This is to certify that the

dissertation entitled

COSMOS: A FRAMEWORK FOR REPRESENTATION AND RECOGNITION OF 3D FREE-FORM OBJECTS

presented by

Chitra Dorai

has been accepted towards fulfillment
of the requirements for

PhD degree in Computer Science

Anil Kumar Jr
Major professor

Date 6/28/96

COSMOS: A FRAMEWORK FOR REPRESENTATION AND RECOGNITION OF 3D FREE-FORM OBJECTS

By

Chitra Dorai

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

Department of Computer Science

1996

ABSTRACT

COSMOS: A FRAMEWORK FOR REPRESENTATION AND RECOGNITION OF 3D FREE-FORM OBJECTS

By

Chitra Dorai

This dissertation presents a new approach to automated representation and recognition of 3D free-form rigid objects using dense surface data. We describe a computer vision system that recognizes arbitrarily curved 3D rigid objects from a single view when (a) the viewpoint can be arbitrary, (b) the objects may vary in shape and complexity, and (c) no restrictive assumptions are made about the types of surfaces on the objects. We assume that a range image of a scene is available which contains a view of a rigid 3D object without occlusion. Availability of CAD models of 3D objects, although not a necessity, is considered an advantage and exploited in our system to easily generate multiple views of the object for its model construction. Our surface representation scheme, COSMOS, describes an object concisely in terms of maximal surface patches of constant *shape index*. These maximal patches that represent the object are mapped onto the unit sphere via their orientations, and aggregated via *shape spectral functions*. Surface properties such as area, curvedness, and connectivity which are required to capture local and global information about the object are also built into the representation. The scheme yields not only a meaningful and rich description useful for the recoverability of several classes of objects, but also provides a set of powerful matching primitives for recognition.

We present a recognition strategy which consists of a multi-level matching mechanism employing shape spectral analysis and features derived from the COSMOS representations of objects for both fast and efficient object identification and pose esti-

mation. Shape spectra based object view representations are employed for efficient view grouping and model organization with large databases. Given a range image of an uncluttered view (allowing self-occlusion) of an object, the shape spectrum based model selection scheme short-lists a few promising candidate views from a database of object views. The COSMOS-based view verification scheme then establishes the correct object identity of the input by comparing the COSMOS representations of the views in detail using a "patch-group graph" matching technique. Estimation of the pose of the recognized object is formulated as registration of the sensed data with the range image of the best matched view of the object. We present a minimum variance estimator to robustly register two range images of a complex object and compute their relative view transformation accurately. All theoretical aspects of this work have been experimentally validated via a prototype system, which has been tested on a database of over 6,000 object views generated from CAD models and surface triangulations and 100 range images of several different complex objects acquired using a 3D laser range scanner.

To ***Sunil***

Acknowledgments

It is a great pleasure to record my sincere appreciation of the support and encouragement shown by my colleagues, friends and family during the course of my Ph. D. First of all, I wish to express my gratitude to Prof. Anil K. Jain, my thesis advisor for his generous support, guidance and willingness to help throughout my research. I have benefited greatly from the extensive research training and opportunities he has provided me, and from his breadth and depth of experience. His insightful questions and remarks on the significance of various aspects of my research have influenced and improved the presentation of my work immensely. His vision and the care he gives to all his students have been a source of inspiration to me.

I would like to thank Profs. George Stockman, John Weng, William Punch and Hira Koul for serving on my Ph. D committee. Special thanks to Dr. Stockman for his steady encouragement and for always keeping an open door for me to discuss research with him at any time throughout these years. Right from the moment I arrived at MSU, he has shown me a great deal of warmth and friendliness which I truly appreciate. I am very grateful to Dr. Weng for his helpful suggestions and constructive criticism that contributed significantly to Chapter 6 of this thesis. Thanks to Dr. Punch for his cheerful disposition and his willingness to make time for our discussions. Dr. Koul initially suggested the analogy between the shape spectral distribution and the probability distributions. I also want to thank all the faculty members from Computer Science, Mathematics and Statistics departments, especially late Prof. Dubes for the excellent instruction that I received. Dr. Patrick Flynn at WSU played a key role in the early development of my interest in free-form sur-

face matching. I would like to thank both Pat Flynn and Tim Newman for sharing their 3D data and source code, and for their prompt assistance whenever I turned to them with questions about the laser range scanner. I also gratefully acknowledge Dr. S.W. Chen's help with the cobra model.

My PRIP friends and colleagues from all over the world contributed to a stimulating and friendly work environment, full of lively discussions and cheerful banter. I would like to thank all of them, in particular Sateesha Nadabar, N. S. Raja, Deborah Trytten, Sushil Bhattacharjee, Timothy Newman, Hansye Dulimarta, Qian Huang, Jianchang Mao, Marie Pierre Jolly, Jinlong Chen, Dan Swets, Sally Howden, Marilyn Wulfekuhler, Shanti Vedula, Aditya Vailaya, Yonghong Li, Karissa Miller, Lin Hong, and Gang Wang. Special thanks to Jayashree, Ram, Geeta, Natraj, Sujatha, and Shankar for the moral support and fun times outside school. Many old friends such as Madan and Shreekumar often sent their affection and encouragement via e-mail.

I am grateful to Innovision Corporation for a Graduate Fellowship award, MSU Graduate School for the Dissertation Completion Fellowship, Northrop Corporation, NASA Lewis Research Center and the Department of Computer Science for their generous financial support over the years. I am thankful to Dr. Anbalagan Reddiar at Innovision Corporation, Dr. Harpreet Sawhney and Dr. Byron Dom at the IBM Almaden Research Center for their interest. I wish to thank all the system administrators in the department, especially Lisa Lees for maintaining excellent computing facilities in PRIP and instructional laboratories. I appreciate the enthusiastic helpfulness of the departmental support staff, particularly Cathy Davison, Linda Moore, and Lora Mae Higbee at all times.

Special thanks to my parents and brothers whose encouragement, love and noninterference made this all possible. My mother in particular, was a source of inspiration to me in many aspects. I am also most thankful for Sunil's unfailing support and love over the last eleven years, from IIT to MSU. I am indebted to him for serving as a (very!) critical sounding board at many points during my research, for his generous sacrifice of our personal time to accommodate long discussions on COSMOS and finally for being always there. His brilliance and his keen sense of perfection have influenced

this work significantly. I dedicate this thesis to him with deep love and gratitude.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

> ...Today's small piece is basically about arbitrary smooth lumps of three-dimensional space such as could be occupied by your typical potato or favorite torso, say... [Jan J. Koenderink, "Solid Shape"]

One of the major goals of computer science is to build machines that mimic human capabilities. Towards realizing this objective, research in building intelligent systems has traditionally concentrated on advanced cognitive skills such as reasoning, problem solving, and natural language understanding. However, a crucial component of intelligent behavior is the ability to sense and affect the world. The field of computer vision, which studies perception and how it can be combined with action is, therefore, an important adjunct to constructing intelligent machines. The growing importance of computer vision is evident from the fact that it was identified as one of the "Grand Challenges" [57] and also from its prominent role in the National Information Infrastructure [52]. The primary goal in computer vision is to build a system that can automatically *interpret* a scene, given a snapshot (image) of the scene in terms of an array of brightness or depth values. While human vision is an existence proof of a system that operates flexibly in multiple environments, computer vision research so far has attempted to deal separately with images of outdoor and indoor scenes.

1.1 Automatic Model-Based Scene Analysis

Our interest lies in building systems to automatically interpret images of a scene, primarily in an industrial setting. We define a scene as consisting of one or more 3D man-made objects. An *interpretation* of an image (data) of a scene is defined as knowing *which* 3D objects are *where* in the scene. This interpretation conceptually binds the entities in the scene to objects that we already have knowledge about. Thus, we are dealing with *model-based* scene analysis. Deriving an interpretation of a scene involves solving two interrelated problems. The first is *identification* or *classification*, where a label is assigned to an object to indicate the category to which it belongs. The second problem involves the *estimation* of the *pose* (position and orientation) or *localization* of the recognized object with respect to some global coordinate system attached to the scene. The term "recognition" is used in computer vision to describe the entire process of automatic identification and localization of objects from the sensed images of scenes in the real world.

Automatic scene analysis is indeed a difficult problem as a recognition system has to make sense out of the pixels of an image array which by themselves contain very little information. Some knowledge of image formation and how the objects in the world are structured is essential to make any assertion about the scene that is being viewed. *Object modeling* is the first step that makes this knowledge of the object structures and how they appear in an image explicit. For example, objects can be modeled as volumes, or sets of bounding surfaces, or just lists of salient local features, and the image formation process can be described using either perspective or orthographic projection of the 3D scene into a 2D image array. The resulting image has to be processed to extract the regularities in the sensed 2D data, organize them as connected entities or "blobs" and characterize them in a way that is indicative of the objects and their spatial interrelations which are in fact manifest in the scene. Note that this process is confounded by practical problems such as sensor inaccuracies, variations in ambient illumination, clutter and occlusion owing to objects being close to or overlapping one another in a scene.

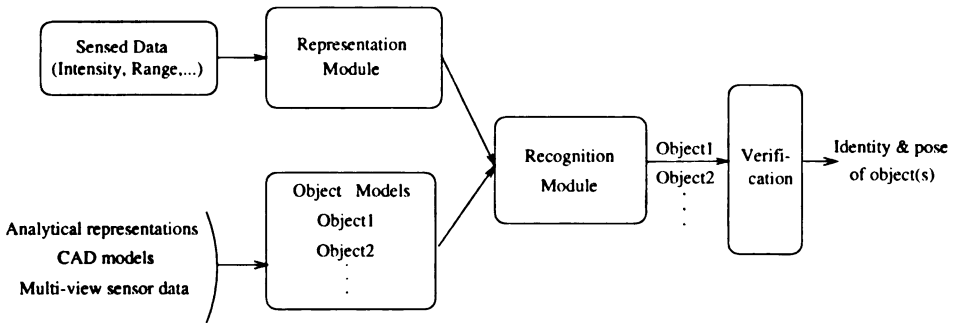


Figure 1.1: Key components of a 3D object recognition system.

Recognition is the processing step that compares a derived description from the image, which is most likely to be incomplete, with stored models or representations of objects in order to identify what is present in the scene. A recognition module has to search among the possible candidate object representations to identify the best match and then verify whether the candidate solution is indeed *correct* or *incorrect*. This search procedure can be very time-consuming because the search space of possible feature-correspondences and view-transformations is very large. The time complexity of matching depends crucially on the number of stored objects, how detailed or sophisticated the stored representations are, and how they are organized. In addition, the matching process has to deal with missing information in the input scene representation due to occlusion, and sometimes spurious additional information resulting from incorrect merging of connected "blobs". Automatic recognition of objects from images has to be performed accurately and quickly for practical use in the real world. The integrated representation of various types of visual information in the scene along with intelligent associations and real-time retrieval mechanisms are some of the challenges in building an automatic recognition system. Figure 1.1 presents the important stages in the design and development of a recognition system.

All the processing steps in an automatic recognition system have to be performed reliably and efficiently in order to be employed in many challenging real-world applications such as robot bin-picking, automated inspection of assembly parts, face recognition for security analysis, and autonomous navigation. Reconstruction and recognition of various aspects of shape and other physical properties of the objects in

the real world are some of the fundamental issues that need to be addressed during the creation of vision augmented virtual environments. Medical diagnosis from X-ray and ultrasound images, and robot assistance in complicated surgeries are some of the important areas where automatic recognition can make a significant impact and be beneficial to humanity.

1.2 Challenges in Three-Dimensional Object Recognition

The dominant and popular paradigm in 3D object recognition [135] assumes two stage processing of scene data: First, an internal representation of the scene is derived from the input data that may have been obtained from one or more sensors such as CCD cameras and range scanners. At the second stage, it is matched against stored representations or models of known objects. This paradigm is also hypothesized to underlie human visual processing [113].

During the model building stage, analytical or CAD models of objects, if available, are typically used to construct descriptions of objects that may be present in the scene. Alternatively, information about 3D objects in the scene is gathered from various viewing directions and organized to construct their descriptions. An explicit *model* or a *computer representation* of a 3D object is important to subsequent recognition by the machine as it influences the processing of sensed data that attempts to derive a similar description from the measurements in an image. A poor scene representation would place a heavy burden on the recognition algorithm when it matches the input against a collection of stored object representations, and would most likely lead to incorrect identification of the object. Therefore, while designing an object recognition system for a specified task, care should be taken to develop approaches that solve both representation and matching problems in an integrated manner.

A number of representational and matching themes have been recognized by several researchers [70] recently as challenging and important, following a survey of a

decade or so of intense activity in the computer vision field. These themes are related to the complexity, speed and generality issues that need to be addressed by a recognition system. We discuss below some of the emerging themes that are of interest to us.

1. *Object shape complexity*: 3D object recognition has so far dealt mainly with geometric entities in two and three dimensions, such as points or groups of points [106, 139], planar patches and normals [80, 78], straight edges and polylines [112, 147], polyhedral and quadric surfaces [22, 59, 30, 68, 164], and superquadrics [124, 144, 132]. The success of several existing object recognition systems can be attributed to the restrictions they impose on the classes of geometrical objects that can be handled. However, there has been a notable lack of systems that can handle arbitrary surfaces with very few restrictive assumptions about their geometric shapes. Interest has recently emerged in matching arbitrarily curved surfaces that cannot be modeled using volumetric primitives and that may or may not have easily detectable landmark features such as vertices and edges of polyhedra, vertices of cones and centers of spheres. A sculpted object may possess various smoothly blended surface features which may not lead to easy object segmentation into simple analytical primitives. Given the complexity of the arbitrary shapes one can encounter in practical situations and the difficulty of representing them in a general fashion, it is not surprising that most computer vision systems have sought to address recognition of only a very restricted class of objects. However, complex real-world applications of computer vision systems have recently begun to stimulate the development of general 3D recognition systems that can handle arbitrarily curved objects.

2. *Size of object model database*: The organization of knowledge is strongly tied to efficient retrieval of pieces of information in an application. The analogue of stored knowledge in a recognition system is the set of descriptions, models, or representations of objects of interest that might be encountered by the system. Efficient retrieval here implies faster matching. In real-world appli-

cations, one typically finds the need to handle databases containing a large number of complex object models. By "large" we mean typically a thousand or more models. As the number of objects to be recognized by a system increases, the computational time to perform recognition of even a simple input image becomes discouragingly high. This is primarily due to the fact that in most systems, the input representation is matched against all the object models in the database. There is an increased awareness of this computational cost and it has resulted in substantial research to prune the number of matches needed either by using focus features [21] or by indexing a hash table based on invariant features [105, 23, 69, 147, 32] during recognition. In addition, approaches that can organize the representations in a hierarchical fashion to eliminate unlikely matches quite early during recognition and subsequently present only a few candidate objects for final verification of their identity and pose have begun to be considered seriously.

3. *Learning*: As the need for adaptation and flexibility in vision systems grows, we find a growing interest in systems that incorporate at least some aspect of learning [17, 170]. Most object recognition systems are built without the ability to reject portions of the scene as unrecognizable, and are therefore limited to domains wherein the set of objects to be recognized is pre-specified. A recognition system may perform well within the scope of its knowledge; but any slight deviation such as noisy segmentation or representation outside the narrow expertise of the system causes the performance to deteriorate rapidly. Many application domains of computer vision systems are by and large unstructured. Less controlled environments, therefore, mandate construction of systems that can automatically build models of "unexpected" objects. It is desirable to have the recognition system learn the description of an unknown object so that future instances of the object when encountered by the system will not make the system break down [127]. The ability to learn to recognize new inputs as well as to remember the previous instances of objects is known as the adaptation

or plasticity property. This property is lacking in current recognition systems which are thus rendered quite brittle. It is also vital to be able to learn generalized representations of an object from multiple training instances in order to recognize a degraded or noisy instance of the object in an image. Learning can aid in overcoming the disadvantages of fixed parameters [137] and representations, that are typical of current vision systems. Further, learning enforces the use of a large variety of real data in performance evaluation of recognition systems, due to the need for feedback that is necessary for valid generalizations.

4. *Individual and generic object categories*: A recognition system can represent and recognize either individual objects or store and match descriptions of generic classes of objects. For example, descriptions can be stored for individual chairs and/or the generic class of chairs. Recognition of the latter category is a more difficult problem since there is no unique structural description that characterizes the entire class of chairs although a functional description of the class of chairs is available and simple. Functionality is tied to the description of geometric features that need to be present in an object for it to be recognized as an instance of a generic category. Although it has been known that function-based representations are appropriate for constructing "generic" models of objects, issues of implementation and experimentation with systems that use function-based representations to represent and recognize elements of a generic class of objects have begun to be addressed only recently [146].

5. *Articulated objects*: Some objects such as a pair of scissors, have movable parts. Such objects are commonly referred to as articulated objects. The representation of 3D objects should encode the range of relative movement between the parts of articulated shapes. The representation that appears suitable here is parts-based, i.e. the individual components or parts of an object and their interrelationships have to be extracted reliably and used to represent the object as a whole. A family of shapes for a single object can be specified by parameterizing the point patterns representing the shape [163]. A multi-level representation

can also be used to describe the parts in an object, their adjacency relationships and also allow some parameterized range on the geometry of the connections between the parts.

6. *Non-rigidity of objects*: Almost all object recognition systems assume that the objects under consideration are rigid. Flexible objects such as organs (for example, heart and lung) in a human body are those whose shape need not remain constant with time. A deformable object, unlike an articulated object is one which is entirely non-rigid. Since most current representation schemes use either volumetric or surface-based primitives, it appears that they are inappropriate for representing flexible objects. It still remains to be investigated if there exist primitives of fixed volume or of fixed surface shape at suitable scales to describe non-rigid objects. Applications in medical imaging have spurred some of the research in deformable shape representations such as deformable superquadrics and finite-element methods [158, 157, 86].

7. *Information-preserving representations*: An information preserving representation of an object is one from which the original object can be reconstructed. Representations based on constructive solid geometry possess this feature. Extended Gaussian Image representations [85] of convex polyhedra can be used to recover them uniquely. Information preserving representations are interesting as they can be used to synthesize or reconstruct objects. The other class of representations is the discriminatory representation where only those features that discriminate between objects are captured. Here, reconstruction is not possible. An obvious drawback of this representation is that addition of new classes of objects to the database would require redesigning and rebuilding of the knowledge base of object descriptions.

8. *Occlusion*: Another major issue in the design of a recognition system is how to reliably recognize partial data of objects in the scene. Recognition systems that deal with multiple objects present within the field of view of a sensor need

to be able to recognize objects that may be partly occluded by others [161]. Self occlusion is also possible with objects that are complex-shaped. The object description stored should be such that recognition of objects is possible even from partial information. A suitable representation possessing this property is in terms of local attributes or features (i.e., those that require information from only a small neighboring region around their locations) of objects. However, if a representation utilizes only local information, then it may fail to capture the global shape of an object. The presence of "salient" local features would certainly be a crucial and deciding criterion in recognizing the object. Even if an object is partially seen, a total lack of distinguishing features may render its recognition impossible.

1.3 Free-Form Object Recognition

This dissertation deals with one of the main challenges described above. We concentrate on building a recognition system that can handle arbitrarily shaped objects. We present here our motivation to address this problem, and the advantages that result from the proposed solution to the problem.

1.3.1 Motivation and Statement of the Problem

As mentioned before, most current vision systems tend to be restrictive about the shape of the objects that can be handled. As Flynn and Jain note in [70], an obstacle to the widespread acceptance of 3D object recognition systems is representational. Most current systems cannot accommodate sculpted surfaces and large model databases. Current recognition systems are limited to domains where the set of objects to be recognized is pre-specified and they tend to be mostly polyhedral, quadric and superquadric. We focus on the need to represent and recognize arbitrarily curved objects without restricting our recognition system to limited geometrical shapes. Free-form object recognition is also sometimes referred to as sculpted object

recognition. Observe that we do not include in our study statistically defined shapes such as textures and foams, arborizations (trees and bushes), crumpled surfaces (fractals), regular and quasi-regular tessellations of space and surface patches, and objects described using integral geometry. We also exclude from consideration surfaces that possess self-intersections and non-orientable surfaces [116] such as Möbius strip and Klein bottle. In general the complexity of the scenes analyzed by a recognition system can be characterized in terms of (a) scenes containing multiple occluding objects, (b) scenes containing multiple non-overlapping objects, and (c) scenes containing single objects with possible self-occlusion. We restrict ourselves to recognizing scenes containing a single object. Further, the focus of this research is automatic identification of objects in industrial applications, and not automated inspection or face recognition.

1.3.2 Definition of Free-Form Surfaces

A free-form surface S is defined to be a smooth surface such that the surface normal is well defined and continuous almost everywhere, except at vertices, edges, and cusps [11]. Figure 1.2 shows a smooth free-form surface. Since there is no other restriction on S, it is not constrained to be polyhedral or piecewise-quadric. As Besl points out in his seminal paper [11], discontinuities in surface depth, surface normal or curvature may be present anywhere on the object and the curves that connect these points of discontinuity may meet or diverge smoothly. The shape of the object can be arbitrary. Some representative objects with free-form surfaces are human faces, cars, boats, airplanes, sculptures, etc. In this thesis, we use the terms "objects with free-form surfaces" and "free-form objects" interchangeably. Figure 1.3 shows a set of 3D objects with free-form surfaces that is representative of objects that we wish a vision system to recognize automatically. The range images of objects shown in Figure 1.3 were obtained using a laser range scanner (Technical Arts White scanner) that produces depth data in an $X - Y$ grid. The figures show surface depth as pseudo intensity displaying the relative orientation of the surfaces; points

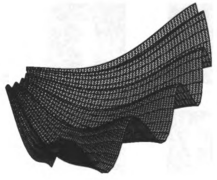


Figure 1.2: Example of a free-form surface.

oriented almost vertically are shown in darker shades. Observe that the surface data obtained from a single-view range imaging sensor typically take the form of a graph surface and hence the surface parameterization takes a very simple form: $\vec{x}(u,v) = [u, v, f(u,v)]^T$ where T indicates the transpose. However, we note that our proposed representation and recognition schemes work on any collection of (x, y, z) points on which the fundamental notions of metric, tangent space, curvature and natural coordinate frames can be suitably defined.

Free-form surfaces are extensively used in the design of smooth objects in automotive, aerospace, and ship building industries. Computer aided design (CAD) packages with capabilities to model free-form surfaces allow users to design, analyze, and test parts, without the need to build an object prototype. Recognition of free-form surfaces assists in automated machining of complex parts, inspection of arbitrarily curved surfaces, and path planning for robot navigation. Free-form surfaces are also used in terrain modeling for cartographic applications. An emerging area of

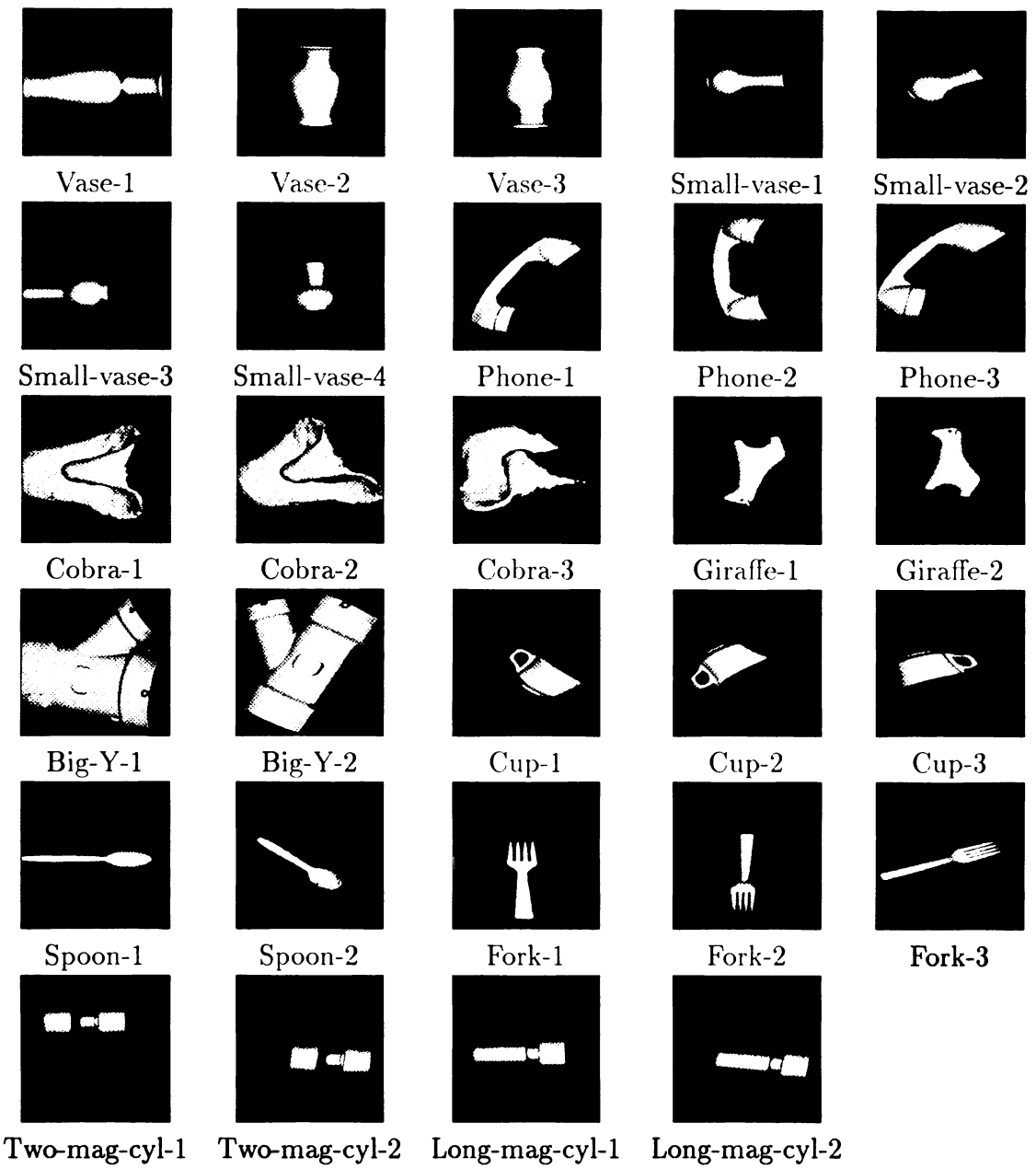


Figure 1.3: Range images of objects with free-form surfaces.

interest is automated re-engineering in which CAD models of manufactured objects that do not possess any geometric models can be built from their multiple views. The insights gained in sculpted object recognition may aid us in a better understanding, for example, of human face recognition.

1.3.3 Problem Definition

Our interest lies mainly in addressing several important issues in designing a recognition system for free-form 3D rigid objects from range data. We address the following important questions: (i) how to represent arbitrarily curved objects compactly without restricting the object shape and structure; (ii) how to recognize objects present in varying poses in the input scene from the sensed data; and (iii) how to organize representations of the model database of objects for faster recognition and also efficient storage? We believe that these issues are intercoupled in terms of the representation and matching schemes that can be used.

1.3.4 Problem Difficulty

Design of an appropriate representation scheme for 3D objects is a crucial factor that influences the ease with which they are recognized. Most of the object representation schemes surveyed in Chapter 2 have adopted some form of surface or volumetric parametric models to characterize the shapes of objects. Current volumetric representations rely on representing objects in terms of spatial occupancy, generalized cylinders, superquadric or set-theoretic combinations of volume primitives as in constructive solid geometry (CSG). However, objects with free-form surfaces, in general, may not have simple volumetric shapes that can be easily expressed with, for example, a superquadric primitive, even though it may contain eight (including bending and tapering) parameters. Further, the difficulty with recognizing an object via matching volumetric descriptions is that many views of the object must be used because there is uncertainty in the extent of the object in a direction parallel to the line of view. However, humans can identify objects even from a partial view. This aspect is

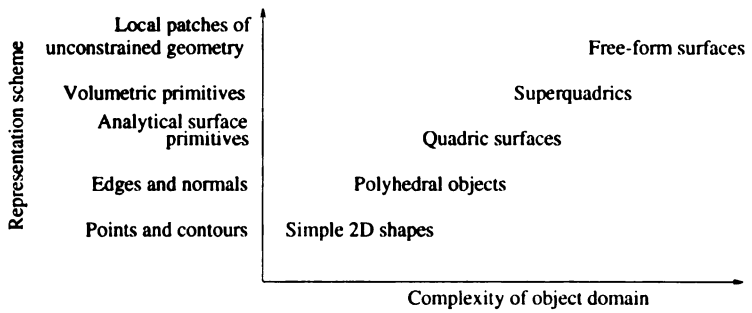


Figure 1.4: Representation and object shape complexity.

a motivating factor for matching objects using "surface-based" representations that describe an object in terms of the properties of the surfaces bounding the object, such as the surface normals, curvatures, etc. This is more commonly employed for recognition since the representation directly corresponds to features easily derived from the sensed image of the scene. In addition, matching only the observed surface patches with those of a stored representation can aid in recognition of partially seen objects.

Surface representations are mostly based on a small set of analytical surface primitives that either exclude sculpted objects from their domains, or allow free-form surfaces at the expense of approximating the object with a very large number of simple primitives such as quadric and bicubic surface patches. Such an approximation tends to be coarse or fine depending on the number of primitives used. If it is coarse, then it may not capture the shape of the object accurately and hence can be ambiguous. If it is too fine, the number of primitives will be too large leading to a loss of global shape. Global representations such as the extended Gaussian image (EGI) [85] and other orientation-based descriptors [118, 96, 109, 115] describe 3D objects in terms of their surface-normal distributions on the unit sphere, with appropriate support functions. They handle convex polyhedra efficiently, but arbitrarily curved objects have to be either approximated by planar patches or divided into regions based on the Gaussian curvature. Figure 1.4 portrays various classes of objects and the representation schemes that are commonly adopted to describe them.

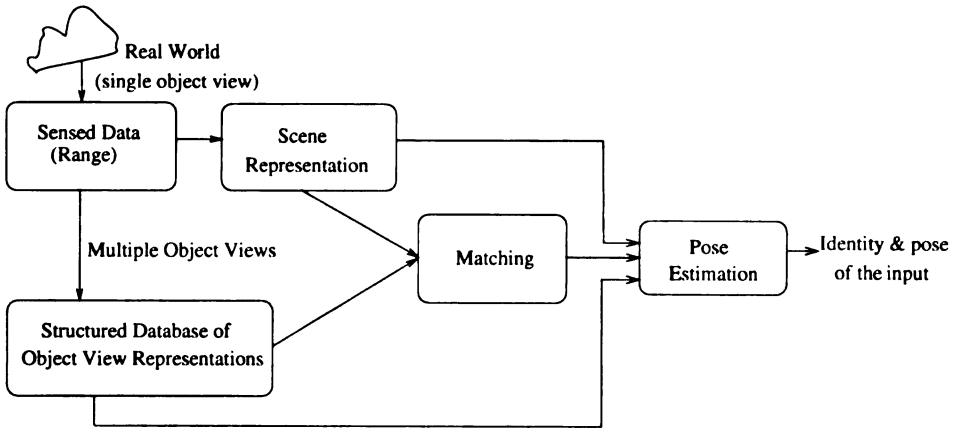


Figure 1.5: An overview of the proposed approach.

1.4 Overview of the Thesis

The goals of the proposed 3D object recognition system are the following:

- Design a representation scheme that can be used to describe sculpted objects with free-form surfaces as well as objects composed of simple analytical surface primitives. The scheme should be as compact and expressive as possible for accurate recognition of objects from a single range image.
- Recognize objects from a single view using range data.
- Estimate the pose and orientation of the recognized object from range data robustly.

Figure 1.5 presents an overview of the proposed approach. In our system the figure/ground separation problem has been alleviated by considering uncluttered scenes and by using range images. Object recognition is performed from a single range image of a view of the object. The geometric approach we propose is based on surfaces and their shapes. We employ a modified definition of *shape index* originally proposed by Koenderink for graphical visualization of surfaces [101] to identify the shape category to which each surface point on an object belongs. Our approach makes use of the shape index to represent complex objects for recognition. An object is concisely characterized by a set of maximally sized surface patches of constant shape index and

their orientation-dependent mapping onto the unit sphere. This spherical mapping of the maximal patches not only results in a description of the object's rotation in 3D space but also aggregates those patches that get assigned to the same point on the sphere using shape spectral functions; this allows us to summarize objects by the shape categories of the surface components, especially when multiple components of the same shape index and orientation are present. The points on the unit sphere that are mapped by the maximal patches are further characterized by a set of appropriate support functions describing the shape, average curvedness and surface area of the patches. The average curvedness of a surface patch specifies whether it is *highly* or *gently* curved; the surface area quantifies its extent or spread in three-dimensional space; the orientation (mean surface normal) of the patch describes how it is oriented or directed in 3D space. The relative spatial arrangement of the surface patches as captured by their adjacency is also built into the representation. We refer to our representation scheme as COSMOS (*C*urvedness-*O*rientation-*S*hape *M*ap *O*n *S*phere).

The main strength of our scheme is the integration of local and global shape information that can be computed easily from sensed data and is reflective of the underlying surface geometry. Thus, our scheme provides a meaningful and rich description maintaining the recoverability of several classes of objects. The representation is compact for many classes of objects that contain only a few distinguishable surface patches of constant shape index, i.e, whose surface shapes do not change rapidly over large regions of the object. It is also a general scheme capable of representing arbitrarily curved 3D objects and objects with holes, as it does not rely on analytical surface primitives to approximate regions. A novel concept of *shape spectrum* of an object is also introduced within the framework of COSMOS for object recognition. We propose a new shape-spectral feature based scheme for grouping object views of sculpted objects, that obviates object segmentation into parts and edge detection. These features allow object views to be grouped meaningfully in terms of the shape categories of the visible surfaces and their surface areas. By exploiting view-grouping in model databases, a small number of plausible correct matches can be quickly retrieved for more refined matching. We have demonstrated that in a database containing 6400

views of 20 different objects, only 20% of the database was examined, on the average when 2,000 independent test views were tested for their correct classification. The proposed model view selection scheme is general and relatively easy to use.

A novel multi-level matching strategy that employs shape spectral analysis and features derived from the COSMOS representations of objects is proposed for fast and accurate recognition of free-form objects. Given a range image of an uncluttered view (allowing self-occlusion) of an object, the shape spectrum-based model selection scheme short-lists a few promising candidate views from a database of object views. During view hypothesis verification, we use the COSMOS representations of object views to determine the scene-model feature correspondences using a combination of search methods, and thus identify and localize the object in the input view. Object pose estimation is formulated as registration of the sensed data with the range image of the best matched view of the object. We present a minimum variance estimator to robustly register two range images of a complex object and compute their relative view transformation accurately. Experiments on a database of over 6,000 object views generated from CAD models and surface triangulations and 100 range images of several complex objects acquired using a 3D laser range scanner have demonstrated the strengths of our COSMOS based 3D object recognition system.

1.5 Main Components of the Recognition System

We describe here some of the key traits that distinguish our proposed 3D recognition system from most other work in computer vision.

- *Free-form surfaces*: Our system can handle general 3D rigid objects that may be arbitrarily curved.

- *Representation*: Our representation provides a shape-based description of objects and is suitable for representing free-form surfaces without requiring complex 3D analytical modeling of objects. It can describe complex shaped objects concisely in terms of the surface patches that are mapped to a unit sphere by

their orientations, along with a set of support functions specifying their geometric attributes. The spherical mapping of surface patch normals allows us to derive a global orientation of objects and at the same time provides us an elegant method to derive high-level geometric feature summaries of multiple surface patches with the same orientation that may map to identical points on the unit sphere (a realistic situation with nonconvex objects) in terms of their various shape categories and their local surface attributes.

- *Recognition*: Our multi-level recognition strategy uses a powerful pruning technique at the first level to eliminate the non matching candidate object models quickly by matching "shape spectral" features of an input object view with those of the views present in a structured database. At the second level it performs detailed matching of components of the COSMOS representations of the retrieved candidate views to establish the model-scene feature correspondences. The second level thus determines the correct object identity of the input view. The 3D rotation of the object is computed using the corresponding surface normals established from the CSMP correspondences between the scene and the stored object view.

- *Pose estimation as registration*: In order to accurately estimate the pose of the object in the scene, we derive a robust minimum variance estimator to compute the transformation between the scene view and the matched object view using their range data. The initial estimate of the object pose is refined using an iterative minimization scheme. We have proposed an error model to take into account uncertainties in z measurements at different orientations of the surfaces in order to handle numerical errors, surface orientation effects, etc. We show that the transformation parameters estimated using our weighted objective function [50] are significantly more accurate than those obtained using an unweighted distance criterion [38].

- *Object database*: The model database used in our experiments consists of two categories of range images of object views that are used to test the strengths of our representation and recognition schemes. The first category consists of 6,400 object views generated from 320 viewing directions using 20 different models (available as CAD models and surface triangulations) of free-form objects. These 3D surface models were obtained from an on-line database resource (Section 4.5.3). The other category includes range images of multiple views of ten different objects, and these were obtained by scanning the objects using a laser range scanner.

1.6 Organization of the Thesis

The rest of this thesis details the key ideas that have been outlined above. Chapter 2 presents a literature survey of previous work related to 3D object recognition. Separate sections are devoted to 3D object representation, recognition strategies and free-form object matching techniques. Chapter 3 describes our proposed COSMOS representation scheme. The COSMOS representation characterizes an object by a set of new surface primitives referred to as CSMPs (Constant Shape Maximal Patches). Definitions of this and other components of COSMOS, and properties of this representation scheme are provided in Chapter 3. This chapter introduces the concept of the shape spectrum of an object and describes techniques for deriving the COSMOS representation of an object view from surface depth data obtained using a laser range scanner. It also presents experimental results with real range images of several different objects. Chapter 4 describes a multiple-view based object model and addresses the problem of constructing view aspects of free-form objects for efficient matching during recognition. It introduces a novel view representation based on shape spectral features and proposes a general and powerful technique for organizing multiple views of objects of complex shape and geometry into compact and homogeneous clusters. It also describes the structuring of a large model base of views for quick matching against input object views. Chapter 5 proposes a multi-level recognition strategy that

exploits the representational power of COSMOS to establish the correct identity of the objects in the scene and their spatial pose, and presents experimental results using real range images. Chapter 6 looks at object pose estimation as a registration problem and proposes a new minimum variance estimator for accurate pose estimation from a given pair of views of an object. It also presents and discusses experimental results. Chapter 7 summarizes the important results of this work and outlines possible directions for future research.

Chapter 2

Three-Dimensional Object Recognition

The object recognition problem discussed in this thesis addresses a number of significant research issues in computer vision: representation of a 3D object, identification of the object, robust estimation of its pose, and registration of multiple views of the object for automatic model construction. This chapter surveys the previous work in three important topics of computer vision that relate to this thesis: representation, matching and pose estimation of a 3D object. It also presents an overview of the free-form surface matching problem, and describes current research efforts to solve this problem. Previous work in other relevant topics such as registration of multiple object views is discussed in Chapter 6.

2.1 Recognition of 3D Objects

Three-dimensional object recognition is a topic of active interest motivated by a desire to provide computers with "human-like" visual capabilities and also by the pragmatic need to aid numerous real-world applications such as robot bin-picking, autonomous navigation, automated visual inspection and assembly tasks. The dominant paradigm in computer vision system proposes to achieve recognition and localization of 3D

objects [91] from images by a two-stage process: first derive an internal representation of a scene from the sensed input data and then match it against stored representations of objects in the database. Figure 2.1 shows popular approaches to the design and development of these processing stages. Besl and Jain [14], Suetens et al. [152] and Arman and Aggarwal [3] present comprehensive surveys of 3D object recognition systems. The spectrum of 3D object recognition problems is also discussed in [5, 40]. Sinha and Jain [143] also provide an overview of geometry-based representations derived from range data of objects. In this chapter we discuss some of the influential schemes for representation of sensed data and recognition. Model-based 3D object recognition systems differ in terms of a number of factors, namely: (i) the type of sensors used, (ii) the kinds of features extracted from an image, (iii) the class of objects that can be handled, (iv) the approaches employed to hypothesize possible matches from the image to object models, (v) the conditions for ascertaining the correctness of hypothesized matches, and (vi) the techniques to estimate the pose of the object. We discuss these various design issues in our descriptions of the popular schemes prevalent for recognition and representation.

2.1.1 Sensors

Among the various type of image sensors used for 3D object recognition, the two most commonly used are: (a) *intensity* sensors that produce an image (a 2D array) containing brightness measurement $I(x, y)$ at each pixel location (x, y) of the image, and (b) *range* sensors which provide the range or the distance $z(x, y)$ of a visible surface point (pixel) on the object from the sensor. The brightness measurement of a scene at a location in an image is a function of the surface geometry, the reflectance properties of the surfaces, and the number and positions of the light sources that illuminate the scene. Range sensors are often calibrated to result in images with coordinates that are directly comparable with the coordinates used in object representations. A big advantage with range data over intensity (brightness) data is that it explicitly represents surface information. This makes it easier to extract and fit mathematical

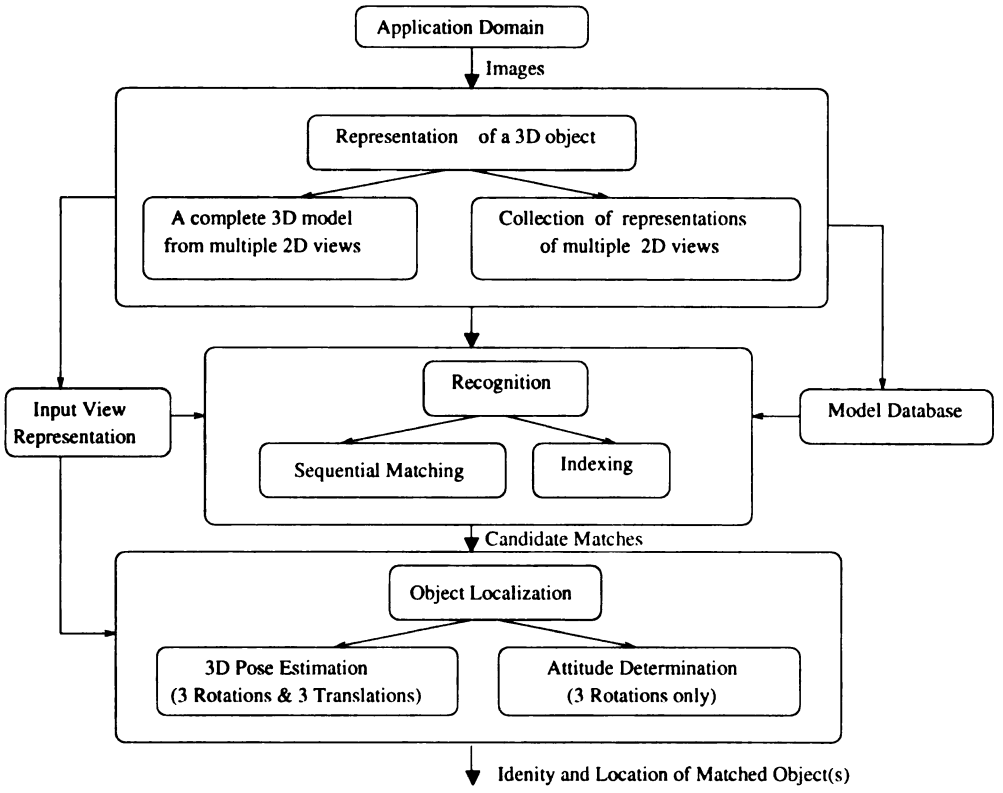


Figure 2.1: Approaches to building a 3D object recognition system.

surfaces to the data. Range data are also usually not sensitive to ambient lighting. Regions of interest belonging to objects can be separated from the background easily in range images based on the distinctive background depth value provided by the sensor, as opposed to intensity images which can have complex backgrounds with varying grey-levels that are similar to those of the objects themselves. A taxonomy and detailed descriptions of range sensing methods can be found in [94]. A comprehensive description of the Technical Arts 100X scanner that was used to obtain depth data for the experiments reported in this thesis is found in [66]. Recent surge of interest in medical applications has also motivated the use of magnetic resonance imaging (MRI) and other types of 3D data obtained through medical imaging modalities.

2.1.2 3D Object Representations and Models

We now discuss various geometric approaches to representing objects and we discuss their applicability to different object domains. We hasten to note that the perfor-

mance of a recognition system can in general be improved by incorporating information about other cues such as color and texture of objects. However, in this thesis we concentrate only on geometry-based representations. Any representation employed in object recognition should have the following properties: (i) it should be *rich* so that similar objects can be clustered together easily from their descriptions, (ii) it should be *stable* such that local changes do not radically alter the description, and (iii) it should also have *local support* so that partially visible objects can be identified. We discuss below some of the well-known representation schemes that attempt to satisfy one or more of the above conditions.

Representations used in computer vision can be fundamentally classified into object-centered and view-centered categories. Geometric techniques that use an object-centered representation attempt either to describe the entire volume of the 3D space occupied by a solid opaque object or use intrinsic or viewpoint-independent features of objects such as corners, holes and straight edges that are projected onto the image under non-accidental viewing conditions and are detectable by various image processing operations. On the other hand, viewer-centered representations rely on specifying the "appearance" of an object from a single or a set of multiple viewpoints and use viewpoint-dependent features such as occluding contours, silhouettes and T-junctions of a shape that are not intrinsic to an object. Further distinction can be made among these classes of representations depending on whether they are local or global shape descriptors.

Among the object-centered representations are the boundary-based methods, volumetric descriptors and sweep representations. The boundary-based local methods represent objects as lists of faces, edges and vertices. Viewpoint-independent features such as long and straight edges on the objects that are projected onto the image are included in this representation. Early systems [112, 78] represented polyhedral object domain using edges, straight line segments and normals and reported promising results. Since polyhedral representations of curved objects require large amounts of space to adequately approximate them, both planar and quadric equations were then used to describe the surfaces [59, 173, 58]. Some early work also extracted sur-

face patches enclosed by boundaries that were orientation discontinuities from range data [122]. Bolles and Horaud [22] used cylindrical and planar surfaces surrounded by circular arcs and straight lines. Surfaces were also classified into primitive shapes such as peak, pit, saddle, etc. based on the signs of Gaussian and mean curvatures [10, 167]. A structural representation [147] of objects that specifies edges and local surface patches in terms of their surface normal distributions has been recently advocated to handle general free-form surfaces. The local boundary and surface-based methods, in general, are sensitive to noise in the sensed data and depend on reliable extraction of primitives describing the objects from input images. In situations where the data sampled from a surface are sparse, the surface can be represented as a collection of triangular patches, and more advanced surface reconstruction techniques can then be used to interpolate from these patches. Triangular approximations of surfaces provide very little information about object parts or components, but are nonetheless useful when no other method is suitable.

The volumetric methods describe the subset of points in 3D space that are contained within an object by representing the object as an implicit or parametric function in an object-centered coordinate system. As the implicit function represents the entire shape of the object, the representation is *global*. Representations using voxels, octrees [136] and superquadrics [6] belong to the class of global, volumetric representations. In voxel representation, an object is described by the union of non overlapping cubes, where the voxels (cubes) are oriented in a rectilinear fashion, and are positioned in a 3D square lattice. Octrees describe objects in a hierarchical manner with the root of the tree being a cube that encloses the object completely and it is further subdivided hierarchically into octants to decompose the space occupied by the object into a very fine resolution. A superquadric primitive is a generalization of a class of ellipsoids called super ellipsoids and it has been used in computer graphics [6]. Pentland proposed it for shape representation in computer vision applications [124] and demonstrated its use in building realistic-looking object models. A superquadric representation for an object from range data is obtained by fitting an implicit equation [144] to a set of input data points. The limited set of shapes repre-

sented by superquadric primitives can be extended to build more complex volumetric primitives by adding parameters, global deformations such as tapering, bending, and twisting to the generic implicit equations. The disadvantage, however, is that the fitting process becomes much more expensive and numerically unstable. One of the proposed solutions is to segment objects into a set of parts that can be modeled using superquadrics [61]; however the computational complexity still remains prohibitively high. Although powerful in describing shapes, superquadrics have some drawbacks. They are intrinsically symmetric along x, y, and z axes and their geometric bounds are just simple cartesian cubes. While original parts-based representation used superquadric functions with multiplicative deformations, Pentland and Sclaroff [123] used spheres augmented with "modal" deformations that are additive. This approach, when tested with the recognition of human head shapes yielded a high accuracy (96%), but it required that parts segmentation and an initial orientation estimate were available. Objects have also been modeled using constructive solid geometry (CSG) based approach [37] where an object is represented as a binary tree; each leaf represents an instance of simple volumetric primitives and each internal node in the tree represents a regularized Boolean operation of its children. A recent effort [20] emphasizes multiple surface representations, ranging from quadric to superquadrics to generalized cylinders to handle a large class of natural objects. A particular choice of representation is made based on the nature of information available from the range image.

While volumetric representations describe objects as solid 3D primitives, sweep representations define objects by combining 2D surfaces with a *sweeping* rule. The generalized cylinder (GC), also called generalized cone is a representative of this class. A generalized cylinder is defined by a 3D space curve that serves as an axis, a 2D closed cross-sectional shape and a sweeping rule along the axis. GCs are specially suited for elongated shapes containing axial symmetry [25]. However, recognition using generalized cylinders, in general, has been hard due to the difficulty of extracting GCs from input images. GCs may not be a natural representation for non-elongated shapes such as polyhedra and for other shapes that may not have any elongated part.

