and Localization of 3-D Objects. *Int. J. Robotics Research.* Vol. 5, No. 3, pp. 27-52, 1986.

[38] O.D. Faugeras, S. Laveau, L. Robert, G. Csurka and C. Zeller. *3D Reconstruction of Urban Scenes from Sequences of Images.* INRIA Report 2572, 1995.

[39] F.P. Ferrie and M.D. Levine. Integrating Information from Multiple Views. *Proc. IEEE Workshop on Computer Vision.* pp. 117-122, 1987.

[40] H. Gagnon, M. Soucy, R. Bergevin and D. Laurendeau. Registration of Multiple Range Views for Automatic 3D Model Building. *Proc. IEEE CVPR.* pp. 581-586, 1994.

[41] S. Ghosal and R. Mehrotra. Segmentation of Range Images: An Orthogonal Moment-based Integrated Approach. *IEEE Trans. Robotics and Automation.* Vol. 9, No. 4, pp. 385-399, August, 1993.

[42] G.H. Golub and C.F. Van Loan. *Matrix Computation.* 2nd Edition. The Johns Hopkins University Press. 1989.

[43] J. Hancock and C. Thorpe. *ELVIS: Eigenvectors for Land Vehicle Image System.* CMU-RI-94-43, 1994.

[44] T. He, S. Wang and A. Kaufman. Wavelet-based Volume Morphing. *Proc. Visualization'94.* pp. 85-91, 1994.

[45] M. Hebert, K. Ikeuchi and H. Delingette. A Spherical Representation for Recognition of Free-form Surfaces. *IEEE Trans. PAMI.* Vol. 17, No. 7, pp. 681-690, 1995.

[46] P. Heckbert. Survey of Texture Mapping. *IEEE Computer Graphics and Applications.* pp. 56-67, November, 1986.

[47] P. Heckbert and M. Garland. Multiresolution Modeling for Fast Rendering. *Graphics Interface'94.*

[48] K. Higuchi, O. Ozeki, S. Yamamoto and N. Takahashi. Profile Measuring System for 3-D Object using Newly Developed Vision Sensor, *IEEE Denshi Tokyo.* No. 29, pp. 167-169 , 1990.

[49] K. Higuchi, M. Hebert and K. Ikeuchi. *Building 3-D Models from Unregistered Range Images.* CVGIP-GMIP, Vol. 57, No. 4, pp. 315-333, July, 1995.

[50] K. Higuchi, H.Y. Shum, M. Hebert and K. Ikeuchi. Combining Shape and Color Information for Object Recognition. To appear in *IJCV 1996.*

[51] D. Hilbert and S. Cohn-Vossen. *Geometry and the Imagination.* Chelsea Publishing Company. 1952.

[52] H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schweitzer and W. Stuetzle. Piecewise Smooth Surface Reconstruction. *Computer Graphics SIGGRAPH.* pp. 295-302, 1994.

[53] B.K.P. Horn. Extended Gaussian Image. *Proc. of IEEE.* Vol. 72, No. 12, pp. 1671-1686, December, 1984.

[54] B.K.P. Horn. *Robot Vision.* The MIT Press. 1986.

[55] J. Hughes. Scheduled Fourier Volume Morphing. *Proc. SIGGRAPH.* pp. 43-46, July, 1992.

[56] D. Huttenlocher and K. Kedem. Computing the Minimum Hausdorff Distance for Point Sets Under Translation. *Proc. ACM Symp. Computational Geometry.* pp. 340-349, 1990.

[57] K. Ikeuchi. Recognition of 3D Object using the Extended Gaussian Image", *Proc. 7th IJCAI,* pp. 595-600, 1981.

[58] K. Ikeuchi. Generating an Interpretation Tree from a CAD Model for 3D-Object Reconstruction in Bin-Picking. *Int. J. Computer Vision.* pp. 145-165, 1987.

[59] K. Ikeuchi and M. Hebert. Task-oriented Vision. *Exploratory Vision* (Editted by M. Landy et al.), Springer, 1996.

[60] K. Ikeuchi and M. Hebert. *From EGI to SAI.* CMU-CS-95-197. 1995.