Another set of global representations include a class of orientation-based descriptors such as the extended Gaussian image (EGI) [85], the support-function-based representation (SFBR) [118], the complex EGI (CEGI) [96] and the generalized Gaussian image (GGI) [109]. These map surface normal distributions of an object to the unit sphere with appropriate support functions, thus creating an orientation histogram. The support function at every point on the unit sphere with the EGI representation is the Gaussian curvature. An attractive feature of EGI is that it is based on Minkowski's theorem which states that for a closed convex object, the Gaussian curvature, as a function of the unit surface normal is sufficient to uniquely determine a surface up to a translation [128]. However, it cannot uniquely represent nonconvex objects [85, 128]. In addition, the translation of an object cannot be recovered when the EGI is used for recognition purposes. In the case of SFBR, the support function associated with the spherical mapping of normals on the object is the distance of the tangent plane at a point from a predefined origin. It is less compact, but can uniquely determine a closed surface [118, 109]. The drawback here is its dependence on the choice of the origin, and representations of the same object vary when the origin is chosen differently.

The CEGI has been proposed as a representation for 3D pose estimation of primarily convex objects. It augments the support function of the EGI by also storing the distance of a point from a specified origin in the direction of the normal, and allows one to recover the translation of a convex object uniquely. In the CEGI representation, the support function is stored as a complex number and it is possible that several nonconvex objects can have the same CEGI representation. GGI [110, 109] extends the EGI approach by storing the connectivity information between the immediate neighboring points on the unit sphere, thus ensuring the uniqueness of the representation for all objects. The multiple folds present in the Gaussian image resulting due to concavities in the object are explicitly modeled using a linked-list of neighbors thus preserving the connectivity between the points. This aids in representing nonconvex objects uniquely. Note that except in the case of convex polyhedra (where all the points lying on a planar face map to the same point on the unit sphere), all of these

representations in general map every point on the object onto the unit sphere. They mostly attempt to answer the question of the recoverability of an object from its representation. From the point of view of recognition, however, they are verbose and they fail when parts of objects are occluded. It is also not clear how to segment the Gaussian image extracted from an image of a scene containing multiple objects into distinct regions corresponding to individual objects, except when the object shapes are simple.

Among the recent global approaches for object representation are the techniques that fit bounded algebraic surfaces of a fixed degree to a set of data points [156]. Algebraic surfaces are attractive to use because they can be used to compute the limb edges and other properties of the object. During recognition, invariant quantities are computed from the algebraic equations of observed and reference surfaces [71] and then compared. This is a growing area of research and issues such bounding constraints, convergence of surface fitting and recognition need to be investigated thoroughly. Occlusion again is a problem here as there is no guarantee that the polynomial computed from a partial view of an object is similar to the polynomial computed from its complete view, all around the object. Surface reconstruction using parametric surfaces such as B-spline surface patches has also been employed to approximate a 3D surface [111].

All the above object-centered representations concentrate on describing the complete 3D shape of an object in an intrinsic manner. The goal of a viewer-centered representation scheme is to summarize the set of possible 2D appearances of a 3D object. Motivated by some of the psychophysical findings [154], objects are represented by a set of two-dimensional views rather than a single object-centered three-dimensional model. The aspect graph approach [99] attempts to group what possibly is a set of infinite 2D views of a 3D object into a set of meaningful clusters of appearances. It partitions the viewpoint space of an object into regions of "similar" views" called aspects, separated by a "visual event" that occurs on the boundary of two neighboring class of views. The visual event signals a change in the topology of the silhouette of the object. The aspect-based representations take the form

of a connected graph in which every node denotes an aspect, and every connecting edge is a visual event. The topic of aspect graphs has proved to be a fertile area of research [55, 74, 75, 103, 126, 34, 145, 149, 148, 169, 56]. While some researchers [55, 74, 75, 103, 126, 145] have assumed orthographic projection to compute the aspect graphs of objects, others [54, 149, 148, 169, 56] have attempted to compute the more general perspective projection aspect graph. The enormous size and the complexity of aspect graphs for even simple polyhedral objects are the primary reasons for lack of their widespread use in recognition. Computing aspect graphs of general 3D objects is still an unsolved problem. Efforts have also been made to organize multiple views generated using CAD models into aspects, and in [89] edges and faces of polyhedra are used to classify an object appearance into a small number of aspect groups. Hansen and Henderson [83], Ikeuchi and Kanade [90], Camps et al. [28], and Arman and Aggarwal [2] advocate an "automatic programming" approach in which recognition procedures that exploit both the CAD models and views aspects are constructed by processing the database of object models. The utility measures of features extracted from CAD models in this context are discussed in [31].

View-based recognition strategies [127, 53, 26, 117] have been particularly applied to object domains for which geometric object models can be difficult to obtain. A wing representation [36] models 3D objects using a set of views, each of which is a set of $2\frac{1}{2}$D primitives called wings that describe a pair of surface patches separated by a 2D contour segment. A recent approach [9] that advocates a viewer-centered representation uses a set of silhouettes of objects as models to recognize smooth objects using an alignment approach. The projection of the object's boundaries in other views can be expressed as a linear combination of the three model views, given that the correspondence between points in all views can be specified. Although not geometric-based, a recent technique [119, 117] called *parametric eigenspace* has been proposed to project a large set of 2D appearances of an object into "eigenspace" (parameterized by pose and illumination) using principal component analysis [121, 73, 160]. The eigenspace is constructed by computing the eigenvectors of a complete image set and retaining only a few eigenvectors to capture the variations in the

appearances of the object. Swets and Weng [153] built upon the basic eigenspace features by augmenting them with discriminant analysis to achieve better inter-class separability. Chen and Jain [35] organized the appearances in a hierarchical manner so that only optimal views are examined at every level of representation. The main drawback of most view-centered representations is the lack of terseness in object descriptions. If an object can be specified using a few parametric equations, then a viewer-centered representation is certainly not appropriate. However, in describing complex objects whose shapes cannot be captured by a single analytical form, or by a set of equations compactly, viewer-centered representations can play an important role.

An integrated approach for describing an object using both view-independent attributes and view dependent features has been adopted by Flynn and Jain [68] wherein a relational graph-based description of planar and quadric surface patches obtained from the CAD models of the object is stored along with the patch areas of the object from a large number of (320) viewpoints. The combined information serves as an object model.

Some of the recent and more general shape representation schemes attempt to capture a variety of details about objects: viewing an object as a composition of primitive parts called *geons* [18, 43, 133]; representing articulatedness of an object as a parameterization of relative movement between the parts that comprise the objects; modeling non-rigid objects such as hearts and lungs using deformable superquadrics [157] and methods using finite element analysis [86]; and employing function (utility)-based attributes to describe a generic category of objects [146]. General difficulties with parts-based representation schemes are the lack of consensus to decide the set of part primitives that need to be used, and to justify why they are necessary, sufficient and appropriate. The generality of a parts-based representation often leads to vagueness in terms of its practical application. In addition, the computation of all the primitives from a single (image) view of an object is difficult. Some of these issues are receiving serious attention.

Before we conclude this section, we make a note that most of the object represen-

tation schemes reported in the literature have adopted specific parametric forms to characterize the shapes of objects, thus constraining the applicability of the schemes to a restricted class of objects. Table 2.1 presents an overview of some of the key representation schemes and applicable object domains. An emerging theme of importance is the design of representations that can handle general 3D objects which can be arbitrarily curved with complex shapes.

2.1.3 Matching Strategies

In the previous section we discussed various approaches for representing objects. The next step is the recognition and localization of objects that may be present in the scene. Recognition is achieved by matching features derived from the scene with stored object model representations. Each successful match of a scene feature to a model feature imposes a constraint on the matches of other features and their locations in the scene. A consistent set of matches is referred to as a *consistent scene interpretation*. Approaches vary in terms of how the match between the scene and model features is achieved, how a consistent interpretation is derived from the scene-model feature matches and how the pose is estimated from a consistent interpretation. In the following discussion, "scene" and "image" are used interchangeably and so are "model" and "object model". A "model" indicates a stored representation.

Major Approaches

The popular and important approaches to recognition and localization of 3D objects are the following: (i) hypothesize-and-test, (ii) matching relational structures, (iii) Hough (pose) clustering, (iv) geometric hashing, (v) interpretation tree (I.T.) search, and (vi) iterative model fitting techniques.

In the **hypothesize-and-test** paradigm, a 3D transformation from the object model coordinate frame of reference to the scene coordinate frame of reference is first hypothesized. This generally involves formulating a system of over-constrained linear or non-linear equations that relate the model features with the scene features through

Table 2.1: An overview of popular object representation schemes.

Representation scheme	**Type of shape descriptor**	**Object domain (test)**	**Sensing modality**	**Viewpoint dependency**
Points (corners and inflection points along edge contours) [87]	local	objects with well-defined local features	intensity	stable over changes in viewpoint
Straight line segments [112]	local	polyhedra	intensity	viewpoint-invariant over wide ranges
Points, planar faces and edges [78]	local	polyhedra	range	viewpoint-independent
Silhouettes of 2D views [8, 162, 33]	global	curved	intensity	viewpoint-dependent
Circular arcs, straight edges, cylindrical and planar surfaces [22]	local	planes and cylinders	range	viewpoint-independent
Planar and quadric surface patches [58, 59, 68]	local	planes, quadric surfaces	range	viewpoint-independent
Gaussian and Mean curvatures based patches [10]	local	curved	range	viewpoint-invariant
Generalized cylinders (GC) [24]	global	generalized cylinders	intensity	object-centered
Superquadrics [124, 144]	global	curved objects	range	object-centered
Geons [133, 44]	parts-based	curved including articulated objects	range	object-centered
Constructive surface geometry (simple volumetric primitives) [37]	local	curved objects	range	object-centered

Table 2.1 (cont'd).

Representation scheme	**Type of shape descriptor**	**Object domain (test)**	**Sensing modality**	**Viewpoint dependency**
Extended Gaussian image (EGI) [85]	global	convex objects	intensity	object-centered
Algebraic polynomials [156]	global	curved	range	object-centered
Splash and super (polygonal) segments [147]	local	arbitrarily curved	range	viewpoint-independent
Eigen faces [117, 153]	global	general 3D object	intensity	viewpoint-dependent
Aspect graphs [99, 126, 56]	global	convex polyhedra and a class of curved-surfaces	intensity	viewpoint-sensitive
Convex, concave and planar surfaces [93]	local	arbitrarily curved objects	range	object-centered

a transformation. The system of equations is solved to provide the transformation that minimizes the squared error which characterizes the quality of match between the model and scene features. The transformation is used to verify the match of model features to image features, by aligning the model features (typically points and edge segments) to the scene features. The hypothesized match is either accepted or rejected depending on the amount of matching error. Lowe [112], Huttenlocher and Ullman [87] and Seales and Dyer [138] have presented representative work in this paradigm. An earlier recognition system, 3DPO [22] uses a distinctive scene feature to match the corresponding model feature, generating a hypothesis to search for matches of other scene features adjacent to the already matched distinctive feature. Verification is done by comparing the synthetically generated range map of the model object at the hypothesized pose with the scene data. The measured error during the verification stage in turn drives the hypothesis generation process.

Fischler and Bolles [64] in their RANSAC system solve for a perspective transformation to project a planar model into an image. The transformation is computed using a heuristic and it is verified by projecting a set of coplanar points into image coordinates. Lowe, in his system SCERPO [112], solves for a perspective transformation that relates the three-dimensional constraints to a single set of image measurements. The plausible group of matches between scene and model features that are straight line segments is searched in the image of an object taken from a single viewpoint. The correspondences are established by refining a given estimate of transformation using an iterative least squares technique. In the alignment approach [87] three pairs of non-collinear points, each pair containing a point in the image and its corresponding point on the model, determine the transformation. A model of an object is represented as a combination of a wire frame and local point features (vertices and inflection points). It consists of the three-dimensional locations of the edges of its surfaces (a wire-frame), and the corresponding corner and inflection features as shown in figure 2.2. In [87], different models of an object from different viewpoints were used. Possible alignment of the object model with the image is tested using the computed transformation. Seales and Dyer [138] represent occluding contours of polyhedra as a

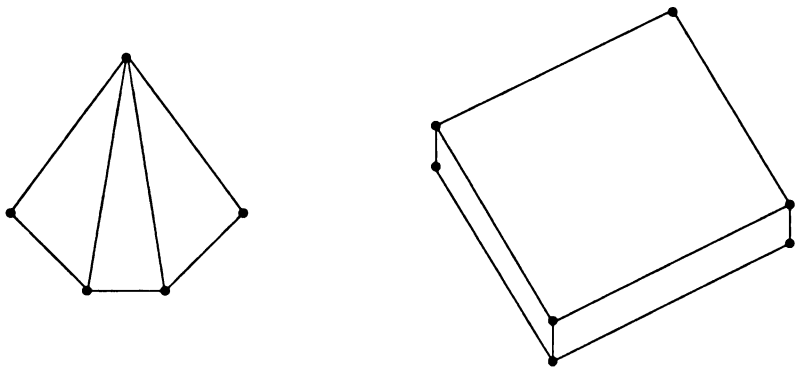


Figure 2.2: Object models with vertices and inflection points as local features.

function of viewpoint, deriving viewpoint constraints associated with each occluding contour feature. The transformations are searched by associating the model and the image contours. Chen and Stockman [33] also employ the alignment approach using both the contour and internal edges for recognition of curved objects.

Representations using **relational structures** attempt to capture the structural properties of objects explicitly for ease of recognition. Both scene and object models are described using attributed-relational graphs (ARGs), where each node in the ARG stands for a primitive scene or model feature and the arc between a pair of nodes represents a relation between the two features. Matching of a scene ARG with a model ARG is carried out using graph-theoretic matching techniques such as maximal clique detection, sub-graph isomorphism, etc. [7]. Relational representations are attractive as they capture both the structural aspects of the objects and their geometrical inter-relationships. However, recognition using relational graphs is difficult because graph matching algorithms are NP-complete. This becomes especially true when scenes contain multiple objects that may be partially occluded. An extension of an ARG is an attributed hypergraph representation (AHR) that contains hyperedges and hypernodes. The hyperedges and hypernodes are themselves ARGs where each hypervertex is associated with an ARG representing a face and each hyperedge corresponds to a primitive block graph representing a primitive block such as a polyhedron, a cylinder, or a conical surface. The AHRs are less difficult to match as we can perform hierarchical matching, thus leading to an overall reduction in complex-

ity. However, the AHR matching problem is still NP-complete, that can result in exponential time algorithms in the worst case. Recognition schemes using relational graphs have been explored extensively. Brooks [24] organizes image feature relations using a graph and matches scene to a model graph using sub-graph isomorphism. An attributed region adjacency graph is constructed from multiple registered object views to form a complete 3D model of an object [166] and matching an input view with the object model graph is performed by identifying a subgraph of the object model. Fan et al. [58] derive relational descriptions of objects in terms of their visible surface patches from dense range data, where nodes characterize the surface patches and the arcs between the nodes specify connectivity and occlusion. The largest subgraph in the model graph that matches the scene graph is found by a depth-first search. Kim and Kak [98] combine a discrete relaxation technique with bipartite graph matching to make the search efficient. Here, scene and model are represented using bipartite graphs. These graphs are used to establish the compatibility between model and scene surfaces. A hypergraph [174] is used to represent an object in a hierarchical fashion, by imposing a grouping on the vertices of a graph in which each vertex corresponds to a surface of the object and each group of vertices to a primitive block of an object. Multiple AHRs obtained from different views of the object are put together to form a complete AHR and it is compared with the stored model AHRs. Recognition using the primitive blocks can quickly eliminate incorrect matches. Shapiro et al. [141] propose a relational pyramid to represent multiple view classes and to rapidly select the view class that best matches an unknown view of the object.

In the **pose clustering** approach, also referred to as generalized Hough transform, evidence is collected for possible transformations (pose) from image-model matches and clustered in the transformation space to select a pose hypothesis with the strongest support. Each scene feature is matched with each possible model feature; matches are then eliminated based on local geometric constraints such as angle and distance measurements. A geometric transformation is computed from each successful match and it is stored as a point in the Hough (transformation parameter) space. The Hough space is six-dimensional if we deal with 3D objects with six degrees of

freedom whereas it is three-dimensional for 2D planar objects with three degrees of freedom. Maxima determination or clustering of points in the Hough space results in a globally consistent pose hypothesis of the object present in the scene. Some of the representative work using pose clustering is found in [150, 104, 142]. Grimson and Huttenlocher [76] analyze the sensitivity of Hough clustering using a statistical "occupancy model". They conclude that the probability of false maxima in the Hough accumulators can be fairly high for scenes with multiple cluttered objects and degraded by occlusion and sensor noise. Some bounds on the likelihood of false peaks in the parameter space as a function of sensor noise, occlusion and quantization effects are also presented.

In **geometric hashing**, also known as indexing, feature correspondence determination and model database search are replaced by a table look-up mechanism. Invariant features are computed from an image that can be used as indices into a table containing references to the object models. The pioneering work by Lamdan and Wolfson [105, 106] uses a two-stage methodology: (i) creation of a model hash table and (ii) indexing into the table to match an image. A model hash table is constructed by first selecting a k-tuple ($k = 3$ for a 2D object and $k = 4$ for a 3D object) of model points that forms a basis for a coordinate system into which all other model points are mapped. The mapped model points serve as indices into the hash table where the corresponding k-tuple that functions as the basis is stored. This is repeated for every set of k-tuple of model points, thus mapping all model points in a transformation-invariant manner as they are rewritten in terms of each of the reference frames. During recognition, a basis is first created by selecting a k-tuple of sensor points. Then the remaining scene points are re-mapped into the coordinate system defined by the basis and are used to index into the hash table. Each successful access of the hash table produces a model basis stored in the accessed location and a counter associated with the retrieved basis is incremented. The model basis which has the maximum support is used to compute a rigid transformation from the model to the scene coordinate system. The process is repeated for every possible scene basis definition until the object in the scene is recognized. Structural indexing

is a variant of geometric hashing in which invariant structural or geometric features are extracted from an object model. The invariant features are used to compute an indexing feature that can be used to access an object model database organized as an index table. Each set of invariant features is designed to aid in the recovery of the object pose when matched with the corresponding model in the scene. Stein and Medioni [147] and Flynn and Jain [69] have employed structural indexing and geometric hashing for 3D object recognition. Grimson and Huttenlocher have analyzed the sensitivity of geometric hashing to inexact sensor data. They conclude that it performs well in a scene containing a single object and with noise-free and perfect data, while the presence of noise and occlusion results in a significant reduction in performance. Flynn [67] argues for features with high saliency to reduce the accrual of spurious evidence in the entries of the hash table.

The **interpretation tree** (I.T.) search, or constrained search is a very popular recognition scheme and has been a subject of active work over the past ten years. An I.T. consists of nodes that represent a potential match between a scene feature and a model feature. During search, a scene feature is paired with a model feature and thus a node at level n of the tree characterizes a partial interpretation, i.e., the path from the root to a node at level n specifies an assignment of model features to the first n scene features. Instead of searching the tree exponentially for a complete and consistent interpretation, local geometric constraints such as pairwise angle and distance measurements between features are used to discard or prune inconsistent matches between scene features and model features. A global transformation is computed to determine and verify the pose of the object when a path of sufficient length is found. The control structure of the algorithm takes the form of a sequential hypothesize-and-test with back tracking. The I.T. search has been formulated and well explored by Grimson [80, 78]. A robot vision system, 3D-POLY [30] uses I.T. to recognize occluded objects in a cluttered scene, and exploits a data structure called feature sphere that aids in a fast retrieval of features for verification. Flynn and Jain [68] and Vayda and Kak [164] have employed constrained search of the interpretation tree for CAD-based object recognition. Both unary and binary geometric constraints are

effectively used to prune the incorrect interpretations. Ikeuchi [88, 89] uses an interpretation tree search to classify a scene object into one of the stored aspects of the object models and then estimates the pose of the scene object within this aspect group. CAD models are used to generate the multiple views of the objects and they are clustered into aspects. Since the task addressed in [88] is *bin picking*, only one type of object, same as the model, appears in the scene. Grimson also establishes that under some simplifying assumptions, the I.T. has a search complexity of $O(n^2)$, where n is the number of models and scene features for single-object scenes, but can become exponential in the worst case for multiple-object scenes. He also showed that if indexing is employed within the constrained search paradigm, along with clustering scene features into subsets likely to have arisen from a single object, then an I.T. based search can reduce the complexity to $O(n^3)$ which is otherwise exponential for multiple-object scenes. If the search is terminated with a "good enough" interpretation [79], then the complexity becomes $O(n^4)$ provided the scene clutter is small, and it is still exponential when the scene clutter becomes very high. A wrong object model can be prevented from being chosen as a matched object by selecting a threshold on the fraction of model features that must be matched as a function of the number of model and sensor features [77].

Iterative model fitting is used when 3D objects are represented by parametric representations wherein the parameters are used to specify both the shape and the pose of the objects. There is no feature detection and correspondence determination between model and scene features. Object recognition and pose estimation reduce to estimating the (pose) parameters of the model from the image data, and matching with stored parametric representations. If a sufficient number of image data points is available then the estimation of parameters can be done by solving a system of over-constrained linear or nonlinear equations for a solution that is best in the minimum-sum-of-squared-error sense. Solina and Bajcy [144], Gupta et al. [81, 82] and Pentland [125] have modeled 3D objects as superquadrics with local and global deformations for recognition purposes. Deformable superquadric models [157] have been proposed for modeling and tracking non-rigid organs such as hearts and lungs.

Implicit equations [129] have been used to fit observed image features to the 3D position and orientation of object models. Pose determination here becomes an iterative fitting of the observed data to the stored implicit equation.

In addition to the popular approaches described above that can be used to classify a majority of the existing matching strategies, there are a few others that perform matching using a different approach. One of them is a **rule-based** approach proposed by Jain and Hoffman [93] to recognize 3D objects based on evidence accumulation. Instead of matching object features to scene features, they construct an evidence rule base that stores salient information about surfaces, their morphological and patch attributes and their relational features, along with the evidence of their occurrences in each of the object in the database. The rules are used to compare the similarity between scene features and the support features as specified in the evidence rules for each object. Extensions of this work in terms of using more view-independent features, generating rules using a minimum entropy clustering scheme, estimating evidence weights and matching using a neural network have been proposed by Caelli and Dreier [27]. The other approach views matching as a **registration** problem [15] and it matches sets of surface data with one another directly without any appropriate surface fitting. In this approach, the distance between two point-sets obtained from surfaces is computed and minimized to find the best transformation between the model and scene data. The quality of the alignment between the model and the scene depth data can be used to determine if a model object matched closely with the scene data or not.

Table 2.2 presents an overview of the matching strategies.

Table 2.2: An overview of major matching strategies †.

Matching Paradigm	Representative Systems	Feature Determination	Model Identification	Determining Image-Model Feature Correspondences	Pose Estimation Technique
Hypothesize-and-test	[112, 87, 138, 22, 64]	typically no exhaustive search performed to group primitive features	no exhaustive search to determine possible object models in the database	exhaustive search employed to establish feature correspondences	pose estimation through least-squares optimization
Relational matching	[24, 58, 98, 174]	an exhaustive search to group features can be used	can be an exhaustive search to match object models	constrained search for determining feature-correspondences	can be an exhaustive search in the 3D transformation parameter space
Pose clustering	[150, 104, 142]	no exhaustive search to detect groupings of image features	no exhaustive search performed to identify object models	exhaustive search to relate image-model features	ordered search in the 3D transformation space
Geometric hashing & structural indexing	[105, 106, 147, 69]	no exhaustive search	no exhaustive search	exhaustive search to select object models	ranked search in the 3D transformation space
Interpretation tree search	[80, 78, 30, 68, 164, 88, 89]	can be an exhaustive search	can be an exhaustive search	constrained search	can be an exhaustive search
Iterative model fitting	[144, 81, 82, 125, 157, 129]	no exhaustive search	no exhaustive search	exhaustive search	search through least-squares optimization
Evidence-based approach	[93]	—	can be an exhaustive search	exhaustive search in "rule" space	—

† "—" indicates "Not applicable".

2.1.4 Difficulties and Challenges

In summary, some of the issues that affect the existing 3D object representation and recognition systems are the following: The first issue is "representational"; although several representations schemes have been proposed, (see Section 2.1.2) none of them seems to satisfy the conflicting requirement of global shape description with local feature support. The need to handle partially visible objects which is frequently encountered in practice dictates the use of local features but they are ineffective in capturing the complete object shape. They are easy to extract from sensed data but may not be discriminating enough. On the other hand, global representations are more descriptive, but fail to be useful in recognizing partially visible instances of objects in images. They have greater discriminating ability but are difficult to be computed with a purely data-driven approach. Most of the representation schemes are limited by the class of shapes that can be described by them. They cannot handle free-form surfaces in particular, in a compact and unambiguous manner.

The matching strategies such as the Hough clustering compare global features or shapes, and are relatively fast. However, they are error-prone when there is occlusion. The local feature-based matching schemes can handle occlusion but are computationally expensive. Recognition systems have to be made faster (e.g., by parallelizing both segmentation and matching algorithms) in order to handle large data bases. A preferred solution to the problem of building a robust and fast 3D object recognition system is to combine representations of the 3D objects at both numerical and symbolic levels, to describe objects using hierarchical representations, to derive mechanisms to match the hierarchical representations efficiently via indexing, and to design strategies that are general and reduce the search.

2.2 Free-Form Object Recognition

The success of several object recognition systems described in the previous section can be attributed to the restrictions they impose on the geometry of objects. However,

there has been a notable lack of systems that can handle arbitrary surfaces with very few restrictive assumptions about their geometric shapes. To our knowledge, there have been only a few systems to date that address the problem of matching general surfaces that may or may not possess easily detectable point or curve features. Besl in his seminal article [11] pushed forward the idea of recognizing objects containing free-form surfaces as an emerging theme of importance since objects found in natural environments are arbitrarily shaped, complex and curved. These objects may not be modeled easily using volumetric primitives and they may not have easily detectable landmark (salient) features such as edges, vertices of polyhedra, vertices of cones and centers of spheres.

The approach that has been commonly adopted for the recognition of free-form objects falls into the class of model-based recognition. For a definition of free-form surfaces, see Section 1.3.2. The free-form surface recognition task is generally formulated as that of establishing correspondences between features detected in the *scene* and features of similar types previously stored in *models* of the objects of interest. Since the underlying surfaces can be arbitrarily curved, most approaches attempt to define features of interest that are not constrained by any assumption of analytical forms present in the objects.

Based on this formulation, there are several important questions that need to be addressed, regarding (i) the type of image data to be used, (ii) the kind of features to be extracted from the input images, and (iii) the strategy to be used to match image features to model features.

2.2.1 Representation Schemes

Parametric representations are mainly employed in CAD systems to design and analyze free-form surfaces. However, vision researchers have sought to employ both global and local *structural* descriptions based on local surface curvatures or normals to represent free-form surfaces.

Parametric Representations

Parametric free-form surface representations [11] include

- Piecewise-polynomials (splines) over rectangular domains.
- Piecewise-polynomials (splines) over triangular domains.
- Polynomials over rectangular domains.
- Polynomials over triangular domains.
- Non-polynomial functions (e.g., exponentials, sinusoids, etc.) defined over arbitrary domains.

The IGES standard used in CAD representations for free-form surfaces is NURBS (Non-Uniform Rational B-spline Surfaces) [11]. Surfaces represented by NURBS form a superset of commonly used surfaces such as spheres, cylinders, and cones. A NURBS surface entity of order (m, n) (with degree $(m-1, n-1)$) is given by

$$\vec{r}(u,v) = \frac{\sum_{i=0}^{N_u-1}\sum_{i=0}^{N_v-1} w_{ij} \times B_i^m(u;T_u) \times B_j^n(v;T_v) \times \vec{p}_{ij}}{\sum_{i=0}^{N_u-1}\sum_{i=0}^{N_v-1} w_{ij} \times B_i^m(u;T_u) \times B_j^n(v;T_v)},$$

where $\vec{p}_{ij}$ are the 3-D surface control points with N_u control points in the u-direction and N_v control points in the v-direction for a total of $N_u N_v$ independent control points, w_{ij} are the rational weight factors, and $B_i^m(u;T_u)$ is the m-th order B-spline defined on the knot sequence T_u, consisting of a set of $K_u = (N_u + m)$ non-decreasing constants T_u that subdivide the interval $[u_{m-1}, u_{n_u}]$ of evaluation. Use of NURBS has not yet become prevalent in the computer vision community because of the difficulty in obtaining non-proprietary algorithms to fit a NURB surface to an arbitrary set of 3-D points and also due to the difficulty in matching NURBS-based representations of objects.

Local and Global Geometry-based Representations

Some recent approaches have specifically sought to address the issue of representing sculpted surfaces using local and global geometry. Table 2.3 presents an overview of these approaches.

Table 2.3: Current representation schemes for complex curved objects.

Representation scheme	**Type of descriptor**	**Object domain**	**Sensing modality**
Algebraic polynomials [155, 97, 130]	global	objects described by quartic curves and surfaces	range
Splash and super segments [147]	local	arbitrarily curved objects	range
Simplex angle image [42]	global	objects topologically equivalent to the sphere	range
Registration using point sets [15]	global	arbitrarily curved objects	range
Registration using image contours [107]	global	free-form surfaces	2D X-ray projections
2D silhouettes with internal edges [33]	global	arbitrarily curved objects	intensity
Triangles and crease angle histograms [13]	global	free-form objects	range
HOT (High Order Tangent) curves [95]	global	arbitrarily curved objects	video sequences
Convex, concave and planar surfaces [93]	local	arbitrarily curved objects	range

Global Representations

Algebraic surfaces [155] are more flexible than quadric or superquadrics in representing complex curved objects and are used to describe and segment objects in terms of volumetric primitives. Keren et al. [97] describe the use of implicit fourth-degree polynomials to represent arbitrary shapes. Ponce et al. [130] derive geometric constraints that are also algebraic polynomials for estimating the pose of the object. Issues such as bounding constraints and convergence of surface fitting need to be investigated thoroughly as surface approximations using implicit functions are less stable generally and can be more computation-intensive than approximations using parametric forms. Occlusion is a problem with these approaches since there is no guarantee that the polynomial computed from a partial view of an object is similar to the polynomial computed from its complete view. Ultimately, even these representations are limited by their scope. Joshi et al. [95] propose a non-parametric representation called HOT (High Order Tangent) curves. It is a collection of the parabolic, flecnodal, limiting and asymptotic bitangent, and tritangent curves that capture the structure of image contours of an object. Using this representation, viewer-centered image features such as the inflection points and bitangents of the contours are computed and used for recognition and pose estimation. Note that the recognition accuracy can be poor if the inflections points are not localized accurately.

The simplex angle image (SAI) [41, 42] is also a global mapping of a curved object wherein a mesh covering the surface of the object is mapped to a mesh on the unit sphere. Using deformable surfaces [41], free-from objects can be reconstructed using a mesh. A general parametric surface is initialized in the vicinity of observed features and the surface is deformed with smoothness constraints to recover the object shape.

Each mapped node on the unit sphere stores the simplex angle, a measure of the local surface curvature at the corresponding node on the object. The SAI is independent of the translation of the object and it also preserves the connectivity of the surface patches on the object. The SAIs of an observed scene and a model object are matched by minimizing the sum of the squared differences between the simplex

angles at the nodes of the scene and model SAIs under every possible rotation. The SAI representation can be used only for those objects that are topologically equivalent to the sphere. In addition, appropriate mesh resolution for each object needs to be determined.

Curved objects have also been modeled using a set of viewpoint-dependent features such as 2D silhouettes, along with internal edges [33]. The edge map derived from a set of model images is aligned with the scene edge map to estimate the pose of the object in the scene. Segmentation is a critical issue here.

Local Structure-Based Representations

Local structural descriptions of free-form surfaces can be derived from small surface patches directly without using volumetric primitives or planar, quadric, and superquadric surfaces. The representation using *splash* and *super segments* [147] is an example of this approach. Splash is a virtual feature encoding a local Gaussian map, thus characterizing the distribution of surface normals on a surface patch and a super segment is a group of line segments resulting from approximating the edges present on the surfaces. Figure 2.3 shows an example of a 3D super segment and a splash. Although this scheme can be applied to general surfaces, the representation does not provide a higher level description of the object. The features derived can be sensitive to noise in the sensed data and also to occlusions, thus affecting the reliability of the matching. A similar approach employs a curvature map which is essentially a layered structure containing the Gaussian curvature and the mean curvature at each point on a surface, and these can be derived from the principal curvatures estimated at surface points [168].

Besl [13] advocates employing triangles as a unifying representation for all kinds of 3D objects, including very precise curved surfaces used in manufacturing. Since such a description is typically verbose, in order to efficiently match these representations for object recognition, he proposes the crease angle histogram that provides information about the crease angles at the edges between adjacent triangles. The stability of this feature depends on a good triangulation of an surfaces and also on smoothing

algorithms that would retain the important features of the surface with minimal number of triangles. Occlusion is a potential problem with crease angle histograms.

Registration

An alternate representational approach that does not require feature detection and correspondence determination is to match surface data directly without any appropriate surface fitting [15]. In this approach, the distance between two points sets obtained from surfaces is computed and minimized to find the best transformation between the model and scene data. The main feature of the method is that it avoids any surface segmentation or surface fitting for recognition. It also does not require an explicit correspondence between models and scene features for recognition. This scheme, however, requires specification of a procedure to find the closest point on a geometric entity such as a curve, to a given point. The main disadvantage is that, just as in any iterative minimization technique, it is not guaranteed to find the global optimum especially if the scene contains occlusions, or if the point densities in model and representation are different, and if there are numerous spurious points from different objects. Note that this approach assumes that the object identity is known beforehand, and the primary interest here is to estimate the pose of the object in the input scene.

Recent work by Lavalle and Szeliski [107] also studies the problem of matching 3D surfaces using 2D X-ray images as a registration problem where the segmented 3D anatomical structures are matched using two or more contours of the same structure derived from its X-ray projections using a least-squares minimization technique. The emphasis of this work is towards recovering the spatial pose accurately rather than discrimination or recognizing multiple objects.

In summary, current representation schemes work best for a limited class of 3D objects. The generality of the shape of free-form objects is a major difficulty that is not easily overcome with analytical representations. Edges need not be present or detected reliably on smooth objects in order to be used as landmark features with edge-based representations. These difficulties have motivated us to find other means

of capturing the object shape in a general manner.

2.2.2 Recognition Techniques

Approaches that use parametric descriptions tend to match a (scene) surface description extracted from a range image with model (surface) descriptions derived from CAD models of objects in the database. Some of the difficulties in comparing a scene surface description with a model surface description using parametric forms are: (a) it may not be possible to parameterize the scene surface accurately such that the derived parameters can be *directly* compared with those of the model surface and (b) The surfaces may not be aligned in 3-D space. Hence it is assumed in these approaches that a one-to-one correspondence between all points on the scene surface and all points on the model surface is not required. Matching is done by computing the "shape distance" between a surface description and a subset of another surface description [11].

Approaches that use local structural descriptions tend to derive invariant descriptions from input scene data, and match these local descriptions with the stored model descriptions. Matching schemes use indexing [147] or search methods employing heuristics to reduce the search complexity [168]. Since these approaches do not utilize CAD-based representations of object models, the range images of objects themselves serve as models. The model descriptions are constructed using range data of different views of the objects. The object may then be rotated arbitrarily and sensed to provide the scene data.

Jain and Hoffman [93] presented an *evidence-based* approach for identifying 3D rigid objects that are arbitrarily shaped. A rule-based database was developed using the evidence conditions based on the presence of distinguished features that represent objects. In their approach, the rules were used to compare the similarity between scene features and the support features as specified in the evidence rules. They also developed a learning process for automatically extracting evidence conditions from different views of objects.

2.3 Summary

We have presented a survey of 3D object representation and recognition schemes in this chapter. We also discussed the inadequacy of a number of prevalent representation techniques in handling objects with free-form surfaces. We presented an overview of the current research efforts that specifically attempt to solve the free-form surface matching problem.

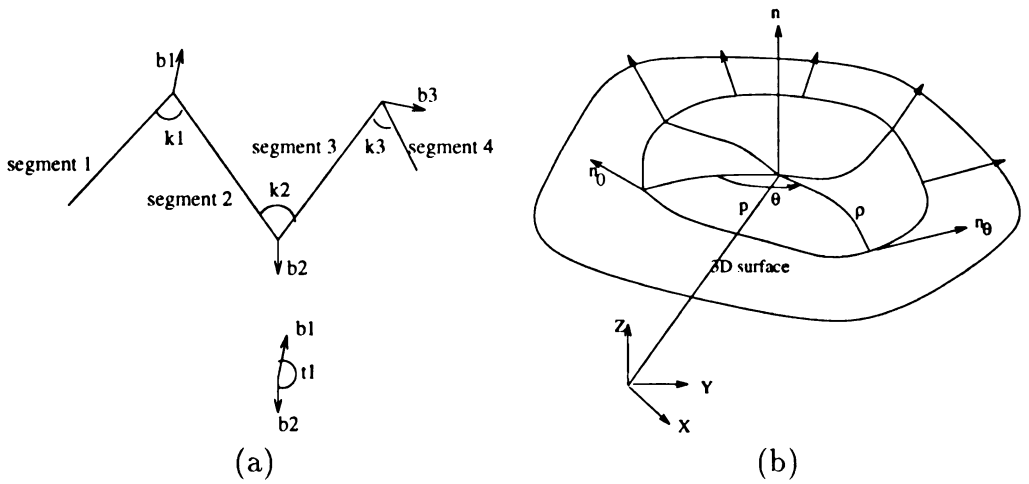


Figure 2.3: Examples of super segments and splash adopted from [147]: (a) a 3D super segment with 4 grouped segments; k1, k2, k3 are the curvature angles, t1 is the torsion angle; (b) a splash with n, the reference normal, ρ the geodesic radius, p, the location vector, θ, the angle.

Chapter 3

COSMOS — A New Representation Scheme for Free-Form Objects

In this chapter, we describe a new representation scheme called COSMOS for handling general 3D objects using surface depth data. We reiterate that by the term "general objects" we imply that we make very few restrictive assumptions about their shapes; the objects of interest to us are free-form surfaces such as cars, turbines, human faces, sculptures, etc. However, we do not include statistically defined shapes such as foams, crumpled objects such as fractals, and arborizations such as trees or bushes. Note that the class of 3D free-form objects includes convex and nonconvex smooth surfaces. Since the emphasis of this chapter is on deriving an effective representation of a sculpted object for its recognition, the topic of multi-object segmentation in the scene will not be discussed here. We assume that a range image of a scene is available which contains a view of a single rigid 3D object without occlusion for the purpose of model construction. However, we briefly note here that our object recognition system based on the COSMOS scheme described in Chapter 5 can handle multiple objects in the scene. Our proposed representation scheme has been designed to aid identification of objects and discrimination between them, and hence it has not been evaluated for other industrial tasks such as object inspection.

Objects can be represented using descriptions at several levels as shown in Fig-

ure 3.1. The kinds of objects that need to be handled and the specific task that has to be carried out by an automatic system would determine the appropriate level of description that needs to be used. If the goal is to render surfaces graphically for visualization or reconstruct general surfaces as accurately and in as much detail as possible, then the lowest level of point-based representation might be appropriate. If the task is to create graphical user interfaces, and the object domain is limited to blob-like entities, then we can choose a volumetric representation to describe objects realistically to the extent possible. The task addressed in our system is *recognition*, and the object domain to be handled is general and not restricted to certain classes of surfaces such as polyhedra or quadrics. Low-level descriptions of free-form surfaces are not stable with respect to viewing directions and are also not robust at handling noise in the image data. On the other hand, higher levels of descriptions such as edge-based representations, parametric-form fitting, and volumetric representations are also not suitable for handling general free-form surfaces. The difficulty is primarily due to the fact that the shape of the objects under consideration can be arbitrary; currently popular parametric forms seem to fit only a restricted class of shapes. Edges need not be present in a general smooth object in order to represent it. Volumetric representations are not appropriate as it is usually difficult to infer the extent of a 3D object from a single view accurately without making simplifying assumptions to compensate for the unseen part. The parts-based representation is difficult as we do not know what part primitives should be chosen as building blocks that can help us in recognizing general free-form surfaces. These difficulties motivate us to find descriptions that are based on visible surfaces and also determine other means of capturing the shapes of objects in a general manner.

3.1 How Do We Describe Rigid 3D Objects?

In order to represent objects in a way that leads to their recognition, one is forced to answer the following question:

What is the intrinsic property of an object that can aid us in its recognition?

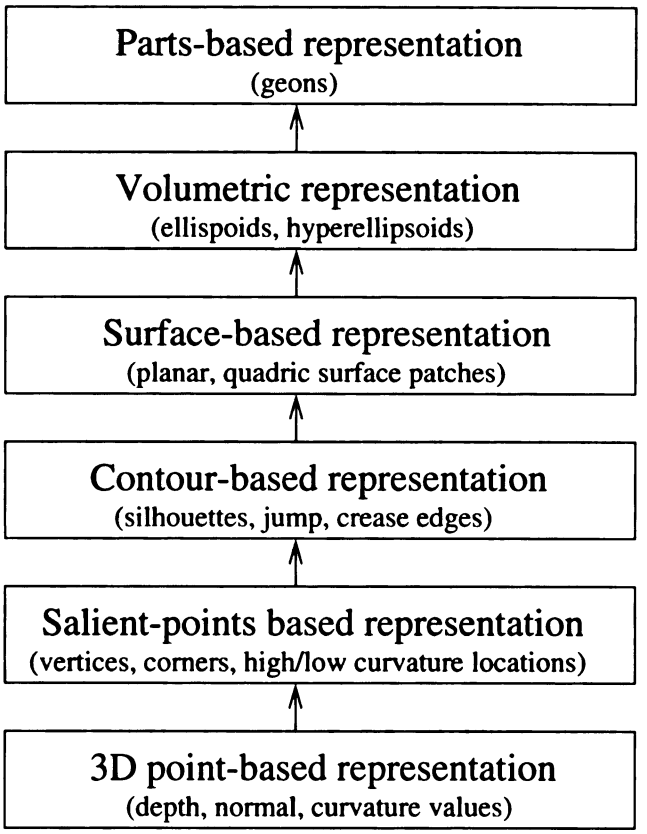


Figure 3.1: Representing objects: several levels of abstraction.

Humans seem to disambiguate objects in terms of their shapes. The Webster's Dictionary defines *shape* as follows:

> *shape* n 1a: the visible makeup characteristic of a particular item or kind of item 1b1: spatial form 1b2: a standard or universally recognized spatial form 2: the appearance of the body as distinguished from that of the face : FIGURE 3a: PHANTOM, APPARITION 3b: assumed appearance : GUISE 4: form of embodiment 5: a mode of existence or form of being having [sic] identifying features 6: something having a particular form 7: the condition in which someone or something exists at a particular time.

The shape is the visible or perceived form of an object that differentiates a cylinder from a sphere, a vase from a bottle. The shape of a rigid object does not depend on its position and orientation in space. It is purely a geometric structure that does not depend on isometrics [102]. The notion of shape is also independent of the scale; two

spherical bowls of large and small sizes are both usually described as spherical-shaped. Thus, shape seems to be one of the major criteria that characterize the geometry or structure of an object and it seems to play a conspicuous and important role in differentiating one object from another. Note that since many naturally occurring objects (e.g., human faces, terrain, etc.) are not single-shaped but made of regions of distinguished shapes that often merge with one another in a smooth fashion, even the shapes of the local regions in an object can characterize it. An object can be partitioned into a number of different shapes and this division is often done in a local manner, i.e., we not only describe objects as being either entirely cylindrical or spherical but also sometimes as those containing a planar face on the top, a large convex patch on the bottom and so on. Thus, we are able to capture the global shape if the object has a single, easily describable and distinguished shape, or a pattern of local shapes if many easily perceived shapes are present in an object.

3.1.1 Local Surface Attributes

Shape alone does not constitute a complete description of an object. What differentiates between a soccer ball (a sphere of large radius) and a cricket ball (a sphere of smaller radius) is the *curvedness* (scale) or the amount of curvature in an object. We should also be able to characterize how curved an object is. A fast curving convex object and a gently curving convex object are often perceived as distinct shapes. Attneave [4] points out how humans concentrate on points of high and low curvatures on a surface for recognizing it. With 2D line drawings of objects, an indication of how curved each portion of a line segment is provides us clues about what part it can possibly constitute in the overall shape of an object. A sharply curved protrusion of an object captures our attention before the rest of the object. Thus, a characterization of curvedness of a surface is also an important factor that aids in its recognition.

The next crucial characterization of an object is how a surface is oriented in the three-dimensional space. This helps us to distinguish between a bowl resting on its curved sides and a bowl resting on its flat side. The experiments conducted by

Pinker and Tarr [154] demonstrate how humans are able to rotate objects mentally in order to recognize them. These experimental results provide a deeper insight into the representation problem. Unless a representation scheme characterizes how objects or their sub-parts are oriented with respect to one another in a three-dimensional space, it is difficult to perform mental rotations of the stored representations to recognize an input in a new orientation. Often, in our attempts to describe objects, we resort to saying, "assume that the vase is vertical, the bottle is horizontal, lying on its side" and so on. The *orientation* of a point, a region, or the entire object seems to be intertwined with its description. In addition to the orientation information, the *extent* or the spread of the object in 3D space is usually included in its representation, characterizing how big it is. This can be measured quantitatively as the *area* occupied by an object.

Note that although the curvedness, or the average amount of curvature in a region of an object captures the scale of the object, both area and curvedness are needed to describe a region because area only provides the extent of the region in 3D space, whereas curvedness characterizes the curvature of the entire region. It is possible to imagine a long and highly curved, truncated (by planes on both ends) cylinder having the same area as that of a short and less curved, truncated cylinder. What disambiguates these two objects is the amount of curvedness present in each of them. Shape, curvedness and area are *intrinsic* quantities that do not change with reparameterizations of an object or with changes in coordinate (rigid) transformations of the object. However, the orientation of an object or regions present on the object can change with coordinate transformations. It gives us a sense of how the object is localized in space and it should be, therefore, sensitive to the transformations of the coordinate system.

3.1.2 Combining Local and Global Descriptions

The next important question is whether these local descriptors that capture (a) the type of surface, (b) how curved it is, (c) how big it is, and (d) how it is oriented in

three-dimensional space, are enough to discriminate between objects. In other words, do these local descriptors suffice to uniquely represent all kinds of surfaces, convex and nonconvex? The answer is in the *negative* for the following reason: Consider a nonconvex object O that has two convex regions R_1 and R_2 on it that are rigid translations of one another. An example of such an object is shown in Figure 3.2. The two regions R_1 and R_2 are identical; they have the same shape (convex), same curvedness, same area and same orientation.

Figure 3.2: An example of a nonconvex object.

Create a new object O_1 by distorting a small surface patch in only R_1; the distortion can be as simple as multiplying the curvatures of the points in the patch by a constant. Now perform the same distortion in $R_2 \in O$ and call the resulting object O_2. The representations of both O_1 and O_2 are identical in terms of their local descriptors alone - shape, curvedness, area, and orientation; however, O_1 and O_2 are dissimilar to each other as the distortions are in two different regions. Unless we encode adjacency information explicitly in object O_1 (for example, the region containing the distorted portion R_1 is on the right of the undistorted region R_2), it is difficult to characterize the two objects uniquely. Adjacencies of the regions in an object specify how they are present together in the object, and clearly different arrangements of regions can

result in many dissimilar objects. This motivates the necessity of characterizing the *connectivity* of regions or patches explicitly along with local descriptions, if one is interested in designing a representation to discriminate objects. Note that maintaining connectivity information is tantamount to characterizing the global structure of an object in a sense.

In the above discussion, it is essential to note that the descriptors we discussed are applicable to all classes of objects. A free-form surface can thus easily be described using the above quantities. However, we do not claim that these descriptors form a *complete* set. Compactness of a representation is also vital in designing an automatic recognition system. Hence it is important how the geometric information present in different attributes should be specified; simple objects should have simple representations, while complex objects are represented in as much detail as required.

3.2 COSMOS — A New Representation Scheme

The novelty of our scheme [45] lies in its description of an object as a smooth composition or arrangement of regions of arbitrary shapes that can be detected regardless of the complexity of the object.

3.2.1 Definitions

Each of the local and global attributes used in the COSMOS scheme captures a specific geometric aspect of the object and is defined using differential geometry-based concepts such as shape index, curvedness, and surface normals. A general parametric form of a surface S with respect to a known coordinate system is given by

$$S = \left\{ \vec{x} \in \mathbb{R}^3 : \vec{x} = \begin{bmatrix} x(u,v) \\ y(u,v) \\ z(u,v) \end{bmatrix}, (u,v) \in \Omega \subseteq \mathbb{R}^2 \right\}. \tag{3.1}$$

We assume without any loss of generality that the surface S can be adequately modeled as being at least *piecewise smooth*, i.e., it contains smooth surface patches separated by discontinuities in depth and orientation, where the extent of smoothness depends on the type of attributes that need to be made explicit at a point P on the surface S. Most free-form surfaces are smooth. Differential geometry [116] is used for describing local behavior of a surface in a small neighborhood. To represent orientation and curvature, for example, the functions used to represent S must at least be of class C^2 (twice differentiable). Then the first and second fundamental forms [116] of a surface are well-defined on these patches. These fundamental forms are used to define geometric quantities that are of interest to us.

Shape Index

A quantitative measure of the shape of a surface at a point p, called the *shape index* S_I, is defined as

$$S_I(p) = \frac{1}{2} - \frac{1}{\pi} \tan^{-1} \frac{\kappa_1(p) + \kappa_2(p)}{\kappa_1(p) - \kappa_2(p)}, \tag{3.2}$$

where κ_1 and κ_2 are the principal curvatures of the surface, with $\kappa_1 \geq \kappa_2$. Note that in Koenderink's definition [101] of the shape index, its range is $[-1, 1]$ whereas with our definition, all shapes can be mapped on the interval $S_I = [0, 1]$, and assigned non-negative values conveniently, allowing aggregation of surface patches based on their shapes. The shape index values need to be non-negative in our formulation because we use them in the definition of the shape spectral functions of surface patches. These spectral functions are used in the aggregation of attributes of the surface patches in terms of their shape index values as shown in Section 3.2.4. With mirror shapes such as the spherical cap and the cup, whose S_I values are +1 and -1 with Koenderink's definition, shape index values interact during attribute aggregation and local information about each shape category is not maintained distinctly. Hence it is necessary to redefine the shape index to take only non-negative values.

Every distinct surface shape corresponds to a unique value of S_I, excepting the

planar shape. Note that a planar surface has an indeterminate shape index as $\kappa_1 = \kappa_2 = 0$. For computational purposes in our implementation, a symbolic label (or sometimes, a shape index value of 2.0) is used to indicate surface planarity. The umbilic points on the object ($\kappa_1 = \kappa_2 \neq 0$) are not affected by the permutation of principal curvatures, i.e., by a rotation of the shape by ninety degrees. The endpoints of the S_I scale represent the concave ($\kappa_1 = \kappa_2 = -a$, where a is some positive constant) and convex umbilics ($\kappa_1 = \kappa_2 = b$, where b is some positive constant). All convex shapes have values of S_I greater than 0.5 and concave shapes have values less than 0.5. This also generalizes the common concave-convex surface classification to anticlastic patches (surfaces that have concave and convex curvatures that are opposite, for example, a saddle surface, where the surface lies on both sides of its tangent plane). Their shape indices lie between $[0.25, 0.75]$. The symmetrical saddle shape with $S_I = 0.5$ is neither convex nor concave. Nine well known shape categories and their locations on the shape index scale are shown in Figure 3.3. The representative shapes from each category are graphically illustrated in Figure 3.4.

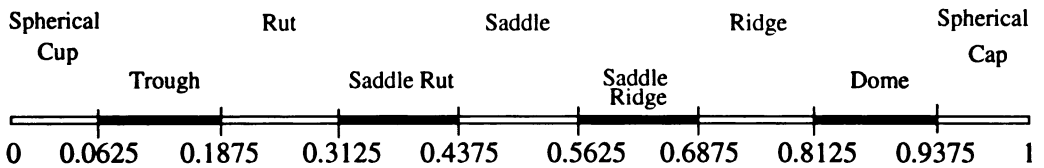


Figure 3.3: Nine well known shape types and their locations on the S_I scale.

The shape index captures the intuitive notion of 'local' shape of a surface. The shape classification into eight basic surface types based on the signs of Gaussian and mean curvatures that was employed by Besl [10] for surface segmentation can be carried out within our framework by quantizing the continuous S_I scale into eight categories. Since Gaussian curvature is intrinsic to a surface, bending a surface without stretching preserves the Gaussian curvature, although the "shape" is modified by this action. Any operation except an isometry or similarity (change of scale) may destroy the shape, and therefore the Gaussian and mean curvatures are less useful for characterizing the notion of "extrinsic shape" [101]. The shape index provides a more continuous gradation between salient shapes such as convex, saddle and concave and

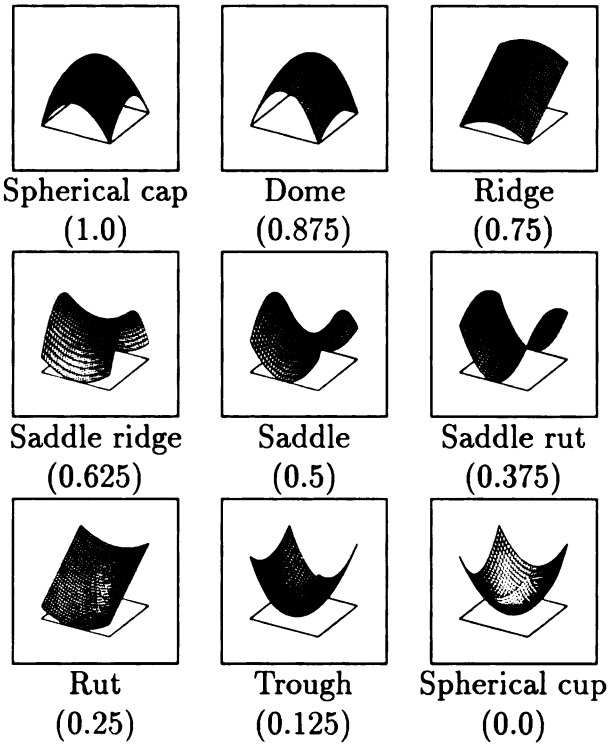


Figure 3.4: Nine representative shapes on the S_I scale.

thus has a large vocabulary to describe subtle shape variations very well.

Curvedness

The shape of a rigid object is not only independent of its position and orientation in space, but also independent of its scale. In order to capture the scale differences between objects (e.g., a soccer ball and a cricket ball) we use the *curvedness* or the amount of curvature in a region. The curvedness [102] of a surface at a point p is defined as

$$R(p) = \sqrt{(\kappa_1^2(p) + \kappa_2^2(p))/2}. \tag{3.3}$$

It is a measure of the scale of the surface, and its dimension is that of the reciprocal of length. A unit sphere has unit curvedness and so does a unit saddle surface ($|\kappa_2| = |\kappa_1| = 1$). A unit cylinder has $R = 1/\sqrt{2}$ ($\kappa_1 = 1$ and $\kappa_2 = 0$). The

curvedness becomes zero only for planar patches unlike the Gaussian curvature which vanishes on parabolic surfaces (e.g., cylindrical ridge and rut) although these surfaces appear definitely curved to humans.

We can observe that as the curvatures of a surface tend to 0 it becomes planar. Surfaces that are identically shaped may have different amounts of curvedness. For example, bell-shaped objects with different widths have the same shape index but different values of R. The same is true for two spheres of differing radii. Table 3.1 presents the shape index and curvedness values of different objects such as spheres, ridge and rut surfaces shown in Figure 3.5.

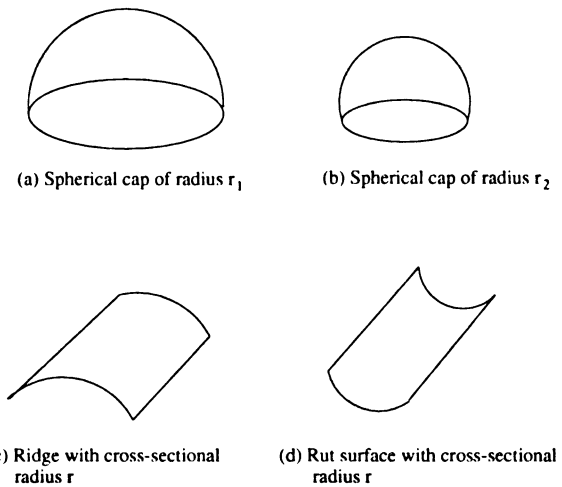


Figure 3.5: Simple surfaces with different shape index and curvedness values.

Relationship between S_I-R and κ_1-κ_2

The new parameters (S_I, R) can be viewed as polar coordinates in the (κ_1, κ_2)-plane, with planar points mapped to the origin. The direction indicates the surface shape, whereas the distance from the origin captures the size. All shapes, except for the plane, can be mapped to the unit circle whose center is at $(0, \frac{1}{2})$ in the κ_1-κ_2-plane as shown in Figure 3.6. The unit circle contains shapes with unit curvedness. Rays through the origin contain identical shapes that merely differ in their curvedness. The shape index conveniently shows "which" shape a surface has, and the curvedness indicates "how much" of that shape is present.

Table 3.1: Shape index and curvedness values of the surfaces shown in Figure 3.5.

Object	Shape Index, S_I	Curvedness, R
Spherical cap of radius, r_1	1.0	$\frac{1}{r_1}$
Spherical cap of radius, r_2	1.0	$\frac{1}{r_2}$
Ridge surface, with cross-sectional radius r	0.75	$\frac{1}{r}$
Rut surface, with cross-sectional radius r	0.25	$\frac{1}{r}$

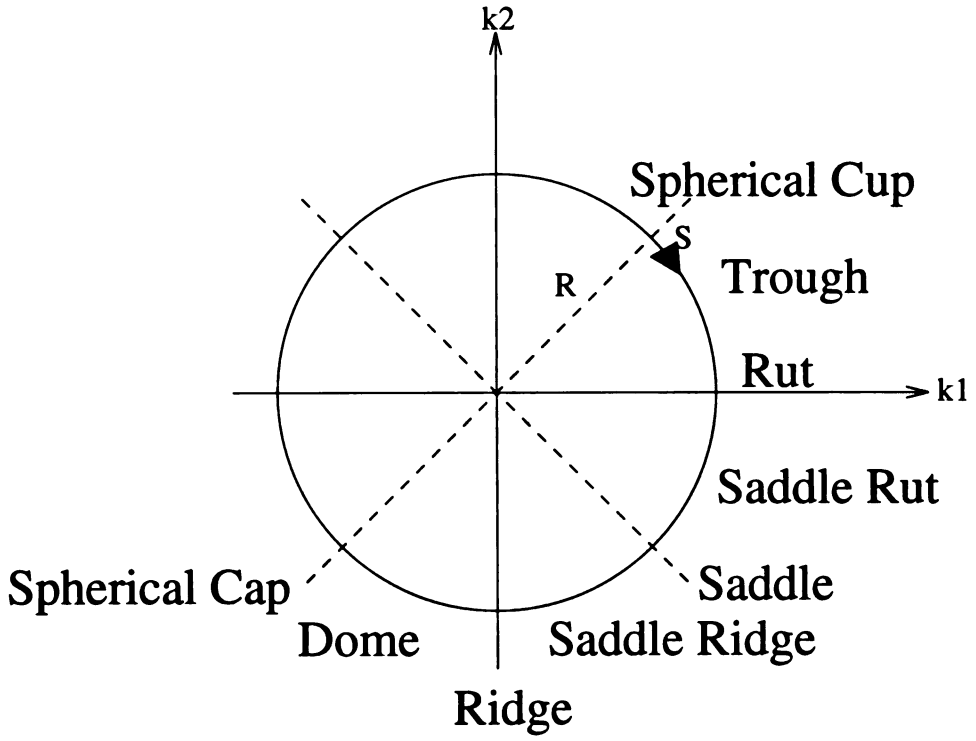


Figure 3.6: Shape index (S_I) and curvedness (R) in the (κ_1, κ_2)-plane.

Koenderink [101] provides several good reasons for using the shape index for surface classification over those schemes based on the (κ_1, κ_2)-plane: Parameterizing 'shape' by the two-sided rays of the (κ_1, κ_2)-plane provides the shape space with the topology of a projective line. In this case, we cannot distinguish *inside* from *outside.* Nor can the half-rays at the origin be used. This would lead to a topology such as that of a unit circle and the same shape would appear twice in shape space. Rather, the space of shapes has the topology of a one dimensional disc or a 'linear segment' and this is clearly brought out with the definition of shape index. Koenderink states, "...in the majority of applications dealing in one way or another with the perception of form, the shape index is the most valuable measure and the curvedness comes next" [102].

Since (S_I, R) are polar coordinates in the (κ_1, κ_2)-plane, κ_1 and κ_2 can be recovered from S_I and R in the usual manner. The principal curvature κ_1 can be derived using S_I and R as

$$\kappa_1 = R\sqrt{(1 + \sin(2\pi S_I))}, \tag{3.4}$$

and κ_2 as

$$\kappa_2 = R\sqrt{(1 - \sin(2\pi S_I))}. \tag{3.5}$$

Thus it can be seen that no loss of information occurs when one goes from (κ_1, κ_2) representation to (S_I, R) space. What we have gained is the decoupling of shape (quality) of a surface from its scale (quantity). Qualitative information about the surface is provided by the shape index whereas the curvedness provides the quantitative characterization. Besl [12] also discusses four different sets of functions based on the surface curvatures and these include a polar transformation of the κ_1 and κ_2 plane akin to (S_I, R). In his formulation, the functions (ρ, ψ) are used where ρ measures the total "bending energy" of the surface in both curvature directions and the ψ is the angle in the principal curvature plane.

Continuity of Surface Shapes and S_I values

A measure like the "shape index" is better than classical measures for identifying visually meaningful local shape features since the shape index scale maps continuously to the shape space, i.e. the local neighborhood relations are preserved. This is illustrated in Figure 3.7. As the shape index value varies continuously from $0.75 \leq S_I \leq 1$, the local shapes of surfaces become ellipsoidal; they start from the ridge shape where one of the curvatures is zero and the other is positive ($S_I = 0.75$), turn ellipsoidal (the curvature parameter that was zero becomes slowly positive) and tend towards the spherical shape as S_I approaches unity (where both the curvatures become equal and positive). As the shape index changes from 0.75 towards 0.5, the local surfaces achieve a saddle shape with the curvature parameter that was zero taking on more and more negative values. Similarly, the continuity of surface shapes from saddle to spherical cup is demonstrated in Figure 3.8.

Even if the scale of the object changes, the shape of the object surface remains the same, and so does its shape index value. Figures 3.9 and 3.10 show the same surfaces present in Figures 3.7 and 3.8, but in a smaller scale.

Shape Spectral Function

We define the shape spectral function as $S_S : \mathcal{P}_\mathcal{O} \rightarrow \mathcal{C}$, where $\mathcal{P}_\mathcal{O}$ is a set of surface patches and $\mathcal{C}$ is the space of complex-valued functions such that

$$S_S(P) = e^{\jmath t S_I(P)},\ P \in \mathcal{P}_\mathcal{O} \tag{3.6}$$

where $S_I(P)$ is the shape index of the patch P and t is a parameter that allows us to study shape properties of objects in a suitable transform domain (see Section 3.2.3), just as time-varying waveforms can be studied conveniently in the Fourier transform domain. It is used for the aggregation of geometric attributes of surface patches when multiple patches of identical shapes are found on different parts of an object (a realistic situation in the cases of nonconvex and symmetrical objects) and need to be

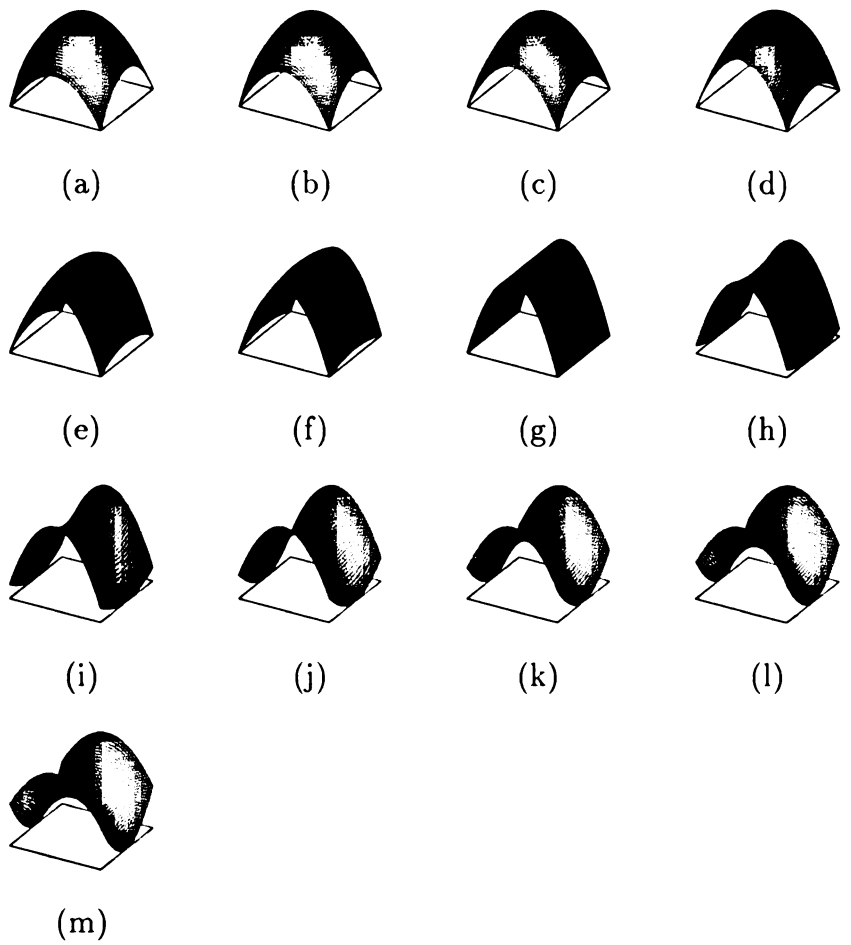


Figure 3.7: Continuous spectrum of various surface shapes and their shape index values ranging from spherical cap to saddle: (a) $S_I = 1.0$; (b) $S_I = 0.96$; (c) $S_I = 0.92$; (d) $S_I = 0.87$; (e) $S_I = 0.81$; (f) $S_I = 0.78$; (g) $S_I = 0.75$; (h) $S_I = 0.72$; (i) $S_I = 0.69$; (j) $S_I = 0.63$; (k) $S_I = 0.58$; (l) $S_I = 0.54$; (m) $S_I = 0.5$.

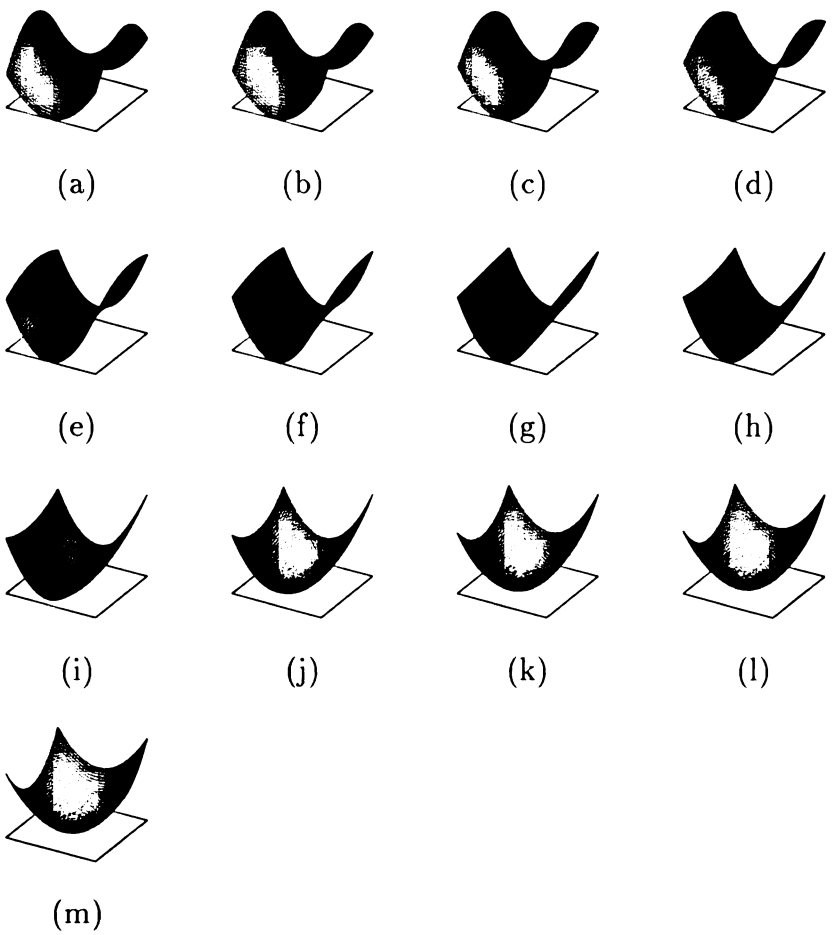


Figure 3.8: Continuous spectrum of various surface shapes and their shape index values ranging from saddle to spherical cup: (a) $S_I = 0.5$; (b) $S_I = 0.46$; (c) $S_I = 0.42$; (d) $S_I = 0.37$; (e) $S_I = 0.31$; (f) $S_I = 0.28$; (g) $S_I = 0.25$; (h) $S_I = 0.22$; (i) $S_I = 0.19$; (j) $S_I = 0.13$; (k) $S_I = 0.08$; (l) $S_I = 0.04$; (m) $S_I = 0$.

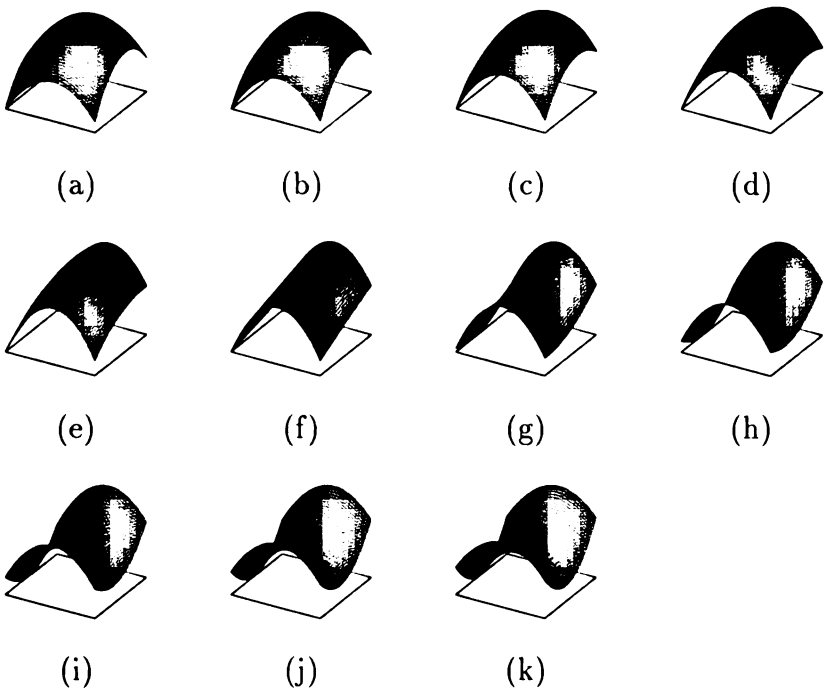


Figure 3.9: Surface shapes from spherical cap to saddle when the object scale changes: (a) $S_I = 1.0$; (b) $S_I = 0.96$; (c) $S_I = 0.92$; (d) $S_I = 0.87$; (e) $S_I = 0.81$; (f) $S_I = 0.75$; (g) $S_I = 0.69$; (h) $S_I = 0.63$; (i) $S_I = 0.58$; (j) $S_I = 0.54$; (k) $S_I = 0.5$.

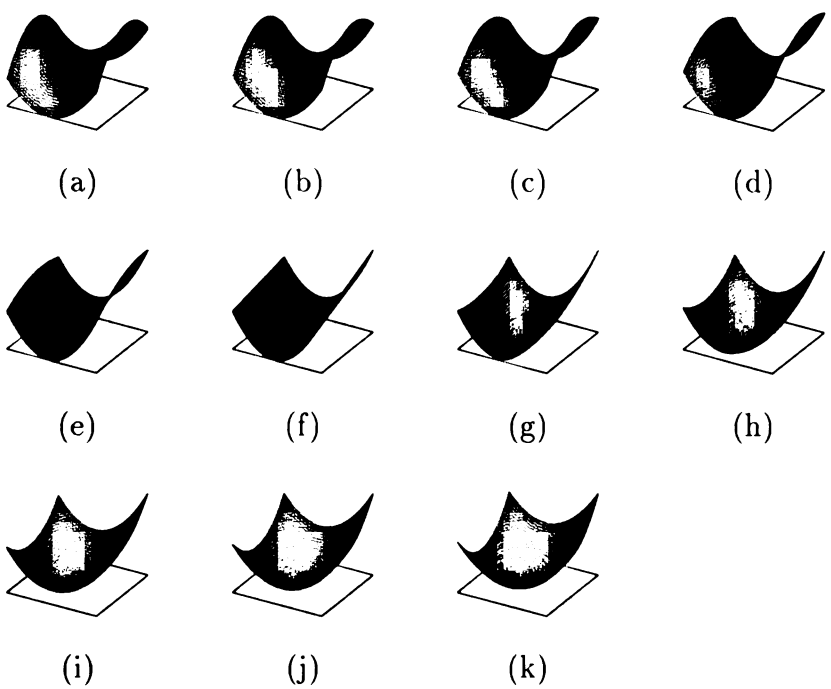


Figure 3.10: Surface shapes from saddle to spherical cup when the object scale changes: (a) $S_I = 0.5$; (b) $S_I = 0.46$; (c) $S_I = 0.42$; (d) $S_I = 0.37$; (e) $S_I = 0.31$; (f) $S_I = 0.25$; (g) $S_I = 0.19$; (h) $S_I = 0.13$; (i) $S_I = 0.08$; (j) $S_I = 0.04$; (k) $S_I = 0.0$.

summarized without any loss of information. It thus provides a way to qualitatively characterize "which shape categories are present and how much of each shape category is present in an object". This characterization is developed in Section 3.2.3.

Other Geometric Attributes

COSMOS also characterizes how objects and their sub-parts are oriented with respect to one another in a three-dimensional space in terms of their average surface normals. The extent or spread of the object in 3D space is also encoded quantitatively in our representation by the *surface area* occupied by an object. Along with these local descriptors, our representation scheme encodes the relative arrangement of surface patches on the object by their adjacencies. Note that maintaining connectivity information is tantamount to characterizing the global structure of an object in a local manner. These descriptors are applicable to arbitrarily curved objects as they are quite expressive and do not depend on the presence of any analytical primitives.

Since the compactness of a representation scheme is crucial to the performance of object recognition systems, it is equally important to determine how the geometric information present in various surface attributes can be specified together such that the complexity of the representation in a way reflects the complexity of shapes present on the object. Our representation scheme composes the attributes together in a novel way to characterize an object compactly. It treats both convex and nonconvex objects alike.

3.2.2 Definition of the COSMOS of a 3D Object

The fundamental component of COSMOS representation of an object is the description of an object in terms of surface patches that are of different shapes. We denote a maximal patch of constant shape index in an object O by the name CSMP (Constant-Shape Maximal Patch).

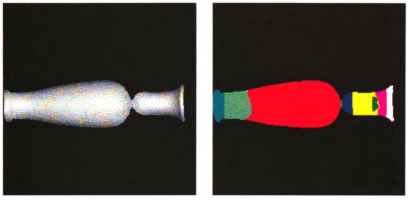
Definition: A CSMP is a maximally sized surface patch $P \subseteq O$ on the object such that $\forall p$, $\forall q \in P$, (i) $S_I(p) = S_I(q)$ and (ii) $\exists$ a path from p to q consisting of points $r \in P$ such that $S_I(r) = S_I(p) = S_I(q)$.

The second condition imposes *connectedness* of the points in P. An object can now be compactly described in terms of the CSMPs that are present on it. For example, a spherical surface has a single CSMP of spherical cap (convex) shape, whereas a truncated cylinder bounded by hemispherical caps at its ends has three CSMPs, one with cylindrical ridge shape and the other two of spherical cap shape.

A n-faced convex polyhedron would contain n CSMPs of planar shape, separated by edges of surface normal discontinuities (shape index is indeterminate for these edges). In theory, edges and vertices would form their own patches (CSMPs) in COSMOS. An edge has infinite curvature in one direction (between the faces it separates) and finite curvature in the other direction (along the edge). As a special case, in a polyhedron an edge can be viewed as the limit of a cylinder whose radius approaches zero (implying that curvedness has approached infinity while shape index has stayed constant). Thus each edge forms a CSMP of its own (recall that planar faces have indeterminate shape index, which is representationally distinct from the shape index of a cylinder) and would separate each face. Vertices would not aggregate into either edges or faces since they have a different shape index—they can usually be viewed as the limiting case of a cup or cap with curvedness approaching infinity. Therefore, a polyhedron theoretically gets segmented into its individual faces in COSMOS. Some of the issues that need to be addressed in practice are discussed in Section 3.5.2.

In a digital implementation of the representation scheme with the surface depth data of an object obtained using a laser range scanner that produces data on a $X-Y$ grid, a CSMP would be computed as a region containing surface points (pixels) whose shape indices are the same and which are *eight-connected* with one another. With other types of surface data, the connectedness can be suitably defined in order to determine the CSMPs on the object. The shape index of a CSMP is the same as that of the surface points contained within it. Note that when the shape index varies at each surface point on the object, each CSMP contains a single point instead of a set

of points. Figure 3.11 shows the constant-shape maximal patches detected in a range image of a view of a vase, obtained using the segmentation technique described in Section 3.5.1.



(a) (b)

Figure 3.11: Maximal patches of constant shape index (colors indicate different shape index values): (a) Range image of a vase; (b) CSMPs detected on the vase.

The COSMOS representation of an object segmented into a set of homogeneous patches (CSMPs) is comprised of two sets of functions discussed below: the Gauss patch map and surface connectivity list $\langle G_0, V \rangle$, and support functions $\langle G_1, G_2 \rangle$. The first set captures the orientation and connectivity information of the object and the latter captures salient local surface information of the CSMPs.

Given an object O segmented into a set of patches $\mathcal{P}_\mathcal{O}$ according to the CSMP criterion, we define the Gauss patch map as a function $G_0 : \mathcal{P}_\mathcal{O} \rightarrow \mathcal{S}^2$ into the unit sphere ($\mathcal{S}^2$) such that

$$\vec{G_0}(P) = \frac{\iint_{p \in P} \hat{n}(p)\, dO}{\iint_{p \in P} dO}, \tag{3.7}$$

where $\hat{n}(p)$ is the normal at $p \in P$, dO is the differential area element in P and $P \in \mathcal{P}_\mathcal{O}$. The integral $\iint_{p \in P} dO$ denotes the total surface area of the CSMP P. $\vec{G_0}(P)$ is the average surface normal over P. For example, in a computer implementation

with discretely sampled object surface data , this will be computed as

$$\begin{cases} \hat{n}(p) & \text{for} \quad P = \{p\} \\ \frac{1}{N}\Sigma_i \hat{n}(p_i) & \text{for} \quad P = \{p_1, \cdots, p_N\}. \end{cases}$$

When we refer to the "unit sphere" we also include an extra point, the center of the sphere to which we map zero normals. Thus G_0 maps each CSMP, P, on O to a point on the unit sphere whose normal corresponds to the *orientation* (mean surface normal) of the patch P. Viewed inversely, the Gauss patch map associates each point $s \in \mathcal{S}^2$ on the unit sphere with a (possibly empty) *set of patches* at s,

$$G_0^{-1}(s) = \{P \in \mathcal{P}_O \mid G_0(P) = s\}. \tag{3.8}$$

As a notational convenience, we generalize this and define the inverse Gauss patch map of a *region* of the unit sphere $S \subseteq \mathcal{S}^2$ as

$$G_0^{-1}[S] = \{P \in \mathcal{P}_O \mid \quad \exists s \in S, \quad G_0(P) = s\}. \tag{3.9}$$

Figure 3.12 shows a telephone handset, the CSMPs on the object and their spherical mapping as given by G_0.

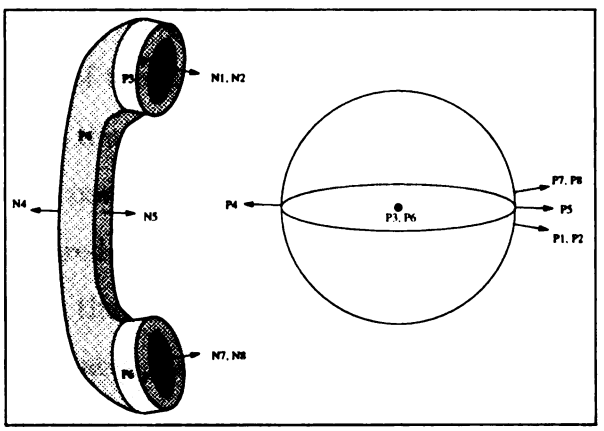


Figure 3.12: Example of a 3D free-form object and its spherical mapping $G_0(P)$.

The surface connectivity list $V : \mathcal{P}_{\mathcal{O}} \rightarrow 2^{\mathcal{P}_{\mathcal{O}}}$ is defined as

$$V(P) = \{Q \in \mathcal{P}_{\mathcal{O}} \mid Q \neq P, \quad \forall(\delta > 0)\ \exists(p \in P, q \in Q) \quad \|p - q\| < \delta\}, \qquad (3.10)$$

where $2^{\mathcal{P}_{\mathcal{O}}}$ is the power set of $\mathcal{P}_{\mathcal{O}}$ and $\|p - q\|$ is the Euclidean distance between the points p and q. That is, V associates each patch $P \in \mathcal{P}_{\mathcal{O}}$ with the set of patches $V(P) = \{Q\} \subseteq \mathcal{P}_{\mathcal{O}}$ that are adjacent to P, and thus represents connectivity information about the segmented object.

It can be seen that the traditional *region adjacency graph* data structure can be easily abstracted from the set of CSMPs, $\mathcal{P}_{\mathcal{O}}$, and the surface connectivity list V of an object, where each CSMP P on the object serves as a node in the graph and $V(P)$ provides information about the edges (or the connectivity) that link the nodes in the region adjacency graph.

The orientation of the patches determined by G_0 associates the CSMPs with points on the unit sphere. The mapping G_0 for any given convex object is one-to-one since surface normals are unique on the convex object and no two CSMPs on the object will have the same surface orientation. However, in the case of a nonconvex object, it is possible to have identical surface normals at multiple points on the object's surface (see Figure 3.12). So, several CSMPs may map to the same point on the unit sphere, leading to multiple folds on the sphere. The surface connectivity list V maintains the connectivity information of the patches by keeping a list of adjacent patches for each patch on the object, and thus identifies each fold with unique information that is in many cases useful for a coarse approximation of the object's surface, even when full recoverability is not assured.

Furthermore, observe that for symmetrically closed smooth objects containing a single CSMP, the Gauss patch map of the object is a single point on the unit sphere, its center. In a practical implementation, this mapping may turn out to be unstable depending on how the object is discretely sampled and also on the amount of noise that may be present in computing the surface normal. However, as will be shown in Section 3.5, we have adopted an object model which is a collection of views of an

object and hence only 2D views (appearances) of objects are dealt with in our system. Since 2D object views need not provide complete information on the object (e.g., the back of the object may not be visible in some views), we are not likely to encounter situations where all the surface normals sum to zero and thus, the problem of possible instability of the Gauss patch map of objects subject to sampling is avoided in our implementation.

While the domain of the first set of functions was $\mathcal{P}_O$, i.e., G_0 and V are defined over the object itself, the second set of functions are defined over the unit sphere $\mathcal{S}^2$. In essence, these functions summarize at each point in $\mathcal{S}^2$ the local information about all the patches that have been mapped by G_0 to the point.

Let $s \in \mathcal{S}^2$ be a point on the unit sphere and $dS(s)$ be a small neighborhood of s on $\mathcal{S}^2$. Several CSMPs from different parts of the object O may have been mapped to s by G_0, i.e., the Gauss patch map may have several folds over s. Then $G_0^{-1}(s)$ is the set of patches from O mapped at s and $G_0^{-1}[dS]$ is the inverse Gauss patch map of dS, i.e, the set of all patches of O that map to points within dS. For any patch $P \in G_0^{-1}(s)$, let us denote the restriction of $G_0^{-1}[dS]$ to neighbors of P as

$$G_0^{-1}[dS]\big|_P = G_0^{-1}[dS] \cap V(P). \tag{3.11}$$

This restriction, $G_0^{-1}[dS]\big|_P$ defines a single continuous region of O (since all the patches in $G_0^{-1}[dS]\big|_P$ are neighbors of P, they are joined into one large patch whose image is entirely within dS),

$$dO|_P = \bigcup_{Q \in G_0^{-1}[dS]\big|_P} Q, \tag{3.12}$$

corresponding to a single fold over s. We then define the *patched* Gaussian object density $D(s|\,P)$ on the single fold corresponding to P as

$$D(s|\,P) = \lim_{dS \to 0} \frac{dO|_P}{dS}. \tag{3.13}$$

Note that $D(s|\,P)$ becomes equal to the inverse of Gaussian curvature of a point on the object when the patch P consists of a single point. When the patch is of finite nonzero area, $D(s|\,P)$ can be written as the product of the surface area of the patch and a Dirac delta function whose spike is at s (see Section 3.2.3 for a definition of the Dirac Delta function), as explained later.

We define the two support functions, $G_1 : \mathcal{S}^2 \rightarrow \mathcal{C}$ and $G_2 : \mathcal{S}^2 \rightarrow \mathcal{C}$, where $\mathcal{C}$ is the space of complex-valued functions such that

$$G_1(s,t) = \sum_{P \in G_0^{-1}(s)} D(s|\,P) S_S(P) \tag{3.14}$$

and

$$G_2(s,t) = \sum_{P \in G_0^{-1}(s)} R_m(P) D(s|\,P) S_S(P), \tag{3.15}$$

where $S_S(P)$ is the shape spectral function of a CSMP P, and $R_m(P)$ is the mean curvedness over P given by

$$R_m(P) = \frac{\iint_{p \in P} R(p)\, dO}{\iint_{p \in P} dO}. \tag{3.16}$$

The support functions $\langle G_1, G_2 \rangle$ defined on the unit sphere capture the local geometric attributes of the mapped CSMPs. $G_1(s,t)$ integrated over a region on $\mathcal{S}^2$ maintains a summary of the surface areas of all the mapped CSMPs in each shape category. As a special case, the integral of $G_1(s,0)$ over a region on $\mathcal{S}^2$ provides the surface area of the CSMPs that are mapped into the region. The term $D(s|\,P)$ in G_1 becomes equal to the inverse of Gaussian curvature of a point on the object when the CSMP P is a point, and it is equal to the product of the area and the Dirac delta function when the CSMP is of finite, nonzero area for the following reason. Recollect that the Gaussian curvature K at a surface point p on O, is defined by

$$K(p) = \lim_{dO \rightarrow 0} \frac{dS}{dO},$$

where dS is the area on the unit sphere to which a region $dO \subset O$ has been mapped. The inverse of the Gaussian curvature $(1/K)$ is given by

$$\frac{1}{K(p)} = \lim_{dS \to 0} \frac{dO}{dS}.$$

If the shape index changes everywhere on O, then each single point $p \in O$ becomes a CSMP P on the object. Then G_1, defined at the point on the unit sphere where P is mapped to, is equal to $(1/K(p))e^{\jmath t S_I(p)}$. In the case of objects where the shape index is distinct and constant over different regions, the CSMPs are well-defined and their surface areas are nonzero. Then $\lim_{dS \to 0} \frac{dO}{dS}$ for a CSMP P becomes equal to the surface area of P multiplied by the Dirac delta function because dO is nonzero for P and dS tends to zero; it is equivalent to the area dO of P multiplied by a Dirac delta function $\delta(t)$. Using the sifting property of the Dirac delta function, we define that for CSMPs of nonzero area, G_1 at a point s_0 on the unit sphere is equal to the area of the mapped patch multiplied by the Dirac delta function and also by its shape spectral function. Then, the integral of $G_1(s_0, 0)$ in a region around s_0 on $\mathcal{S}^2$ provides the surface area of the CSMPs that are mapped into the region. From now on, for ease of discussion, when we refer to G_1 defined for patches of finite surface area we will simply state that G_1 contains information about the surface area of the mapped patches.

Similarly, $G_2(s, 0)$ when integrated over a region around a point on the unit sphere and normalized by the area of the mapped patches provides the mean curvedness of the patches mapped into the region. When a point on the unit sphere is the image of multiple CSMPs on the object, $G_2(s, t)$ after integration and normalization provides a weighted summary of the mean curvedness of these patches categorized in terms of the shape index. Note that in the definitions of both G_1 and G_2 at a point on the unit sphere, the area and the mean curvedness of each patch P are multiplied by the shape spectral function, $e^{\jmath t S_I(P)}$. The shape spectral function aids in maintaining the surface area and curvedness of each patch individually without aggregating them if their shapes are different, although many patches may have been mapped to the same

point. When multiple patches of identical shape index map onto the same point on the unit sphere, then their areas and curvedness add up appropriately, resulting in a summary of area and curvedness of each class of shape present in the object as explained in Section 3.2.4. This can be used effectively for indexing during the matching stage of an object recognition system.

The above definitions can be illustrated using a few examples: For a convex polyhedron, each face of the polyhedron becomes a CSMP (the faces are planar everywhere, and the edges bound these surface patches of constant planar shape). The orientations of the CSMPs are given by the surface normals of the planar faces. Each CSMP gets mapped individually to a point (as the object is convex) on the unit sphere. The support function G_1 defined at the points on the unit sphere where the planar faces are mapped specifies the area of the planar faces. Note that for planar shapes, the curvedness R is zero. In the case of general convex objects where several CSMPs of different shapes and areas exist, each CSMP P gets mapped to a unique point s on the unit sphere with $G_1(s,t) = (Area(P))e^{jtS_I(P)}$. In the cases of nonconvex objects, where multiple CSMPs $P_i, i = 1, 2, \cdots$ are mapped to the same point s_0 on the unit sphere, $G_1(s_0,t) = ((Area(P_1))e^{jtS_I(P_1)} + (Area(P_2))e^{jtS_I(P_2)} + \cdots)$. If the shape indices of the CSMPs are the same, then their areas are aggregated. Similarly, the support function G_2 results in a summary of the mean-curvedness of the CSMPs in each shape category.

A suitable transform of the support functions result in high level feature summaries that characterize the local geometric attributes of an object. This will be established in the following sections.

3.2.3 Definition of Shape Spectrum

We define the shape spectrum $H : [0,1] \rightarrow [0,\infty]$ as follows:

$$H(h) = \iint_O \delta(S_I(p) - h)\, dO \tag{3.17}$$

where h is the shape index variable, dO is a small region containing a point p on object O, $S_I(p)$ is the shape index at p, and $\delta()$ is the Dirac delta function. The latter is defined by

$$\int_a^b \delta(x-k)\, f(x)\, dx = \begin{cases} f(k) & \text{if } a \le k \le b \\ 0 & \text{otherwise.} \end{cases}$$

for all functions $f(x)$.[1] As a consequence of its definition, $\delta(x-k)$ is zero everywhere except at $x = k$ where it has an infinite value (a "spike"). Clearly, the delta function has a 'sifting' or 'sampling' property, in the sense that it picks up the value of $f(x)$ at the point where its spike appears. Our objective in defining the shape spectrum is to determine "how much" of the object's surface area has a particular shape index value h, and therefore equation (3.17) utilizes the Dirac delta to accumulate the area of O for each shape index value h.[2]

In practice, we need a discretized definition of the shape spectrum since we work with pixels when we deal with range images of the object; we call this the *shape histogram.* Let us partition the shape index scale $[0, 1]$ into n "bins," such that the kth bin is the half open interval $[(k-1)/n, k/n)$ (the shape index value 1 is included in the nth bin by definition). The value of the shape histogram in bin k is the number

[1] Strictly speaking, for a suitably chosen set of all "test functions' on the domain of x [151], which places no restriction on us in practice.

[2] An alternative way of defining the shape spectrum is as the derivative of a ***shape distribution function***, in analogy with the way probability density functions can be defined from probability distribution functions:

$$H(h) = \frac{d}{dh}\mathrm{SDF}(h) \stackrel{\text{def}}{=} \frac{d}{dh}\left(\iint_O u(S_I(p) - h)\, dO\right)$$

where $u(x)$ is the unit step function (also known as the Heaviside function):

$$u(x) = \begin{cases} 1 & x \ge 0 \\ 0 & x < 0. \end{cases}$$

Thus $\mathrm{SDF}(h)$ is the surface area of O that has shape index value less than or equal to h. Moving the differentiation inside the integral and observing that $du(x)/dx = \delta(x)$, we obtain equation (3.17).

of pixels whose shape index falls in that bin:

$$H(h = \frac{k}{n}) = \sum_{i=1}^{N} \chi_k(S_I(p_i)) \tag{3.18}$$

where p_i is a pixel, N is the total number of object pixels in the range image, and χ is the characteristic function of a bin:

$$\chi_k(x) = \begin{cases} 1 & \frac{k-1}{n} \leq x < \frac{k}{n} \\ 0 & \text{otherwise} \end{cases}$$

While equation (3.17) is the precise definition required for our theory, equation (3.18) is not as important, in the sense that we are free to employ other discretizations that may display algorithmically better properties. For example, instead of "binning" the shape index scale into n fixed-width bins, we often employ a more flexible binning scheme which has narrower bins (i.e., "higher resolution") near values of the shape index where a large number of pixels accumulate. Similarly, when we work with patches (the CSMPs used to segment an object) explicit bin boundaries are stored in each patch and inverted to approximate the shape spectrum. All these practical methods are qualitatively equivalent to the concept of "how much of the object has a given shape value," formally embodied in equation (3.17). For computational purposes such as comparing spectra of two different objects (Chapter 4), we use a normalized (with respect to the object area) shape histogram where each bin contains *percentage* surface area of the object.

3.2.4 Relationship between Shape Spectrum and G_1

As has been signaled by the term "spectrum" itself, there is a fundamental connection between the support function $G_1(s,t)$ and the shape spectrum $H(h)$: the shape spectrum is the Fourier transform of the integral of G_1 over the entire unit sphere. In other words, if "shape" is a "frequency-domain" concept, $g_1(t) = \iint G_1(s,t)\, dS$ is the corresponding "time-domain" concept. Indeed, this is what originally motivated

our definition of the support function G_1; G_1 spreads the information over the unit sphere, which makes it much easier to work with patches (our CSMPs), whereas H spreads the information over the shape index scale, which makes it much easier to work with object view correlations.

To formally prove the relationship between G_1 and H, we begin with the integral of G_1 summarizing it over the entire unit sphere $\mathcal{S}^2$:

$$g_1(t) = \iint_{\mathcal{S}^2} G_1(s,t)\, dS = \iint_{\mathcal{S}^2} \sum_{P \in G_0^{-1}(s)} \left[\lim_{dS \to O} \frac{dO|_P}{dS} \right] e^{\jmath t S_I(P)}\, dS \qquad (3.19)$$

To simplify the right hand side, we make the following observations. Whereas in a normal integral we would replace $(dO/dS)dS$ by dO, the term $(dO|_P/dS)dS$ is included, because we wished to consider point sets P on each fold in $G_0^{-1}(s)$ *separately.* This need to consider each fold of O separately is also the reason for the presence of the summation: the right hand side of the definition of $G_1(s,t)$ (equation 3.14) aggregates all the CSMPs that get mapped to a point s on the unit sphere. We now note that the integral is over the *entire* unit sphere $\mathcal{S}^2$, and therefore each fold (and CSMP P) on O will be duly considered in turn, and will be considered exactly once. Therefore, in changing domain of the integral to dO, we may safely drop the restriction $dO|_P$, as well as the summation sign, and just traverse the entire object O. For the same reason, we can change the variable and integrate over points $p \in O$ instead of considering point sets $P \in G_0^{-1}(s)$. Thus, the above equation can be simplified to

$$g_1(t) = \iint_O e^{\jmath t S_I(p)}\, dO.$$

Taking the Fourier transform of g_1 with h as the transform variable,

$$\begin{aligned}
\mathcal{F}(g_1(t)) &= \int_{-\infty}^{\infty} \left(\iint_O e^{\jmath t S_I(p)}\, dO \right) e^{-\jmath t h}\, dt \\
&= \iint_O dO \int_{-\infty}^{\infty} e^{-\jmath t (S_I(p) - h)}\, dt \\
&= \iint_O dO\, (2\pi\, \delta(S_I(p) - h)) \\
&= 2\pi\, H(h),
\end{aligned}$$

where we have used the facts that the Fourier transform of $e^{\jmath \alpha t}$ is $2\pi\, \delta(\alpha - h)$ for any constant α, and $\delta(x) = \delta(-x)$. The factor of 2π is of no particular significance as it can be made to disappear with an alternative definition of the Fourier transform (see e.g., the treatment in [151]).

In passing, we observe that the above treatment of "shape" as an analogue of "frequency" explains our need to relocate the shape index scale to the interval $[0, 1]$ from Koenderink's original $[-1, +1]$. When all the "frequencies" are nonnegative numbers, terms with different frequencies in the expressions for G_1, H, etc. do not interact with each other. On the other hand, transform theory makes it clear that two spikes, one at -1 and another at $+1$, do interact to define a single signal, the sinusoid. We need to keep the shape content in an object at $S_I = -0.5$ (rut shape) distinct from (i.e., unaggregated with) the shape content at $S_I = +0.5$ (ridge shape), and hence the redefinition of the shape index.

Observe that a similar formulation can be used to define "how much" of the object's curvedness has a particular shape index value h, and this can be seen to provide a qualitative measure of the scale of each surface shape present in the object.

3.3 Properties of the COSMOS Representation

In this section we briefly highlight some of the important properties of the COSMOS representation. Table 3.2 discusses the features of various orientation-based descriptors and contrasts them with the COSMOS scheme. As detailed in Section 2.1.2, all the other orientation-based descriptors are verbose from the point of view of recog-

nition and their main emphasis is on abstracting an object description that can aid in the recoverability of the object. The matching can fail when parts of objects are occluded as only spherical maps of the objects are available to establish object similarities. However, COSMOS provides local surface features, CSMPs and their various local and global information that can be potentially used for recognition even when objects are occluded.

Some important issues in the design and evaluation of a representation scheme as put forth by Requicha [134] are as follows: (i) object domain (objects that can be modeled with the scheme), (ii) validity of the representation (i.e., whether it avoids nonsensical representations), (iii) uniqueness and completeness (whether the mapping from objects to representations is one-to-one) and (iv) conciseness of the representation. The properties of the COSMOS scheme are discussed and evaluated with respect to each of these issues.

Although the decomposition of an object in terms of the surface patches has been studied earlier, the COSMOS scheme is new because of its use of the continuous-scale shape index for segmenting and matching free-form objects. The continuous scale increases the expressiveness of our scheme and makes it suitable for representing complex objects. Observe that the previous approaches which use the signs of the Gaussian and mean curvatures [10, 109] can describe a surface in terms of only eight categories (based on user-specified thresholds) whereas the shape index scale can be divided into as many classes as needed to represent an object depending on the complexity of its shape. In addition, the mapping of the segmented CSMPs on the object surface based on their mean surface normals onto a unit sphere is novel. The representation combines both local and global aspects of describing an object via support functions defined on the unit sphere and the CSMP connectivity that can be used for reconstruction and recognition of surfaces.