[61] A. Johnson and M. Hebert. *Recognizing Objects by Matching Oriented Points.* CMU-RI-TR-96-04, May 1996.

[62] R. Johnson and D. Wichern. *Applied Multivariate Analysis.* Prentice-Hall. 1982.

[63] I.T. Jolliffe. *Principal Component Analysis.* Springer-Verlag. 1986.

[64] S. Kang and K. Ikeuchi. The Complex EGI: New Representation for 3D Pose Determination. *IEEE Trans. PAMI.* Vol. 15, No. 7, pp. 707-721, July, 1993.

[65] S. Kang, J. Webb, C. Zitnick and T. Kanade. *An Active Multibaseline Stereo System with Real-time Image Acquisition.* CMU-CS-94-167. 1994.

[66] M. Kass, A. Witkin and D. Terzopoulos. Snakes: Active Contour Models. *Int. J. Computer Vision.* Vol. 1, No. 4, pp. 321-331, 1987.

[67] Y. Kawai, T. Ueshiba, T. Yoshimi and M. Oshima. Reconstruction of 3D Objects by Integration of Multiple Range Data. *Proc. 11th ICPR.* pp. 154-157, September, 1992.

[68] J. Koenderink. *Solid Shape.* The MIT Press. Cambridge, 1990.

[69] K. Kupeev and H. Wolfson. On Shape Similarity. *Proc. Int. Conf. Pattern Recognition.* pp. 227-237, 1994.

[70] S. Laveau and O. Faugeras. *3D Scene Representation as a Collection of Images and Fundamental Matrices.* Inria Report 2205, 1994.

[71] A. Lerios, C. Garfinkle and M. Levoy. Feature-based Volume Metamorphosis. *Proc. SIGGRAPH.* pp. 449-456, August, 1995.

[72] J. Little. Determining Object Attitude for Extended Gaussian Image. *Proc. IJCAI.* Los Angeles, California, pp. 960-963, August, 1985.

[73] W. Lorensen and H. Cline. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. *Proc. SIGGRAPH.* pp. 163-169, July, 1987.

[74] D. Marr. *Vision.* W.H. Freeman and Company. 1982.

[75] R. Malladi, J. Sethian and B.C. Vemuri. Shape Modeling with Front Propagation: A Level Set Approach. *IEEE Trans. PAMI.* Vol. 17, No. 2, pp. 158-175, February, 1995.

[76] T. McInerney and D. Terzopoulos. A Finite Element Model for 3D Shape Reconstruction and Nonrigid Motion Tracking. *Proc. Fourth Int. Conf. Computer Vision.* pp. 518-523, 1993.

[77] T. McInerney and D. Terzopoulos. Topologically Adaptable Snakes. *Proc. Fifth Int. Conf. Computer Vision.* pp. 840-845, 1995.

[78] T. McInerney and D. Terzopoulos. Deformable Models in Medical Image Analysis: A Survey. 1996. *Medical Image Analysis* 96/1.

[79] M. L. Merriam. Experience with the Cyberware 3D Digitizer. *Proc. of the National Computer Graphics Association.* March 9-12, 1992, Anaheim, CA.

[80] L. McMillan and G. Bishop. Plenoptic Modeling: An Imaged-Based Rendering System. *Proc. SIGGRAPH'95.* pp. 39-46.

[81] T. Morita and T. Kanade. *A Sequential Factorization Method for Recovering Shape and Motion from Image Streams.* CMU-CS-94-158, May, 1994.

[82] M. Mortenson. *Geometric Modeling.* John Wiley and Sons. 1985.

[83] D. Mumford. Mathematical Theories of Shape: Do They Model Perception? *SPIE Vol. 1570 Geometric Methods in Computer Vision.* pp. 2-10, 1991.

[84] J. Mundy. *Object Recognition: The Search for Representation.* In Lecture Notes in Computer Science 994. Springer. 1995.

[85] S. Muraki. Volumetric Shape Description of Range Data using Blobby Model. *Proc. SIGGRAPH*, pp. 227-235, 1991.

[86] H. Murase and S. Nayar. Learning and Recognition of 3D Objects from Appearance. *Int. J. Computer Vision.* pp. 1-24, January, 1995.