A COSMOS of an object is independent of the position of the origin of the object coordinate system. It is independent of the position of the origin because orientations of the CSMPs get projected on the unit sphere in parallel transformation. The scale information is captured in terms of the curvedness of the maxpatch. A rotation of

Table 3.2: COSMOS and orientation-based representations.

Representation scheme	Mapping	Support functions on the unit sphere	Applicable object domain	Salient features
EGI [85]	spherical mapping of surface normals at all points	Gaussian curvature at a surface point	convex closed objects	objects can be recovered; translation cannot be recovered uniquely.
SFBR [118]	spherical mapping of surface normals at all points	distance of the tangent plane at a point from a predefined origin	convex closed objects	closed surfaces can be uniquely determined; representation depends on choice of origin.
CEGI [96]	spherical mapping of surface normals of all points	complex number storing the Gaussian curvature and the distance of a point from a specified origin in the direction of the normal	convex closed objects	the translation of a convex object can be recovered uniquely.
GGI [109]	spherical mapping of surface normals of constant Gaussian curvature patches	connectivity information	convex and nonconvex objects	nonconvex objects can be represented uniquely and recovered.
OBR [108]	the Gauss map (surface normals) and the dilation map	distance function; first and second curvature functions; radial distance function	convex and star shaped objects	convex and star shaped objects can be recovered from their representations.
COSMOS [45]	spherical mapping of orientations of CSMPs	the Gaussian curvature; mean curvedness	convex and nonconvex objects	shape-based analysis of objects; connectivity information is maintained as a list; convex polyhedra and convex objects whose shape index varies continuously can be recovered.

the coordinate systems is reflected by the rotation of the orientations of the CSMPs mapped to the unit sphere.

3.3.1 Compactness

The COSMOS representation compactly captures the geometry of an object in terms of a set of easily detectable maximal patches. Objects of simple shapes such as polyhedra, cylinders, and spheres have very compact representations. For example, for a full sphere, we have a single CSMP ($S_I = 1$) and it is mapped to the center of the unit sphere ($\zeta = 0, \eta = 0$) which is then associated with the appropriate support functions for area and curvedness. The adjacency set V is empty as there is only a single CSMP. This is compact in comparison with other orientation-based descriptors where every surface normal of a closed convex object would be mapped to the sphere. It is also compact for many classes of objects that contain only a few distinguishable surface patches of constant shape index, i.e, whose surface shapes do not change rapidly over large regions of the object. If an object is complex and composed of different shapes with rapid protrusions and indentations, then its shape complexity is reflected in an increased number of CSMPs on the object.

3.3.2 Convex Objects

The COSMOS representation is equivalent to the EGI scheme in the case of convex polyhedra. For a convex polyhedron O, each planar face (f_i) of the polyhedron becomes a maximal patch of constant shape separated by edges consisting of points of surface normal discontinuity, $f_i \in O \Leftrightarrow P_i \in O$. G_0 maps each P_i to the unit sphere, based on the direction of its surface normal (orientation). The adjacencies of the planar faces are stored in V. Since the curvedness of a planar surface is zero, the support function $G_2 = 0$ for all patches P_i (i.e., for all faces f_i in the polyhedron) and G_1 provides the area of each of the planar faces. Thus COSMOS for a convex polyhedron is a mapping of the normals of each of the planar faces of the polyhedron onto the unit sphere, along with their associated areas and connectivity (denoted by

A and V) as shown in Figure 3.13. The EGI representation for the polyhedron is also shown in this figure.

We can now define a one-to-one mapping $F : \mathcal{S}^2{}_{\text{COSMOS}} \rightarrow \mathcal{S}^2{}_{\text{EGI}}$ between the spherical map $\mathcal{S}^2{}_{\text{EGI}}$ generated by the EGI representation and $\mathcal{S}^2{}_{\text{COSMOS}}$ by the COSMOS description, such that every normal defined on the $\mathcal{S}^2{}_{\text{EGI}}$ has a normal from $\mathcal{S}^2{}_{\text{COSMOS}}$ associated with it. The support function stored at the normals on $\mathcal{S}^2{}_{\text{EGI}}$ is also included among the support functions stored at the points on $\mathcal{S}^2{}_{\text{COSMOS}}$. Thus, COSMOS is isomorphic to EGI for a convex polyhedron.

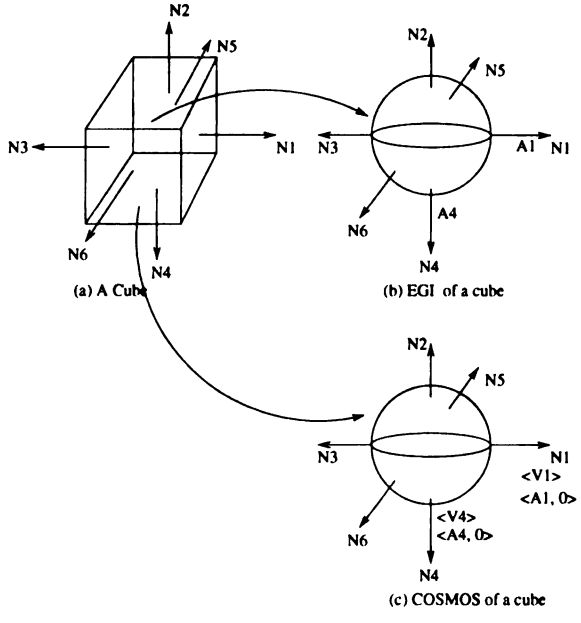


Figure 3.13: COSMOS and EGI of a convex polyhedron. (The support functions are shown only for normals N1 and N4 for clarity.)

The COSMOS reduces to EGI in the case of a continuous smooth convex closed object in which $S_I(p)$, $p \in O$ is different at every point p on the object.

Let O be an object that is convex (for a convex object $S_I \geq 0.5$.) and whose $S_I(p)$ differs at every point $p \in O$; then a maximal patch P_i with constant shape index reduces to a point p_i as the shape index is not constant over any neighborhood around p_i. The orientation of P_i is then the same as the normal N_{p_i} at p_i. Then each

p_i on the surface gets mapped to a point on the unit sphere by the function G_0,

$$\vec{G_0}(P_i) = \vec{G_0}(p_i) = N_{p_i}.$$

As explained earlier, we associate the inverse of the Gaussian curvature of the point $1/K(p_i)$ as the coefficient of the support function G_1. G_2 stores the curvedness of the point, $R(p_i)e^{jtS_I(p_i)}$.

Since the EGI of a closed smooth convex object is a spherical map $\mathcal{S}^2{}_{\text{EGI}}$ containing the normal at each point p on the object, along with a support function of $1/K(p)$, it can be observed that we can define a function $F_1 : \mathcal{S}^2{}_{\text{COSMOS}} \rightarrow \mathcal{S}^2{}_{\text{EGI}}$ that associates each point on the spherical map $\mathcal{S}^2{}_{\text{COSMOS}}$ to that on $\mathcal{S}^2{}_{\text{EGI}}$. The support function stored on $\mathcal{S}^2{}_{\text{EGI}}$ is included among the support functions stored at the points on $\mathcal{S}^2{}_{\text{COSMOS}}$.

Thus, both convex polyhedra and smooth convex closed objects whose shape index varies continuously are recoverable (up to a translation) from their COSMOS representations as the surface recoverability theorems of *EGI* apply in these cases.

Though it may be possible that the COSMOS representation of a convex object containing many CSMPs that have nonzero area is unique up to a translation, the recoverability is not guaranteed. The uniqueness property appears intuitively plausible as no two surface normals have the same direction on a convex object, leading to a non-identical mapping of the orientation of each patch on the unit sphere. Thus, the mapping of a convex object in terms of its maximal patches onto the unit sphere is one-to-one. A coarse approximation of an object in terms of its CSMPs can be reconstructed.

3.3.3 Nonconvex Objects

In dealing with a nonconvex object, global information or adjacency is required due to the nonuniqueness of the surface normals on the object. The main difficulty with EGI and CEGI in representing nonconvex objects is that the connectivity (neighborhood) information between surface points is lost. Global connectivity or adjacency

information of the surface points is required to reconstruct the object unambiguously.

Consider two objects, one convex and the other nonconvex as shown in Figure 3.14. Both the surface normals $N1$ and $N2$ on Object2 map to the same point on the unit sphere with its EGI representation. Object1 and Object2 have identical EGI representations if we assume that the surface areas stored at the points on the unit sphere corresponding to these normals are the same. This example clearly demonstrates that

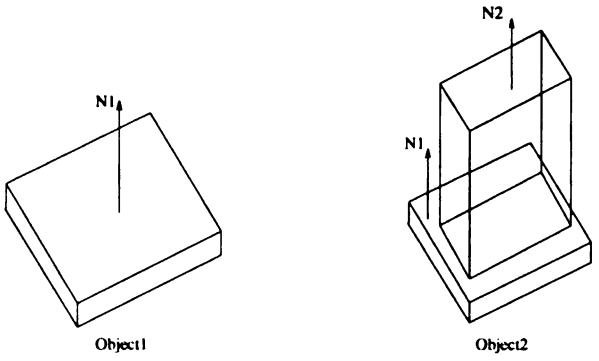


Figure 3.14: A convex (Object1) and a nonconvex object (Object2) that have identical EGI representations.

it is impossible to uniquely recover a nonconvex object from its EGI.

It may be possible that the COSMOS representation of a nonconvex object containing many CSMPs that have nonzero surface area can be used to reconstruct the object coarsely in terms of the patches up to a translation. However, we hasten to add that the recoverability is not always guaranteed. Recollect that our representation of an object not only has the local descriptions of the CSMPs on the object, but also maintains their adjacencies explicitly. Since the connectivity information between the patches is captured, even when more than one patch maps onto an identical point on the unit sphere, there is a sufficient amount of information preserved to obtain the inverse map of the unit sphere unambiguously. The surface adjacency information helps in discriminating between different patches with identical orientation. However, note that since we use the average surface normals in our representation, information is lost about the orientation of each surface point and this can easily complicate the recoverability of the object. Our representation does not model the holes that may be present in an object and hence, recoverability of such an object from its COSMOS

representation is not possible.

3.4 3D Objects and their COSMOS Representations: Examples

In this section, we give examples of 3D object surfaces and their COSMOS representations. Note that we omit an explicit mention of the Dirac delta function in specifying the support functions G_1 and G_2 on the unit sphere for patches of finite, nonzero area.

3.4.1 Simple Objects

- A sphere of radius a, $(a \geq 1)$: Its COSMOS representation is shown in Figure 3.15. The entire object is described by a a single patch $P1$, whose surface orientation

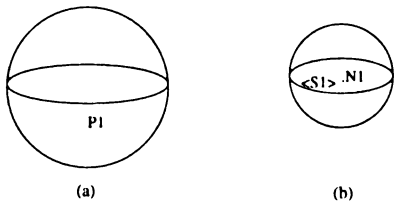


Figure 3.15: COSMOS representation: (a) a sphere of radius a, $(a \geq 1)$; (b) the Gauss patch map with the support functions indicated by $\langle S1 \rangle$.

$N1$ maps it to the center of the unit sphere $(0,0,0)$. The shape index of $P1$ is that of a spherical cap, with $S_I = 1$ (convex umbilic). The support functions indicated by $\langle S1 \rangle$ in Figure 3.15 at $(0,0,0)$ are $G_1 = (4\pi a^2)e^{\jmath t}$ and $G_2 = \frac{1}{a}e^{\jmath t}$. Note that the adjacency set V is empty as there is only a single patch on the object.

- A convex polyhedron: The object has six CSMPs, each of which is a planar shape and mapped to the unit sphere as shown in Figure 3.16. The surface connectivity information and the support functions indicated by $\langle Si \rangle, i = 1, \cdots, 6$ in Figure 3.16 are listed in Table 3.3. Note that for a planar patch P, $e^{\jmath t S_I(P)}$ is

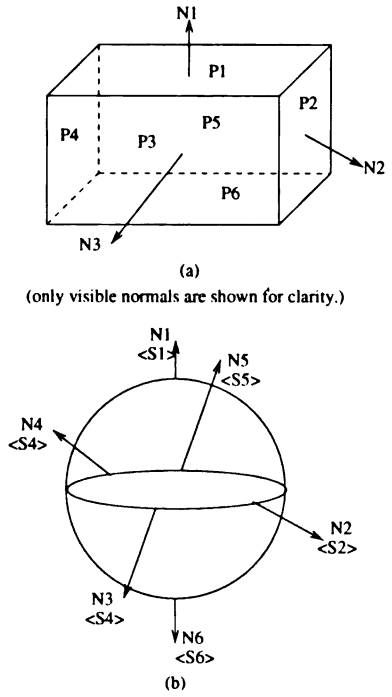


Figure 3.16: COSMOS representation. (a) a convex polyhedron; (b) the Gauss patch map with the support functions.

defined to be $e^{j\pi/2}$ as the shape index of a planar shape is indeterminate and it is assigned an arbitrary value of 2.0 in our implementation. Observe that the curvedness of a planar patch is zero, resulting in $G_2 = 0$ for all CSMPs on this object.

- A nonconvex polyhedron: The eight planar shaped CSMPs are mapped to the unit sphere as shown in Figure 3.17. The support functions on the unit sphere denoted by $\langle Si \rangle, i = 1, \cdots, 8$ in Figure 3.17 are specified in Table 3.4. Observe that the CSMPs $P2$ and $P4$ map to the same point on the unit sphere, thus causing a fold in the spherical mapping. However, the surface patch adjacency information stored explicitly in the connectivity list V for each CSMP aids in distinguishing them as different patches on the object.

- A truncated cylinder with spherical caps: This object has 3 CSMPs; one of cylindrical ridge shape, and two patches of spherical cap shape. The cylinder has a radius of a and a height of h. The spherical mapping of the orientations

Table 3.3: Surface connectivity and support functions on the unit sphere for a convex polyhedron.

CSMP P	Connectivity list $V(P)$	$\iint G_1$ Surface area	$\iint G_2$ Mean curvedness
$P1$	$\{P2, P3, P4, P5\}$	A1 e^{jt2}	0
$P2$	$\{P3, P1, P5, P6\}$	A2 e^{jt2}	0
$P3$	$\{P2, P4, P1, P6\}$	A3 e^{jt2}	0
$P4$	$\{P3, P5, P1, P6\}$	A5 e^{jt2}	0
$P5$	$\{P2, P4, P1, P6\}$	A5 e^{jt2}	0
$P6$	$\{P2, P3, P4, P5\}$	A6 e^{jt2}	0

Table 3.4: Surface connectivity and support functions on the unit sphere for a non-convex polyhedron.

CSMP P	Connectivity list $V(P)$	$\iint G_1$ Surface area	$\iint G_2$ Mean curvedness
$P1$	$\{P2, P5, P6, P7\}$	A1 e^{jt2}	0
$P2$	$\{P1, P3, P5, P7\}$	A2 e^{jt2}	0
$P3$	$\{P2, P4, P5, P7\}$	A3 e^{jt2}	0
$P4$	$\{P3, P5, P7, P8\}$	A4 e^{jt2}	0
$P5$	$\{P2, P3, P4, P6, P8, P1\}$	A5 e^{jt2}	0
$P6$	$\{P1, P5, P7, P8\}$	A6 e^{jt2}	0
$P7$	$\{P1, P2, P3, P4, P6, P8\}$	A7 e^{jt2}	0
$P8$	$\{P4, P5, P6, P7\}$	A8 e^{jt2}	0

of the patches is shown in Figure 3.18. Note that the orientation of the entire cylindrical patch reduces to $(0, 0, 0)$ and hence maps to the center of the unit sphere. The support functions on the unit sphere denoted by $\langle Si \rangle, i = 1, \cdots, 3$ in Figure 3.18 and the surface adjacency list are listed in Table 3.5.

- A truncated cylinder with planar ends: It contains 3 CSMPs; one of cylindrical ridge shape, and two patches of planar shape. The cylinder has a radius of a and a height of h. The spherical mapping of the orientations of the CSMPs is shown in Figure 3.19. The orientation of the cylindrical surface in the object

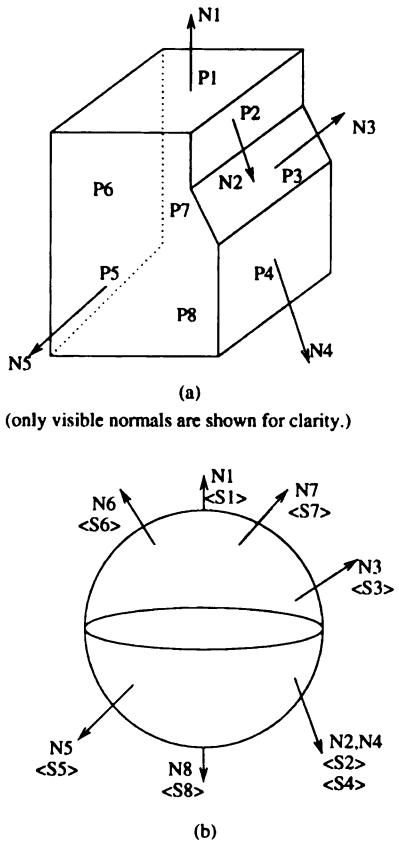


Figure 3.17: COSMOS representation. (a) a nonconvex polyhedron; (b) the Gauss patch map with the support functions.

maps to the center of the unit sphere. The support functions on the unit sphere denoted by $\langle Si \rangle, i = 1, \cdots, 3$ in Figure 3.19 and the surface adjacency list are listed in Table 3.6. Note that for the objects illustrated in Figures 3.18 and 3.19, only the CSMPs $P2$ and $P3$ are different in their shape index and curvedness values.

- A telephone handset: A simplified drawing of a telephone handset as shown in Figure 3.20 reveals 6 CSMPs; two surface patches of planar shape ($P1, P6$), three CSMPs of cylindrical ridge shape ($P2, P3, P5$), and one of saddle rut shape ($P4$). $P2$ and $P5$ are cylindrical ridge surfaces with a radius of a_1 and a height h_1. $P3$ is a ridge shaped surface with a radius a_2 and height h_2. The spherical mapping of the patches is illustrated in Figure 3.20. Note that the mean orientation of the entire cylindrical patch, for example, $P2$ sums to

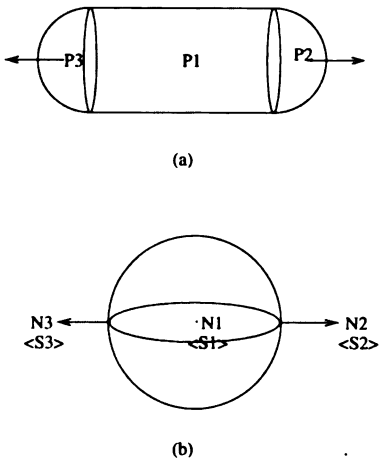


Figure 3.18: COSMOS representation. (a) a truncated cylinder with spherical caps; (b) the Gauss patch map with the support functions.

Table 3.5: Surface connectivity and support functions on the unit sphere for a cylinder truncated with spherical ends.

CSMP P	Connectivity list $V(P)$	$\iint G_1$ Surface area	$\iint G_2$ Mean curvedness
$P1$	$\{P2, P3\}$	$(h\pi a^2)e^{\jmath t 0.75}$	$(\frac{1}{\sqrt{2}a})e^{\jmath t 0.75}$
$P2$	$\{P1\}$	$(2\pi a^2)e^{\jmath t}$	$(\frac{1}{a})e^{\jmath t}$
$P3$	$\{P1\}$	$(2\pi a^2)e^{\jmath t}$	$(\frac{1}{a})e^{\jmath t}$

$(0,0,0)$ and thus, maps to the center of the unit sphere. The support functions on the unit sphere denoted by $\langle Si\rangle, i = 1, \cdots, 6$ in Figure 3.20 and the surface adjacency list are listed in Table 3.7.

3.4.2 Torus

In this section, we derive the shape index values on a torus as an example of a continuous smooth nonconvex object whose parametric representation is available. The parametric equation of the torus is given by

$$torus(u,v) = ((a + b\cos v)\cos u, (a + b\cos v)\sin u, b\sin v). \tag{3.20}$$

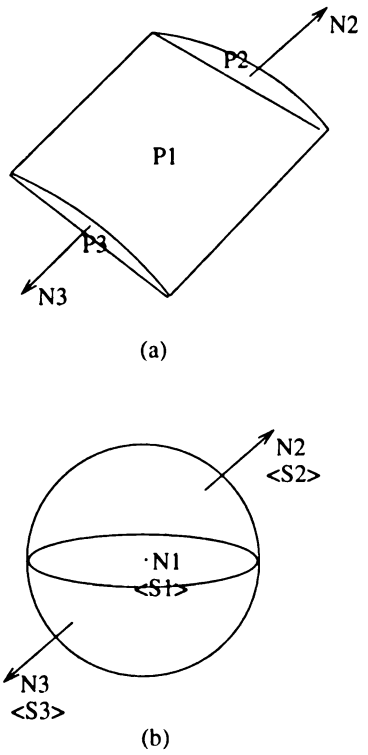


Figure 3.19: COSMOS representation. (a) a truncated cylinder with planar ends; (b) the Gauss patch map with the support functions.

The principal curvatures κ_1 and κ_2 are given by

$$\kappa_1 = -\frac{\cos v}{(a + b\cos v)}, \; \kappa_2 = -\frac{1}{b}.$$

Thus we note that κ_1 of the torus vanishes along the curves given by $v = \pm\frac{\pi}{2}$. These values correspond to convex cylindrical shape category locally. The set of hyperbolic points (saddle-shaped points) is given by

$$\left\{ torus(u,v) \mid \frac{\pi}{2} < v < \frac{3\pi}{2} \right\},$$

and the set of elliptic points on the surface (dome shaped points) is

$$\left\{ torus(u,v) \mid -\frac{\pi}{2} < v < \frac{\pi}{2} \right\}.$$

Table 3.6: Surface connectivity and support functions on the unit sphere for a truncated cylinder with planar ends.

CSMP P	Connectivity list $V(P)$	$\iint G_1$ Surface area	$\iint G_2$ Mean curvedness
P1	$\{P2, P3\}$	$(h\pi a^2)e^{\mathcal{J}t0.75}$	$(\frac{1}{\sqrt{2}a})e^{\mathcal{J}t0.75}$
P2	$\{P1\}$	A2 $e^{\mathcal{J}t2}$	0
P3	$\{P1\}$	A2 $e^{\mathcal{J}t2}$	0

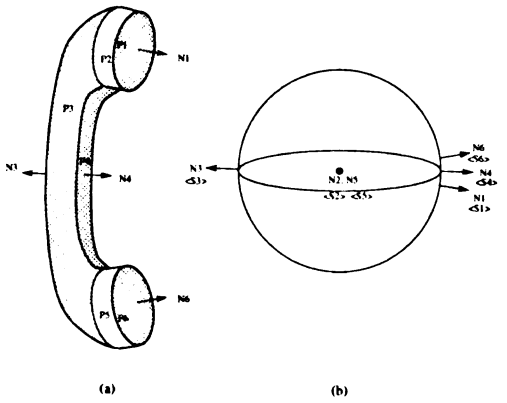


Figure 3.20: COSMOS representation. (a) a telephone handset; (b) the Gauss patch map with the support functions.

Figure 3.21 shows the values of the shape index computed on the surface of the torus. The shape index values are color coded and displayed in Figure 3.21. The blue points are elliptic, the green points are hyperbolic and the parabolic points are shown in red.

3.5 Deriving COSMOS Representation of an Object from Range Data

A complete description of a 3D object may be available as an analytic function, for example, with simple 3D objects such as a sphere, a hyperboloid, or a polyhedron. In such situations, we can construct the COSMOS of the complete object easily. Let the input object be convex or nonconvex. We compute the shape index of the points

Table 3.7: Surface connectivity and support functions on the unit sphere for a telephone handset.

CSMP P	Connectivity list $V(P)$	$\iint G_1$ Surface area	$\iint G_2$ Mean curvedness
$P1$	$\{P2\}$	A1 e^{jt2}	0
$P2$	$\{P1, P3, P4\}$	$(h_1\pi {a_1}^2)e^{jt0.75}$	$(\frac{1}{\sqrt{2}a_1})e^{jt0.75}$
$P3$	$\{P2, P4, P5\}$	$(h_2\pi {a_2}^2)e^{jt0.75}$	$(\frac{1}{\sqrt{2}a_2})e^{jt0.75}$
$P4$	$\{P2, P3, P5\}$	A4 $e^{jt0.375}$	(R4) $e^{jt0.375}$
$P5$	$\{P3, P4, P6\}$	$(h_1\pi {a_1}^2)e^{jt0.75}$	$(\frac{1}{\sqrt{2}a_1})e^{jt0.75}$
$P6$	$\{P5\}$	A6 e^{jt2}	0

on the object using its analytic form, and we then determine the maximal patches of constant shape index present on the object. We compute the area A_{P_i}, the orientation and the mean curvedness of each of the CSMPs P_i, $0 < i \leq n$. The orientation O_{P_i} provides the position (ζ, η) on the unit sphere, thus specifying the spherical mapping of the CSMP. At the mapped points on the unit sphere, the support functions G_1 and G_2 are computed. In the case of a convex object, since the surface normal at each point on the object is unique, the mapping from the object to the unit sphere is one-to-one and G_1 and G_2 take simple forms in their computation. However, in the case of a nonconvex object, it is possible that there are multiple CSMPs whose orientations map to the same point on the unit sphere. In such a case, if the shape index is the same for the multiple patches mapped together, their support functions are added depending on their shape categories as specified in Section 3.2.2.

In practice, however, with free-form surfaces we may not have complete object models nor their descriptions in the form of analytic functions. Hence, we need to build an internal representation of the object from range data of its multiple views. The COSMOS representation of objects derived from complete 3D surface data of objects is viewpoint-independent. However, the COSMOS derived from a given range image of an object is view-dependent due to the following two factors: (i) the orientations of the CSMPs, and (ii) the curved nature of the object. For a point p

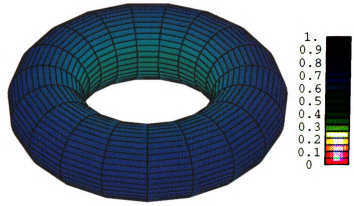


Figure 3.21: Shape index values on the surface of the torus.

on a surface of the object to be visible from a point q in space, the outward-pointing surface normal at p must make a positive projection on the vector from p to q. Surface normals on the "outer side" of a surface project out of the volume occupied by the surface. In the cases of polyhedra, either the planar face is fully visible or it is not at all visible. A planar face is never visible partially because the normals at every point on the planar face make the same angle with the viewing direction. However, in the case of curved objects, this is not necessarily true. Since the normals on a single CSMP can possibly vary, it is possible that only a part of the complete CSMP present on the entire object is visible, and thus the area and the curvedness of the CSMP seen in a specific view of the object are reflective of only what is visible in that view. Hence multiple views are needed to fully describe a curved object.

Given multiple views of an object we compute the COSMOS of each view of the object from its range data. We assume that the scene contains only a single object. This is not a restrictive assumption, as a cluttered scene can be broadly segmented

using depth discontinuities into several connected components, each of which serves as an object of interest. A set of COSMOS representations derived from various views of an object forms a 3D model of a single object. We have adopted a "collection of views" description of an object instead of building the complete COSMOS representation of the object after registering and integrating the multiple views. This is primarily because an input to a recognition system is typically a 2D appearance of an object. Since object identification has to proceed from a single view of an object, our approach is to maintain a set of views in our object database and to match only observed surface patches with those of the model views. The range data of the views of the object are obtained either using a laser range sensor or using the CAD models or surface triangulations, if available. The catalog of possible views of objects is based only on a chosen tessellation of the view sphere. Therefore, the number of views chosen to represent an object is model-driven, based on the complexity of the object. For a simple object, a very coarse tessellation resulting in a few views may be sufficient. The issue of determining the number of views necessary for recognition is addressed in Chapter 4.

3.5.1 Construction of the COSMOS of a Single View of an Object

We first present a scheme to derive the COSMOS representation from the range data from a single view of an object. For the discussion below, we assume that a range image of an object from an arbitrary viewpoint is obtained using a Technical Arts White scanner [66]. The computational scheme outlined is applicable to data obtained using other kinds of range scanners with suitable definitions of surface curvatures, pixel connectedness, etc. The processing steps are shown in Figure 3.22.

The range image obtained using the laser range scanner available in our laboratory provides the surface depth data of the object visible to the camera with the pixels in the image arranged in a cartesian $X - Y$ grid. The accuracy of depth value is of the order of 0.001 inches. Since the camera used in the scanner does not use square

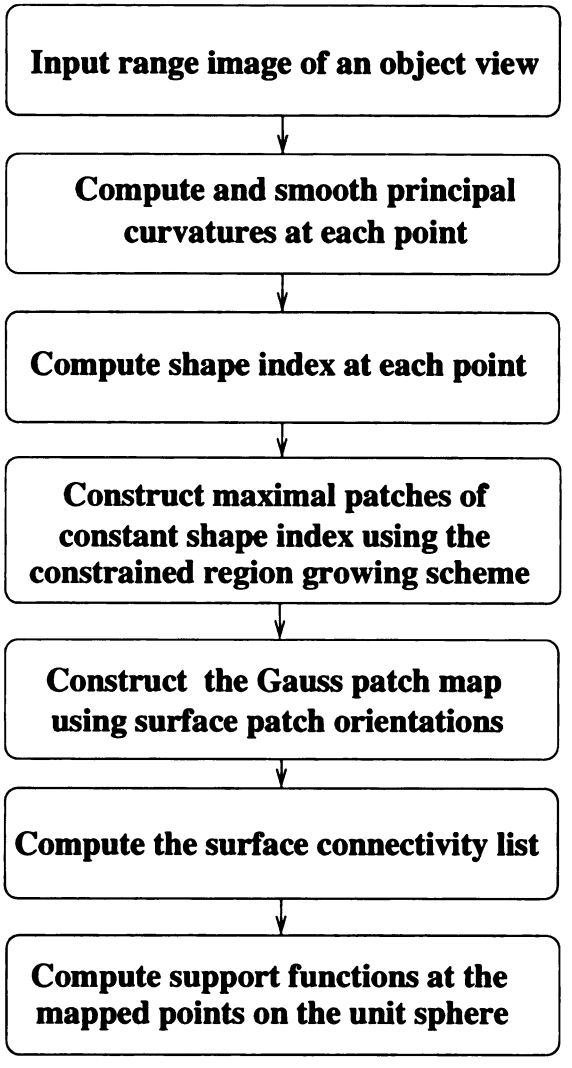


Figure 3.22: Construction of COSMOS from range data of an object.

pixels and since the camera is mounted at an angle to the object to be scanned (it is not placed directly overhead) some views of the object appear compressed in their length while some of the views appear stretched. This is also aggravated by the fact that the image sampling along the X direction is not the same as in the Y direction. Currently, no preprocessing is done on our data to correct for this distortion as our intent is to demonstrate that our recognition system can perform well in spite of the distortions that may be present in the data. We treat these distortions as different forms of noise in the sensed surface data.

We first compute the surface curvatures at each pixel (object point) in the image

by estimating them using a bicubic approximation of the local neighborhood. In our implementation, we use a neighborhood of 5×5 pixels to locally approximate the underlying surface. Then we apply the iterative curvature smoothing algorithm based on the curvature consistency criterion [63] to improve the reliability of the estimated curvatures. We then process all the data by running the curvature smoothing algorithm for 15 iterations and it can be seen from the segmentation results shown in Section 3.5.2 that this smoothing is sometimes excessive for some object views.

Once the curvatures are reliably estimated at each pixel, the shape index values are computed. The next challenging task is to obtain as few maximally connected patches of constant shape index as possible while retaining sufficient information about different shapes that may be present. Since the range data are of finite resolution, and since curvature estimates can be noisy, we need to take into account the possibility of noise in shape index values of surface points belonging to the same shape. The connected points whose shape indices are similar have to be merged to obtain a CSMP.

An obvious approach to find maximal patches of constant shape index in the range image is to partition the shape scale *a priori* (independent of the actual distribution of shape index values in the image) into a finite number of bins of either fixed or varying width. However, with such a method, the bin boundaries are artificial and may not correspond to a more "natural" segmentation of the image. In other words, a region of the image containing pixels with similar shape index values may be split up into two maximal patches just because the values crossed the *a priori* bin boundary. Therefore, the shape index boundaries should depend on the contents of the image. More generally, the problem is to group image pixels into CSMPs with the following objectives: (i) minimize the total number of CSMPs in the image (to avoid fragmentation into hundreds of small sized patches); and (ii) minimize some measure of the spread of shape index values of the pixels within a CSMP. These two objectives obviously conflict. Global information is required to achieve objective (i) subject to some constraints. A brief description of our segmentation algorithm is given below.

3.5.2 Constrained Region Growing

Our segmentation algorithm constructs maximal patches by repeatedly merging smaller patches, starting with very small (pixel-sized) patches. Since in each step it merges the two (adjacent) patches that would result in the least merged *shape diameter* (the difference between the minimum and maximum shape index values within the patch) of all the feasible (connected) pairs of patches, the spread of shape indices of the pixels within a patch is minimized. In addition, since the algorithm is applied until the maximum merged shape diameter would exceed the constrained value for every remaining pair of patches, it minimizes the total number of patches. Using the shape index in this way improves the segmentation over methods that use only fixed-width and fixed-threshold quantized bins. The algorithm can be supplemented with a post-processing step in which purely local adjustments are done to pixels on the boundaries of patches. For example, a boundary pixel can be moved to another patch depending on its influence on the means and deviations of the maximal patches.

To illustrate the effectiveness and the generality of our scheme, we show the maximal patches for different objects in Figure 3.23. In this figure, part (a) shows the range image of an object and part (b) shows the image of the various CSMPs (shown in different colors) on the object. The Gauss map of the CSMPs is shown in part (c). A shape diameter of 0.25 yielded good CSMPs in most images. An increased shape diameter resulted in bigger patches in the cases of the cobra-head and the cup. This parameter can be adaptively adjusted depending on the size of the smallest CSMP that is detected in a given image. If the design philosophy is that the smallest CSMP found in an object view should contain at least 10% of the visible surface area in the view, then the shape diameter can be adaptively adjusted depending on the object present in the view. We note that the perceptual accuracy of the segmentation is not the only criterion to evaluate object segmentation in our case; one can have visually unsatisfactory segmentation (e.g., over-segmentation) which is, however, adequate for object recognition. Our recognition system (Chapter 5) has been designed to handle imperfect segmentation results by merging connected CSMPs if necessary, while

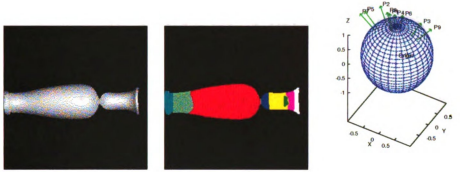
Table 3.8: The COSMOS representation: Support functions for Vase2-1 on the unit sphere.

Point on the unit sphere (ζ, η)	CSMP	Mean shape index	Surface area	Mean Curvedness
$(3.106989, 0.677436)$	P1 (red)	0.854807	5725	0.593455
$(-3.079618, 0.482079)$	P2 (green)	0.720356	112	0.366058
$(0.065550, 0.607387)$	P3 (blue)	0.473433	289	0.454395
$(-2.893622, 0.616532)$	P4 (yellow)	0.656924	793	0.737030
$(3.091251, 0.926560)$	P5 (magenta)	0.576004	343	0.571970
$(0.230828, 0.580261)$	P6 (cyan)	0.821453	520	0.932052
$(-3.129172, 0.983077)$	P7 (white)	0.747665	326	1.105722
$(-3.115084, 0.647045)$	P8 (light green)	0.678569	1088	0.474518
$(0.053756, 0.681923)$	P9 (pine green)	0.666768	17	0.394001

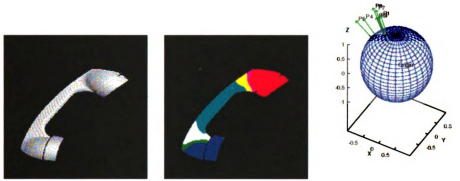
establishing the model-scene patch correspondences during recognition.

A few remarks about the role of edges on the objects (surface depth and surface normal discontinuities) are in order here. In range images of objects, edges are never knife-sharp nor are corners needle-like. Particularly, with free-form sculpted objects, the edges present on the objects are typically smooth, without sudden jumps in the depth or normals (contrast a human face with a polyhedron!). Since our current segmentation algorithm does not explicitly detect edges prior to region growing, our experimental results indicate that the detected CSMPs gracefully blend into neighboring CSMPs without a sharp boundary between them. Future work in improving the segmentation results should integrate edge detection schemes along with the region growing algorithm to obtain stable region boundaries and also prevent leaking of a CSMP across the discontinuities to its surrounding regions.

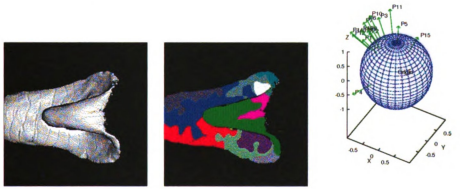
The surface attributes stored by the coefficients of the support functions on the unit sphere for the vase are given (without the shape spectral functions) in Table 3.8. The support functions are defined at points on the unit sphere as indicated in the table. At all other points on the unit sphere, they are undefined.



Vase2-1



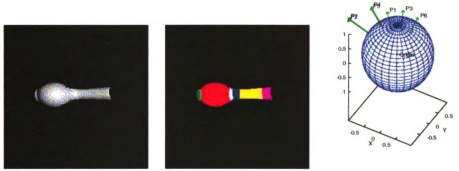
Phone-1



Cobra-1

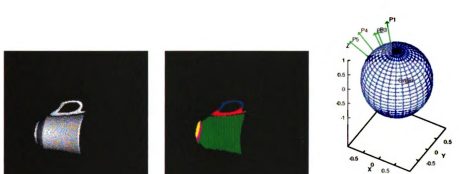
(a) (b) (c)

Figure 3.23: Representation of objects with free-form surfaces: (a) range image; (b) constant-shape maximal patches; (c) the Gauss patch map.



Small-vase-1

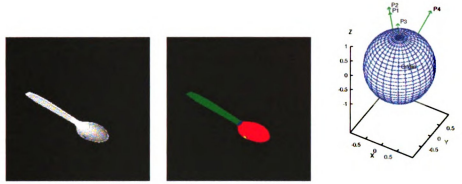
Giraffe-2



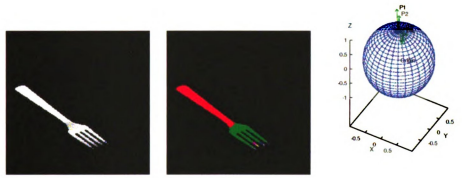
Cup-4

(a) (b) (c)

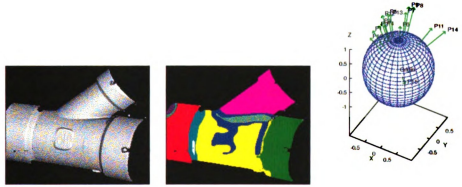
Figure 3.23 (cont'd): Representation of objects with free-form surfaces: (a) range image; (b) constant-shape maximal patches; (c) the Gauss patch map.



Spoon-2



Fork-4



Big-Y-1

(a) (b) (c)

Figure 3.23 (cont'd): Representation of objects with free-form surfaces: (a) range image; (b) constant-shape maximal patches; (c) the Gauss patch map.

3.5.3 Sensitivity Analysis of Shape Index, S_I

When shape index is used to segment the digital range images of objects into CSMPs, it is apparent that any inaccuracy in surface curvature values resulting due to sensor noise and numerical errors introduced by the curvature estimator affect the accuracy of the shape index computed and the CSMPs. The curvatures computed from range images are sensitive to noise in the surface depth values as they are second-order differentials of the surface depth functions.

We studied the sensitivity of the shape index to small changes in the principal curvatures to improve our segmentation results. The sensitivity is defined as the change in the shape index with an infinitesimal change in the principal curvatures, κ_1 and κ_2, and is given by

$$\frac{\partial S_I}{\partial \kappa_1} = \frac{\kappa_2}{\pi(\kappa_1^2 + \kappa_2^2)} \tag{3.21}$$

$$\frac{\partial S_I}{\partial \kappa_2} = -\frac{\kappa_1}{\pi(\kappa_1^2 + \kappa_2^2)} \tag{3.22}$$

The plot of the differential sensitivity of the shape index S_I with respect to both κ_1 and κ_2 is shown in Figure 3.24. It can be seen from Figure 3.24 that when the principal curvatures are zero or near zero, any small variation in their values result in a very large change in the shape index. In fact, when the principal curvatures are nearly zero, the sensitivity goes to infinity as shown in the figure. Thus the shape index is very sensitive to the variations in the curvature values when they are almost zero. This information can be utilized in identifying the planar surfaces. Observe that in practice, to detect the planar points on a surface, we can only assert that the principal curvatures have to be less than some value; they are never zero in an implementation using a digital computer. The difficulty in choosing the threshold near zero is avoided by computing the sensitivity of the shape index at all points, and declaring all those points that have sensitivity values higher than a threshold to be planar points. This threshold is easier in practice to set, as it is a relatively

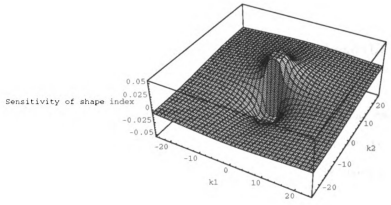
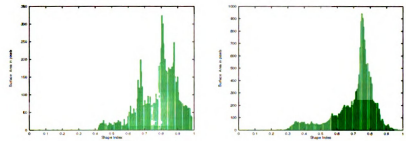


Figure 3.24: Sensitivity of the shape index to principal curvatures.

large value. In addition, pixels where the curvature values are unreliable can also be detected and excluded from further analysis. When the shape sensitivity values computed at the pixels exceed a certain threshold, it indicates that the curvature values are too low to reliably compute shape index. Further processing of such points is then avoided.

3.5.4 Shape Spectrum of an Object View: Examples

As described in Section 3.2.3, the shape spectrum of an object is similar to that of the frequency spectrum of a time-domain signal. It characterizes the shape content



(a) (b)

Figure 3.25: Shape spectra of (a) Vase2-1 shown in Figure 1.3 and (b) Big-Y-1 shown in Figure 1.3.

of an object by summarizing the area on the surface of an object at each shape index value. The shape spectrum of an object view is obtained from its range data by constructing a histogram $H(h)$ of the shape index values—we used 0.005 as the bin width—and accumulating all the object pixels that fall into each bin. Note that the shape spectrum could also have been constructed using the surface area and shape index values of the CSMPs generated by segmentation. However, we preferred to directly use the original shape index values computed for each pixel in the image (instead of using the mean and variance information stored in the CSMPs) and thus avoided being affected by any segmentation imperfections. Under theoretically ideal conditions (noiseless curvatures, zero CSMP shape diameter) these two ways of constructing the spectrum would yield identical results, but in practice the former is desirable. Since the shape index for planar points on an object surface is indeterminate, we have assigned a symbolic label to the planar points for our consideration. For plotting and computational purposes, this label is arbitrarily assigned a value of 2.0, and therefore, the range of shape index values for computing the shape spectrum of a view takes a value in the range $[0, 1]$ and also 2.0.

The non-planar shape spectral plot of the vase (Vase2-1), shown in Figure 3.25(a), indicates that the main shape category present in this object is *dome*, along with a few smaller peaks in the *ridge* shape type and *saddle-ridge* category. From Figure 3.25(b), it can be seen that the dominant shape category in Big-Y-1 object is *ridge* ($0.6875 \leq$

$S_I \leq 0.8125$). A few concavities in the object are characterized by the nonzero bins below the 0.5 shape index level.

It can be observed that based on the shape spectral information alone, it is difficult to discriminate between objects that are only polyhedral. However, the concept of shape spectrum can be effectively used for categorizing objects into purely polyhedral and non-polyhedral; polyhedral object views exhibit a peak (maximum surface area) only at the planar shape index (which has been chosen to be 2.0, in our experiments). Since there is a huge body of techniques available for polyhedral object recognition, we will not emphasize on polyhedral object recognition further. As the main focus of this thesis has been the representation and recognition of free-form surfaces, we will study mainly the use of non-planar shape spectra for grouping object views and for fast matching as discussed in Chapter 4. Whenever, the shape spectral information utilizes the planar surface information explicitly, the reader will be alerted to this fact.

Note that occlusion of an object in a view by other objects or due to self-occlusion results in some of the surface patches being visible only partially in the image and this affects the shape spectrum by influencing the surface area count (percent) stored at various shape categories. We have devised a novel technique for comparing two shape spectra that can tolerate the presence of occlusion in an object to a certain extent and it will be presented in Chapter 4.

3.6 Summary

In this chapter, we have presented our new representation scheme, called COSMOS for 3D objects. It is a shape-based description of objects suitable for representing free-form surfaces without requiring complex 3D analytical modeling of objects. In the next chapter, we discuss how a model of a 3D free-form object can be constructed as a collection of COSMOS representations of its multiple views and also present results on clustering a large number of views of a 3D object into a small number of salient groups that can be used effectively for view matching during recognition. Chapter 5

presents a recognition strategy which consists of a multi-level matching mechanism that employs shape spectral analysis and features derived from the COSMOS representations of objects for fast and efficient object identification and pose estimation. It also presents experimental results on real range images of complex free-form objects.

Chapter 4

Object View Grouping and Model Database Organization

As discussed in Chapter 3, a multiple-view based description of a free-form object has been adopted in this thesis for recognition. A 3D rigid object can give rise to arbitrarily many different 2D appearances (views). For objects with free-form or sculpted surfaces, only a part of a surface will typically be visible from a single viewpoint, due to the object's curvedness. Variations in viewing directions and angles can result in very distinct views of the object, with more of the curved surface(s) either coming into view or disappearing from view. Thus, a sculpted object can give rise to infinitely many different views owing to its smoothly curved nature. In practice, only a finite number of such views can be stored. Therefore, an important issue is which and how many of these object views are actually necessary and useful for *recognition*. The problem we address in this chapter is as follows. Given a set of views of a free-form 3D rigid object, how do we represent and organize these views in a meaningful and efficient manner?

4.1 Object-centered versus Viewer-centered Representations

Previous approaches to 3D object representation can be categorized as either *viewpoint-independent* (object-centered) or *viewpoint-dependent* (viewer-centered). A viewpoint-independent representation attaches a co-ordinate system to an object; all points or object features are specified with respect to this system. The description of the object thus remains canonical, independent of the vantage point. Although, this approach has been favored by Marr and Nishihara [114] and others, it is difficult to derive an object-centered representation from an input image. A unique co-ordinate system needs to be first identified from the input images and this becomes difficult when the object has many natural axes. Practical implementation becomes a complicated task as only 2D or 2.5D information is usually available from a single image and perspective projection effects have to be corrected in the image, before building the representation. Note that this approach is well suited to simple 3D objects that can be specified by analytic functions.

A viewer-centered approach on the other hand, describes an object relative to the viewer and as one does not have to compensate for the viewpoint, view representations can be easily computed from images. A major disadvantage is that a large number of views needs to be stored for each object, since different views of an object are in essence treated as containing distinct objects. However, representing an object with multiple views is quite useful in view-based matching, alleviating the need for expensive 3D model construction.

A viewer-independent scheme suffers more difficulty when representing an object with free-form surfaces. Such an object may neither have a complete geometric model, nor a description in terms of analytic functions. As noted earlier, it may not be an assembly of simple surfaces like planes and cylinders. The constituent surfaces can be of such a high degree that reliable surface segmentation from image data is difficult. Therefore, a practical solution would be to build a multiple view based representation

of the object. However, a sculpted object gives rise to infinitely many different views owing to its curved nature. In practice, only a finite number of such views can be stored. Therefore, an important issue is: which and how many of these object views are actually necessary and useful for *recognition*? The number of views chosen to represent an object for quick and accurate identification entirely depends on the complexity of the object.

This chapter addresses the following question specifically: How do we generate a *representative and adequate grouping* of the views such that a new object view can be indexed effectively and efficiently to one of the stored views in the database? We place emphasis on automatically obtaining the clusters of views without requiring segmentation of object surfaces. Such a set of view-clusters will serve as a view-based representation of each object in the database. Efficient retrieval of a cluster of views provides a small set of plausible correct matches for further refined matching. The term *view* refers to a range image of an unoccluded object obtained from any arbitrary viewpoint. For the purposes of this chapter, two views of an object are not considered distinct if they produce appearances of the object that merely differ from each other by a rotation about the view plane.

4.2 3D Object Model as a Collection of Representations of Multiple Views

When constructing a multiple-view based description of an object, we adopt an "approximate visibility technique" to restrict the set of possible viewpoints to a sphere of large but finite radius, centered around the object. The surface of the viewing sphere is tessellated in a quasi-regular fashion to provide a discrete set of points. Each of these points provides a viewpoint vector in the "approximate visibility" space. Range data of an object surface seen from each of the sampled view directions are obtained using the laser range scanner or from the CAD model or surface triangulation of the object. The collection of representations of these 2D views then constitutes a

multi-view description of the object.

View clustering for a single object has been addressed by several researchers under the topics of characteristic views (CVs) [72, 34] and aspect graphs (AGs) [100]. A lot of research has been directed towards obtaining the aspect graph representation for different classes of objects [55, 54, 74, 75, 103, 126, 34, 145, 149, 148, 169, 56]. Although there exists an extensive body of work dealing with aspect graphs, a major difficulty that confounds the use and implementation of AGs for object recognition is that complicated objects can result in enormous and complex AGs. To derive aspect graphs of manageable sizes, appropriate heuristics need to be designed. The problem of computing the aspect graph of an arbitrary object still remains unsolved. Ikeuchi's practical approach [88] relied on detecting the planar and curved faces of an object using photometric stereo in order to form the aspects of the object containing topologically similar views. However, such an approach is difficult with a free-form object since each viewpoint gives rise to a slightly different view of the object, because of its smoothly curved nature. It is also hard to define a *single face* in a sculpted object. A recent paper [140] organizes the model base hierarchically using parametric structural descriptions built from the CAD models of objects where it is assumed that a complete 3D description of an object is available for its recognition.

4.3 View Sensitivity of the Shape Spectrum

As shown in Chapter 3, with the COSMOS representation scheme, an object's shape and surface area can be characterized quantitatively in terms of its *shape spectrum.* The shape spectrum of an object, derived from complete 3D surface data of the object is viewpoint-independent. However, the spectrum derived from a single range image of an object is view-dependent. The view sensitivity of this high level feature is exploited for object view grouping as shown in section 4.4.1. Figure 4.1 shows a set of object views to demonstrate how shape spectra of views of various objects differ and how spectra computed from range images obtained by observing an object at nearby viewpoints are similar to one another. Recollect that as explained in Section 3.5.4, we

construct the shape spectrum of a view directly using the original shape index values computed for each pixel in its image (thus avoiding segmentation of object into parts). Figure 4.1(b) shows the non planar shape histogram of a view of Vase2 and it indicates that the main shape category present in this view is *dome* ($S_I = 0.875$), along with a few smaller peaks in the *ridge* (0.75) shape type and *saddle-ridge* (0.625) category. Figures 4.1(d) and 4.1(e) show the strong similarities between the spectral plots of two different views of Cobra. These plots also indicate the predominance of *rut* (0.25), *ridge* (0.75) and *trough* (0.125) shapes in Cobra.

Since the spectra of purely polyhedral objects exhibit a single peak at the shape index value of 2.0 (all the planar patches contribute to this bin), it is difficult to discriminate between various views of these objects. However, shape spectrum based classification can be used to categorize object views in a database into two classes: those that are purely planar and those that contain non-planar shapes on the object surfaces.

4.4 Organizing Object Views

We now describe how the non-planar shape spectra of object views (spectra computed without taking the planar points on the surfaces into account) can be used efficiently for (a) view grouping and (b) view matching. Note that when the model database is populated during its construction, the object identities of the views to be stored in the database are known. We first investigate whether multiple views of the same object can be clustered into meaningful groups based on their shape spectra. We have chosen to perform clustering instead of supervised classification to find out whether there is any inherent clustering tendency present among the training set views. Secondly, the object view-grouping can be repeated with each set of object views, and the model database can thus be structured into a collection of distinct groups of views of each object. For example, if n_i object views are used to originally represent an object i, then with the grouping scheme, it may result in n_{i_r} groups of views, where $n_{i_r} << n_i$. We propose to determine the matching efficiency and accuracy by hierar-

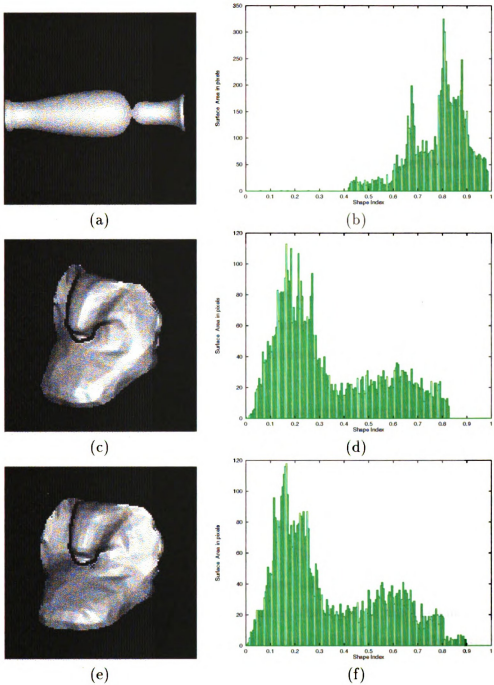


Figure 4.1: Shape spectrum: (a) Range image of Vase2; (b) shape spectrum of Vase2; (c) a view of the cobra head - Cobra-1; (d) shape spectrum of Cobra-1; (e) another view - Cobra-2; (f) shape spectrum of Cobra-2.

chically comparing an input view with the view cluster representatives first, followed by matching it with the views within the clusters themselves. Our primary concern is to structure a large database of object views in order to eliminate matching the input view with all the stored views before the object identity can be ascertained, and to narrow down the possible set of views that need to be matched more comprehensively as described in chapter 5.

4.4.1 Feature Representation and Similarity between Shape Spectra

A group of object views organized on the basis of "similarity" of shape spectral features would contain views that exhibit the characteristics of the same set of visible surfaces of the object [46]. Note that views that can be obtained by rotations about the viewing direction are likely to possess similar shape spectral features and are hence grouped together. No surface segmentation or edge detection is required with this approach.

We have proposed a feature representation that emphasizes the spread characteristics of the spectral distribution. Our feature vector representation $\mathbf{R}$ of a view is based on the first ten moments of the normalized (with respect to the visible object surface area) shape spectral distribution, $\bar{H}(h)$, of an object view. By normalizing the spectrum with respect to the total object area, we remove the scale (size) differences that may exist between different objects. The first moment is the mean which provides the average concentration of the shape index values of the surfaces visible in the view; the second feature is the variance, the third is the skewness, and the fourth is the kurtosis (peakedness around the mean) of the shape index distribution. Features five through ten are the central moments of the shape spectrum of orders five through ten, respectively. These features are best understood if we observe the likeness between the shape spectrum of an object view and a probability density function of a random variable [60].

The first moment is computed as the weighted mean of $\bar{H}(h)$, and is defined as

$$m_1 = \sum_h (h)\, \bar{H}(h). \tag{4.1}$$

The other moments, m_p, $2 \leq p \leq 10$ are computed as follows:

$$m_p = \sum_h (h - m_1)^p\, \bar{H}(h). \tag{4.2}$$

Then the feature vector is denoted as $\mathbf{R} = (m_1, m_2, \cdots, m_{10})$. Note that the range of each of these moments is $[-1, 1]$.

Let $\mathcal{O} = \{O^1, O^2, \cdots, O^n\}$ be a collection of n 3D objects whose views are present in the model database, $\mathcal{M}_D$. The jth view of the ith object, O^i_j in the database is represented by $\langle L^i_j, \mathbf{R}^i_j \rangle$, where L^i_j is the object label and $\mathbf{R}^i_j$ is the shape spectral moment vector. Given a set of object representations $\mathcal{R}^i = \{\langle L^i_1, \mathbf{R}^i_1 \rangle, \cdots, \langle L^i_m, \mathbf{R}^i_m \rangle\}$ that describe m views of the ith object, the goal is to derive a partition of the views, $\mathcal{P}^i = \{C^i_1, C^i_2, \cdots, C^i_{k_i}\}$. Each cluster in $\mathcal{P}^i$ contains views that have been adjudged similar based on the dissimilarity between the corresponding moment features of the shape spectra of the views. The measure of dissimilarity between $\mathbf{R}^i_j$ and $\mathbf{R}^i_k$ is defined as

$$\mathcal{D}(\mathbf{R}^i_j, \mathbf{R}^i_k) = \sum_{l=1}^{10} \left(R^i_{jl} - R^i_{kl}\right)^2. \tag{4.3}$$

4.4.2 Object View Grouping

In order to provide a meaningful categorization of views of the object O^i, views are clustered based on their dissimilarities $\mathcal{D}(\mathbf{R}^i_j, \mathbf{R}^i_k)$ using a hierarchical clustering scheme such as the complete-link algorithm [92]. The partition $\mathcal{P}^i$ is obtained by splitting the hierarchical grouping of O^i at a specific level of dissimilarity in the dendrogram. The split level is chosen at the dissimilarity value of 0.1 or less to result in a set of compact and well-separated clusters. If the number of resultant clusters is pre-specified as a design criterion, then the cut level can be automatically selected.

Once the partition $\mathcal{P}^i$ is determined from the training views of O^i, the database $\mathcal{M}_D$ is organized into a two-level structure, $\mathcal{M}_D = \{\mathcal{P}^1, \cdots, \mathcal{P}^n\}$, where each $\mathcal{P}^i$ is itself a set of view clusters. A summary representation for each view cluster C^i_j is abstracted from the moment vectors of its constituent views such as the centroid of the view cluster. Given an input view, its object label and best matching view are identified quickly and accurately in two stages: (i) the object identity is established by first comparing the moment vector of the input view with the cluster summary representations and selecting the best matched cluster; (ii) comparison of the input view with the moment vectors of the views in the best matched cluster determines the view that matches most closely with the input.

4.5 Experimental Results

We have used a database containing 3,200 range images of 10 different sculpted objects with 320 views per object [47]. The range images from 320 possible viewpoints (determined by the tessellation of the view-sphere using the icosahedron) of the objects were synthesized either from the CAD models when available or from the hand-constructed object triangulations. Figure 4.2 shows a subset of the collection of views of Cobra used in the experiment. We computed the shape spectrum of each view and then determined its feature vector $\mathbf{R}$. We clustered the views of each object based on the dissimilarity measure $\mathcal{D}$ between their moment vectors using the complete-link hierarchical clustering scheme [92]. The hierarchical grouping obtained with 320 views of the Cobra object is shown in Figure 4.3. The view grouping hierarchies of the other objects are similar to the dendrogram in Figure 4.3. These clusterings demonstrate that the views of each object fall into several distinguishable clusters. The hierarchical grouping obtained from each object was then cut at a dissimilarity level of 0.1 or less to result in compact view clusters. The centroid of each of these clusters was determined by computing the mean of the moment vectors of the views falling into the cluster. Figure 4.4 shows the visualization of the centroids of the view clusters of Cobra (obtained by splitting its hierarchy at a dissimilarity level of 0.05) using

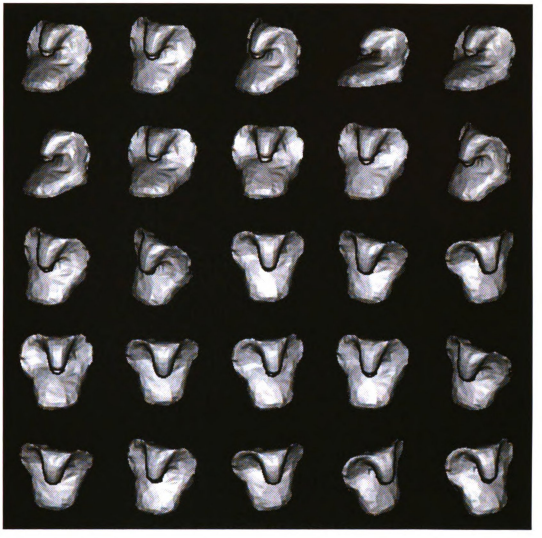


Figure 4.2: A subset of views of Cobra chosen from a set of 320 views.

Chernoff faces [39] where the ten moment features were depicted using the area of the face, shape of the face, length of nose, location of the mouth, curve of the smile, width of the mouth, location, separation, angle and shape of the eyes, respectively. Figure 4.5 shows some of the clusters obtained with 320 views of Cobra and the views contained in them. Observe that the shape spectra of these views do not change with a rotation of the views about a single axis and this leads to a more concise method for grouping multiple views.

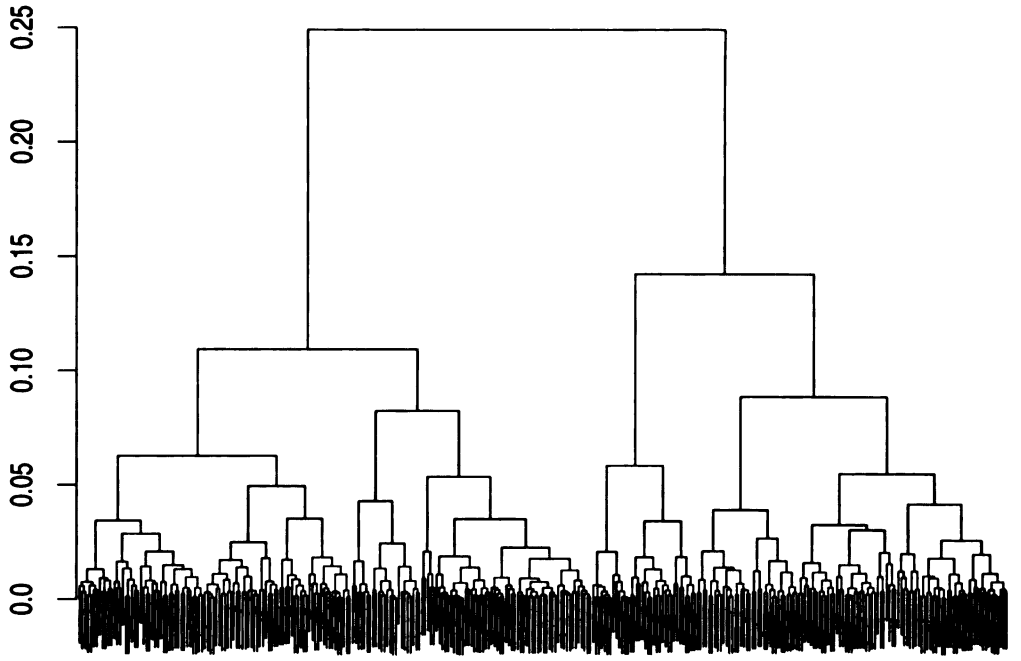


Figure 4.3: Hierarchical grouping of 320 views of Cobra.

4.5.1 Matching Accuracy: Resubstitution

The goal of our experiment is to examine how view grouping facilitates matching in terms of classification accuracy and the number of matches necessary for correct classification of views.

View Groups of a Single Object

For this experiment, the database was assumed to contain view groups of only a single object at a time. Each training pattern (feature vector of a view) from the object class was matched with each of the cluster centroids in the database, and the best matched cluster was identified. Then the input pattern was matched with all the constituent patterns in the cluster and once again, the stored pattern giving rise to the minimum distance with the input pattern was identified. Since the viewpoint of each input view was known, the viewpoint of the best matched pattern was compared with the actual object viewpoint to compute the view classification error. The input

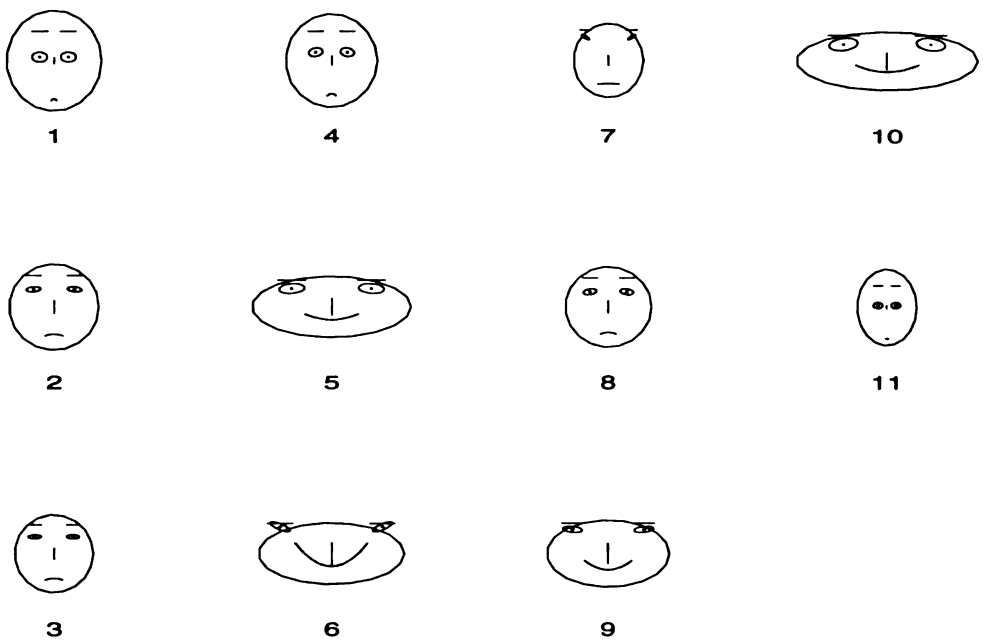


Figure 4.4: Visualization of the centroids of eleven view clusters of 320 views of Cobra using Chernoff faces.

view was matched with the views belonging to only the best matched cluster; this is the "top-one-cluster" strategy. Table 4.1 illustrates the correct classification rate obtained when tested with 320 input views. It can be seen that a view classification accuracy of at least 90% can be obtained even by matching views from the best matched cluster alone.

We now modified our matching strategy to allow the input pattern to match with all the patterns falling into the *two* best clusters that resulted in the smallest and the next smallest distances among all view groups. We refer to this as the "top-two-clusters" strategy. This improved our view classification accuracy to 100% in eight object classes. These results show that object views are grouped into compact and homogeneous view clusters, thus demonstrating the discriminatory power of the shape spectrum-based feature representation.

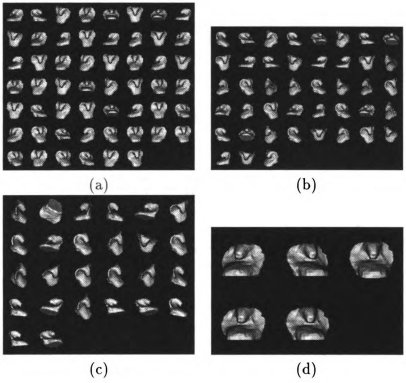


Figure 4.5: View clusters of Cobra and their views: (a) views belonging to Cluster5; (b) Cluster6; (c) Cluster9; (d) Cluster10.

View Groups of Several Objects

We now consider a database containing views of different objects, which has been organized into a two-tiered structure: the first level containing all the view-groups obtained from clustering views of each object individually, and the second level consisting of the views themselves in these clusters.

We verified the accuracy of view matching in the presence of groups of views arising from various objects in the database. We collected together view groups, for example, of four objects obtained with a training set of 320 views per object, resulting in a total of 48 groups. For each view group, the centroid-based representative pattern was determined. Each input pattern was first matched with all the 48 centroid patterns, and the group with the least distance from the input was chosen for further examination to identify the final match. We repeated this experiment, increasing the

Table 4.1: View classification accuracy with view groups of a single object.

Object class	No. of clusters	Correct view classification (%)	
		Top 1 cluster	Top 2 clusters
Vase2	12	96.9	100
Vase1	10	92.8	100
Big-Y	9	93.4	100
Two-mag-cyl	10	91.9	100
Long-mag-cyl	16	91.3	99.7
Cobra	11	94.1	100
Cup	10	96.6	100
Apc	10	91.9	99.4
Jeep	11	90.0	100
Truck	11	91.3	100

number of object classes in the database to five, eight and finally ten. Figure 4.6 presents the view classification accuracy obtained when tested with the training patterns themselves with different numbers of objects in the database. The horizontal axis contains the number of best (in terms of the dissimilarity between the input pattern and the centroid patterns) matched clusters that were examined for the final match, for a given level of accuracy. As shown in Figure 4.6, as the number of objects in the database increased, 14 best-match clusters had to be examined to obtain 100% accuracy with correct view identification. We observe that even when tested with a large number (3,200) of object views, an accuracy of 99% can be achieved by choosing only the top 8 clusters. Only 20% of the views in this large database were examined on the average, even when allowing the top 14 clusters and their constituent patterns to match with the input in order to achieve a 100% correct object identification and view determination accuracy. As the number of object classes increases, the percentage of comparisons performed decreases, thus confirming the efficiency of matching with a structured database of views.

In the resubstitution mode, an error in the classification of an input view arises from only being placed in the wrong object cluster in the first place. This further

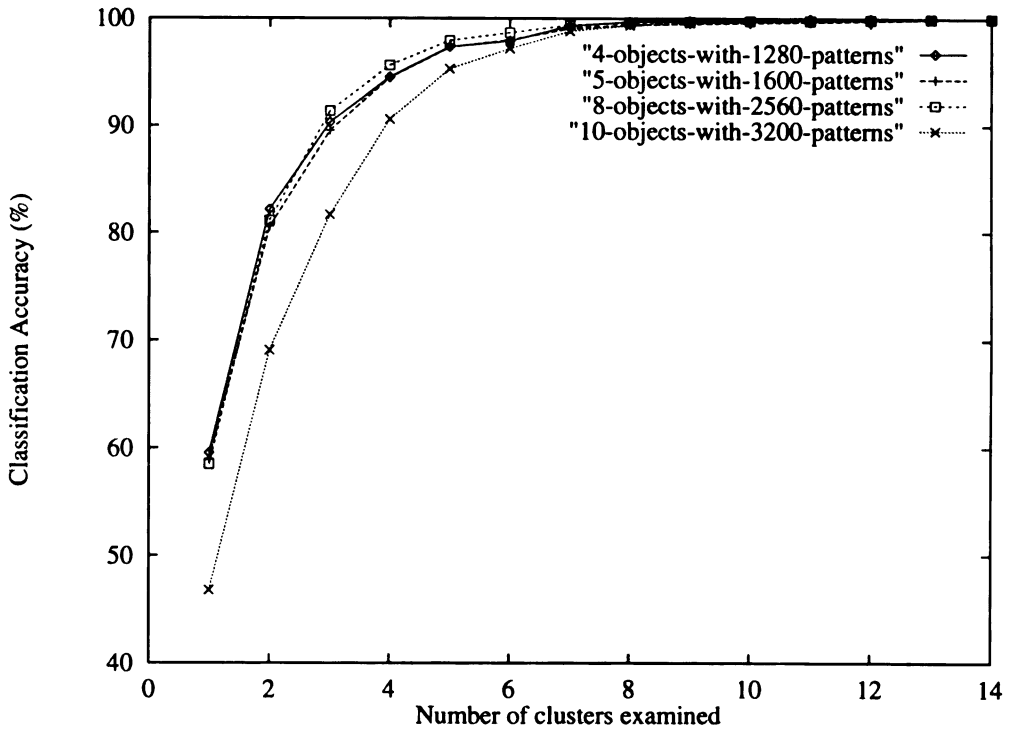


Figure 4.6: View classification accuracy vs. number of clusters examined in the database.

indicates that the simple centroid-based generalization that we adopted is a reasonable scheme and the clusters are tight enough that after a view falls into a cluster, the view very rarely matches with a wrong pattern within the cluster.

4.5.2 Matching Accuracy: Testing Phase

We trained the view grouping system with 3,200 training views (320 views per object) and tested it with 1,000 independent test views (100 per object). At the top level in the database, there were 110 cluster centroids to match with the input test pattern. We studied the number of clusters that had to be examined in order to attain several levels of misclassification rates. The computational steps are summarized in Figure 4.7. Figure 4.8 provides several interesting observations. First, for eight complex objects (e.g., Cobra) the correct object identity of 97% or more of their test views can be determined within the top 10 best cluster matches. With the other two objects, 20 best matched clusters are needed to be examined before the correct

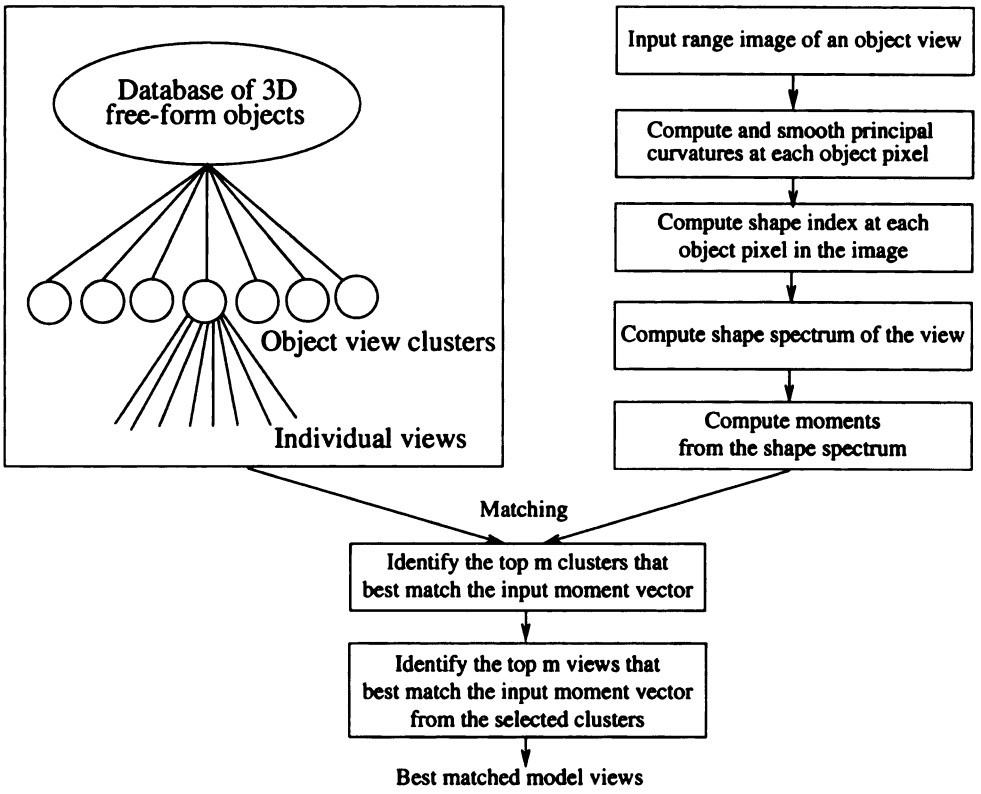


Figure 4.7: Model view selection with the view-grouping and matching system.

object identity of an input can be obtained; this is due to the fact that these objects contain mostly surfaces of predominantly cylindrical ridge shape. Twenty one best matched clusters were needed to be examined for 100% correct classification of the test views when the percentage planar surface area of the object was incorporated as an eleventh feature (see Section 4.5.5). Even in the worst case, allowing the best 20 out of 110 clusters to be examined, only about 23.5% of the 3,200 view comparisons were performed. In addition, on the average, across all the ten object classes, only 15% of the database was examined to identify the correct object identity of an unknown input view. Each test pattern took about 20 ms to be correctly classified on a SPARCstation 20. This demonstrates the efficiency of the view grouping and matching system even with simple centroid-based cluster summaries. With more sophisticated methods of generalizing the cluster patterns, further reduction of these matching costs can be expected.

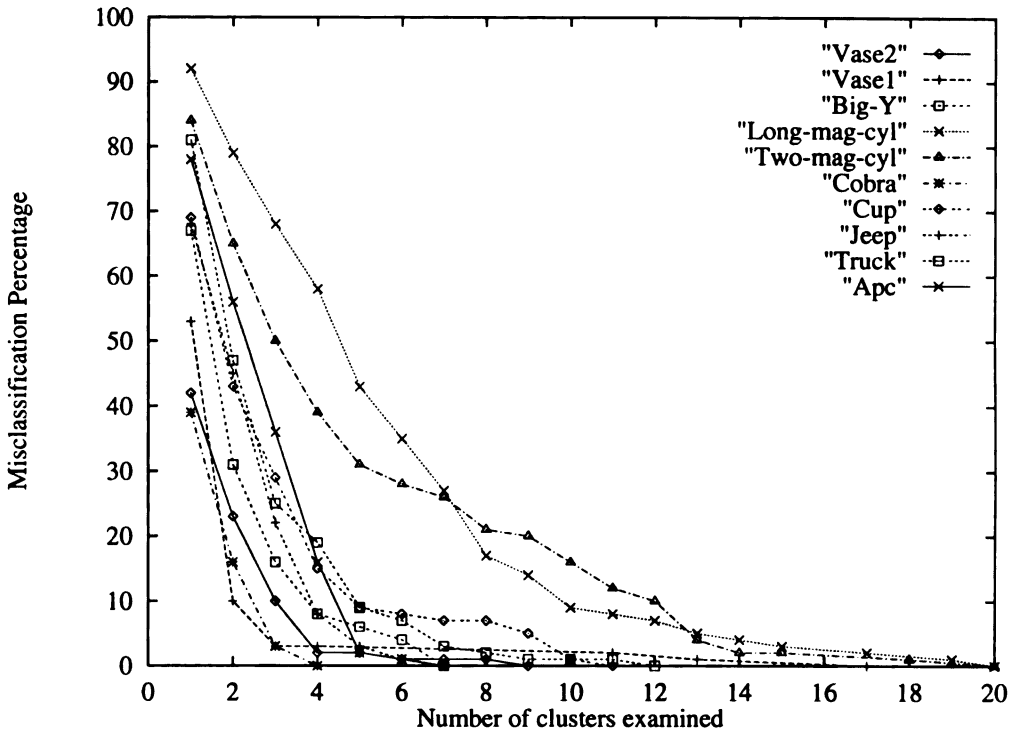


Figure 4.8: Misclassification vs. number of clusters examined.

4.5.3 Testing with 6400 Object Views

In order to study the performance of the shape spectrum-based view grouping and matching system in the presence of a larger model database of views, we added views of ten additional free-form objects, resulting in a total of 6,400 training views in the database. We discuss below the results of some of our experiments conducted with the enlarged database. Figure 4.9 shows the twenty complex objects, each of which was modeled using 320 different views to populate the database. The polyhedral models of the ten objects added were collected from a public domain (http://www.eecs.wsu.edu/~flynn) database on the Internet and were used to generate the views from multiple viewpoints.

As before, we generated 100 random views of the twenty free-form objects in the database for testing. At the top level, the database contained 229 view clusters obtained by grouping the views of all the objects. The second level contained the training views themselves. Each of the 2000 test views was used as a query view

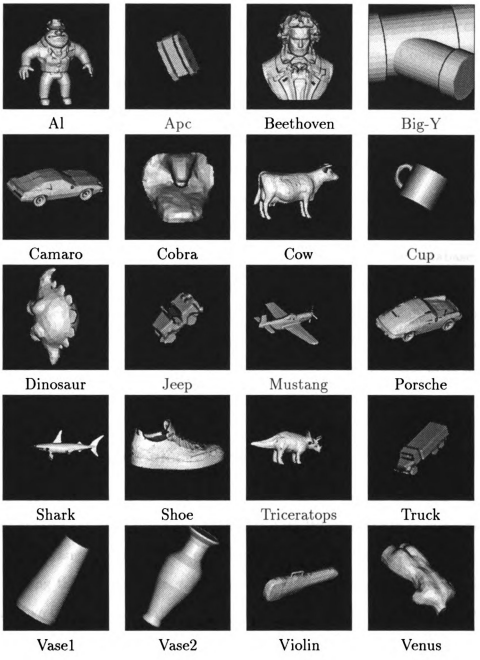


Figure 4.9: Range images of objects generated from arbitrary viewing directions from twenty object models.

and the number of best-matched clusters that needed to be examined in order to correctly identify the object class of the query view was noted. Table 4.2 summarizes the results of this experiment. It can be observed that despite the increase in the size of the database, only ten (about 4.4%) of the view clusters needed to be examined to obtain an accurate classification of 95% of the test views. Complex free-form objects such as the Cobra, Vase2, Beethoven, Cow, Violin, Venus etc. required fewer number of clusters (15 top clusters or less) to be examined to obtain 100% correct classification of the test views drawn from their object categories. Model views from the vehicle category such as Porsche and Camaro were retrieved often as the best matched views for test views from either of these two classes. It can be observed from the table that the number of best matched clusters that need to be examined for 100% object classification accuracy depends on the size of the database. These results further demonstrate that the shape spectrum-based moment vector for view representation can serve as a useful pruning primitive during matching with a model database containing many complex free-form objects. Only 20% of the database was matched for view classification, on the average, over 2000 test views, even when top 30 clusters were examined.

4.5.4 Model View Selection with Real Range Data

The shape spectrum based matching scheme was also tested on real range images of free-form objects obtained using the Technical Arts White scanner in our laboratory. A total of 100 range images of different free-form objects (10 views per object) were collected using the White scanner. The views in the database were randomly separated into two different categories: (i) a model database containing 50 views with five views obtained from each of the ten different objects and (ii) an independent test set containing 50 views of the objects (5 views per object). Figure 4.10 shows the range images of model views from each of these ten object classes and Figure 4.11 shows fifty test views.

Object class	Correct object classification (%) when K best-matched clusters were examined							
	K=1	K=2	K=5	K=10	K=15	K=20	K=25	K=30
Vase2	30.0	43.0	79.0	98.0	100.0			
Vase1	27.0	51.0	84.0	97.0	97.0	97.0	97.0	97.0
Big-Y	3.0	7.0	35.0	68.0	92.0	100.0		
Cobra	61.0	84.0	100.0					
Cup	31.0	56.0	90.0	98.0	100.0			
Apc	21.0	44.0	97.0	100.0				
Jeep	25.0	48.0	94.0	100.0				
Truck	17.0	48.0	88.0	98.0	98.0	99.0	99.0	99.0
Al	25.0	51.0	86.0	99.0	99.0	99.0	99.0	99.0
Beethoven	33.0	63.0	91.0	99.0	100.0			
Cow	29.0	62.0	96.0	100.0				
Dinosaur	23.0	46.0	70.0	92.0	98.0	99.0	99.0	99.0
Porsche	16.0	44.0	75.0	98.0	99.0	100.0		
Shark	36.0	46.0	80.0	93.0	96.0	96.0	98.0	99.0
Shoe	39.0	59.0	78.0	91.0	95.0	100.0		
Triceratops	23.0	41.0	77.0	97.0	99.0	100.0		
Venus	26.0	46.0	81.0	100.0				
Violin	66.0	95.0	99.0	100.0				
Camaro	21.0	26.0	66.0	97.0	99.0	99.0	100.0	
Mustang	19.0	23.0	47.0	77.0	85.0	91.0	96.0	100.0

Table 4.2: Object matching accuracy with an independent test set of 2,000 views.

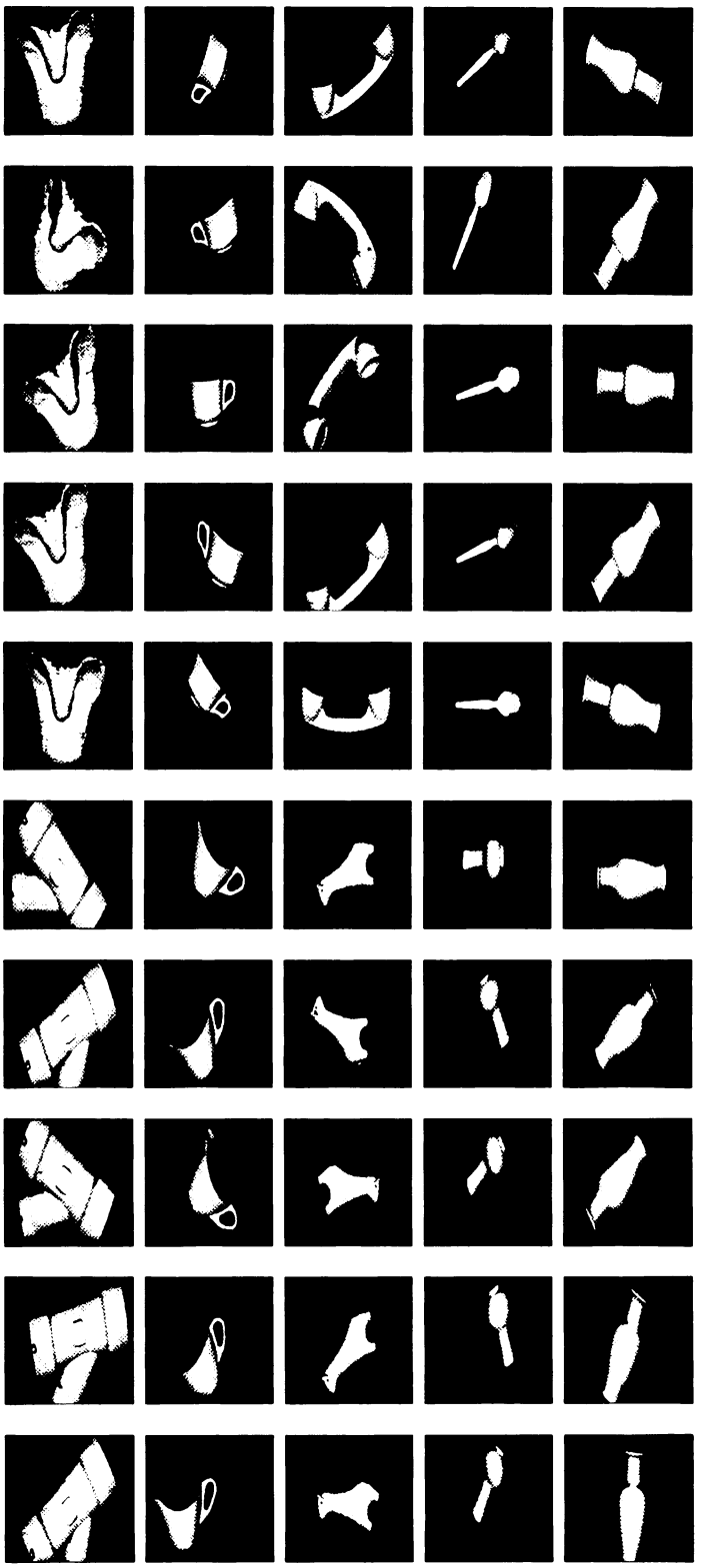


Figure 4.10: Range images of 50 model views.

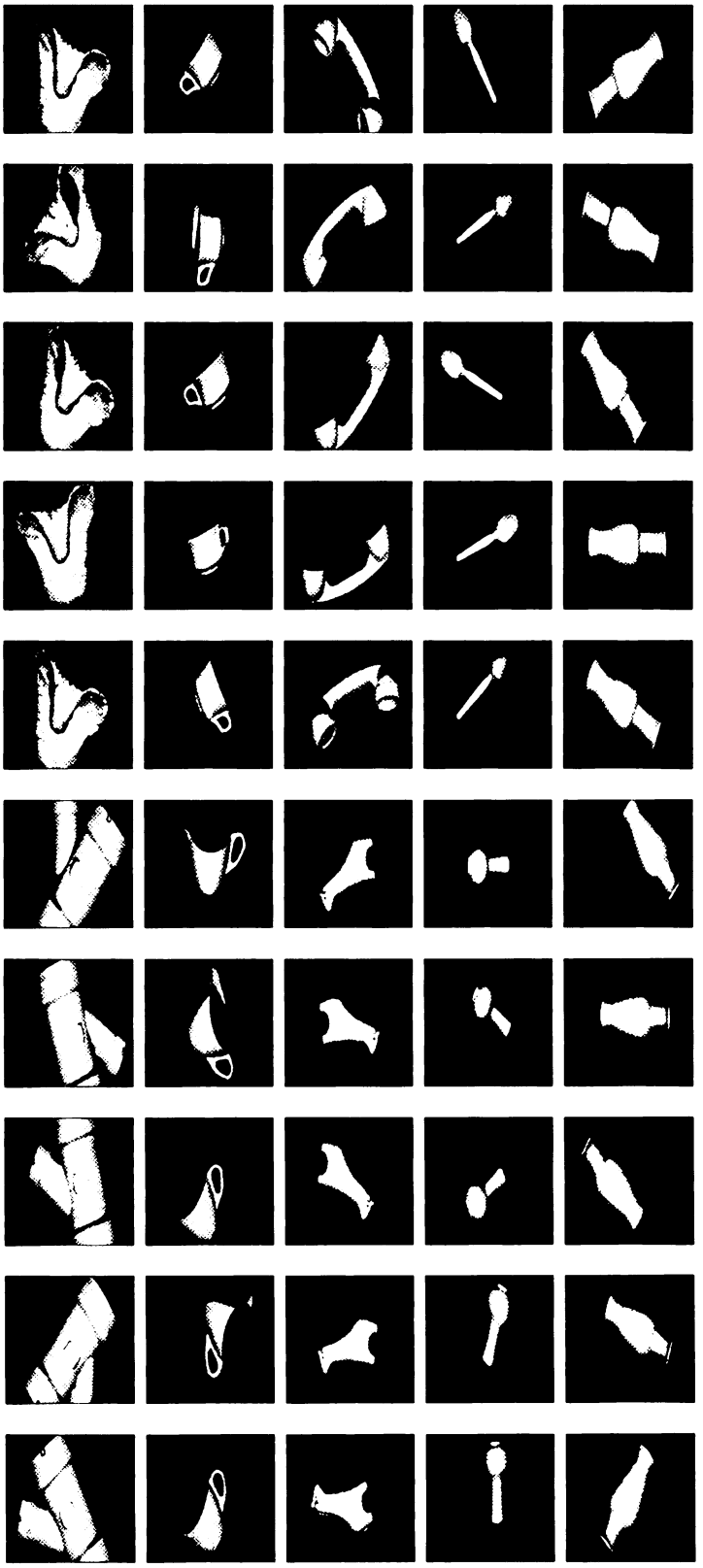


Figure 4.11: Range images of 50 test views.

The shape spectral moment representation was derived for each of the views. The model database was then structured into two levels: at the first level were ten view clusters, with each cluster containing five views from its object class at the second level. Each test view was first matched with the ten view clusters to rank the best matched 3 view clusters and then the views falling into these three view groups were examined clusterwise to select a best matched view from each of the three clusters. This resulted in a view classification accuracy of 92%, with only 4 views out of 50 test views failing to select even a single model view from their correct object classes among their top three matches. When five best matched clusters were examined to select a view from each of them, the accuracy increased to 98% with only one test view incorrectly classified. The wrongly classified test view was a view belonging to the Creamer class. The range image of the incorrectly classified view and the five candidate views that matched the best are shown in Figure 4.12. The shapes of the surfaces visible in this view of Creamer were shared by views from other objects, leading to an incorrect classification. The average number of view comparisons performed in retrieving the top five hypotheses that matched a test view was 35, which is smaller than the number of view comparisons required when linear matching of the test view with all 50 model views is performed. However, observe here that for each test view that was correctly classified, there was only a single model view from its correct object class among its top five view hypotheses. More sample views in each object class in the model base are needed to increase the discrimination between the object shapes visible in the views.

We also repeated the matching experiment with a slight variation in the way the best matched views were chosen from the selected clusters in the first level. Each test view was used as a probe to select the five best matched view clusters as before. However, instead of choosing one best matched view from each of these clusters, five best matched views among all the twenty five views collected from the selected view clusters were determined. Although this method incurs the additional computational cost of sorting the dissimilarity values of twenty five views to select the top five views, it increases the possibility of more than one view from the correct model object class

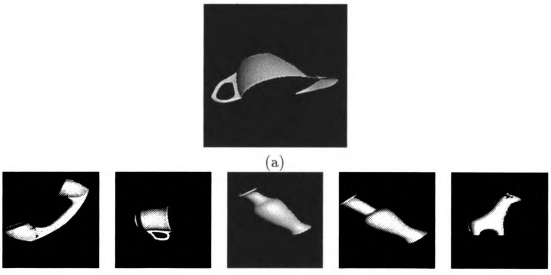


Figure 4.12: Incorrect model view selection: (a) Test view; (b) top 5 view hypotheses generated by the model selection scheme.

featuring among the best matched ones. Table 4.3 shows the number of test views in each object class that elicited at least one model view from their correct object class and also the number of test views that did not find even a single model view from their correct object classes. Three views out of the test set of 50 views did not short-list even a single model view from their correct object classes among the 5 views selected for each of them, thus resulting in a model view selection accuracy of 94%. However, thirty seven views out of the correctly classified 47 test views retrieved two or more model views from their correct object classes.

Comparison of shape spectral moments-based representation of an object view with the approach proposed by Dudani et al. [51] where rotation and scale-invariant moments are derived to identify aircraft types brings forth an important observation: Although both schemes are global representations of object views, the shape-based representation provides a richer discrimination between objects (e.g., disks and spheres) in terms of their surface shapes which would not be possible with the scheme proposed by Dudani et al.

Table 4.3: Shape spectrum based selection of five best matched model views among all the twenty five views at the second level.

Object class	Number of correct and incorrectly classified test views in each object class	
	No. of views correctly classified	No. of views wrongly classified
Big-Y	4	1
Cobra	5	0
Creamer	3	2
Cup	5	0
Giraffe	5	0
Phone	5	0
Small-vase	5	0
Spoon	5	0
Vase2	5	0
Vase3	5	0

4.5.5 Shape Spectrum of Objects with Planar Surfaces

We also studied the performance of the view grouping and matching system using the spectral information of the object views that included the amount (the percentage) of planar surface area found in the object visible in a view. The non-planar shape spectrum of a view was augmented by accumulating the percentage planar points on the surface into a bin with the shape index value 2.0. Using this augmented spectrum, a eleventh feature was added to the 10 original moments derived from the non-planar shape spectrum to form the view feature vector. The additional feature described the percentage of surface area present in the planar shape category. We obtained comparable performance when the experiments described in Section 4.5.1 were repeated. Note that all of the objects in the database used were mainly smooth objects, with planar surfaces present only in a very few views. Hence, our results did not change drastically.

4.6 Summary

We addressed the problem of constructing view clusters of free-form objects. By exploiting view-grouping in model databases, a small number of plausible correct matches can be quickly retrieved for more refined matching. We have proposed a novel shape-spectral feature based scheme for grouping views that obviates object segmentation into parts and edge detection. These features allow object views to be grouped meaningfully in terms of the shape categories of the visible surfaces and their surface areas. The proposed approach is general and relatively easy to use. We demonstrated that in a database containing 110 view clusters of 10 different objects, only the top 20 best-matched clusters need to be examined for 100% recognition accuracy. Only 23.5% of the database was examined in the worst case for a 100% classification accuracy of test views. Even with a larger database containing 6,400 views of 20 different objects, only 20% of the database was examined, on the average, over 2,000 independent test views for correct classification. We also demonstrated the effectiveness of the scheme on real range images of views of free-form objects.

Chapter 5

Multi-level Matching for Free-Form Object Recognition

This chapter focuses on utilizing the COSMOS representation scheme for recognizing a given object view and estimating its position and orientation with respect to the views stored in the model database. Given a database consisting of different objects and their views obtained from many different viewpoints, our goal is to recognize and estimate the pose of an object view using its range image. The difficulties that make this task challenging are as follows: (i) The recognition procedure should efficiently be able to cope with our view-based representation scheme and (ii) object identification and pose estimation must be achieved accurately and quickly. We address the following specific issues in this chapter: (i) What sort of indexing (model view selection) mechanism should be used for identifying a subset of candidate object views that can then be matched in detail with the input view? (ii) What recognition strategy should be used to match the COSMOS representation derived from an input view of an object with the selected model view representations?

There are a number of processing steps involved in a COSMOS-based 3D object recognition system and they can be divided into two stages: model construction and recognition. During the model construction stage, the following steps are carried out: (i) acquisition of dense depth data of an object from multiple viewpoints, (ii) seg-

mentation of the range images of the object into maximal patches of constant shape index, (iii) construction of the COSMOS representation using the CSMPs detected in each range image, and (iv) building a collection of representations of multiple views of the object to be stored as its model.

During the recognition stage, a range image of an unidentified object view is presented to the system. Given the sensed data, the following actions need to be taken: (i) determine the identity of an unknown object view from its COSMOS representation, and (ii) estimate the pose of the recognized object view. Note that the models of objects can be constructed and stored *a priori* and the recognition can then be performed on-line. The performance of the recognition stage, in terms of its accuracy and speed, essentially determines the usefulness of the system in real world situations. Figure 5.1 shows the various modules that comprise our recognition system.

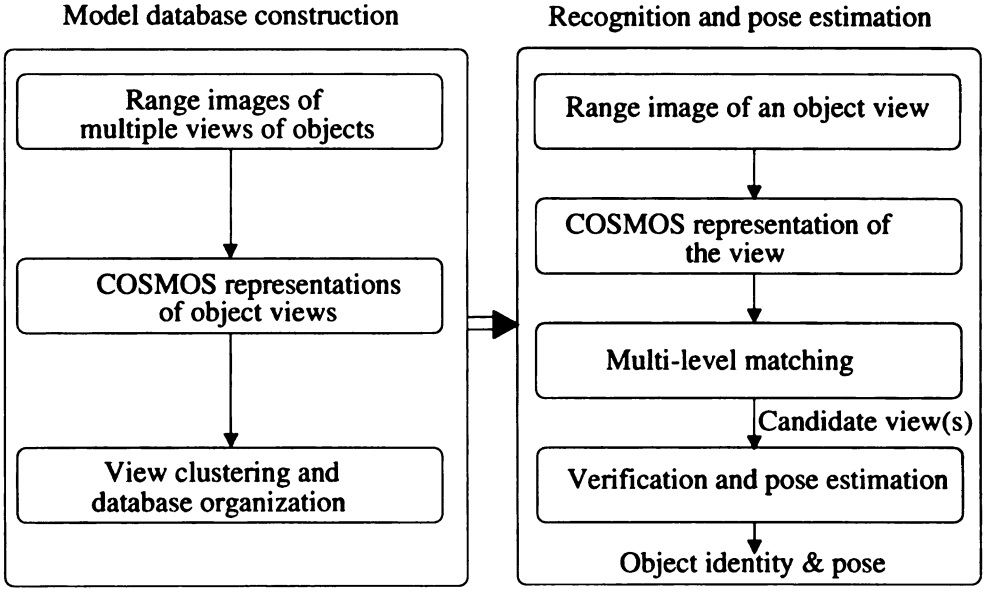


Figure 5.1: Overview of our 3D object recognition system.

During recognition, as the model database is updated with an increasing number of model objects, the computational time required to establish the identity of a given input also grows prohibitively high. With free-form objects especially, this

computational cost can be crucial, as the sculpted objects themselves may use complex representations and hence require large computational resources even to match a single pair of object representations. A preferred solution to building a robust and fast 3D object recognition system, therefore, is to to derive strategies to efficiently prune the model database (i.e., indexing or model selection) and to design methods that are general and reduce the search while matching the candidate object representations with the input in a detailed manner. In this spirit, we propose a multi-level matching strategy that employs shape spectral analysis and features derived from the COSMOS representations of objects (including patches of constant shape index, mean curvedness, orientation, etc.) for fast and accurate recognition of arbitrarily curved 3D objects using range data.