[87] V. Nalwa. Representing Oriented Piecewise C2 Surfaces. *Proc. 2nd ICCV.* pp. 40-51, December, 1988.

[88] S. Nayar, M. Watanabe and M. Noguchi. Real-time Focus Range Sensor. *Proc. ICCV.* pp. 995-1001, 1995.

[89] M. Okutomi and T. Kanade. A Multiple-baseline Stereo. *Proc. IEEE CVPR.* pp. 63-69, 1991.

[90] OpenGen Inc. *OpenFlight Format Specification.* 1992.

[91] B. Parvin and G. Medioni. B-rep from Unregistered Multiple Range Images. *Proc. IEEE Int. Conf. R & A.* pp. 1602-1607, May, 1992.

[92] X. Pennec and J-P. Thirion. Validation of 3-D Registration Methods based on Points and Frames. *Proc. ICCV.* pp. 557-562, June, 1995.

[93] C.J. Poelman and T. Kanade. *A Paraperspective Factorization Method for Shape and Motion Recovery.* CMU-CS-93-219, December, 1993.

[94] A. Pope. *Model-Based Object Recognition: A Survey of Recent Research.* University of British Columbia, Department of Computer Science, TR-94-04, January, 1994.

[95] I. Porteous. *Geometrical Differentiation.* Cambridge University Press. 1994.

[96] F.P. Preparata and M.I. Shamos. *Computational Geometry.* Springer-Verlag. 1988.

[97] W. Press, S. Teukolsky, W. Vetterling and B. Flannery. *Numerical Recipes in C.* 2nd Edition. Cambridge University Press. 1992.

[98] R. Reddy. The Automated Machine Shop Project. *CMU CS Faculty Research Guide.* 1993.

[99] M. Rioux. Laser Range Finder based on Synchronized Scanners. *Applied Optics.* Vol. 23, No. 21, pp. 3837-3844, 1984.

[100] P. Rousseeuw and A. Leroy. *Robust Regression Outlier Detection.* John Wiley & Sons. New York. 1987.

[101] A. Ruhe. *Numerical Computation of Principal Components when Several Observations are Missing.* Tech Rep. UMINF-48-74, Dept. Information Processing, Umea Univ., Umea, Sweden. 1974.

[102] M. Rutishauser, M. Stricker and M. Trobina. Merging Range Images of Arbitrarily Shaped Objects. *Proc. IEEE CVPR.* pp. 573-580, 1994.

[103] K. Sato and S. Inokuchi. Range-imaging System Utilizing Nematic Liquid Crystal Mask. *Proc. ICCV.* 1987.

[104] J. Schwartz and M. Sharir. Identification of Partially Obscured Objects in Two and Three Dimensions by Matching Noisy Characteristic Curves. *Int. J. Robotics Research.* Vol. 6, No. 2, pp. 29-44, Summer 1987.

[105] S. Sclaroff and A. Pentland. Object Recognition and Categorization Using Modal Matching. *Proc. 2nd CAD-Based Vision Workshop.* pp. 258-265. Champion, Pennsylvania. Feb. 8-11, 1994.

[106] H. Shum, K. Ikeuchi and R. Reddy. Principal Component Analysis with Missing Data and Its Application to Polyhedral Object Modeling. *IEEE Trans. PAMI.* Vol. 17 No. 9, pp. 854-867, 1995.

[107] H. Shum, K. Ikeuchi and R. Reddy. Virtual Reality Modeling from a Sequence of Range Images. *Proc. IROS'94.* October, 1994.

[108] H. Shum, M. Hebert, K. Ikeuchi and R. Reddy. *An Integral Approach to Free-form Object Modeling.* CMU-CS-95-135, May, 1995. Also appeared in *Proc. ICCV*'95. pp. 870-8715, 1995.

[109] H. Shum, M. Hebert and K. Ikeuchi. *On 3D Shape Similarity.* CMU-CS-95-212, November, 1995. Also appeared in *Proc. IEEE CVPR*'96. pp. 526-531, 1996.

[110] H. Shum, M. Hebert and K. Ikeuchi. *On 3D Shape Synthesis.* CMU-CS-95-213, November, 1995. Also appeared in *Proc. ECCV workshop on Object Representation in Computer Vision*, 1996.

[111] H. Shum and K. Turkowski. View Interpolation with Projective Motion Estimation. Apple Computers, Inc. US Patent pending.

[112] M. Soucy and D. Laurendeau. Multi-Resolution Surface Modeling from Multiple Range Views. *Proc. IEEE CVPR.* pp. 348-353, 1992.

[113] K. Sugihara. *Machine Interpretation of Line Drawings.* The MIT Press. 1986.

[114] S. Sullivan, L. Sandford and J. Ponce. Using Geometric Distance Fits for 3D Object Modeling and Recognition. *Proc. IEEE CVPR*'93. pp. 110-115, 1993.

[115] R. Szeliski. Video mosaics for virtual environments. *IEEE Computer Graphics and Applications.* pp. 22-30, March 1996.

[116] R. Szeliski and D. Tonnesen. Surface Modeling with Oriented Particle Systems. *Proc. SIGGRAPH.* pp. 185-194, July, 1992.

[117] R. Szeliski and S.B. Kang. *Recovering 3D Shape and Motion from Image Streams using Non-linear Least Squares.* DEC CRL 93/3, 1993.

[118] R. Szeliski and H. Shum. Motion Estimation with Quadtree Splines. To appear in *IEEE Trans. PAMI.* 1996.

[119] G. Taubin. Curve and Surface Smoothing Without Shrinkage. *Proc. ICCV.* pp. 852-857, 1995.

[120] G. Taubin. Estimating the Tensor of Curvature of Surface from a Polyhedral Approximation. *Proc. ICCV.* pp. 902-907, 1995.

[121] D. Terzopoulos. Regularization of Inverse Visual Problems Involving Discontinuities. *IEEE Trans. PAMI.* Vol. 8, No. 4, pp. 413-424, July, 1986.

[122] D. Terzopoulos, A. Witkin and M. Kass. Symmetry-Seeking Models and 3D Object Reconstruction. *Int. J. Computer Vision.* Vol. 1, No. 1, pp. 211-221, 1987.

[123] C. Tomasi and T. Kanade. Shape and Motion from Image Streams under Orthography: A Factorization Method. *Int. J. Computer Vision.* Vol. 9, No. 2, pp. 137-154, 1992.

[124] G. Turk and M. Levoy. Zippered Polygon Meshes from Range Images. *Computer Graphics SIGGRAPH.* pp. 311-318, 1994.

[125] A. Tversky. Features of similarity. *Psychological Review.* Vol. 84, No. 4, pp. 453-462.

[126] S. Ullman. *The Interpretation of Visual Motion.* The MIT Press. 1979.

[127] M. Vasilescu and D. Terzopoulos. Adaptive Meshes and Shells. *Proc. IEEE CVPR.* pp. 829-832, 1992.

[128] B.C. Vemuri and J.K. Aggarwal. 3-D Model Construction from Multiple Views Using Range and Intensity Data. *Proc. IEEE CVPR.* pp. 435-437, 1986.

[129] N. Wada, H. Toriyama, H. Tanaka and F. Kishino. Reconstruction of an Object Shape from Multiple Incomplete Range Data Sets Using Convex Hulls. *Computer Graphics International.* pp. 193-203, 1993.

[130] M. Wheeler and K. Ikeuchi. Sensor Modeling, Probabilistic Hypothesis Generation, and Robust Localization for Object Recognition. *IEEE Trans. PAMI.* Vol. 17, No. 3, pp. 252-265, March, 1995.

[131] T. Wiberg. Computation of Principal Components when Data are Missing. *Proc. Second Symp. Computational Statistics.* pp. 229-236, Berlin, 1976.

[132] A. Witkin and P. Heckbert. Using Particles to Sample and Control Implicit Surfaces. *Proc. SIGGRAPH.* pp. 269-278, July, 1994.

[133] Z. Zhang. *Iterative Point Matching for Registration of Free-formed Curves and Surfaces.* Research Report, INRIA, March, 1992.