5.1 COSMOS-Based Free-Form Object Recognition

The proposed multi-level matching scheme [48] makes use of the components of the COSMOS representation to prune the set of possible matches to the input view and also to establish the input-model view feature correspondences and to estimate the pose of the object in the input view. The terms "input view" and "scene view" will be used interchangeably in this chapter as also the terms "model views" and "stored object views."

The input scene is assumed to contain an uncluttered view of an object (allowing self-occlusion). In the first level, objects are matched efficiently on the basis of shape spectral information (see section 5.2). The shape spectrum is an easily computable feature of an object view. The comparison of feature vectors containing the moments derived from the view spectra is also based on a simple measure, thus allowing it to be used for rapid pruning of a model database of object views to obtain a small set of candidate views. As demonstrated in Chapter 4, shape spectra of object views can also be used to sort a large model base into structurally homogeneous subsets, leading to a meaningful organization of the database.

In the second level of matching, we may encounter two kinds of scenarios as

depicted in Figure 5.2: (i) an input scene contains multiple overlapping objects and (ii) the input scene may contain unoccluded objects. Our current study is restricted

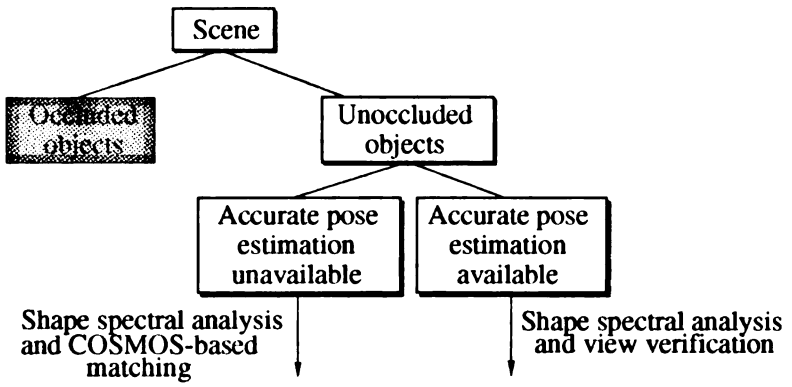


Figure 5.2: 3D object recognition and pose estimation.

to unoccluded object views only. When multiple nonoverlapping objects are present in the scene, our recognition system can encounter two kinds of scenarios. In the first situation, it is possible that the database consists of object views generated either from CAD models of objects or from surface triangulations of objects, as typically encountered in industrial applications. In that event, the potential matches selected from the database are tagged with their identity as well as their 3D pose. The recognition and the pose estimation problem then becomes one of verifying which of the views in the selected subset is most similar to the sensed object. Flynn and Jain [68] propose a simple verification scheme that generates synthetic range images of the hypothesized objects at the hypothesized pose and performs a pixel-by-pixel comparison of this data with the input view to establish the correct identity and the pose of the sensed object.

With the second situation, it is possible that the database consists of object views that are acquired using a laser range scanner without a control device having six degrees of freedom of motion. The 3D positions and orientations of the stored views are, therefore, unknown and need to be estimated from the data during matching. In our system, we exploit the COSMOS representations of object views to determine the image-model feature correspondences using a combination of search methods, and

thus identify and localize the object in the input view.

5.2 Shape Spectrum-Based Model View Selection

A shape spectrum based hierarchical organization of the model database is exploited in our recognition system to efficiently match a given input view with the stored representations in the database. Let $\mathcal{O} = \{O^1, O^2, \cdots, O^n\}$ be a collection of n 3D objects whose views are present in the model database, $\mathcal{M}_D$. The jth view of the ith object, O^i_j is stored in the database with its tag $\langle L^i_j, \mathbf{PO}^i_j \rangle$, where L^i_j is the object label and $\mathbf{PO}^i_j$ is its pose (position and orientation) vector, if already known. The feature vector representation $\mathbf{R}^i_j$ of a view is based on the first ten moments of its shape spectral distribution, $\bar{H}^i_j(h)$, that has been normalized with respect to the visible object surface area [47].

The database $\mathcal{M}_D$ is organized into the two-tiered structure shown in Figure 5.3, $\mathcal{M}_D = \{\mathcal{P}^1, \cdots, \mathcal{P}^n\}$, where each $\mathcal{P}^i$ is itself a set of view clusters, $\{C^i_1, C^i_2, \cdots, C^i_{k_i}\}$. Each cluster in $\mathcal{P}^i$ contains views that have been adjudged similar based on the dissimilarity between the corresponding moment features of the shape spectra of the views belonging to the ith object. A summary representation for each view cluster C^i_l is abstracted from the moment vectors of its constituent views by computing the centroid of the view cluster. Given an input view, a small set of object views that are most similar to the input is determined quickly and accurately in two stages: first compare the moment vector of the input view with all the cluster summary representations and select K best matched clusters; then match it with the moment vectors of the views in these best matched clusters and select the top m closest views. This spectrum based first level matching step results in a set of probable model object views that are similar to the input in terms of visible surface shapes.

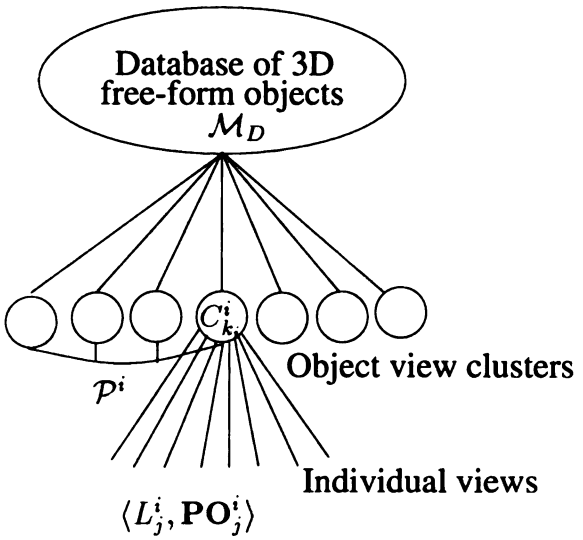


Figure 5.3: Shape spectrum based two-tiered organization of a model database.

5.3 COSMOS-Based Refined View Matching

Having tackled the task of short-listing the model views to a few promising candidate views, we now address the problem of exploiting our COSMOS representation to compare views. The function of this view verification stage is to determine a *correct* object match among the selected model view hypotheses. Specifically, the objective of this section is to elucidate how, given COSMOS representations of two object views, we may establish *correspondences* between the CSMPs in the views. The model view features and the scene features need to be related in a detailed manner to establish the identity and the pose of the sensed object accurately. The matching algorithm however must deal with noise, missing patches, and spurious information due to imperfect segmentation.

Since COSMOS provides a structural description of surface patches in an object view (CSMPs arranged in a definite pattern of organization), scene-model feature matching is formulated as a search and optimization problem exploiting the *region adjacency graph* data structure (Section 3.2.2) that can be abstracted from the COSMOS representation of the view. A *good* and *consistent* correspondence between two

graphs has to be found, where goodness implies similarity of matched components, and consistency means that violation of connectivity relationships between matched components is not allowed, or allowed only to a small degree.

In essence, our solution is to merge some patches into "patch-groups" (which along with their connectivity information define a "patch-group graph"), construct a consistent correspondence between the two patch-group graphs, and compare the resulting graphs, iteratively until they become isomorphic. The goal is to construct a solution that maximizes a given measure of goodness of matched patch-groups in the graphs, given the topological constraints imposed by the surface connectivity information, and to obtain the finest grain mapping possible between the patch-groups in the views.

5.3.1 Patch Grouping, Correspondences and Graph Isomorphism

We now introduce some concepts and terminology used in the matching algorithm. Most of these concepts translate directly into data structures in the implementation of the algorithm.

In a region adjacency graph each vertex of the graph represents a patch—in COSMOS, a CSMP—and each edge represents the fact that the patches represented by the adjoining vertices are directly connected to each other. (We defined direct connectedness as 8-connectedness in the range images obtained using the Technical Arts White scanner in Section 3.2.2; this information is available in COSMOS through the surface connectivity list V.) In Figure 5.4(c), a CSMP is denoted by a circle and a bidirectional arrow between the circles indicates the adjacency of the CSMPs. When a set of *connected* patches is grouped together, we call such a set a *patch-group*. We extend the notion of direct connectedness to patch-groups and say that two patch-groups $\mathcal{P}_1$ and $\mathcal{P}_2$ are directly connected if and only if there exists some patch $P_1 \in \mathcal{P}_1$ and another $P_2 \in \mathcal{P}_2$ such that patches P_1 and P_2 are directly connected. We then define a *patch-group graph* as a graph in which each vertex represents a patch-group,

and each edge represents direct connectedness between the two patch-groups denoted by the adjoining vertices. For example, in Figure 5.4(c) where the correspondence shown has been obtained from matching two views of Vase2, the ellipses denote patch-groups. We are interested only in patch-group graphs in which the patch-groups are disjoint, i.e., a given CSMP appears inside one and only one vertex in the graph. We make use of groups rather than individual patches because it is possible that there are excess patches in the input (scene) view or in the model view, so CSMPs need to be combined in both the scene and the model views to identify a good equivalent on the other side. Thus, the algorithm has the robustness to noise, missing data, and spurious information resulting from imperfect segmentation of object views, built into its design.

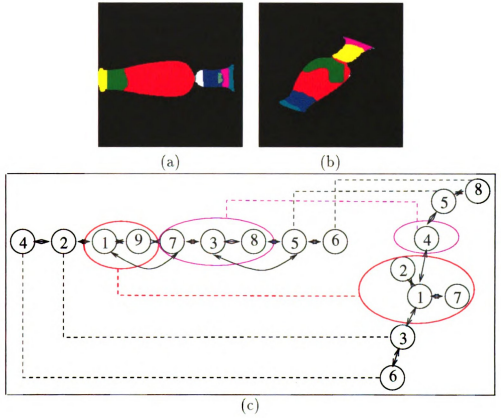


Figure 5.4: Correspondence between patch-group graphs: (a) View 1 of Vase2; (b) view 2 of Vase2; (c) correspondence established between the CSMPs in the views.

Given two object views, each of which has been partitioned into a patch-group graph, we need to uniquely associate a patch-group in one view with a patch-group in the other view. Such an ordered pair of CSMP groups is called a *MappedPair* structure in the COSMOS implementation; the two sides of a MappedPair may of course contain different numbers of patches. In Figure 5.4(c), a dashed line associates each patch-group established in one view to its matching patch-group in the other view.

Given two objects decomposed into patch-group graphs with equal numbers of vertices, a bijective mapping between the vertex sets of the two graphs is called a *correspondence.* Thus a correspondence is a set of MappedPairs that fully covers both object views. In Figure 5.4(c), the correspondence shown between the views of Vase2 is given by

$$\{\langle\{P_{14}\},\{P_{26}\}\rangle,\langle\{P_{12}\},\{P_{23}\}\rangle,\langle\{P_{11},P_{19}\},\{P_{21},P_{22},P_{27}\}\rangle,$$
$$\langle\{P_{17},P_{13},P_{18}\},\{P_{24}\}\rangle,\langle\{P_{15}\},\{P_{25}\}\rangle,\langle\{P_{16}\},\{P_{28}\}\rangle\}$$

where P_{1i} denote CSMPs from the scene (View 1 of Vase2) and the patches P_{2j} from the model view (View 2 of Vase2).

There are many ways of constructing correspondences between two images and we need to identify one that best detects any similarity between the two objects in the presence of noise, displacement, etc. Therefore, our search for a good match is conducted conceptually over the entire space of possible correspondences, and each correspondence is a *candidate solution* to the problem of image comparison. In practice, we will use heuristics and examine only a small part of this space.

In general, a correspondence arbitrarily associates each patch-group in one image with a unique patch-group in the other image. However, we are primarily interested only in *feasible correspondences* which are defined as those that satisfy *patch-group graph isomorphism.* To recapitulate the concept of isomorphism, we say that two patch-group graphs are isomorphic if we can establish a one-to-one mapping (i.e., a correspondence) from the patch-groups in one image to the patch-group in the other

image in such a way that adjacency is maintained (i.e., for every edge in one image, there is a unique corresponding edge in the other image, whose adjoining pair of vertices have been mapped from the adjoining pair of vertices of the first edge). Thus if the correspondence preserves local connectivity information between the two graphs it is called feasible.

Since the grouping of patches can usually be done in many ways, different correspondences may contain different sizes of patch-groups given the same two images. Some correspondences may be *fine grained*, i.e., each patch-group on each side of the correspondence may contain just one or a few patches. Other correspondences may be relatively *coarse grained*, comprised of a few large MappedPairs. The extreme case of a coarse correspondence is the *trivial correspondence* in which all patches in an image have been collected into a single patch-group, and mapped to a similar single all-inclusive patch-group in the other image.

Clearly the trivial correspondence always exists, and is always a feasible correspondence. It should be intuitively evident that it is more desirable for us to produce a fine grain feasible correspondence than a coarse one, since the former indicates a higher degree of similarity between the parts of the two images. But since there is no guarantee in general that the region adjacency graphs of the two images even resemble each other, we may have to terminate our matching algorithm with a fairly coarse correspondence (possibly even with the trivial correspondence). Therefore, a measure of "fineness of grain" of a correspondence is a necessary component in any definition of the overall goodness measure of the correspondence, and coarse correspondences must be penalized implying that the images are probably of different objects.

We have thus shown how connectivity information is used as a hard constraint in the sense that any correspondences that do not preserve patch-group graph isomorphism are promptly eliminated as being infeasible. The fineness of a correspondence, and other measures based on surface attributes S_I, R and surface area act as soft constraints, i.e., we attempt to maximize those measures. However, the fineness of a correspondence is also treated in a way that is central to our matching algorithm, whose details are presented in the next section. The algorithm begins with the trivial

correspondence and then iteratively attempts to refine the correspondence to a finer grain by splitting patch-groups on each side and reassigning them creating new mappings. Thus, the main loop of the algorithm tries to progressively increase the fineness of the correspondence *while maintaining feasibility at all times* and terminates when it is not possible to further refine the correspondence without losing isomorphism. Thus, only those model view hypotheses that are geometrically consistent with the input are retained and their goodness measures ranked to determine the object view that results in the best scene-model correspondence.

5.3.2 The Matching Algorithm

We will present the matching algorithm that we employed using pseudo code. "//" denotes the beginning of a comment, which continues to the end of the line. The top level function, **match**, takes two image views represented using the COSMOS scheme and returns a complete feasible correspondence that is in some practical sense the "best" correspondence that could be established between the views:

```
match (view1, view2)
// Start with the trivial correspondence
current-correspondence = {⟨{merge all CSMPs in view1},
                           {merge all CSMPs in view2}⟩};
loop:              // Refine current-correspondence
{
   // Identify where to split current-correspondence:
   mp = select-MappedPair-from (current-correspondence);
   // Replace the selected portion with its refinement
   new-correspondence = current-correspondence − mp + refine (mp);
   if (quality(new-correspondence) > quality(current-correspondence))
     current-correspondence = new-correspondence;
} until all pairs in current-correspondence are fully refined;
return current-correspondence.
```

Match simply keeps trying to refine each of its constituent MappedPairs until none of them is further refinable. A MappedPair is unrefinable when either side contains a single patch, or when connectivity constraints between neighboring patch-groups would be violated for every possible way of breaking up the MappedPair into smaller MappedPairs; this is more fully explained in the definition of the **refine** function below. **Refine** takes a MappedPair as its argument, examines each side of the MappedPair (i.e., the patch-group from view 1 that has been assigned to the patch-group from view 2 by the MappedPair), and tries to split both patch-groups in such a way that the resulting patch-group graphs are isomorphic.

refine (mp)

```
left-patch-group = left (mp);
right-patch-group = right (mp);
loop:
{
    lp = select-patch-from (left-patch-group)    // E.g., largest patch
    left-components = split (left-patch-group, {lp})
    for all rp in right-patch-group
        right-components = split (right-patch-group, {rp})
        if (# left-components = # right-components)
          sub-corresp = correspond each left component
                to some right component (feasibly!)
          record best sub-corresp
        for all rp2 = neighbor of rp
            right-components = split (right-patch-group, {rp, rp2})
            repeat as above
            for all rp3 ...
                ... till some groupsize limit
    for all lp2 = neighbor of lp
        left-components = split (left-patch-group, {lp, lp2})
```

repeat as above ...

... till some groupsize limit

} until some maximum number of correspondences checked

return best sub-corresp

To describe the effect of **refine** more precisely, we need to understand how a patch-group is split. A patch-group $\mathcal{P}$ is by definition a set of connected patches. Given a strict subset $\mathcal{P}_1 \subset \mathcal{P}$ of these patches, two or more graph-components get defined on the underlying patch adjacency graph, namely the subgraph identified by the patches in $\mathcal{P}_1$ itself, and the components[1] in the rest of the graph (i.e., the subgraph obtained by deleting all vertices in $\mathcal{P}_1$ from $\mathcal{P}$). Thus selecting a subset $\mathcal{P}_1$ is equivalent to splitting patch-group $\mathcal{P}$ into some n smaller patch-groups including $\mathcal{P}_1$, based on connectivity information between the patches. Further, examining the group-connectivity (described in section 5.3.1) between these new patch-groups results in the establishment of a unique patch-group graph for a given $\mathcal{P}_1$.

Having split left-patch-group into n_l components, and right-patch-group into n_r components, we wish to establish a correspondence (sub-corresp) between the components, i.e., to create MappedPairs $\langle \mathcal{P}_{l1}, \mathcal{P}_{r1} \rangle$ etc., which will be finally returned as the replacement for old MappedPair mp. Clearly both left-patch-group and right-patch-group must be split in such a way that they have an equal number of component patch-groups; otherwise there would be no way to establish an isomorphism between the equivalent patch-group graphs. Further, having $n_l = n_r$ is not enough; connectivity relationships between patch-groups must also be identical on both sides to satisfy isomorphism. Therefore the **refine** function attempts to pair every component on the left with every component on the right and thus tries out all possible sub-correspondences. If a patch-group graph edge on one side cannot be associated with a corresponding edge on the other side, the sub-correspondence is rejected. If more than one isomorphic sub-correspondence can be created, the "best" one is

[1] A *component* C of a graph G is a subgraph such that all vertices of C are connected to each other, and no vertex in G that is not in C is connected to any vertex in C.

recorded; the measure of goodness is very similar to the **quality** function used in the **match** function (the difference arising from the fact that the match function works with complete correspondences, while the refine function works with partial (sub) correspondences; more criteria can be used on the former).

Since there is combinatorially huge number of ways to split each patch-group, all possible ways are not explored. Heuristics are used to decide how to split; these heuristics are embodied by the "select-patch-from()" function and the "for" loops in the **refine** function. As a rule, large patches will correspond to each other because mismatches between small patches have lower penalties. Therefore we typically pick the largest patch *lp* in left-patch-group and try to find a patch *rp* in right-patch-group that matches it most closely in size, shape index, and all the other quality-of-match criteria. Since there may not be a single patch *rp* that closely resembles *lp*, we need to try combining pairs of adjacent patches in right-patch-group and comparing with *lp*. Then we need to try all possible triplets of adjacent patches in right-patch-group as a possible match for *lp*, and so on. Since this quickly gets combinatorial, we impose a "groupsize limit." Since it is possible for a group of patches from left-patch-group, rather than a single patch *lp* to form a better match with *rp*, we need to try out combinations on the left side too, once again up to some practical computational limit. While we have used simple thresholds, smarter heuristics are obviously possible.

5.3.3 Goodness Measure of a Correspondence

The goodness measure (the **quality** function) evaluates the similarities in the shape index, mean curvedness, area and the neighbours' attributes of matched components in a correspondence. The goodness measure of an entire correspondence is computed as the weighted average of the goodness values of the MappedPairs contained within the correspondence. The weight that multiplies a MappedPair is proportional to the combined area of the patches within the MappedPair; thus larger patch-groups have a greater say in establishing the goodness of the correspondence.

quality (correspondence)

```
total-goodness = 0;
for mp in correspondence
  weight = sum of percentage-area of each patch in left(mp) +
           sum of percentage-area of each patch in right(mp);
  total-goodness = total-goodness + (weight × goodness(mp));
return total-goodness
```

Note that total-goodness is normalized (by using percentage areas) to lie in the $[0, 1]$ interval. A smarter version of this quality function has been employed that adds a factor to reflect the overall coarseness of the correspondence.

The **goodness** function which measures the quality of an individual MappedPair is a product of the goodness measures of various features derived from the MappedPair. We have used

- the area-goodness: how similar is the total percentage-area of the left patch-group to the total percentage-area of the right patch-group?
- the S_I-goodness: how close is the mean shape index of the left patch-group to that of the right patch-group?
- the mean curvedness-goodness: how close are the mean curvedness values?
- neighbor-goodness: how close is the mean shape index of the neighbours adjacent to the left patch-group to that of the neighbours of the right patch-group? This incorporates a measure of similarity between the neighbours adjacent to the left and right patch-groups.

The quantitative definitions of the above criteria are both simple and intuitive. For example, area-goodness is computed as

$$1 - \frac{|\text{area}_1 - \text{area}_2|}{\text{area}_1 + \text{area}_2}$$

where area_1 is the total percentage-area of the left patch-group of mp, and area_2 that of the right patch-group. Thus the area-goodness has the value 1 when both groups are of identical size, and goes to 0 as the areas diverge in size. Thus, an

overall matching score characterizing the similarity of the features of the matched patch-groups established in the correspondence between the two views is returned by the COSMOSbased matching scheme, along with the correspondence.

5.3.4 Highlights of the Matching Algorithm

Our COSMOS-based matching algorithm determines the overall goodness measure of the similarity between the given views both locally (i.e., by examining the constituent patches in the views and their attributes) as well as globally (i.e., by using the neighbor-goodness). Local patch comparisons are difficult when the two views are of very different objects, since it is hard to establish a correspondence between the different graph components of the two images. Therefore a very important requirement of a good matching algorithm is that it should be robust, in the sense that it should display graceful degradation with a decrease in the amount of relevant information in the scene data that is useful for matching.

An appealing feature of our COSMOS-based matching algorithm is that it is robust in the above sense, since it always returns *some* feasible correspondence between the given views along with its goodness measure. The greater the dissimilarity between the two objects, the coarser the correspondence returned by COSMOS. In the worst case, the trivial correspondence is returned. In other words, an intuitively meaningful comparison is performed in all cases, and the goodness measure reflects this. Since verification of consistency with respect to surface connectivity is embedded within the algorithm at all points of the computation, the structures manipulated (candidate solutions) are always feasible. An extension of this approach would be to temporarily permit inconsistencies that would be cleaned out before the algorithm terminates.

Our scheme exploits a combination of bottom-up (merging patches) and top-down (splitting patch-group graph) approaches. It thus combines their advantages namely utilizing both local and global information simultaneously such that a *good* correspondence can determined.

Recognition schemes using relational graphs have been explored extensively in the

literature as described in 2.1.3. Although our matching scheme bears resemblance to the hypergraph approach [174] where a grouping was imposed on the vertices of a graph extracted from an object view just as in our scheme, the motivating reasons behind the grouping are very different. In [174], each group of vertices (hypernode) corresponds to a face of a geometric primitive, and a collection of face graphs correspond to a a primitive block of an object, and thus forms a hyperedge in the AHRs. Recognition proceeds by merging multiple AHRs obtained from different views of the object to obtain a complete AHR and it is compared with the stored model AHRs by matching the subgraphs depicted by hyperedges. However, in our scheme, patch merging is carried out to offset segmentation errors and to obtain the best possible correspondence between the views.

The problem of finding the isomorphism between an arbitrary graph and a subgraph of another graph falls into the class of NP-complete problems. Since patch-group graph isomorphism is required by our matching algorithm, without heuristics it would incur this theoretical complexity. However, our matching algorithm using heuristics at several places (as described in the context of the **refine** function) to limit the exploration of isomorphic correspondences (ultimately trading off perfection in establishing the finest grain correspondence possible in favor of efficiency). Because of these heuristics and groupsize limits, all steps of the algorithm are of polynomial complexity, with the sole exception of the step during refinement (immediately after the split step) where two sets of components are mapped to each other in establishing the graph isomorphism. The complexity of this step is exponential because all permutations of the components on one side are tried against the other side. Nevertheless, this step's complexity is not significant in practice because the number of graph components generated by splitting out a patch or group of adjacent patches is likely to be of the order of 2–5. Thus the effective performance of the algorithm is a (large) polynomial in the size (number of patches and edges) of the graphs.

5.3.5 Estimation of Object Pose

Once we ascertain the stored model view in the database that best matches with the input view, we can estimate the rotational component of the pose of the object in the view by aligning the surface normals of *corresponding* CSMPs. We adopt the technique proposed by Flynn and Jain [66] for the estimation of the rotation of the model. Given a single pair of MappedPairs, $(\mathcal{P}^s_{i1}, \mathcal{P}^m_{j1})$ and $(\mathcal{P}^s_{i2}, \mathcal{P}^m_{j2})$, let their mean orientation vectors be $\hat{n}_{s1}$, $\hat{n}_{m1}$, $\hat{n}_{s2}$, and $\hat{n}_{m2}$, respectively. Our goal is to find a 3×3 rotation matrix which, when applied to $\hat{n}_{m1}$, aligns it with $\hat{n}_{s1}$, and also aligns $\hat{n}_{m2}$ with $\hat{n}_{s2}$ due to the rigid nature of the object. An alternate representation of this rotation matrix consists of an *axis of rotation*, $\mathbf{r}$ and an *angle* θ which can be determined in the following manner [80]. Given two non parallel model normal vectors and their corresponding scene vectors, the rotation axis is given by

$$\mathbf{r} = (\hat{n}_{m1} - \hat{n}_{s1}) \times (\hat{n}_{m2} - \hat{n}_{s2}).$$

The angle of rotation, θ is given by

$$\theta = \tan^{-1}\left[\frac{(\mathbf{r} \times \hat{n}_{s1}) \cdot \hat{n}_{m1}}{1 - (\hat{n}_{s1} \cdot \hat{n}_{m1})}\right].$$

The range of values for the components of $\mathbf{r}$ and θ are examined for each pair of scene-model MappedPairs to ensure that the rotation estimated is valid. If the rotation axis and the angle vary only by a small amount (less than 19 degrees in our experiments), we average these estimates to get an average axis $\bar{\mathbf{r}}$ and $\bar{\theta}$. Once a coarse estimate of the rotation is obtained, it can then be refined using an optimal range image registration algorithm [50]. The translational component of the object pose can directly be estimated using the view registration algorithm. Note that the correctness of the object identity of the input view as determined by COSMOS can be further confirmed by registering the input range image with the range image of the best-matched model view using our registration algorithm.

5.3.6 Experimental Results

We demonstrate the performance of our COSMOS-based matching algorithm with several pairs of views obtained from different objects. Figure 5.4 shows the CSMPs obtained from two different views of Vase2 and the correct correspondence determined by the COSMOS-based matching algorithm between the patch-group graphs. Figure 5.5 shows the correspondences between the CSMPs detected in the two views of Phone. Observe that since our matching algorithm does not model symmetry explicitly, the correspondence shown in Figure 5.5(c) inversely matches the symmetrical structures in the two views.

In our second experiment, we tested the performance of our complete recognition strategy using a model database containing 10 different object views obtained using a laser range scanner. Figure 5.6 shows these model views. A view of Cobra (Figure 5.7(a)) was independently obtained and used as a test view. The moments computed from its shape spectrum were compared with those in the model database to select the top five best matched model views for further verification. Figures 5.7(b)-(f) show the five model views selected on the basis of their low dissimilarity values which were computed using the shape spectrum based matching technique.

The COSMOS representation of the input test view of Cobra was then matched with the five model view hypotheses using our view verification algorithm. Figures 5.8(a) and (b) show the segmentation of the test view and the stored model view with the highest matching score of 0.447. Observe that despite the differences in the segmentation results of these two views, our matching algorithm was able to successfully merge patches overcoming the imperfections arising from segmentation, and provide a structurally correct correspondence between the scene and model patches as shown in Figure 5.8(c). Table 5.1 lists the matching scores obtained when the COSMOS representation of the test view was compared in detail with those of the five model view hypotheses. Notice that the matching score for the correct hypothesis is significantly higher than the matching scores for the incorrect hypotheses.

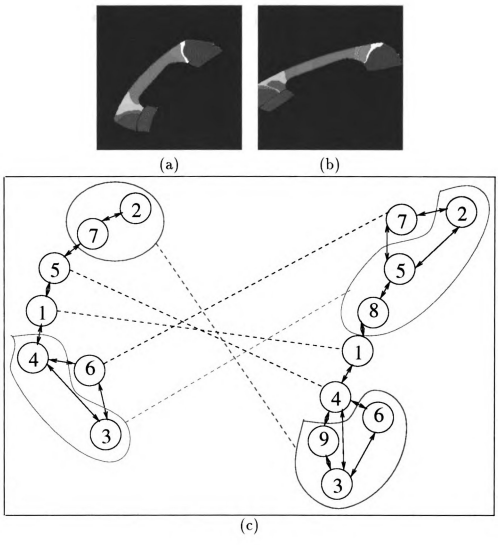


Figure 5.5: Correspondence between the views of Phone: (a) View 1; (b) view 2; (c) correspondence established between the CSMPs visible in the views.

Table 5.1: Matching scores of the five model hypotheses determined by the view verification stage.

View Hypothesis shown in Figure 5.7	COSMOS based matching score
1	0.447
2	0.166
3	0.192
4	0.071
5	0.273

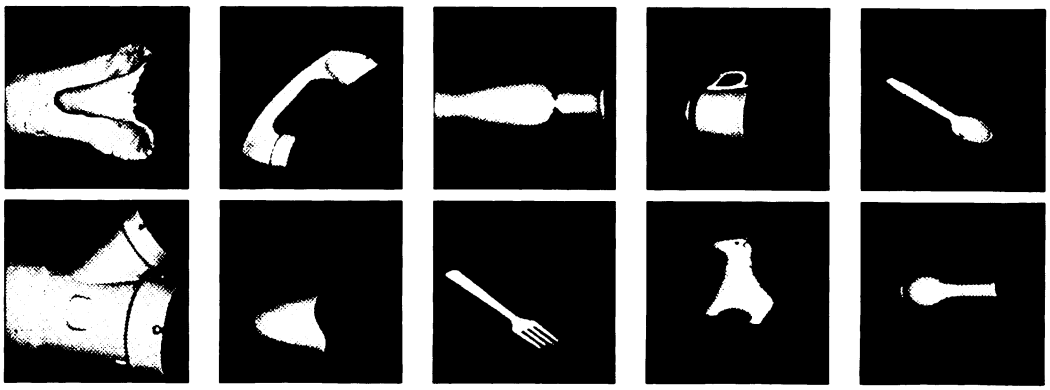


Figure 5.6: Range images of object views stored in the model database.

5.4 Performance of the Recognition System

Three primary components can be identified in our recognition strategy for handling 3D free-form rigid objects: (i) COSMOS as a representation scheme for handling free-form rigid objects, (ii) the concept of shape spectrum, and its use in establishing view clusters when given a large number of views of a free-form object, and also its potential use as a "fast" matching primitive when a large model database of object views is available, and (iii) a graph-based matching scheme for comparing COSMOS representations of two object views for establishing the correct object interpretation via correspondences of surface primitives and for estimation of object pose.

Experimental results obtained from testing each of these components individually have been presented in Chapters 3, 4 and 5. In this section, we concentrate on testing the complete system in an integrated manner. For our discussion below, we categorize the objects used in our experimentation available into three kinds: (i) Type I—real objects that are available for obtaining view data using a laser range scanner and whose geometric models (in the form of CAD models or surface polygonal models) are also available to generate multiple views from arbitrary viewpoints, (ii) Type II—objects for which only their geometric models are available (typically polygonal/CAD models obtained electronically from multiple *ftp* sites), and (iii) Type III—real objects that can be used for obtaining data with a laser scanner, but no geometric models

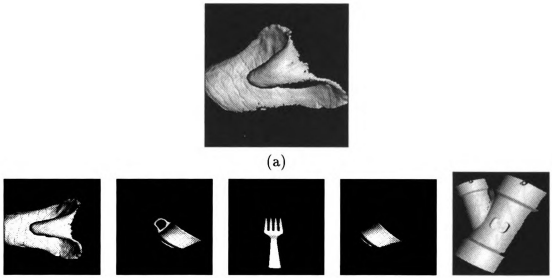


Figure 5.7: Five model hypotheses (b)-(f) generated using shape spectral analysis for a test view (a) of Cobra.

are available for synthetic view generation. Although there are many data resources available for collecting Type II objects, these models are available in various image formats, and one encounters the practical difficulty of implementing or searching for various filters to convert these formats into one uniform format that conforms with the local computing environment. With Type III objects, since our laser scanner is not equipped with six degree motion device, images of multiple views of objects can be obtained to a limited extent by rotating them about the Z axis and this renders the task of obtaining the ground truth information about the objects' orientation harder.

5.4.1 The COSMOS Representation Scheme

Currently, data belonging to Type III objects have been used to illustrate the various components of the COSMOS representation scheme—shape index based surface primitive extraction, building the Gauss patch map, surface connectivity list and the support functions. A total of 21 object views obtained using 11 different real objects

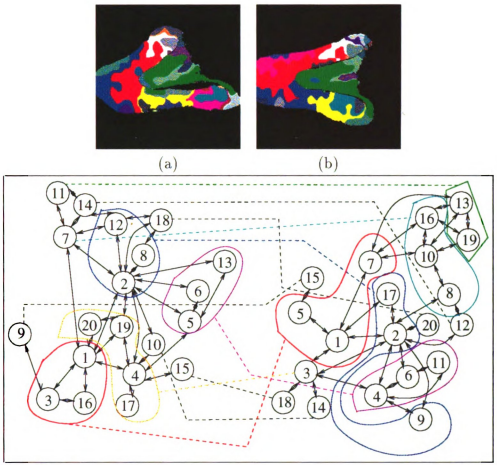


Figure 5.8: COSMOS-based matching: (a) CSMPs on the test view; (b) CSMPs on the model view with the highest matching score; (c) scene-model patch correspondence established between the views of Cobra.

have been used to demonstrate the strengths of the COSMOS representation.

5.4.2 View Grouping and Model View Selection

Here, we have used a mixture of objects belonging to Type I and II categories to illustrate the strengths of shape spectral based matching, view grouping, and model database organization. We have used 20 different complex shaped objects, with 5 from Type I category and 15 from Type II category. We populated an object view database with 6,400 views (320 views/object), trained the view grouping system using this database and tested the performance of our spectral matching scheme with

2,000 independent test views (100 views/object). We also demonstrated the good performance of the scheme on 100 real range images of arbitrarily curved (Type III) free-form objects.

5.4.3 Matching of Object Views using COSMOS

In Section 5.3.6, the strengths of this matching module have been demonstrated with pairs of views of two complex free-from objects. In addition, the integrated recognition strategy involving shape spectral feature-based model selection and detailed COSMOS-based matching was illustrated using an image of Cobra as a test view on a database of multiple views of ten free-form objects (belonging to Type III category). The pose estimation results for views of Vase2, Phone and Cobra are shown in Section 5.4.4.

The representation and recognition system was tested as a whole using 50 independent test images on a database containing 50 model views obtained from ten different free-form objects. The range images of these 100 views were obtained using a laser range scanner (Section 4.5.4). Figure 4.10 shows the range images of the model views in the database. The identity and pose of the objects in the test views shown in Figure 4.11 was established as shown in Figure 5.9.

For each of the 50 test views, its non-planar shape spectrum was computed and a moment vector was derived from the shape spectrum. The database was organized into two levels of hierarchy with the first level containing 10 view clusters corresponding to ten different object classes. The second level contained the fifty model views with five views in each cluster. Comparison of the moment vector of a single test view with those in the database using the shape spectral matching yielded five model view hypotheses that matched most closely with the input moment vector among all the views present within the selected clusters. Forty seven of the test views were found to retrieve at least one model view from their correct object classes among the selected five view hypotheses, thus resulting in a view classification accuracy of 94%.

The COSMOS representation was then computed for each of the fifty test views. During CSMP detection, the shape diameter was adaptively determined to derive

a maximum of fifteen surface patches in each object view. The matching scheme presented in 5.3.1 was used to determine the best object view match among the five candidate view hypotheses short-listed using the shape spectral analysis for each test view. For each test view, the matching scores of the view correspondences returned by the "patch-group" graph matching algorithm were ranked to determine the view with the highest goodness measure among the five model hypotheses. The recognition system was able to identify 82% of the fifty test views correctly by returning a model view from the correct object class with the highest matching score. Out of the nine test views incorrectly matched, three views did not have any view from the correct object class present among the five hypotheses that were examined using the detailed matching scheme. The remaining six errors were mainly caused due to errors in the surface connectivity information introduced by noisy small patches.

Observe that the correct object identity and pose of the input are determined in our recognition system by examining only the few best-matched view hypotheses returned by the shape spectral matching scheme. Hence, the system can fail to recognize the object in the input scene when only model views from incorrect object classes are presented as hypotheses to the COSMOS-based detailed matching stage by the spectrum-based pruning strategy. In addition, the current version of our matching algorithm does not tolerate any violation of connectivity relationships between matched patch-groups, and as observed in our experiments, noisy small patches can introduce serious errors in the adjacency relationships between the patches thus affecting the recognition accuracy. Note also that in the current implementation of the recognition system, we have not incorporated a reject option to prevent matches with high shape spectral dissimilarity values from being examined in detail. Even among the best subset of view hypotheses determined using shape spectral features, it is possible examine only a few hypotheses in detail using the COSMOS-matching stage by enforcing a reject option using a threshold on the dissimilarity levels of the hypotheses. The algorithm can be improved by allowing violations of the connectivity to a small degree depending on the strength of the adjacency as determined by the number of boundary pixels that are shared between a pair of patches.

5.4.4 Pose Estimation

In this section, we continue to use the views of Vase2, Phone and Cobra to illustrate the strength of our pose estimation technique.

The rotational component of the test view of Vase2 (View 1 shown in Figure 5.4(a)) with respect to the model view (View 2 shown in Figure 5.4(b)) was estimated using surface normals of corresponding patch-groups. For each patch-group established in the correspondence, the average surface normal was computed as the mean of the surface normal vectors of the patches present in the group. A total of 10 pairs of MappedPairs was used to estimate the average rotation axis and the angle of rotation. These rotation parameters ($\mathbf{r} = (0.005288, -0.004433, 0.024552)$ and $\theta = 0.180429$ radians) were used to compute the 3×3 rotation matrix [66] which was then used as an initial guess to register the model view (View 2) with the test view (View 1) of Vase2 using the registration technique presented in Chapter 6. We note here that the computational procedure described in Chapter 6 has been further augmented using a verification mechanism during its implementation [49]. We derive the results presented in this section using this augmented procedure. Figure 5.10 shows the iterative registration of the model view with the scene view. It can be seen that the views are in complete registration with one another at the end of seven iterations.

Figure 5.11 shows the registration of the model view with the scene view of Phone through several iterations of the algorithm. The registration scheme converged with the lowest error value at the sixth iteration.

The initial transformation matrix (incorporating both the rotation and translation components of the pose) for aligning the best matched model view (View 2 shown in 5.8(b)) with the test view of Cobra (View 1 shown in 5.8(a)) was computed as

$$T_{init} = \begin{bmatrix} 0.9838 & -0.0044 & -0.0008 & 0.0 \\ 0.0044 & 0.9838 & -0.0009 & 0.0 \\ 0.0008 & 0.0009 & 0.9838 & 0.0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.1}$$

and after three iterations of the registration algorithm, the final transformation matrix was given by

$$T_{fin} = \begin{bmatrix} 0.8849 & -0.4501 & -0.0041 & 1.8645 \\ 0.4499 & 0.8848 & -0.0126 & 0.6214 \\ 0.0094 & 0.0094 & 0.9927 & -0.02292 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.2}$$

Figure 5.12 shows the evolution of the registration of the model view with the scene view of Cobra. It can be seen that even with a coarse initial estimate of the rotation (see Figure 5.12(a)), the registration technique can align the two views successfully within a few iterations.

Computational Time Requirements

Given a range image, estimation of local curvatures at each surface point and the smoothing of curvatures using the curvature consistency algorithm takes between 5 and 15 minutes on the average on images of size 240×240 containing about 15,000 surface points. The CSMP extraction step, once the shape index values are determined using the curvatures requires computational time of the order of an hour. Obtaining the COSMOS representation once the CSMPs have been determined and the computation of shape spectrum takes about a few seconds on the average. The matching of shape spectral features to identify a small set (10) of object views out of 3,200 views takes about 20 msec. View verification using graph-matching scheme was carried out with only fifteen largest (in surface area) patches in the views and this takes of the order of several minutes to compare a pair of views. Given a coarse correct initial guess, the registration stage in our system on the average takes about 30 seconds to register two range images whose sizes are 640×480 on a SPARCstation 10 with 32MB RAM.

With our current two-stage matching strategy comprising shape spectral moments comparison and COSMOS based detailed matching, the system scales up linearly with

respect to the number of objects in the database. By adding more levels to the database hierarchy (as mentioned in Section 7.2.4), the cost of matching can be further reduced. Constructing such a search tree based on COSMOS features remains an interesting line of future research.

5.5 Summary

We addressed the problem of recognizing 3D rigid free-form objects using the COSMOS representation scheme in this chapter. We proposed a novel multi-level matching strategy that employs shape spectral analysis and features derived from the COSMOS representations of objects for fast and accurate recognition of sculpted objects. A small subset of candidate model views are isolated from the database using shape spectrum based model selection scheme. During view hypothesis verification, we use the COSMOS representations of object views to determine the scene-model feature correspondences using a combination of search methods, and thus identify and localize the object in the input view. Experiments on a database containing 100 real range images of views of ten objects have demonstrated the encouraging performance of our COSMOS-based 3D object recognition system. Although we have assumed unoccluded views of the objects, we expect that small amounts of occlusions will be tolerated by the graph-based matching scheme in our recognition system as long as the salient surfaces that provide the characteristic shape information about the objects are visible.

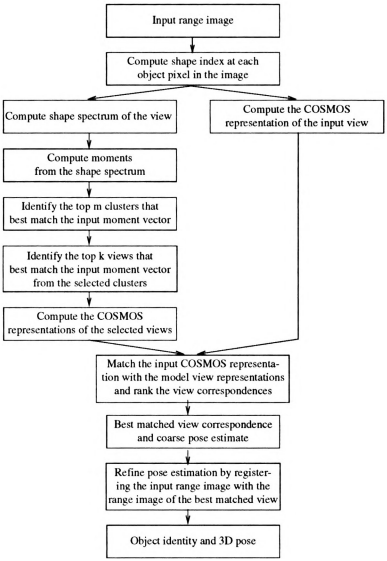


Figure 5.9: Matching a test view in the COSMOS-based recognition system.

Figure 5.10: Pose estimation: (a) Model view registered with the test view of Vase2 at the end of first iteration; (b) registered views after 3 iterations; (c) Registered views after 4 iterations; (d) registered views after 5 iterations; (e) registered views at the convergence of the algorithm.

Figure 5.11: Pose estimation: (a) Model view registered with the test view of Phone at the end of first iteration; (b) registered views after 2 iterations; (c) registered views after 3 iterations; (d) registered views after 4 iterations; (e) registered views at the convergence of the algorithm.

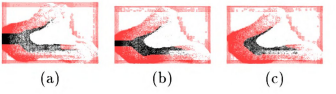


Figure 5.12: Pose estimation: (a) Model view registered with the test view of Cobra at the end of first iteration; (b) registered views after the second iteration; (c) registered views at the convergence of the algorithm.

Chapter 6

Pose Estimation by Registering Object Views

This chapter presents a technique to estimate the pose of a free-form object once its identity has been ascertained using the COSMOS-based recognition scheme. Pose estimation is cast as a registration problem. We assume that the object in the input range image has been recognized as one of the objects stored in the model database. The sensed range image and the range image of the stored view are then registered, thereby estimating the transformation between them. An initial estimate of the transformation between the sensed and the stored images is determined using the surface normals once the correspondences between the maximal patches visible in both the views have been established using our COSMOS based graph-matching scheme and it is then refined using the method presented in this chapter. If it is not known, then our method can estimate it using the range data themselves. A more accurate pose is computed by refining the initial estimate using an iterative minimization scheme.

6.1 Robust Object View Registration

Our approach to pose estimation is to formulate it as the problem of obtaining an accurate registration between the input and the matched object views, thereby es-

timating the transformation between the range images of a free-form object even in the presence of noise or uncertainties in surface data. A registration-based approach is especially suitable for free-form surfaces as it does not rely on the presence of any salient features. In addition, a registration-based approach can offset errors introduced in the initial estimate of the object pose due to poor feature localization and noise-corrupted surface normals.

However, view registration can be affected by the noise in the sensed data to a certain extent. Most approaches to range image registration assume that the surface depth data are accurate and hence do not take into account the sensitivity of the estimated transformation to noise. In order to provide a view registration algorithm that is reliable in the presence of uncertainties in the surface depth measurements, we derive a a minimum variance estimator (MVE) [50] for computing the transformation parameters from range data of views of an object.

Another important application of our registration technique is automatic construction of object models from multiple range views. Automatic construction of models of objects involves three steps: (i) data acquisition, (ii) registration of different views, and (iii) integration. Data acquisition involves obtaining either intensity or depth data of an object from multiple views. Integration of multiple views is dependent on the representation chosen for the model description and also requires knowledge of the transformation relating the data obtained from multiple views. The intermediate step, registration, is also known as the *correspondence problem* and its goal is to find the transformations that relate multiple views. In this chapter we focus on the issue of registering multiple range images of different views of an object.

In order to register object views accurately, we model errors introduced due to sensor inaccuracies in range data explicitly. We have not seen any work that reports to date, establishing the dependencies between the orientation of a surface, noise in the sensed surface data, and the accuracy of surface normal estimation and how these dependencies can affect the estimation of transformation parameters that relate a pair of object views. We present a detailed analysis of this "orientation effect" with geometrical arguments and experimental results.

6.2 Previous Work

There have been several research efforts directed at solving the registration problem. They fall into two categories: (i) the first kind relies on precisely calibrated data acquisition device to determine the transformations that relate the views and (ii) the second involves techniques to estimate the transformations from the data directly. Bhanu [16] describes an object modeling system in which objects are rotated through known angles to obtain multiple views. Ahuja and Veenstra [1] use orthogonal views to construct octree object models. The correspondence problem is solved easily with the calibration of data acquisition facilities in this work. Vemuri and Aggarwal [165] used a base-plane pattern to estimate the interframe rotation of objects by *corresponding* the control pattern in intensity images which are acquired simultaneously with the range images. These techniques are inadequate for constructing a complete description of complex shaped objects because views are restricted to rotations or to some known viewpoints only. Therefore, we cannot make use of the object surface geometry in the selection of vantage views to obtain measurements.

Inter-image correspondence has also been established by matching surface features derived from the data [62]. The accuracy of the feature detection method employed determines the accuracy of feature correspondences. Potmesil [131] matched multiple range views using a heuristic search in the view transformation space. Though quite general, this technique involves searching a huge parameter space, and even with good heuristics, it may be computationally very expensive. Chen and Medioni avoid the search by assuming an initial approximate transformation for the registration, which is improved with an iterative algorithm [38] that minimizes the distance from points in a view to tangential planes at *corresponding* points in other views. Besl and Mckay [15] proposed an iterative closest point algorithm for registration of free-form surfaces which requires the specification of an appropriate procedure to find the closest point on a geometric entity to a given point. Blais and Levine [19] propose a reverse calibration of the range-finder to determine the point correspondences between the views directly and use stochastic search to estimate the transformation. These

approaches, however, do not take into account the presence of noise or inaccuracies in the data and its effect on the estimated view-transformation. A related work by Turk and Levoy [159] that describes an entire system for registration and integration of object views uses a variant of the *iterated closest-point* algorithm [15]. Our registration technique also uses a distance minimization algorithm to register a pair of views, but we do not impose the requirement that one surface has to be strictly a subset of the other.

6.3 Error in Surface Measurements

Range data are often corrupted by measurement errors and sometimes lack of data. The errors in surface measurements of an object include scanner errors, camera distortion, and spatial quantization. The missing data can be due to self-occlusion, overlapping objects, or sensor shadows. Therefore, the fusion of multiple views during 3D model construction should take into account different uncertainties in observations. The registration errors affect the integration stage in model construction. It can also affect the surface classification. Even if the noise in range data is small, it is important to ensure that the estimated transformation is accurate because when data from different views are merged based on an inaccurately estimated transformation, it may result in holes as the merged data may have gaps and may also introduce discontinuities in the surfaces when multiple data points are mapped to represent the same physical point on the surface. Due to noise, it is generally impossible to obtain a solution for a rigid transformation that fits two sets of noisy three-dimensional points exactly. The least-squares solution in [38] is non-optimal as it treats all surface measurements with different reliabilities equally. Our objective is to derive a transformation that globally registers the noisy data in some optimal sense.

With range sensors that provide measurements in the form of a graph surface $z = f(x, y)$, it is assumed that the error is present along the z axis only, as the x and y measurements are usually laid out in a grid. The effects of errors in the z measurements on the estimation of surface attributes such as normals may vary depending

on the orientation of a surface patch. For example, the noisy z measurement on a horizontal surface patch affects the estimation of the surface normal more than an erroneous z value on an inclined surface patch, as shown in Figures 6.2 and 6.3. We discuss this aspect more in detail in Section 6.5.1. There are different uncertainties along different surface orientations and they need to be handled appropriately during view registration.. Furthermore, the measurement error is not uniformly distributed over the entire image. The error may depend on the position of a point, relative to the object surface. A measurement error model dealing with the sensor's viewpoint has been previously proposed [84] for surface reconstruction where the emphasis was to recover straight line segments from noisy single scan 3D surface profiles.

In this chapter we investigate the effect of measurement noise on the *registration* of multiple views to accurately estimate the relative transformation between them and propose a new method that improves upon the approach of Chen and Medioni [38]. The registration technique is an iterative least squares algorithm minimizing the sum of weighted distances between a set of control points chosen from range data of an object observed from some viewpoint and the tangential planes fitted at the *corresponding* control points in a range image of another view of the object. We use the terms "view" and "image" interchangeably in this chapter. We formulate a noise model that characterizes the error in estimating tangent planes from noisy range data, and present a minimum variance estimation of the transformation parameters that relate the data from two views. Our model handles numerical errors in z values and surface orientation effects. The minimum variance estimator algorithm proposed here handles the inaccuracies introduced in the range data by the sensor. It only assumes that noise distributions of the data are well behaved and possess short tails. When a Gaussian distribution is assumed to model the noise in the data, the minimum variance estimator becomes equivalent to a weighted linear least-squares algorithm.

6.4 A Non-Optimal Algorithm for Registration

Two views of a surface are said to be in registration [38] when *any* pair of points, $\mathbf{p}$ and $\mathbf{q}$ from the two views representing the same object surface point, can be related to each other by a *single* rigid 3D spatial transformation T, such that

$$\forall \mathbf{p} \in P, \quad \exists \mathbf{q} \in Q \text{ such that } \|T\mathbf{p} - \mathbf{q}\| = 0, \tag{6.1}$$

where P and Q are two views of the same surface, $T\mathbf{p}$ is a point obtained by applying the transformation T to $\mathbf{p}$, and T is a transformation expressed in homogeneous coordinates as given below:

$$T = T(\alpha, \beta, \gamma, t_x, t_y, t_z) =$$

$$\begin{bmatrix} \cos\gamma\cos\beta & \cos\gamma\sin\beta\sin\alpha - \sin\gamma\cos\alpha & \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma & t_x \\ \sin\gamma\cos\beta & \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma & \sin\gamma\sin\beta\cos\alpha - \cos\gamma\sin\alpha & t_y \\ -\sin\beta & \cos\beta\sin\alpha & \cos\beta\cos\alpha & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.2}$$

where α, β and γ are rotation angles about the x, y and z axes, respectively, and t_x, t_y and t_z are the translation parameters. The transformation T needed to bring the two views into registration has 6 degrees of freedom. Thus, the problem of registration is to search for such a transformation in the six-dimensional parameter space, which satisfies Eq. (6.1).

The approach of [38] is based on the assumption that an approximate transformation between two views is already known, i.e., data from the two views are approximately registered, and the goal is to refine the initial estimate to obtain more accurate global registration. Given a set of N pairs of *corresponding* points called *control points* in two views, $\mathbf{p}_i \in P$ and $\mathbf{q}_i \in Q$, $\quad i = 1 \ldots N$, the transformation can

be estimated by minimizing

$$e = \sum_{i=1}^{N} \|T\mathbf{p}_i - \mathbf{q}_i)\|^2, \tag{6.3}$$

where $N \geq 3$.

Since $\mathbf{q}_i \in Q$ *corresponding* to a point $\mathbf{p}_i \in P$ is not usually known, Chen and Medioni [38] used the following objective function to minimize the distances from points on one surface to another iteratively:

$$e^k = \sum_{i=1}^{N} d_s^2(T^k\mathbf{p}_i, S_i^k), \tag{6.4}$$

where T^k is the 3D transformation applied to a control point $\mathbf{p}_i \in P$ at the kth iteration, $l_i = \{\mathbf{a} \mid (\mathbf{p}_i - \mathbf{a}) \times \mathbf{n}_{\mathbf{p}_i} = 0\}$ is the line normal to P at $\mathbf{p}_i$, $\mathbf{q}_i^k = (T^k l_i) \cap Q$ is the intersection point of surface Q with the transformed line $T^k l_i$, $\mathbf{n}_{\mathbf{q}_i}^k$ is the normal to Q at $\mathbf{q}_i^k$, $S_i^k = \{\mathbf{s} \mid \mathbf{n}_{\mathbf{q}_i}^k \cdot (\mathbf{q}_i^k - \mathbf{s}) = 0\}$ [1] is the tangent plane to Q at $\mathbf{q}_i^k$ and d_s is the signed distance from a point to a plane as given in Eq. (6.5). Figure 6.1 illustrates the distance measure d_s between surfaces P and Q.

The registration algorithm thus finds a T that minimizes e^k, using a least squares method iteratively. The tangent plane S_i^k serves as a local linear approximation to the surface Q at a point. The intersection point $\mathbf{q}_i^k$ is an approximation to the actual *corresponding* point $\mathbf{q}_i$ that is unknown at each iteration k. An initial T^0 that approximately registers the two views is used to start the iterative process. The signed distance d_s, from a transformed point $T\mathbf{p}_i$, $\mathbf{p}_i \in P$ to a tangential plane $S_i^k \in Q$ is given by

$$d_s = -\frac{\mathcal{A}x + \mathcal{B}y + \mathcal{C}z + \mathcal{D}}{\sqrt{\mathcal{A}^2 + \mathcal{B}^2 + \mathcal{C}^2}}, \tag{6.5}$$

where $T\mathbf{p}_i = (x, y, z)^T$ and $S_i^k = (\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D})^T$ define the transformed point and the tangential plane respectively. Note, $(x, y, z)^T$ is the transpose of the vector (x, y, z).

[1] '$\cdot$' stands for the scalar product and '$\times$' for the vector product.

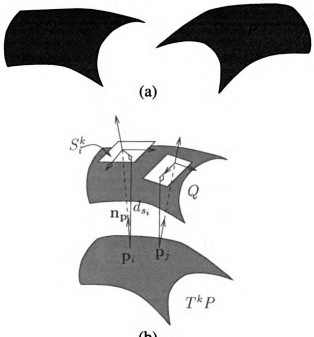


Figure 6.1: Point-to-plane distance: (a) Surfaces P and Q before the transformation T^k at iteration k is applied; (b) distance from the point $\mathbf{p}_i$ to the tangent plane S_i^k of Q.

By minimizing the distance from a point to a plane, only the direction in which the distance can be reduced is constrained.

The algorithm that performs this minimization is as follows:

1. A set of control points $\mathbf{p}_i \in P, i = 1, 2, \cdots, N$ is selected and the surface normal $\mathbf{n}_{\mathbf{p}_i}$ is computed at each point. Let an initial transformation be T_o.

2. The following procedure is repeated for every iteration k for $k = 1, 2, \cdots$ until the process converges.

   (i) For each control point $\mathbf{p}_i$,

   - The transformation T^{k-1} is applied to both the control point $\mathbf{p}_i$ and the normal $\mathbf{n}_{\mathbf{p}_i}$ to get $\mathbf{p}_i'$ and $\mathbf{n}_{\mathbf{p}_i}'$.
   - The intersection $\mathbf{q}_i^k$ of surface Q and the normal line l_i defined by $\mathbf{p}_i'$ and $\mathbf{n}_{\mathbf{p}_i}'$ is computed.
   - The tangent plane S_i^k of Q is computed at $\mathbf{q}_i^k$.

- The distance d_s between $\mathbf{p}_i$ and S_i^k is determined.

(ii) The transformation T that minimizes e^k in Eq. (6.4) is estimated with a least squares method.

(iii) Now let $T^k = TT^{k-1}$.

The convergence of the process can be tested by verifying that the difference between the errors e^k at any two consecutive iterations is less than a pre-specified threshold. The line-surface intersection given by the intersection of the normal line l_i and Q is found using an iterative search near the neighborhood of prospective intersection points.

6.5 Registration and Error Modeling

Chen and Medioni's algorithm is not optimal because it does not handle the errors in z measurements. Though there are no actual outliers in the range data, occluded and noisy surface points can be considered such. Then, a simple least-squared-error estimation procedure is not desirable since the estimated transformation parameters will be affected more by outliers than the actual data. The least-squares estimation method treats outliers equally since all errors are equally weighted proportional to their magnitude. Instead, we would like an estimation procedure that throws out or gives low weight to the noisy measurements.

We show that the noise in z values affects the estimation of the tangential plane parameters differently depending on how a surface is oriented. Since the estimated tangential plane parameters play a crucial role in determining the distance d_s (which is being minimized to estimate T), it is important to study the effect of noise on the estimation of the parameters of the plane and the minimization of d_s. Note that the error in the iterative estimation of T is a combined result of errors in each control point $(x, y, z)^T$ from view 1 and errors in fitting a tangential plane at the *corresponding* control points from view 2.

6.5.1 Fitting Planes to Surface Data with Noise

Figures 6.2 and 6.3 illustrate the effect of noise in the values of z on the estimated plane parameters. For the horizontal plane shown in Figure 6.2, an error in z (the uncertainty region around z) directly affects the estimated surface normal. In the case of an inclined plane, the effect of errors in z on the normal to the plane is much less pronounced as shown in Figure 6.3. Here, even if the error in z is large, only its projected error along the normal to the plane affects the normal estimation. This projected error becomes smaller than the actual error in z as the normal becomes more and more inclined with respect to the vertical axis. Therefore, our hypothesis is that as the angle between the vertical (Z) axis and the normal to the plane increases, the difference between the fitted plane parameters and the actual plane parameters should decrease. This hypothesis has been verified by our simulations as explained below.

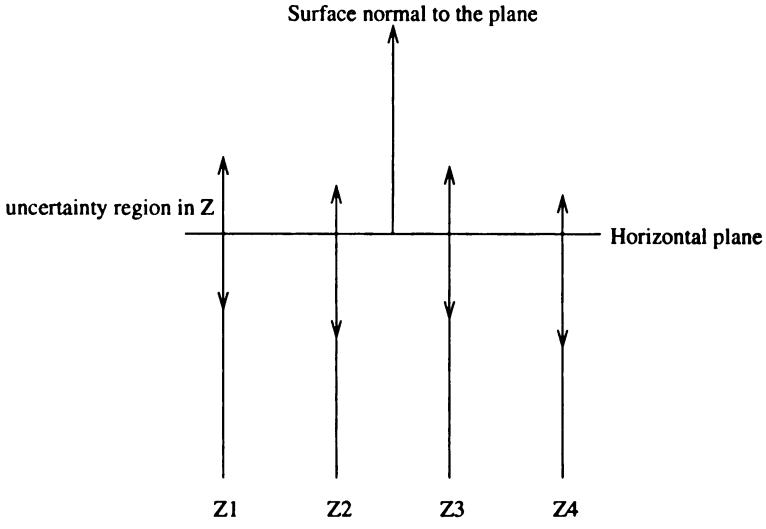


Figure 6.2: Effect of noise in z measurements on the fitted normal when the plane is horizontal. The double-headed arrows indicate the uncertainty in depth measurements.

We carried out the simulations to study the actual effect of the noise in the z measurements on estimating the plane parameters and to verify the above hypothesis. We obtained the planar parameters from the surface measurements using two methods:(i) eigenvector method and (ii) linear regression.

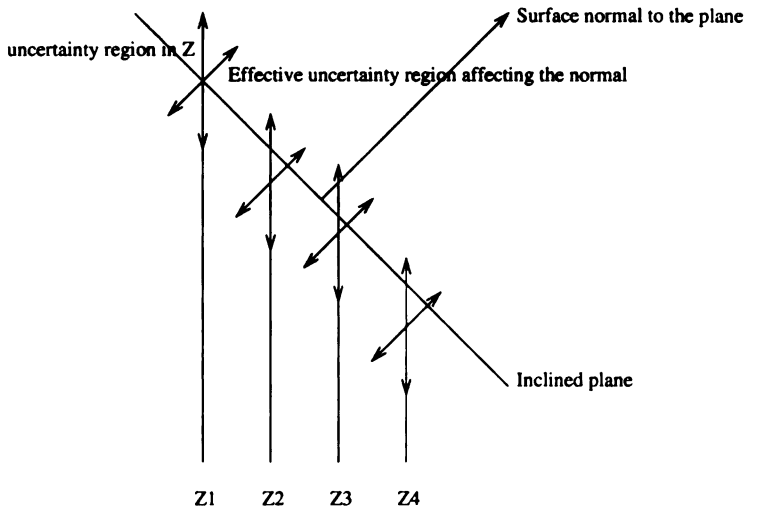


Figure 6.3: Effect of noise in z measurements on the fitted normal when the plane is inclined. The double-headed arrows indicate the uncertainty in depth measurements.

Eigenvector Method for Planar Fitting

The conventional method for fitting planes to a set of 3D points uses a linear least squares algorithm which is described in the following section. Note that the linear regression method implicitly assumes that two of the three coordinates are measured without errors. However, it is possible that in general, surface points can have errors in all three coordinates, and surfaces can be in any orientation. Hence, we use a classical eigenvector method (principal components analysis) [65] that allows us to extract all linear dependencies.

Let the plane equation be $\mathcal{A}x + \mathcal{B}y + \mathcal{C}z + \mathcal{D} = 0$. Let $X_i = (x_i, y_i, z_i)^T, \quad i = 1, 2, \cdots, n$, be a set of surface measurements used in fitting a plane at a point on a surface. Let

$$A = \begin{bmatrix} x_1 & y_1 & z_1 & 1 \\ x_2 & y_2 & z_2 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & z_n & 1 \end{bmatrix} \tag{6.6}$$

and $h = (\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D})^T$ be the vector containing the plane parameters. We solve for the

vector h such that $\|Ah\|$ is minimized. The solution of h is a unit eigenvector of A^TA associated with the smallest eigenvalue. We renormalize h such that $(\mathcal{A},\mathcal{B},\mathcal{C})^T$ is the unit normal to the fitted plane and $\mathcal{D}$ is the distance of the plane from the origin of the coordinate system. This planar fit minimizes the sum of the squared perpendicular distances between the data points and the fitted plane and is independent of the choice of the coordinate frame.

In our computer simulations, we used synthetic planar patches as test surfaces. The simulation data consisted of surface measurements from planes at various orientations with respect to the vertical axis. Independent and identically distributed (*i.i.d.*) Gaussian and uniform noise with different variances were added to the z measurements of the synthetic planar data. The standard deviation of the noise used was in the range 0.001–0.005 *in.* as this realistically models the error in z introduced by a Technical Arts 100X White range scanner [120] that was employed to obtain the range data for our experiments. The planar parameters were estimated using the eigenvector method, at different surface points with a neighborhood of size 5×5. The error E_{fit} in fitting the plane was defined as the norm of the difference between the actual normal to the plane and the normal of the fitted plane estimated with the eigenvector method. Figure 6.4 shows the plot of E_{fit} versus the orientation of the normal to the simulated plane (with respect to the vertical axis) at different noise variances. The plot shows the error values averaged over 1,000 trials.

It can be seen from the Figure 6.4 that the error in fitting a plane decreases with an increase in the angle between the vertical axis and the normal to the plane since the error in z contributes less to the estimation of the normal to an inclined plane. When the plane is nearly horizontal (the angle between the vertical axis and the normal to the plane is small), the error in z entirely contributes to the error in fitting as shown in Figure 6.2. The error plot in Figure 6.4 was observed to have the same behavior for varying amounts of variance with the Gaussian noise model. The results were found to have similar characteristics with a uniform noise model also as shown in Figure 6.4. These simulations confirm our hypothesis about the effect of noise in z on the fitted plane parameters as the surface orientation changes.

Plane Fitting Using Linear Regression

Since we assume errors in z direction only, it may be more appropriate to use a linear least squares method to fit the plane in order to verify our hypothesis instead of the general eigenvector method described in the above section. The linear regression method brings out the errors in z direction explicitly.

With the linear regression method, a plane equation is redefined as $z = ax + by + c$. Let $X_i = (x_i, y_i, z_i)^T, \quad i = 1, 2, \cdots, n$, be a set of surface measurements used in fitting a plane at a point on a surface. Let

$$A = \begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ \vdots & \vdots & \vdots \\ x_n & y_n & 1 \end{bmatrix} \quad \text{and} \quad Z = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix} \tag{6.7}$$

and $m = (a, b, c)^T$. We solve for m such that $Z = Am$ is satisfied in a least square sense. The solution of m is given by $(A^T A)^{-1} A^T Z$ and the covariance matrix Γ_{abc} associated with m is given by $(A^T \Gamma_Z A)$. We compute the planar parameters $(\mathcal{A}, \mathcal{B}, \mathcal{C})^T$ from $(a, b, c)^T$ such that $(\mathcal{A}, \mathcal{B}, \mathcal{C})^T$ is the unit normal to the plane and $\mathcal{D}$ is the distance of the fitted plane from the origin. Note that the planar fit here minimizes the squared differences between the actual z measurements and the z coordinates of the fitted plane.

We repeated our simulations described in Section 6.5.1, but used the linear regression method to fit planes to data. The experimental details for these simulations were identical to those described before. Figure 6.5 shows the estimated E_{fit} between the fitted and actual normals to the plane at various surface orientations.

It can be seen from Figure 6.5 that the error between the actual and the estimated normals again decreases with an increase in the angle between the vertical axis and the normal to the plane, thus confirming our hypothesis. The error plot was observed to exhibit the same behavior for varying amounts of variance with the Gaussian noise model. The results were found to be similar with uniform noise model also. Our model

works for general noise distributions as long as errors in different measurements are uncorrelated and their distributions have short tails.

6.6 An Optimal Registration Algorithm

Since the estimated tangential plane parameters are affected by the noise in z measurements, any inaccuracies in them, in turn influence the accuracy of the estimates of d_s, thus affecting the error function being minimized during the registration. Therefore, we characterize the error in the estimates of d_s by modeling the uncertainties associated with the z measurements using weights. Our approach is inspired by the Gauss-Markov theorem [171] which states that an unbiased linear minimum variance estimator of a parameter vector $\mathbf{m}$ when $\mathbf{y} = \mathbf{f}(\mathbf{m}) + \delta_{\mathbf{y}}$, is the one that minimizes $(\mathbf{y} - \mathbf{f}(\mathbf{m}))^T \Gamma_{\mathbf{y}}^{-1} (\mathbf{y} - \mathbf{f}(\mathbf{m}))$, where $\delta_{\mathbf{y}}$ is a random vector with zero mean and covariance matrix $\Gamma_{\mathbf{y}}$. Based on this theorem, we formulate an optimal error function for registration of range images as

$$e^k = \sum_{i=1}^{N} \frac{1}{\sigma_{d_s}^2} d_s^2, \tag{6.8}$$

where $\sigma_{d_s}^2$ is the estimated variance of the distance d_s. When the reliability of a z value is low, the variance of the distance $\sigma_{d_s}^2$ is large and the contribution of d_s to the error function is small, and when the reliability of the z measurement is high, $\sigma_{d_s}^2$ is small, and the contribution of d_s is large. In other words, d_s with minimum variances affect the error function more. One of the advantages of this minimum variance criterion is that we do not need the exact noise distribution. What we require is that the noise distribution be well-behaved and have short tails. In our simulations, we employ both Gaussian and uniform noise distributions to illustrate the effectiveness of our method. We need to know only the second-order statistics of the noise distribution, which in practice can often be estimated.

In the following section, we present a formulation to characterize and estimate $\sigma_{d_s}^2$.

6.6.1 Estimation of the Variance $\sigma_{d_s}^2$

We need to estimate $\sigma_{d_s}^2$ to model the reliability of the computed d_s. This can then be used in our optimal error function in Eq. (6.8). Let the set of all the surface points be denoted by P and the errors in the measurements of these points be denoted by a random vector ϵ. The error e_{d_s} in the distance computed is due to the error in the estimated plane parameters and the z measurement. It is a function of P and ϵ:

$$e_{d_s} = f(P, \epsilon). \tag{6.9}$$

Our goal is to estimate the error e_{d_s} given the surface measurements P. However, we do not know ϵ. If we can estimate the standard deviation of e_{d_s} (with ϵ as a random vector) from the noise-corrupted surface measurements P, we can use it in Eq. (6.8).

Estimation of $\sigma_{d_s}^2$ Based on Perturbation Analysis

Perturbation analysis is a general method for analyzing the effect of noise in data on the eigenvectors obtained from the data. It is general in the sense that errors in x, y and z can all be handled in this model. This analysis is also related to the general method of plane fitting that we studied - the eigenvector approach. The analysis to estimate $\sigma_{d_s}^2$ is simpler as discussed in the next section if we use linear regression method to do plane fitting. Note that when we use linear regression we can assume that an error is present in z component only.

Since we fit a plane with the eigenvector method that uses the symmetric matrix $C = A^T A$ computed from the (x, y, z) measurements in the neighborhood of a surface point, we need to analyze how a small perturbation in the matrix C caused by the noise in the measurements can affect the eigenvectors. Recall that these eigenvectors determine the plane parameters $(\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D})^T$ which in turn determine the distance d_s. We assume that the noise in the measurements has zero mean and some variance and that the latter can be estimated empirically. The correlation in noise at different points is assumed to be negligible. Estimation of correlation in noise is very difficult

but even if we estimate it, its impact may turn out to be insignificant. We estimate the standard deviation of errors in the plane parameters and in d_s on the basis of the first-order perturbations, i.e., we estimate the "linear terms" of the errors.

Before we proceed, we discuss some of the notational conventions that are used: I_m is a $m \times m$ identity matrix; $diag(a, b)$ is a 2×2 diagonal matrix with a and b as its diagonal elements. Given a noise-free matrix A, its noise-matrix is denoted by Δ_A and the noise-corrupted version of A is denoted by $A(\epsilon)$, i.e.,

$$A(\epsilon) = A + \Delta_A.$$

The vector δ is used to indicate the noise vector,

$$X(\epsilon) = X + \delta_X.$$

We use Γ with a corresponding subscript to specify the covariance matrix of the noise vector/matrix. For a given matrix $A = [A_1\ A_2\ \cdots\ A_n]$, a vector $\mathbf{A}$ can be associated as

$$\mathbf{A} = \begin{bmatrix} A_1 \\ A_2 \\ \vdots \\ A_n \end{bmatrix}.$$

In other words, $\mathbf{A}$ consists of the column vectors of A that are lined up together.

As proved in [172], if C is a symmetrical matrix (A^TA) formed from the measurements and h is the parameter vector $(\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D})^T$ given by the eigenvector of C associated with the smallest eigenvalue, say λ_1, then the first-order perturbation in the eigenvector (parameter vector) h is given by

$$\delta_h \cong H \Delta H^T \Delta_{A^TA} h, \tag{6.10}$$

where

$$\Delta = diag\{0, (\lambda_1 - \lambda_2)^{-1}, (\lambda_1 - \lambda_3)^{-1}, (\lambda_1 - \lambda_4)^{-1}\}, \tag{6.11}$$

and H is an orthonormal matrix such that

$$H^{-1}CH = diag\{\lambda_1, \lambda_2, \lambda_3, \lambda_4\}. \tag{6.12}$$

Δ_{A^TA} is a 4×4 noise or perturbation matrix associated with A^TA. If the noise matrix Δ_{A^TA} can be estimated, then the perturbation δ_h in h can be estimated by a first-order approximation as given in Eq. (6.10).

We estimate Δ_{A^TA} from the perturbation in the surface measurements. We assume, for the sake of simplicity of analysis that only the z component of a surface measurement $X_i = (x_i, y_i, z_i)^T$ has errors, with this general model. This analysis is easily and directly extended to include errors in x and y if their noise variances are known.

Let z_i have additive errors δ_{z_i}, for $1 \leq i \leq n$. We then get

$$\Delta_A^T = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \delta_{z_1} & \delta_{z_2} & \cdots & \delta_{z_n} \\ 0 & 0 & \cdots & 0 \end{bmatrix}. \tag{6.13}$$

If the errors in z at different points on the surface have the same variance σ^2, we get the covariance matrix

$$\Gamma_{A^T} = \sigma^2 \; diag(P_1, P_2, \cdots, P_n), \tag{6.14}$$

where $P_i, 1 \leq i \leq n$, is a 4×4 sub matrix:

$$P_i = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{6.15}$$

Now, consider the error in h. As stated before, we have

$$\begin{aligned} \delta_h &\cong H\Delta H^T \Delta_{A^TA} h \\ &= H\Delta H^T [\mathcal{A}I_4 \ \mathcal{B}I_4 \ \mathcal{C}I_4 \ \mathcal{D}I_4]\delta_{A^TA} \\ &\triangleq G_h \delta_{A^TA}. \end{aligned} \tag{6.16}$$

In the above equation, we have rewritten the matrix Δ_{A^TA} as a vector δ_{A^TA} and moved the perturbation to the extreme right of the expression. Then the perturbation of the eigenvector is the linear transformation (by matrix G_h) of the perturbation vector δ_{A^TA}. Since we have $\Gamma_{A^T}(= \Gamma_A^T)$, we need to relate δ_{A^TA} to δ_{A^T}. Using a first-order approximation [172], we get

$$\Delta_{A^TA} \cong A^T \Delta_A + \Delta_A^T A. \tag{6.17}$$

Letting $A^T = [a_{ij}]^T \triangleq [A_1 \ A_2 \ \cdots \ A_n]$, we write

$$\delta_{A^TA} \cong G_{A^TA} \delta_{A^T}, \tag{6.18}$$

where G_{A^TA} is easily determined from the equation $G_{A^TA} = [F_{ij}] + [G_{ij}]$, where $[F_{ij}]$ and $[G_{ij}]$ are matrices with $4 \times n$ submatrices F_{ij} and G_{ij} respectively; $F_{ij} = a_{ji} I_4$, and G_{ij} is a 4×4 matrix with the ith column being the column vector A_j and all other columns being zero. Thus, we get

$$\delta_h \cong G_h \delta_{A^TA} \cong G_h G_{A^TA} \delta_{A^T} \triangleq D_h \delta_{A^T}. \tag{6.19}$$

Then the covariance matrix of h is given by

$$\Gamma_h \cong D_h \Gamma_{A^T} D_h^T. \tag{6.20}$$

The distance d_s is affected by the errors in the estimation of the plane parameters $(\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D})^T$ and the z measurement in $(x_i, y_i, z_i)^T$. The error variance in the distance d_s is, therefore, given by

$$\sigma_{d_s}^2 = \begin{bmatrix} \frac{\partial d_s}{\partial \mathcal{A}} \\ \frac{\partial d_s}{\partial \mathcal{B}} \\ \frac{\partial d_s}{\partial \mathcal{C}} \\ \frac{\partial d_s}{\partial \mathcal{D}} \\ \frac{\partial d_s}{\partial z} \end{bmatrix} \times [\Gamma_{hz}] \times \begin{bmatrix} \frac{\partial d_s}{\partial \mathcal{A}} & \frac{\partial d_s}{\partial \mathcal{B}} & \frac{\partial d_s}{\partial \mathcal{C}} & \frac{\partial d_s}{\partial \mathcal{D}} & \frac{\partial d_s}{\partial z} \end{bmatrix}. \tag{6.21}$$

The covariance matrix Γ_{hz} is given by

$$\Gamma_{hz} = \begin{bmatrix} \Gamma_h & 0 \\ 0 & \sigma^2 \end{bmatrix}. \tag{6.22}$$

Once the variance of d_s, $\sigma_{d_s}^2$ is estimated, we rewrite our error function that is to be minimized to estimate T as

$$e^k = \sum_{i=1}^{N} \frac{1}{\sigma_{d_s}^2} d_s^2(T^k \mathbf{p}_i, S_i^k). \tag{6.23}$$

Figure 6.6 shows the plot of the *actual* standard deviation of the distance d_s versus the orientation of the plane with respect to the vertical axis. Note that the mean of d_s is zero when the surface points are in complete registration and there is no noise. We generated two views of synthetic planar patches with transformation T between the views being an identity transformation. We experimented with the planar patches at various orientations. We added uncorrelated Gaussian noise independently to the two views. Then we estimated the distance d_s at different control points from Eq.(6.5)

and computed its standard deviation. The plot shows the values averaged over 1,000 trials. As indicated by our hypothesis, the actual standard deviation of d_s decreases as the planar orientation goes from being horizontal to vertical. As the variance of the added Gaussian noise to the z measurements increases, σ_{d_s} also increases. The figure also shows the results obtained when we added uniform noise to the data.

We compare the actual variance with the *estimated* variance of the distance (Eq. (6.21)) in order to verify whether our modeling of errors in z values at various surface orientations is correct. We computed the estimated variance of the distance d_s using our error model from Eq. (6.21) with the same experimental setup as described above. Figure 6.7 illustrates the behavior of the estimated standard deviation of d_s as the inclination of the plane (the surface orientation) changes. A comparison of Figures 6.6 and 6.7 shows that both the actual and the estimated standard deviations of d_s have similar behavior with varying planar orientation and their values are proportional to the amount of noise added. This proves the correctness of our error model of z and its effect on the distance d_s.

Estimation of $\sigma^2_{d_s}$ with Linear Regression for Planar Fitting

The analysis in the previous section to estimate the variance of the distance d_s becomes simpler when we use linear regression to fit a plane to the surface measurements. Since we assumed errors in z measurements only, it is possible to obtain the covariance matrix Γ_h directly using the linear regression method. The difficulty with the eigenvector method was in estimating the covariance matrix Γ_h of the fitted planar parameters. Recollect from Section 6.5.1 that the covariance matrix Γ_{abc} of the fitted plane parameters (a, b, c) is given directly by $(A^T\Gamma_Z^{-1}A)$. Thus Γ_h simply becomes equal to $A^T\Gamma_Z A$. Since we assume that the errors in z have zero mean and same variance, $\Gamma_Z = diag(\sigma^2, \sigma^2, \cdots, \sigma^2)$. Therefore $\Gamma_h = \sigma^2 A^T A$. Then $\sigma^2_{d_s}$ can be computed using Eq. (6.21) as before.

We repeated the experiments described in Section 6.6.1 to compute the *actual* variance of d_s using the planar parameters obtained with linear regression. Figure 6.8

shows the plot of the actual standard deviation of the distance d_s versus the orientation of the plane with respect to the vertical axis when linear regression was used. We experimented with planar patches at various orientations. As indicated by our hypothesis, the actual standard deviation of d_s decreases as the planar orientation goes from being horizontal to vertical. As the variance of the added Gaussian noise to the z measurements increases, σ_{d_s} also increases. The figure also shows the results obtained when we added uniform noise to data.

We also estimated the variance of the distance d_s using the simpler formulation described above. Figure 6.9 illustrates the behavior of *estimated* σ_{d_s} with varying planar orientations when linear regression is used to fit a plane. We can observe here too, that the estimated standard deviation of the distance d_s decreases as the planar orientation goes from horizontal to vertical. The plots of estimated variance $\sigma^2_{d_s}$ resemble those of the actual variance, demonstrating the validity of our method of estimating $\sigma^2_{d_s}$ using linear regression.

The two sets of plots shown in Figures 6.6 and 6.8 and Figures 6.7 and 6.9 together illustrate yet another important point. Similar behavior of the *actual* and *estimated* variance in both sets of figures as the planar orientation varies demonstrates the important fact that the actual method used for planar fitting does not bias our results.

6.7 Experimental Results

In this section we demonstrate the improvements in the estimation of transformation parameters using the minimum variance estimator (MVE). Henceforth, we will refer to the technique of Chen and Medioni [38] as C-M method.

6.7.1 Selection of Control Points

We perform uniform subsampling of the depth data to locate the control points in view 1 that are to be used in registration. From these subsampled points we choose only those that are present in smooth surface patches. The local smoothness of the

surface was verified using the value of residual standard deviation resulting from the least-squares fitting of a plane in the neighborhood of a pixel. The algorithm does not rely on the exact correspondence of the control points, but uses the constraints from the geometry or the shape of the surfaces.

6.7.2 Initial Estimate of the Transformation

Even if an initial estimate for the transformation is not available, when the range images contain the entire object surface and the rotations of the object in the views are primarily in the plane, a good initial guess for the iterative algorithm can be determined automatically. It is based on estimating an approximate rotation and translation from the major (principal) axes of the object views. Figure 6.11 depicts the two major axes of the objects. The normalized eigenvectors from view 1 form the column vectors of matrix M_1 and the eigenvectors from view 2 form the matrix M_2. The rotation matrix $R = (\alpha, \gamma, \beta)$ is then given by $R = M_2 M_1^{-1}$. If $t = (t_x, t_y, t_z)$ is the translational vector then a point in view 2, X_i' is related to a point X_i in view 1 by $X_i' = RX_i + t$. Translation vector t is given by $t = \bar{X}' - R\bar{X}$, where $\bar{X}$ and $\bar{X}'$ are the centers of mass of views 1 and 2, respectively. We use this estimated transformation as an initial guess for the iterative procedure in our experiments, since we assume that we do not have the prior knowledge of the sensor placement. This also illustrates the effectiveness of our method in refining such a rough estimate to be close to the ground truth. In all our experiments, the same initial guess was used with both the C-M method and the proposed MVE. We also used Newton's method for minimizing the error function iteratively.

6.7.3 Errors in the Estimated Transformation

In order to measure the error in the estimated rotation parameters, we define an error measure that does not depend on the actual rotation parameters. The relative error of rotation matrix R, E_R is defined to be $E_R = \|\tilde{R} - R\|/\|R\|$, where $\tilde{R}$ is an estimate of R. Since $RI = R$, the geometric sense of E_R is the square root of the

Table 6.1: Estimated transformation for the cobra data.

Parameters	Actual value	Chen and Medioni [38]	New method
α (degrees)	5	0.285706	4.524133
β (degrees)	0	−2.225014	0.014339
γ (degrees)	10	12.97406	10.38106
t_x (inches)	0	0.942902	0.668358
t_y (inches)	0	0.348335	0.021682
t_z (inches)	0	0.226312	−0.297580
E_R		0.084830	0.008691
E_t		1.030348	0.73193

mean squared distance between the three unit vectors of the rotated orthonormal frames. This is illustrated in Figure 6.10. Since the frames are orthonormal, $E_R = \sqrt{(dx^2 + dy^2 + dz^2)}/\sqrt{3}$. The error in translation, E_t is defined as the square root of the sum of the squared differences between the estimated and actual t_x, t_y and t_z values.

6.7.4 Results

Figure 6.11 shows the range data of a cobra head and Big-Y. The figure renders depth as pseudo intensity and points almost vertically oriented are shown darker. View 2 of the cobra head was obtained by rotating the surface by 5° about the X axis and 10° about the Z axis.

Table 6.1 shows the values of E_R and E_t for the cobra images estimated using only as few as 25 control points. It can be seen that the transformation parameters obtained with MVE are closer to the ground truth than those estimated using the unweighted objective function of C-M method. Table 6.2 shows the improved results when more control points were used; even in that case the estimates using our method were closer to the ground truth than those obtained with the C-M method.

We also show the performance of our method when the two views are substantially different and the depth values are very noisy. Figure 6.11 shows two views of Big-

Table 6.2: Registration of cobra data with 156 control points.

Parameters	Actual value	Chen and Medioni [38]	New method
α (degrees)	5	4.297135	4.351065
β (degrees)	0	−0.601008	−0.190772
γ (degrees)	10	10.61468	10.55809
t_x (inches)	0	0.704665	0.686584
t_y (inches)	0	0.060029	−0.020930
t_z (inches)	0	−0.230945	−0.267845
E_R		0.015795	0.012487
E_t		0.743970	0.737276

Table 6.3: Registration of Big-Y views using 81 control points.

Parameters	Actual value	Chen and Medioni [38]	New method
α (degrees)	0	−15.563313	−10.706766
β (degrees)	0	−2.826640	−1.991803
γ (degrees)	45	42.18119	43.29852
t_x (inches)	0	0.040259	0.049479
t_y (inches)	0	0.096469	0.062579
t_z (inches)	0	−0.401167	−0.23081
E_R		0.229121	0.157230
E_t		0.414563	0.244215

Y generated from its CAD model. The second view was generated by rotating the object about the Z axis by 45°. We also added Gaussian noise with mean zero and standard deviation of 0.5 mm to the z values of the surfaces in view 2. Table 6.3 shows E_R and E_t computed with 81 control points. When the number of control points used for registration was increased to 154, the results were better, as shown in Table 6.4. It can be seen from these tables that the proposed MVE method estimates the transformations more accurately in comparison with C-M method in the presence of noise.

The transformation matrix, especially the rotation matrix obtained with the MVE is closer to the ground truth than that obtained using C-M method. The errors

Table 6.4: Registration of Big-Y views using 154 control points.

Parameters	Actual value	Chen and Medioni [38]	New method
α (degrees)	0	1.263640	0.606140
β (degrees)	0	2.015097	1.191350
γ (degrees)	45	44.46735	44.77864
t_x (inches)	0	0.016323	0.000602
t_y (inches)	0	0.131396	0.084499
t_z (inches)	0	0.043766	0.037624
E_R		0.034801	0.019322
E_t		0.139452	0.092498

Table 6.5: Registration of Face1 views with 250 control points.

Parameters	Actual value	Chen and Medioni [38]	New method
α (degrees)	5	5.311695	4.608456
β (degrees)	0	−2.300209	−0.741129
γ (degrees)	0	3.14468	0.7895
t_x (inches)	0	−0.120410	−0.087956
t_y (inches)	0	0.450793	0.366696
t_z (inches)	0	−0.301092	−0.251068
E_R		0.055615	0.016433
E_t		0.555310	0.453031

in translation components in the final estimates of the transformation matrices are mainly due to the approximate initial guess. Our method refined these initial values to provide a final solution very close to the actual values. Our method can also handle large transformations between views robustly.

We show more results on range images of faces. The depth data are noisy owing to the roughness of the face masks used to obtain the range images. Figure 6.12 shows the range data of Face1. View 2 of the face was obtained by rotating the surface by 5° about the X axis. Table 6.5 shows E_R and E_t computed with 250 control points. When convergence was achieved, only 76 control points were used in updating the transformation.

Table 6.6: Registration of Face2 views with 142 control points.

Parameters	Actual value	Chen and Medioni [38]	New method
α (degrees)	5	4.053439	4.669192
β (degrees)	5	5.917358	5.045794
γ (degrees)	5	6.50263	5.08654
t_x (inches)	0	−0.077274	−0.104206
t_y (inches)	0	0.248883	0.317506
t_z (inches)	0	−0.132019	−0.192163
E_R		0.029432	0.005019
E_t		0.292136	0.385481

Figure 6.13 shows the range data of Face2. In this experiment, View 2 was obtained by rotating the face by 5° about X, Y, and Z axes. Table 6.6 shows E_R and E_t computed with 142 control points. When convergence was achieved, only 35 control points were used in updating the transformation.

In our experiments, we found that even when the depth data contained noise owing to the roughness of the surface texture of the object and also due to self-occlusion, more accurate estimates of the transformation were obtained with the MVE. Note that the measurement error is random and we minimize the expected error in the estimated solution. However, the method does not guarantee that every component in the solution will have a smaller error in every single case. Additional results on real range images using the MVE for estimating the object pose of the input with respect to the best matched model view have been presented in Section 5.4.4.

6.8 Surface Geometry and Registration

In this section, we discuss the performance of the weighted and unweighted registration algorithms on surfaces of various geometries. The geometry varies from planar surfaces where the normal is constant everywhere, to elliptical surfaces where the normal changes as we move along the major and minor axes of the surfaces.

Since any registration method that uses estimated normals for its computation

is affected by the noise in z values depending on the orientation of the surfaces, our objective is to study the extent to which the registration accuracy of different surfaces gets affected because of the errors in z. We first define a measure to characterize the average vertical orientation of a surface. A surface may be totally vertical (e.g., a vertically placed half-plane) or may be partially inclined or even fully horizontal. The average vertical orientation V_S of a surface S is defined as the average of the z components of the surface normals n_p computed at each point p on the surface wherever data are available. This measure of the average "vertical orientation" of a surface captures how much a surface is oriented vertically in a three-dimensional space. The orientation measure, V_S, equals 0 if a planar surface is fully vertical as the normals to the surface are parallel to the $x-y$ plane and the z components of the normals are all zero. The value of V_S increases and becomes equal to 1 as the plane goes from being vertical to horizontal.

We expect that a surface that is nearly vertically oriented ($V_S = 0$) will get affected very little by the noise in z. This is due to the fact that surface normals estimated are more or less aligned in the horizontal direction and they are least affected by the noise in z when the surface is vertical. This has already been established by our results in Section 6.5.1. So any registration method that uses surface normals in their computation should estimate the view transformation fairly accurately for vertically oriented surfaces. However, as the surface becomes more horizontally oriented, the errors in surface normal estimation become more pronounced and affect the registration accuracy.

For nearly horizontal planes, the errors in z affect the surface normal computation as established in Section 6.5.1. So the rotational errors are expected to be fairly high near the horizontal planes with both the methods. However, since the MVE compensates for this "orientation effect" by explicitly modeling the uncertainties in z, the relative rotational and translational errors in the estimates computed using the MVE should be much less than those of the C-M method for the same planar orientation. For the noiseless case, the behavior of both the methods should be similar. This orientation effect on registration accuracy is true across several classes

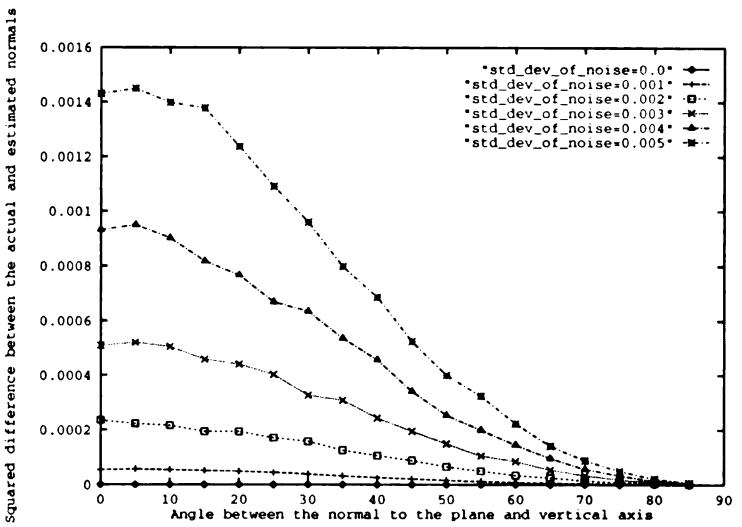
of surfaces irrespective of the surface type (cylinder or plane), based only on the amount of vertical orientation of the surface.

In general, all the orientation parameters of an object will be improved by the proposed MVE method if the object surface covers a wide variety of orientations which is true with many natural objects. This is because each locally flat surface patch constrains the global orientation estimate of the object via its surface normal direction. For example, if the object is a flat surface, then only the global orientation component that corresponds to the surface normal can be improved, but not the other two components that are orthogonal to it. For the same reason, the surface normal of a cylindrical surface (without end surfaces) covers only a great circle of the Gaussian sphere, and thus, only two components of its global orientation can be improved. The more surface orientations that an object covers, the more complete the improvement in its global orientation can be, by the proposed MVE method.

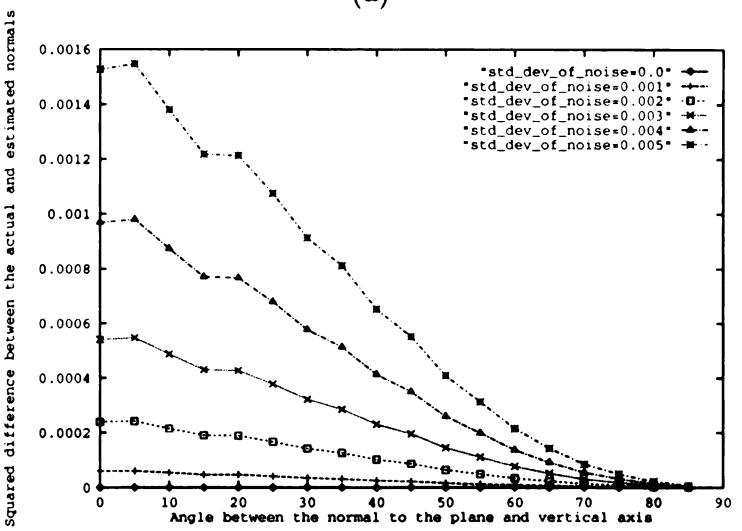
6.9 Summary

We cast the problem of pose estimation as one of registering two range views of an object and we proposed a robust technique for accurate estimation of transformation between the range images. We formulated a noise model that characterizes the effect of error in z measurements upon estimating tangent planes, and we proposed a minimum variance estimator for registration in order to handle uncertainties in z values. Our model handles numerical errors and surface orientation effects. We presented a first-order perturbation analysis of the estimation of planar parameters from surface data. We derived the variance of the point-to-planar distance to be minimized to update the transformation between views. We employed this variance as a measure of uncertainty in the point-to-planar distances resulting from noise in the z values and presented a weighted least squares method to estimate transformation parameters reliably. The results of our experiments on real range images have shown that the pose estimates obtained using our weighted objective function (MVE) generally are significantly more reliable than those computed with an unweighted distance

criterion.

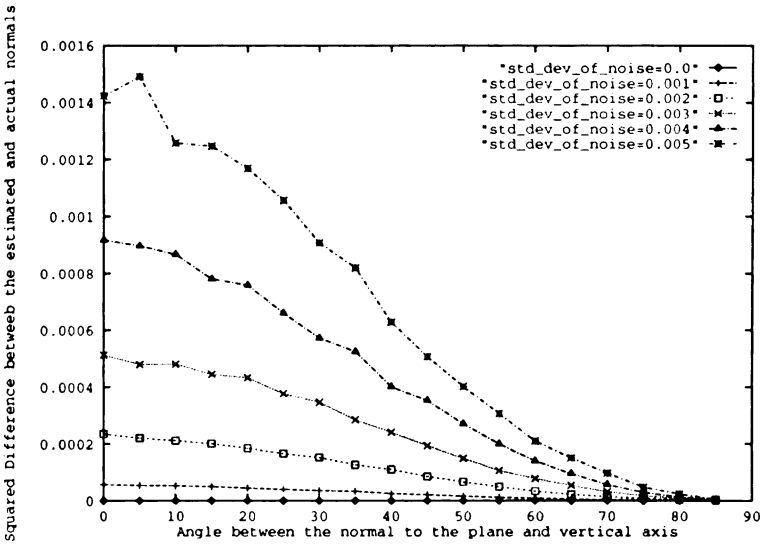


(a)

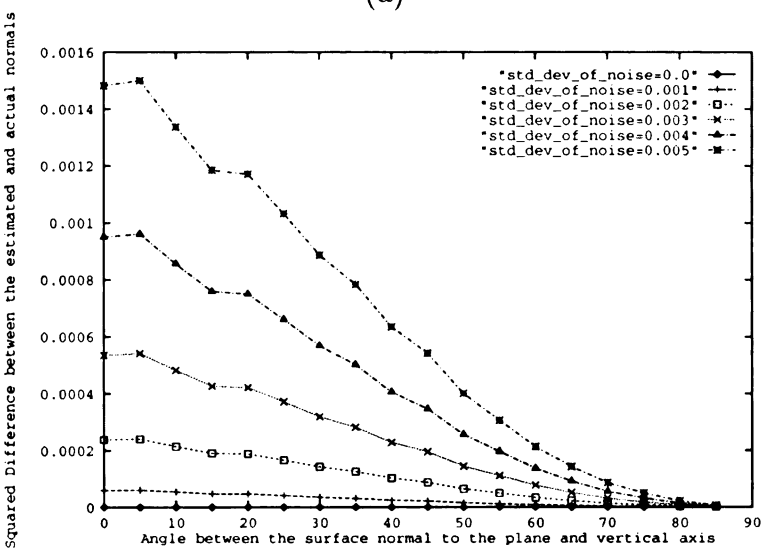


(b)

Figure 6.4: Effect of noise in z measurements on the fitted plane using eigenvector approach: (a) *i.i.d.* Gaussian noise; (b) uniform noise.

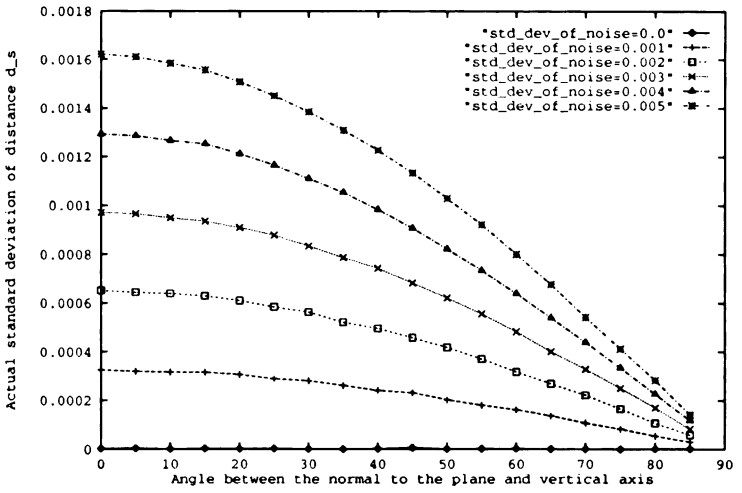


(a)

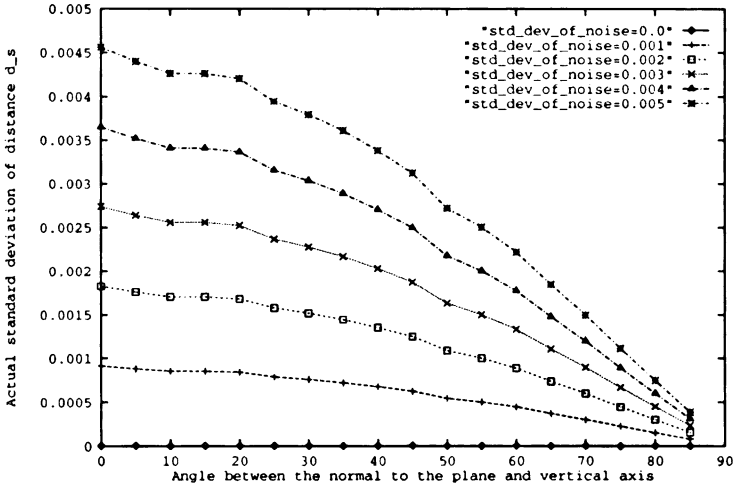


(b)

Figure 6.5: Effect of noise in z measurements on the planar fit using linear regression: (a) *i.i.d.* Gaussian noise; (b) uniform noise.

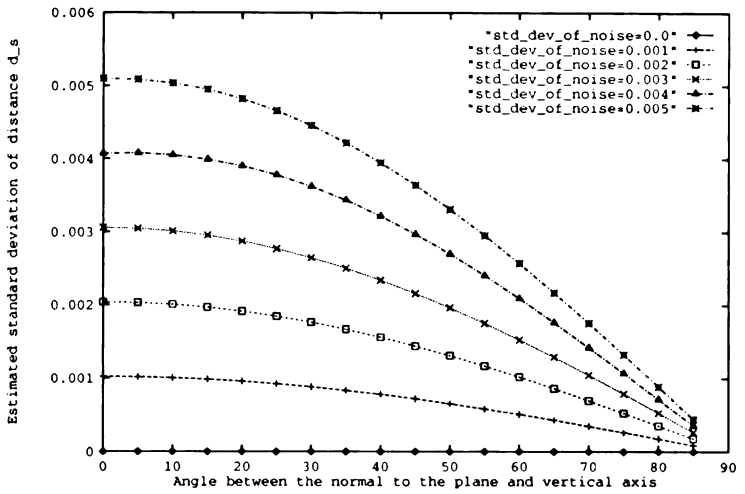


(a)

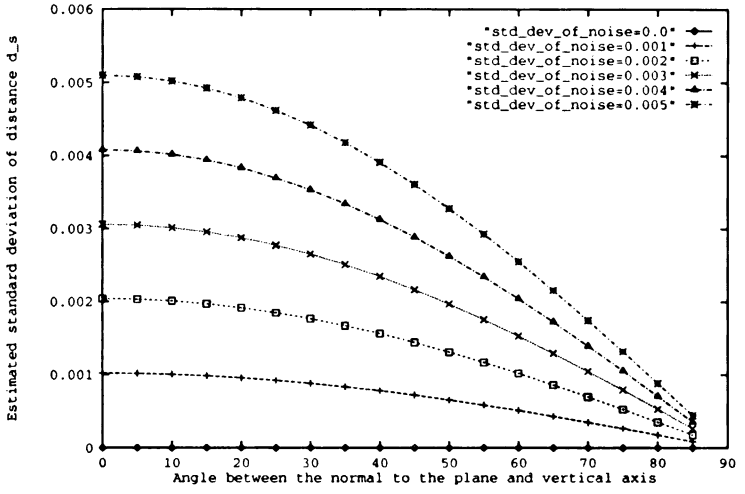


(b)

Figure 6.6: Actual standard deviation of d_s versus the planar orientation: (a) *i.i.d.* Gaussian noise; (b) uniform noise.

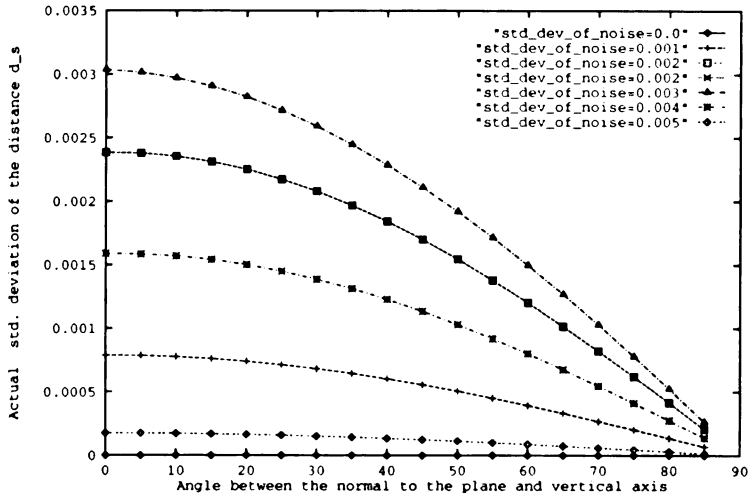


(a)

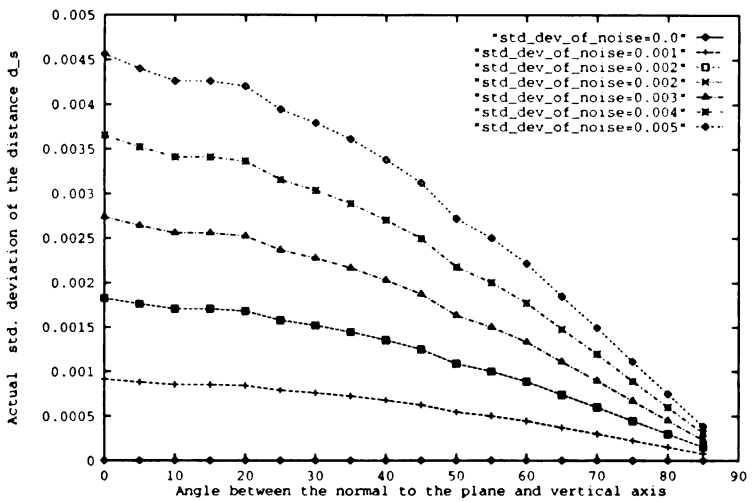


(b)

Figure 6.7: Estimated standard deviation of the distance d_s using the perturbation analysis versus the plane orientation: (a) *i.i.d.* Gaussian noise; (b) uniform noise.

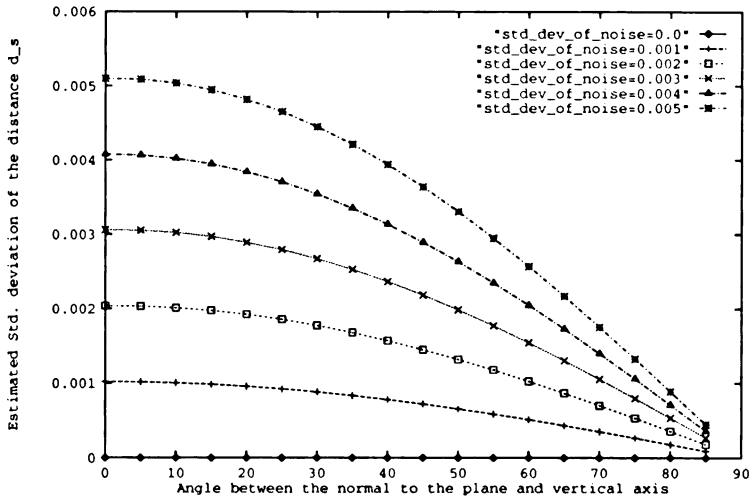


(a)

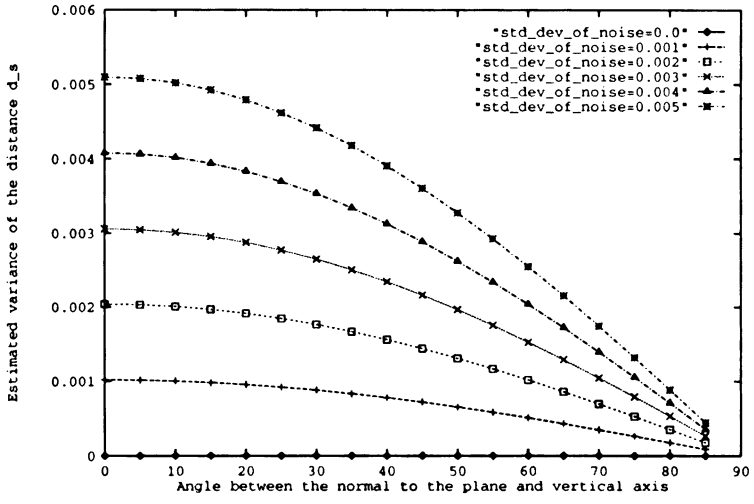


(b)

Figure 6.8: Actual standard deviation of d_s versus planar orientation using linear regression for plane-fitting: (a) *i.i.d.* Gaussian noise; (b) uniform noise.



(a)



(b)

Figure 6.9: Estimated standard deviation of d_s versus planar orientation using linear regression for plane-fitting: (a) *i.i.d.* Gaussian noise; (b) uniform noise.

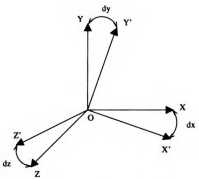


Figure 6.10: Relative error of the rotation matrix R.

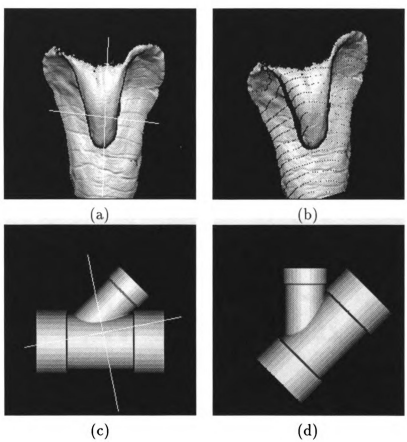
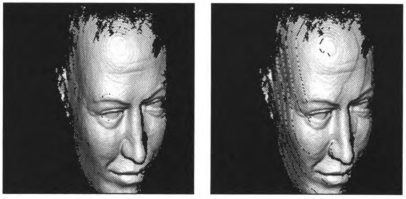
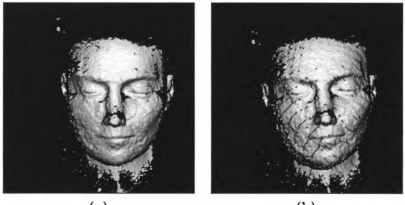


Figure 6.11: Range images and the principal axes: (a) Cobra head with depth rendered as pseudo intensity — view 1; (b) cobra head rotated — view 2; (c) view 1 of Big-Y generated from its CAD model; (d) view 2 of Big-Y.



(a) (b)

Figure 6.12: Range images of Face1: (a) View 1; (b) view 2.



(a) (b)

Figure 6.13: Range images of Face2: (a) View 1; (b) view 2.

Chapter 7

Summary and Directions for Future Research

This chapter summarizes the results of the research reported in this thesis. Several exciting directions for future research are also outlined.

7.1 Summary

The most important contribution of this thesis is the introduction of the COSMOS representation scheme and the associated framework for the representation and recognition of 3D free-form rigid objects. COSMOS addressed the issue of representing and recognizing arbitrarily curved 3D objects using range data when (i) the object viewpoint (2D appearance) is not constrained, (ii) the objects may assume complex shapes and form, and (iii) there is no restriction about the types of surfaces on the object. It is within this framework that the techniques for representing and recognizing arbitrarily shaped 3D objects have been developed.

Given surface depth data of a scene containing multiple nonoccluding objects (in general), the COSMOS-based recognition system derives a shape-based surface representation of each object (region of interest) in the scene. Under the formulation of the COSMOS scheme, an object is described concisely in terms of maximal surface patches

of constant shape index. The maximal patches that represent the object are mapped onto the unit sphere via their orientations. This spherical mapping not only preserves the orientation information about the object, but is also employed to aggregate the local geometric attributes of the CSMPs via shape spectral functions. Geometric attributes such as surface area, curvedness and adjacency which are required to capture local and global information are built into the representation using the connectivity list and the support functions defined on the unit sphere.

The representation scheme has been demonstrated to provide a meaningful and rich description of objects that is useful for recognition of arbitrarily curved objects. We also introduced a powerful matching primitive, the shape spectrum of an object, for fast matching of an input object view with views stored in a model database and for eliminating unlikely views during recognition. It characterizes the shape content of an object by summarizing the area on the surface of the object at each shape index value. We also established that the shape spectrum is the Fourier transform of the integral of the support function G_1 over the entire unit sphere. The concept of shape spectrum is especially appealing as it gives us the ability to construct an intuitive "frequency domain" (shape domain) characterization of spatially varying curvatures. We also discussed the issues of compactness of COSMOS representation of an object. We studied the recoverability of objects from COSMOS representations and we established that recovery of several classes of objects such as convex polyhedra and convex closed surfaces on which the shape index varies continuously at every point is feasible from their COSMOS representations from both theoretical and practical viewpoints.

We adopted a multiple-view based model for an object where the 3D model of the object is a collection of the COSMOS representations of its views seen from different viewpoints. Then we demonstrated how the COSMOS representation can be derived from range data of an object view and illustrated the representation using range images of several complex objects.

Next, we studied the problem of organizing a database of multiple views of a free-form 3D rigid object in a meaningful and efficient manner. This is important because

a complex smooth object can give rise to infinitely many different views owing to its smoothly curved nature. We demonstrated how moment features derived from the shape spectrum of an object view are used to group views of objects of complex shape and geometry into compact and homogeneous clusters. The proposed method is general and easy to use, and offers a practical solution to the construction of view aspects of complex sculpted objects. We demonstrated with experimental results on a database of 6,400 views of 20 objects that view aspects can be determined for sculpted objects easily and effectively.

We also demonstrated that when view-grouping is exploited to structure a large model base of views, even with a relatively flat (two-tiered) arrangement a small set of plausible correct matches to an input object view can be determined quickly. Experimental results on a database of 6,400 views of 20 objects show that when tested with 2,000 independent views, our matching technique examined on the average only 20% of the database for correct classification of the test views.

We proposed and implemented a matching strategy that is a combination of shape spectrum based model database pruning and graph-based search for establishing scene-model feature correspondences, thereby exploiting the advantages of these two techniques for fast and efficient recognition. We tested the generality and the effectiveness of our scheme on a database of 100 range images of several complex objects acquired using a 3D laser range scanner. The shape spectral feature based model selection module yielded a view classification accuracy of 94% over fifty independent test views when the top five out of ten clusters were examined in the database for each input image. The COSMOS-based view verification stage exhibited 82% accuracy in establishing the correct object identity of the test images. Better CSMP detection will aid in increasing this accuracy.

Our approach to pose estimation was that of deriving a robust registration between the input and the matched object views, thereby estimating the transformation between the range images of a free-form object. An initial estimate of the transformation between the sensed and the stored images is determined using the surface normals, once the correspondences between the maximal patches visible in both views have

been established by our recognition strategy. This initial estimate of the pose is then refined to obtain a more accurate set of translation and orientation parameters using an iterative minimization scheme. The registration-based approach thus compensates for the errors that may have been introduced in the initial pose due to poor localization and noise-corrupted surface normals.

The proposed representation, recognition and pose estimation schemes are designed to (i) handle general 3D rigid objects that are arbitrarily shaped, (ii) aid in easy model view selection from a large model database for recognition, and (iii) help in computing the object identity and pose accurately and robustly. The shortcomings of the COSMOS based recognition system are: (i) lack of *explicit* incorporation of edge information within the representation scheme, both in theory and in practical implementation; (ii) an unstable (with respect to changes in viewpoints), data-driven CSMP segmentation technique; (iii) the inability of the shape spectrum based matching scheme to handle model database pruning under occlusion of an input by other objects; and (iv) a graph-matching algorithm for final view verification that is likely to be slow on images that have been segmented into a very large number of CSMPs. It would also be more satisfying to test the representation and recognition schemes exhaustively on a larger set of objects than was possible within the constraints of this thesis. The shortcomings are not critical in the sense that there are clear means of overcoming them via further development as outlined in the next section.

7.2 Future Research

COSMOS is a novel framework within which there are a number of exciting avenues for future research.

7.2.1 Incorporation of Explicit Edge Information within COSMOS

The COSMOS representation currently does not explicitly treat edges (curves of discontinuities of both surface depth and surface normals) on object surfaces. As indicated in Section 3.2.2, an edge can be viewed as the limiting case of a cylindrical surface with infinite curvedness and a corner can be modeled as the limit of a spherical cup or cap shape. On the other hand, traditional edge detection algorithms explicitly represent edges as discontinuities separating homogeneous regions. While edges are likely to be often detected in our system as distinct patches as a byproduct of our segmentation algorithm, we have not made any special effort to detect them as our focus has been on smooth curved objects. However, explicit edge representation may provide additional information when dealing with polyhedra and may also have visual significance (high information content) in interactive/integrated human-computer vision systems. There is a whole body of earlier work [29] on edge detection which can be integrated into COSMOS. Future work can study how best to model edges theoretically and how to use them effectively during shape index based segmentation. One possibility, for example, is to use a traditional edge detection algorithm as a preprocessing step and then use the detected edges as constraints during the region-growing stage in COSMOS.

7.2.2 Improving the Segmentation Algorithm

Since surface depth and normal discontinuities are currently not modeled explicitly within our scheme, we observed that adjacent CSMPs tend to blend with their neighboring regions during the region growing process. An important future direction of research will be to integrate the region-growing segmentation algorithm with edge information to obtain stable (with respect to changes in viewpoints) CSMPs from a range image for effective matching.

In addition, shape index based discontinuity can also be defined and formulated in such a way that distinct shapes of objects do not blend with one another in the

presence of noise in the sensed data.

7.2.3 Deriving COSMOS from a 3D Object Model

We have adopted a "collection of views" approach to modeling a 3D object. Another challenging and related issue is building the COSMOS representation for the entire 3D object from its multiple views which can then serve as an object-centered representation of a free-form object. This problem has several components such as (i) integrating the COSMOS derived from multiple views into a single representation, given the knowledge of the transformations between the multiple views, (ii) valid inferencing about the complete shape, area, curvedness of a maximal patch that is only partially visible in several views, and (iii) accumulating evidence about its attributes from several partial pieces of information present in multiple views and collating them to form a complete and correct representation of the patch.

7.2.4 Better Distance Measures and Matching Efficiency

We informally studied the efficacy of the moment features derived from the shape spectra of object views in classifying an input view correctly and found that only the first four moments significantly contributed to the correct classification of the input. The higher order moments were low in magnitude and did not add much to the Euclidean distances computed for comparison of moment vectors.

Alternative metrics, especially the Mahalanobis distance, could be used instead to compare the feature vectors and measure the similarity of views. The Mahalanobis distance ensures "equal" contributions of individual feature values to the distance computed between views. A thorough comparison can be made between these two distance measures to determine the utility of the high-order moments in the feature vectors derived from the shape spectra of views.

A potential future contribution is to to add more levels to the hierarchical database structure. For example, a set of object views can be organized based on their shape spectra into several categories: those that exhibit planar patches alone and those that

exhibit other shapes in addition to planar patches. Given this broad organization, a fine-grain organization of the latter category into those views that contain purely nonconvex shapes and those that contain purely convex shapes can also be obtained. Given an input object view, its shape spectrum can be computed easily, and then, by descending through this hierarchy, can be compared with only a small subset of views that are likely to best match with it.

7.2.5 Occlusion

Currently, shape spectrum based pruning assumes that the shape spectrum computed from a view belongs to only a single region of interest and performs the classification of that object view during matching. However, when objects overlap one another in the scene, if we cannot separate them into distinct regions, the shape spectrum computed would be that of the entire overlapping surface area in the image. Since adjacency information is not part of the shape spectrum, there are no means of distinguishing whether the spectrum computed is from a single region or from multiple regions in the image. In cases of occlusion, this spectrum-based pruning step can be avoided, and the graph-based search can directly be used to establish patch-group graph isomorphism and thus determine the object identity. Future study should investigate the use of discontinuities in separating the regions of interest, and then computing the shape spectrum of each of these regions sequentially and performing the matching; this is related to our suggestion in Section 7.2.1.

Further, geometric reasoning based on model features has to be employed once a representation of an input image has been derived in order to detect that some features are missing in the sensed data and that this could be due to the object being occluded by others. Occlusion events have to be determined by exploiting the fact that adjacent features in the model must match adjacent features in the image except when occlusion occurs or when there are errors in segmenting the regions of interest. Independent evidence for occlusion can be determined by detecting the loss of support for the model features.

7.2.6 Integrating Color and Texture

Our discussion so far has dealt with the design of a geometry-based representation and recognition system for free-form objects. However, we believe that a robust recognition system benefits from using other cues such as color and texture. Therefore, a very interesting line of future work would be to investigate how color and texture features can be incorporated within the COSMOS representation of an object and to explore the possibility of enhancing our matching scheme using features derived from these additional cues as indexing primitives.

BIBLIOGRAPHY

Bibliography

[1] N. Ahuja and J. Veenstra. Generating octrees from object silhouettes in orthographic views. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(2):137–149, 1989.

[2] Farshid Arman and J. K. Aggarwal. Automatic Generation of Recognition Strategies Using CAD Models. In *Proc. IEEE Workshop on Directions in Automated CAD-Based Vision*, pages 124–133, Maui, Hawaii, 1991.

[3] Farshid Arman and J. K. Aggarwal. Model-based object recognition in dense range images–A review. *Computing Surveys*, 25(1):5–43, 1993.

[4] F. Attneave. Some informational aspects of perception. *Psychological Review*, 61(3):183–193, 1954.

[5] R. Bajcsy and F. Solina. Three-dimensional object representation revisited. In *Proc. First IEEE International Conference on Computer Vision*, pages 231–240, London, 1987.

[6] A. H. Barr. Superquadrics and angle-preserving transformations. *Computer Graphics and Applications*, 1:11–23, 1981.

[7] H. G. Barrow and R. M. Burstall. Subgraph isomorphism, matching relational structures and maximal cliques. *Information Processing Letters*, 4(4):83–84, January 1976.

[8] R. Basri and S. Ullman. The alignment of objects with smooth surfaces. In *Proc. Second IEEE International Conference on Computer Vision*, pages 482–488, Tarpon Springs, FL, 1988.

[9] Ronen Basri. *Viewer-centered representations in object recognition: A computational approach*, chapter 5.4, pages 863–882. Handbook of Pattern Recognition & Computer Vision. World Scientific Publishing Company, 1993.

[10] Paul J. Besl. *Surfaces in Range Image Understanding.* Springer Series in Perception Engineering. Springer-Verlag, 1988.

[11] Paul J. Besl. The free-form surface matching problem. In Herbert Freeman, editor, *Machine vision for three-dimensional scenes*, pages 25–71. Academic Press, 1990.

[12] Paul J. Besl. Geometric signal processing. In Ramesh C. Jain and Anil K. Jain, editors, *Analysis and Interpretation of range images*, chapter 3, pages 141–205. Springer-Verlag, 1990.

[13] Paul J. Besl. Triangles as a primary representation. In Martial Hebert, Jean Ponce, Terry Boult, and Ari Gross, editors, *Object representation in computer vision*, pages 191–206. Springer-Verlag, Berlin, 1995.

[14] Paul J. Besl and R.C. Jain. Three-dimensional object recognition. *Computing Surveys*, 17:75–145, 1985.

[15] Paul J. Besl and Neil D. Mckay. A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

[16] B. Bhanu. Representation and shape matching of 3-D objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(3):340–351, May 1984.

[17] B. Bhanu and T. Poggio. Introduction to the special section on learning in computer vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(9):865–868, September 1994.

[18] Irving Biederman. Recognition-by-components: A theory of human image understanding. *Psychological Review*, 94(2):115–147, 1987.

[19] G. Blais and M. D. Levine. Registering multiview range data to create 3d computer graphics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):820–824, 1995.

[20] A. F. Bobick and R. C. Bolles. The representation space paradigm of concurrent evolving object descriptions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):146–156, 1992.

[21] R. C. Bolles and R. A. Cain. Recognizing and locating partially visible objects: The local feature focus method. *International Journal of Robotics Research*, 1(3):57–82, 1982.

[22] R.C. Bolles and P. Horaud. 3DPO: A three-dimensional part orientation system. *International Journal of Robotics Research*, 5(3):3–26, 1986.

[23] Thomas M. Breuel. Adaptive model base indexing. In *Proc. DARPA Image Understanding Workshop*, pages 805–814, Palo Alto, California, 1989.

[24] Rodney A. Brooks. Model-based three-dimensional interpretations of two-dimensional images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, PAMI-5(2):140–150, March 1983.

[25] Rodney A. Brooks. *Model-Based Computer Vision.* UMI Research Press, 1984.

[26] R. Brunelli and T. Poggio. Face recognition: Features versus templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(10):1042–1052, 1993.

[27] Terry Caelli and Ashley Dreier. Variations on the evidence-based object recognition theme. *Pattern Recognition*, 27(2):185–204, 1994.

[28] Octavia I. Camps, Linda G. Shapiro, and Robert M. Haralick. PREMIO: An Overview. In *Proc. IEEE Workshop on Directions in Automated CAD-Based Vision*, pages 11–21, Maui, Hawaii, June 1991.

[29] John Canny. A computation approach to edge detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, 1986.

[30] C. Chen and A. Kak. A robot vision system for recognizing 3-D objects in low-order polynomial time. *IEEE Transactions on Systems, Man, and Cybernetics*, pages 1535–1563, 1988.

[31] Chien-Huei Chen and Prasanna G. Mulgaonkar. CAD-based feature-utility measures for automated vision programming. In *Proc. IEEE Workshop on Directions in Automated CAD-Based Vision*, pages 106–114, Maui, Hawaii, 1991.

[32] J. L. Chen and G. Stockman. Indexing to 3D model aspects using 2D contour features. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, San Francisco, CA, 1996.

[33] Jin-Long Chen and G. C. Stockman. Determining pose of 3D objects with curved surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(1):52–56, January 1996.

[34] S. Chen and H. Freeman. Computing characteristic views of quadric-surfaced solids. In *Proceedings of the 10th ICPR*, pages 77–82, Atlantic City, N. J., 1990.

[35] Sei-Wang Chen and Anil K. Jain. Strategies of multiview and multi-matching for 3D object recognition. *Computer Vision, Graphics and Image Processing*, 57(1):121–130, January 1993.

[36] Sei-Wang Chen and George Stockman. Wing representation for rigid 3D objects. In *Proc. 10th International Conference on Pattern Recognition*, pages 398–402, Atlantic City, 1990.

[37] Tsu-Wang Chen and Wei-Chung Lin. A neural network approach to CSG-based 3-D object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(7):719–726, 1994.

[38] Yang Chen and Gérard Medioni. Object modelling by registration of multiple range images. *Image and Vision Computing*, 10(3):145–155, April 1992.

[39] H. Chernof. Using faces to represent points in k-dimensional space graphically. *Journal of the American Statistical Association*, 68:361–368, 1973.

[40] Jonathan H. Connell and Michael Brady. Generating and Generalizing Models of Visual Objects. *Artificial Intelligence*, 31:159–183, 1987.

[41] H. Delingette, M. Hebert, and K. Ikeuchi. Shape representation and image segmentation using deformable surfaces. *Image and Vision Computing*, 10(3):132–144, 1992.

[42] H. Delingette, M. Hebert, and K. Ikeuchi. A spherical representation for the recognition of curved objects. In *Proc. Fourth IEEE International Conference on Computer Vision*, pages 103–112, Berlin, 1993.

[43] Sven J. Dickinson, A. P. Pentland, and Azriel Rosenfeld. From volumes to views: An approach to 3-D object recognition. In *Proc. IEEE Workshop on Directions in Automated CAD-Based Vision*, pages 85–96, Maui, Hawaii, 1991.

[44] Sven J. Dickinson, Alex P. Pentland, and Azriel Rosenfeld. 3-D shape recovery using distributed aspect matching. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 14(2):174–198, February 1992.

[45] Chitra Dorai and Anil K. Jain. COSMOS—A representation scheme for free-form surfaces. In *Proc. Fifth International Conference on Computer Vision*, pages 1024–1029, Boston, Massachusetts, June 1995.

[46] Chitra Dorai and Anil K. Jain. Shape spectra based view grouping for free-form objects. In *Proc. IEEE International Conference on Image Processing*, volume III, pages 340–343, Washington, D.C., October 1995.

[47] Chitra Dorai and Anil K. Jain. View organization and matching of free-form objects. In *Proc. IEEE International Symposium on Computer Vision*, pages 25–30, Coral Gables, Florida, November 1995.

[48] Chitra Dorai and Anil K. Jain. Recognition of 3D free-form objects. In *Proc. 13th International Conference on Pattern Recognition*, Vienna, Austria, August 1996, To appear.

[49] Chitra Dorai, Gang Wang, Anil K. Jain, and Carolyn Mercer. From images to models: Automatic 3D object model construction from multiple views. In *Proc. 13th International Conference on Pattern Recognition*, Vienna, Austria, August 1996, To appear.

[50] Chitra Dorai, John Weng, and Anil K. Jain. Optimal registration of multiple range views. In *Proc. 12th International Conference on Pattern Recognition*, pages 569–571, Jerusalem, Israel, October 1994.

[51] Sahibsingh A. Dudani, Kenneth J. Breeding, and Robert B. McGhee. Aircraft identification by moment invariants. *IEEE Transactions on Computers*, C-26(1):39–46, 1977.

[52] D. S. Weld (Ed.). The role of intelligent systems in the National Information Infrastructure. *AI Magazine*, 16(3):45–64, Fall 1995.

[53] S. Edelman and D. Weinshall. A self-organizing multiple-view representation of 3d objects. *Biological Cybernetics*, 64:209–219, 1991.

[54] H. Edelsbrunner, J. O'Rourke, and R. Seidel. Constructing arrangements of lines and hyperplanes with applications. *SIAM Journal on Computing*, 15:341–363, 1986.

[55] D. Eggert and K. Bowyer. Computing the orthographic projection aspect graph of solids of revolution. In *Proc. IEEE Workshop on Interpretation of 3D Scenes*, pages 102–108, Austin, November 1989.

[56] D. Eggert and K. Bowyer. Computing the perspective projection aspect graph of solids of revolution. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(2):109–128, February 1993.

[57] Executive Office of the President, Office of Science and Technology Policy. The Federal High Performance Computing Program. Washington, D.C., September 1989.

[58] Ting-Jun Fan, Gérard Medioni, and Ramakant Nevatia. Recognizing 3-D objects using surface descriptions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(11):1140–1157, 1989.

[59] O.D. Faugeras and M. Hebert. The representation, recognition, and locating of 3-D objects. *International Journal of Robotics Research*, 5(3):27–52, 1986.

[60] W. Feller. *An Introduction to Probability Theory and its Applications*, volume II. John Wiley & Sons, Inc., New York, 2 edition, 1971.

[61] F. P. Ferrie, J. Lagarde, and P. Whaite. Darboux frames, snakes, and superquadrics: Geometry from the bottom-up. In *IEEE Workshop on Interpretation of 3-D scenes*, pages 170–176, Austin, Texas, 1989.

[62] F. P. Ferrie and M. D. Levine. Integrating information from multiple views. In *IEEE Workshop on Computer Vision*, pages 117–122, Miami Beach, FL, 1987.

[63] F. P. Ferrie, S. Mathur, and G. Soucy. Feature extraction for 3-D model building and object recognition. In Anil K. Jain and Patrick J. Flynn, editors,

Three-Dimensional Object Recognition Systems, pages 57–88. Elsevier Science Publishers B.V., Amsterdam, The Netherlands, 1993.

[64] M. Fischler and R. Bolles. Random consensus: A paradigm for model-fitting with applications in image analysis and automated cartography. *Communications of the ACM*, 24:381–395, 1981.

[65] P. J. Flynn and A. K. Jain. Surface classification: Hypothesis testing and parameter estimation. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 261–267, Ann Arbor, Michigan, 1988.

[66] Patrick J. Flynn. *CAD-based computer vision: Modeling and recognition strategies.* PhD thesis, Michigan State University, Department of Computer Science, East Lansing, Michigan, 1990.

[67] Patrick J. Flynn. Saliencies and symmetries: Towards 3D object recognition from large model databases. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 322–327, Urbana, IL, 1992.

[68] Patrick J. Flynn and A. K. Jain. BONSAI: 3D object recognition using constrained search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(10):1066–1075, 1991.

[69] Patrick J. Flynn and A. K. Jain. 3D object recognition using invariant feature indexing of interpretation tables. *CVGIP: Image Understanding*, 55(2):119–129, 1992.

[70] Patrick J. Flynn and Anil K. Jain. Three-Dimensional object recognition. In Tzay Y. Young, editor, *Handbook of Pattern Recognition and Image Processing*, volume 2, chapter 14, pages 497–541. Academic Press, 1994.

[71] D. A. Forsyth, J. L. Mundy, A. Zisserman, C. Coelho, A. Heller, and C. Rothwell. Invariant descriptors for 3-D object recognition and pose. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14:971–992, 1992.

[72] H. Freeman and I. Chakravarty. The use of characteristic views in the recognition of three-dimensional objects. In E. Gelsema and L. Kanal, editors, *Pattern Recognition in Practice.* North-Holland Publishing Co., Amsterdam, 1980.

[73] K. Fukunaga. *Introduction to Statistical Pattern Recognition.* Academic Press, Boston, 1990.

[74] Z. Gigus, J. Canny, and R. Seidel. Efficiently computing and representing aspect graphs of polyhedral objects. In *Proc. Second IEEE International Conference on Computer Vision*, pages 30–39, Tarpon Springs, 1988.

[75] Z. Gigus and J. Malik. Computing the aspect graph for line drawings of polyhedral objects. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 654–661, Ann Arbor, 1988.

[76] W. E. L. Grimson and D. P. Huttenlocher. On the sensitivity of the Hough transform for object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(3):255–274, 1990.

[77] W. E. L. Grimson and D. P. Huttenlocher. On the verification of hypothesized matches in model-based recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(12):1201–1213, 1991.

[78] W. E. L. Grimson and T. Lozano-Pérez. Localizing overlapping parts by searching the interpretation tree. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(4):469–482, July 1987.

[79] W. Eric L. Grimson. The combinatorics of heuristic search termination for object recognition in cluttered environments. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(9):920–935, September 1991.

[80] W. Eric L. Grimson and Tomás Lozano-Pérez. Model-based recognition and localization from sparse range or tactile data. *International Journal of Robotics Research*, 3(3):3–35, Fall 1984.

[81] A. Gupta, L. Bogoni, and R. Bajcsy. Quantitative and qualitative measures for the evaluation of the superquadric models. In *Proc. IEEE Workshop on Interpretation of 3D Scenes*, pages 162–169, Austin, 1989.

[82] Alok Gupta and Ruzena Bajcsy. Surface and volumetric segmentation of range images using biquadrics and superquadrics. In *Proc. 11th International Conference on Pattern Recognition*, pages 158–162, The Hague, The Netherlands, 1992.

[83] C. Hansen and T. Henderson. CAGD-based computer vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(11):1181–1193, November 1989.

[84] Patrick Hebert, Denis Laurendeau, and Denis Pousart. Scene reconstruction and description: Geometric primitive extraction from multiple view scattered data. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 286–292, New York City, NY, 1993.

[85] B. K. P. Horn. Extended Gaussian image. *Proceedings of the IEEE*, 72:1671–1686, 1984.

[86] B. Horowitz and A. P. Pentland. Recovery of non-rigid motion and structure. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 288–293, Maui, Hawaii, 1991.

[87] Daniel P. Huttenlocher and Shimon Ullman. Recognizing solid objects by alignment with an image. *International Journal of Computer Vision*, 5(2):195–212, 1990.

[88] K. Ikeuchi. Generating an interpretation tree from a CAD model for 3D-object recognition in bin-picking tasks. *International Journal of Computer Vision*, 1:145–165, 1987.

[89] Katsushi Ikeuchi and Ki Sang Hong. Determining linear shape change: Toward automatic generation of object recognition programs. *Computer Vision, Graphics and Image Processing*, 53(2):154–170, 1991.

[90] Katsushi Ikeuchi and Takeo Kanade. Automatic Generation of Object Recognition Programs. *Proc. IEEE*, 76(8):1016–1035, August 1988.

[91] A. K. Jain and P. J. Flynn (Eds.). *3D Object Recognition Systems.* Elsevier Science Publishers B.V., Amsterdam, The Netherlands, 1993.

[92] Anil K. Jain and Richard C. Dubes. *Algorithms for Clustering Data.* Prentice Hall, Englewood Cliffs, NJ, 1988.

[93] Anil K. Jain and Richard L. Hoffman. Evidence-based recognition of 3-D objects. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 10(6):783–802, November 1988.

[94] Ray Jarvis. Range sensing for computer vision. In Anil K. Jain and Patrick J. Flynn, editors, *Three-dimensional Object Recognition Systems*, pages 17–56. Elsevier Science Publishers B.V., Amsterdam, The Netherlands, 1993.

[95] T. Joshi, J. Ponce, B. Vijayakumar, and D. J Kriegman. HOT curves for modelling and recognition of smooth curved 3D objects. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 876–880, Seattle, Washington, June 1994.

[96] S. B. Kang and K. Ikeuchi. The complex EGI: A new representation for 3-D pose determination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(7):707–721, 1993.

[97] Daniel Keren, David Cooper, and Jayashree Subrahmonia. Describing complicated objects by implicit polynomials. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16:38–53, 1994.

[98] Whoi-Yul Kim and Avinash C. Kak. 3-D object recognition using bipartite matching embedded in discrete relaxation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(3):224–251, 1991.

[99] J. J. Koenderink and A. J. van Doorn. Internal representation of solid shape with respect to vision. *Biological Cybernetics*, 32(4):211–216, 1979.

[100] J. J. Koenderink and A. J. van Doorn. The internal representation of solid shape with respect to vision. *Biol. Cybern.*, 32:211–216, 1979.

[101] J. J. Koenderink and A. J. van Doorn. Surface shape and curvature scales. *Image and Vision Computing*, 10(8):557–565, October 1992.

[102] Jan. J. Koenderink. *Solid Shape.* The MIT Press, 1990.

[103] D. J. Kriegman and J. Ponce. Computing exact aspect graphs of curved objects: Solids of revolution. In *Proc. IEEE Workshop on Interpretation of 3D Scenes*, pages 116–122, Austin, 1989.

[104] R. Krishnapuram and D. Casasent. Determination of three-dimensional object location and orientation from range images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(11):1158–1167, 1989.

[105] Y. Lamdan and H.J. Wolfson. Geometric hashing: A general and efficient model-based recognition scheme. In *Proc. Second IEEE International Conference on Computer Vision*, pages 238–249, Tarpon Springs, Florida, December 1988.

[106] Yehezkel Lamdan, Jacob T. Schwartz, and Haim J. Wolfson. Affine invariant model-based object recognition. *IEEE Trans. on Robotics and Automation*, 6(5):578–589, 1990.

[107] Stephane Lavalle and Richard Szeliski. Recovering the position and orientation of free-form objects from image contours using 3D distance maps. *IEEE Trans-*

actions on Pattern Analysis and Machine Intelligence, 17(4):378–390, April 1995.

[108] Ying Li and R. J. Woodham. Orientation-based representations of 3-D shape. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 182–187, Seattle, Washington, June 1994.

[109] P. Liang and C. H. Taubes. Orientation-based differential geometric representations for computer vision applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(3):249–258, 1994.

[110] P. Liang and J. Todhunter. Representation and recognition of surface shapes in range images: A differential geometry approach. *Computer Vision, Graphics and Image Processing*, 52:78–109, 1990.

[111] Chia-Wei Liao and Gérard Medioni. Representation of range data with B-spline suface patches. In *Proc. 11th International Conference on Pattern Recognition*, volume 3, pages 745–748, The Hague, The Netherlands, 1992.

[112] David G. Lowe. Three-dimensional object recognition from single two-dimensional images. *Artificial Intelligence*, 31:355–395, 1987.

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

[114] D. Marr and H. K. Nishihara. Representation and recognition of the spatial organization of three-dimensional shapes. In *Proc. Royal Society, London, ser. B*, volume 200, pages 269–294, 1978.

[115] Hiroshi Matsuo and Akira Iwata. 3-D object recognition using MEGI model from range data. In *Proc. 12th International Conference on Pattern Recognition*, pages 843–846, Jerusalem, Israel, October 1994.

[116] Richard S. Millman and George D. Parker. *Elements of Differential Geometry.* Prentice-Hall, 1977.

[117] Hiroshi Murase and Shree K. Nayar. Visual learning and recognition of 3-D objects from appearance. *International Journal of Computer Vision*, 14(1):5–24, 1995.

[118] V. S. Nalwa. Representing oriented piecewise C^2 surfaces. *International Journal of Computer Vision*, 3:131–153, 1989.

[119] Shree K. Nayar, Hiroshi Murase, and Sameer A. Nene. Learning, positioning and tracking visual appearance. In *Proc. IEEE Conference on Robotics and Automation*, volume 4, pages 3237–3244, San Diego, California, 1994.

[120] Timothy S. Newman. *Experiments in 3D CAD-based Inpection using Range Images.* PhD thesis, Michigan State University, Department of Computer Science, 1993.

[121] E. Oja. *Subspace Methods of Pattern Recognition.* Research Studies Press, Hertfordshire, 1983.

[122] Masaki Oshima and Yoshiaki Shirai. Object recognition using three-dimensional information. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, PAMI-5(4):353–361, 1983.

[123] A. Pentland and S. Sclaroff. Closed-form solutions for physically based shape modeling and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:715–729, July 1991.

[124] A. P. Pentland. Perceptual organization and the representation of natural form. *Artificial Intelligence*, 28:293–331, May 1986.

[125] A. P. Pentland. Automatic extraction of deformable part models. *International Journal of Computer Vision*, 4:107–126, 1990.

[126] W. H. Plantinga and C. R. Dyer. Visibility, occlusion, and the aspect graph. *International Journal of Computer Vision*, 5:137–160, 1990.

[127] T. Poggio and S. Edelman. A network that learns to recognize three-dimensional objects. *Nature*, 343:263–266, 1990.

[128] A. V. Pogorelov. *Extrinsic Geometry of Convex Surfaces*, volume 35 of *Translations of mathematical monographs.* American Mathematical Society, Providence, R. I., 1973.

[129] J. Ponce, A. Hoogs, and D. J. Kreigman. On using CAD models to compute the pose of curved 3D objects. *CVGIP: Image Understanding*, 55(2):184–197, 1992.

[130] J. Ponce, D. J. Kriegman, S. Petitjean, S. Sullivan, G. Taubin, and B. Vijayakumar. Representations and algorithms for 3D curved object recognition. In Anil K. Jain and Patrick J. Flynn, editors, *Three-Dimensional Object Recognition Systems*, pages 17–56. Elsevier Science Publishers B.V., Amsterdam, The Netherlands, 1993.

[131] M. Potmesil. Generating models for solid objects by matching 3D surface segments. In *Proc. International Joint Conference on Artificial Intelligence*, pages 1089–1093, Karlsruhe, Germany, 1983.

[132] N. S. Raja and A. K. Jain. Recognizing geons from superquadrics fitted to range data. *Image and Vision Computing*, 10(3):179–190, 1992.

[133] N. S. Raja and A. K. Jain. Obtaining generic parts from range images using a multi-view representation. *Computer Vision, Graphics and Image Processing*, 60(1):44–64, July 1994.

[134] A. A. G. Requicha. Representations for rigid solids: Theory, methods, and systems. *Computing Surveys*, 1980.

[135] L.G. Roberts. Machine perception of three-dimensional solids. In James T. Tippett, David A. Berkowitz, Lewis C. Clapp, Charles J. Koester, and Jr. Alexander Vanderburgh, editors, *Optical and Electro-Optical Information Processing*, pages 159–197. MIT Press, Cambridge, Massachusetts, 1965.

[136] Hanan Samet. *The Design and Analysis of Spatial Data Structures.* Addison-Wesley, 1990.

[137] Steven R. Schwartz and Benjamin W. Wah. Machine learning of computer vision algorithms. In Tzay Y. Young, editor, *Handbook of Pattern Recognition and Image Processing: Computer Vision*, volume 2, chapter 14, pages 319–359. Academic Press, 1994.

[138] W. B. Seales and C. R. Dyer. Viewpoint from occluding contour. *Computer Vision, Graphics and Image Processing*, 55(2):198–211, 1992.

[139] M. Seibert and A M. Waxman. Adaptive 3-D object recognition from multiple views. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):107–123, February 1992.

[140] K. Sengupta and K. L. Boyer. Organizing large structural modelbases. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(4):321–332, 1995.

[141] L. G. Shapiro and H. Lu. The use of a relational pyramid representation for view classes in CAD-to-Vision system. In *Proc. 9th International Conference on Pattern Recognition*, pages 379–381, Rome, 1988.

[142] T. M. Silberberg, L. Davis, and H. Harwood. An iterative Hough procedure for three-dimensional object reocgnition. *Pattern Recognition*, 17(6):621–629, 1984.

[143] Sarvajit S. Sinha and Ramesh Jain. Range image analysis. In Tzay Y. Young, editor, *Handbook of Pattern Recognition and Image Processing: Computer Vision*, volume 2, chapter 14, pages 185–237. Academic Press, 1994.

[144] F. Solina and R. Bajcsy. Recovery of parametric models from range images: The case for superquadrics with global deformations. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 12(2):131–147, February 1990.

[145] T. Sripradisvarakul and R. Jain. Generating aspect graphs for curved objects. In *Proc. IEEE Workshop on Interpretation of 3D Scenes*, pages 109–115, Austin, 1989.

[146] L. Stark and K. W. Bowyer. Achieving generalized object recognition through reasoning about association of function to structure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:1097–1104, 1991.

[147] Fridtjof Stein and Gérard Medioni. Structural indexing: Efficient 3-D object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):125–145, 1992.

[148] J. Stewman and K. Bowyer. Creating the perspective projection aspect graph of polyhedral objects. In *Proc. Second IEEE International Conference on Computer Vision*, pages 494–500, Tarpon Springs, 1988.

[149] J. H. Stewman and K. W. Bowyer. Aspect graphs for convex planar-face objects. In *Proc. IEEE Workshop on Computer Vision*, pages 123–130, Miami Beach, 1987.

[150] George C. Stockman. Object recognition and localization via pose clustering. *Computer Vision, Graphics, and Image Processing*, 40:361–387, 1987.

[151] Robert S. Strichartz. *A Guide to Distribution Theory and Fourier Transforms.* CRC Press, Boca Raton, 1994.

[152] P. Suetens, P. Fua, and A. J. Hanson. Computational strategies for object recognition. *ACM Computing Surveys*, 24(1):5–61, March 1992.

[153] Daniel L. Swets. *The self-organizing hierarchical optimal subspace learning and inference framework for object recognition.* PhD thesis, Michigan State University, Department of Computer Science, East Lansing, Michigan, 1996.

[154] M. Tar and S. Pinker. Mental rotation and orientation-dependence in shape recognition. *Cognitive Psychology*, 21:233–282, 1989.

[155] G. Taubin. Estimation of planar curves, surfaces, and nonplanar space curves defined by implicit equations with applications to edge and range image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(11):1115–1138, Nov. 1991.

[156] G. Taubin, F. Cukierman, S. Sullivan, J. Ponce, and D. J. Kreigman. Parametrized and fitting bounded algebraic curves and surfaces. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 103–108, Champaign, Illinois, 1992.

[157] D. Terzopoulos and D. Metaxas. Dynamic 3D models with local and global deformations: Deformable superquadrics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:703–714, 1991.

[158] Demetri Terzopoulos, Andrew Witkin, and Michael Kass. Constraints on deformable models: Recovering 3D shape and nonrigid motion. *Artificial Intelligence*, 36(1):91–123, 1988.

[159] Greg Turk and Marc Levoy. Zippered polygon meshes from range images. In *SIGGRAPH 94*, pages 311–318, 1994.

[160] M. Turk and A. P. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

[161] Jerry L. Turney, Trevor N. Mudge, and Richard A. Volz. Recognizing Partially Occluded Parts. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, PAMI-7(4):410–421, 1985.

[162] S. Ullman and R. Basri. Recognition by linear combination of models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(10):992–1006, October 1991.

[163] Shinji Umeyama. Parameterized point pattern matching and its application to recognition of object families. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(2):136–144, 1993.

[164] A.J. Vayda and A.C. Kak. A robot vision systems for recognition of generic shaped objects. *Computer Vision, Graphics, and Image Processing: Image Understanding*, 54(1):1–46, July 1991.

[165] B. C. Vemuri and J. K. Aggarwal. 3-D model construction from multiple views using range and intensity data. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 435–437, Miami Beach, FL, 1986.

[166] B.C. Vemuri and J.K. Aggarwal. Representation and recognition of objects from dense range maps. *IEEE Transactions on Circuits and Systems*, CAS-34(11):1351–1363, November 1987.

[167] B.C. Vemuri, A. Mitiche, and J.K. Aggarwal. Curvature-based representation of objects from range data. *Image and Vision Computing*, 4(2):107–114, May 1986.

[168] Wu Wang and S. S. Iyengar. Efficient data structures for model-based 3-D object recognition and localization from range data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(10):1035–1045, October 1992.

[169] N. A. Watts. Calculating the principal views of a polyhedron. In *Proceedings of the 9th ICPR*, pages 316–322, Rome, 1988.

[170] J .Weng. On comprehensive visual learning. In *Proc. NSF/ARPA workshop on Performance vs. Methodology in Computer Vision*, pages 152–166, Seattle, WA., June 24-25 1994.

[171] J. Weng, Paul Cohen, and Nicolas Rebibo. Motion and structure estimation from stereo image sequences. *IEEE Transactions on Robotics and Automation*, 8(3):362–382, June 1992.

[172] J. Weng, T. S. Huang, and N. Ahuja. Motion and structure from two perspective views: Algorithms, error analysis, and error estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(5):451–476, 1989.

[173] Peter R. Wilson. Conic representations for shape description. *IEEE Computer Graphics and Applications*, 7(4):23–30, 1987.

[174] A. K. C. Wong, S. W. Lu, and M. Rioux. Recognition and shape synthesis of 3D objects based on attributed hypergraphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(3):279–290, 1989.