Multi-Scale Representation and Recognition of Three Dimensional Surfaces Using Geometric Invariants

P. C. Yuen

Submitted for the Degree of
Doctor of Philosophy
from the
University of Surrey

Centre for Vision, Speech and Signal Processing
School of Electronics, Computing and Mathematics
University of Surrey
Guildford
Surrey
GU2 7XH
United Kingdom

26th October 2001

Abstract

A novel technique for multi-scale representation and recognition of three-dimensional (3-D) surfaces is presented. This is achieved by iteratively convolving local parametrizations of the surface with two-dimensional (2-D) Gaussian filters. In this technique, semigeodesic coordinates are constructed such that each vertex of a mesh becomes a local origin. A geodesic line from the origin is first constructed in an arbitrary direction such as the direction of one of the incident edges. The smoothing process eliminates surface noise and small surface detail gradually, and results in simplification of the object shape. Using this technique the surface Gaussian curvature (K) and mean curvature (H) values are estimated accurately at multiple scales together with curvature zero crossing contours. Furthermore, local maxima of absolute values of K and H as well as the torsion local maxima of absolute values of the zero crossing contours of K and H are located on the surface. These features are utilized by geometric hashing and global verification processes for robust object recognition. The matching algorithm uses a hash table prepared in the off-line stage. Given a scene of feature points, the measurements taken at scene points are matched to those stored in the hash table. Recognition results are demonstrated for rotated and scaled as well as partially occluded objects. In order to verify matches, 3-D translation, rotation and scaling parameters are calculated and results indicate that the technique is invariant to those transformations. Another advantage is that it is applicable to both incomplete surfaces which arise during occlusion and to surfaces with holes.

This research is also contributed by Dr. Farzin Mokhtarian and Dr. Nasser Khalili.

Key words: Multi-Scale Representation, Open and Closed 3-D Free-form Surfaces, Smoothed 3-D meshes, Curvature and Torsion Estimation, Object Recognition.

Acknowledgements

I would like to thank my supervisors Dr. Farzin Mokhtarian and Professor John Illingworth for the help which they have given me during the research. I would also like to thank Dr. Adrian Hilton for his advice on the ModelMaker 3-D laser scanner. This research was supported by the Engineering and Physical Sciences Research Council (EPSRC).

Further information:

http://www.ee.surrey.ac.uk/Research/VSSP/demos/css3d/index.html/
Email pcyuen@iee.org

Contents

List of Figures

List of Tables

Chapter 1

Introduction

"Vision is the process of discovering from images what is present in the world, and where it is" [106]. Discovering "what and where" are the problems of object recognition and location, and are major tasks in computer vision. Representation and recognition of 3-D objects have been an important and active area of research in computer vision for many years [36, 48]. Applications range from industrial automation to mobile robot navigation. Much work has been carried out on recovering geometric features from 2.5D range images [10, 12] as well as extending the scale space representation to 2-D signals [100, 95, 76]. However, a proper representational framework does not yet exist for 3-D surfaces.

The aim of this research is to develop a multi-scale representation for 3-D surfaces with arbitrary shapes through the use of geometric invariants [187, 138], and to utilize the representations for recognition of the underlying objects from arbitrary viewpoints. A novel feature of the envisaged representation is the combination of multi-scale characteristics with inherent geometric primitives extracted from the underlying surfaces. This gives rise to representations robust with respect to noise and local distortions of shape as well as invariant with respect to the shape-preserving transformations (i.e. uniform scaling, translation and rotation) of the underlying objects.

Another novel feature of the representation in this research is the ability to deal with objects of any shape without having to make simplifying assumptions about the

shape of those objects. As physical objects often do not look like the mathematically defined objects that are commonly used in computer vision, attempts to describe them using these mathematical primitives do not result in robust representations since the approximations obtained are quite poor.

The main goal of the research is to be able to recognize any 3-D object from an arbitrary viewpoint. As a result, each object model must be as complete as possible. However, depth data obtained from a single viewpoint is not sufficient to cover every part of the object surface. In the past, detailed and complete object models were constructed manually [40], but this is a very time-consuming process. Instead, depth data is now commonly obtained from multiple viewpoints and fused together to build a single complete model of the object [58].

With recent major improvement in imaging technology, it is now possible to obtain reliable data quickly using relatively inexpensive equipment. Effective techniques for processing that data open up potential applications in industrial automation/inspection [24] and mobile robot navigation [64]. Various methods such as depth from focus, stereo matching and laser range scanning are used for the recovery of depth values from 3-D objects [189, 101, 109]. Of these techniques, laser range scanning is believed to have the capability to provide the most reliable depth data at reasonable cost and speed.

As a result of recent advances in laser development, there are a number of 3-D laser range scanners (e.g. ModelMaker, Minolta VI-300) available for the automatic construction of complete object models. These 3-D laser scanner systems are based upon the measurement principle of triangulation. A low-intensity laser beam scans over an object whose surface may be treated with a removable white matt spray. The laser beam is reflected from the object surface towards the sensor, then passed through an auto focus lens, and finally recorded by a charge coupled device. This captured data is transferred as a point-cloud (raw-data) for further processing, such as surface registration [178] and surface fusion, to become a complete object model.

Once an object model and scene are obtained either in polygon mesh or some other format, then multi-scale representations of both model and the scene are computed

so that the surfaces from the scene can be compared to the object model. These multi-scale representations are based upon geometric invariants, therefore, robustness with respect to noise and local distortions of shape as well as invariance with respect to shape-preserving transformations are achieved.

In order to obtain descriptions of a 3-D surface at multiple scales, a smoothing filter (such as a 2-D Gaussian kernel) is convolved with a parametric representation at each point of the surface. In the case of a 2-D contour, arc length is utilized as the natural parametrization of the contour [67, 13]. However, there are no such natural parametrizations available for an arbitrary 3-D surface. Instead, it is necessary to rely on local curvilinear coordinates that are defined at the neighbourhood of each point of the surface. It follows that the entire surface must be covered with these local coordinate systems. Successively, filtered versions of the surface are computed by repeating this process using the previously smoothed surface as the input to the next stage.

Local coordinate systems can be defined in various ways on 3-D surfaces. The use of semigeodesic coordinates [44] and polar coordinates will be studied in this research. Furthermore, different types of geometric invariants which can be employed in multi-scale representations of 3-D surfaces are carefully examined. In general, contours of zero Gaussian curvature and zero mean curvature on a 3-D surface are space curves [30, 2]. These curves are in turn represented by the local maxima of absolute values of torsion. Together with the local maxima of absolute values of Gaussian and mean curvature, a rich set of features can be obtained for object recognition.

A common industrial vision problem is to determine the location and pose of objects in a scene. In order to apply this research to solve such problems, robust matching strategies have been developed to find the best match for the object model and the scene. As features taken from multiple scales of representations are used, the matching process is robust with respect to noise that can give rise to spurious features at the fine scales of the representation. Since the matching process must be able to accommodate occlusion [197], all possible local matches must be considered. In order to achieve this efficiently, geometric hashing is utilized to quickly eliminate

very unlikely candidates. Finally, global transformation parameters are used in a verification stage to determine the goodness of fit of the scene data to each model and to choose the best-matching candidate.

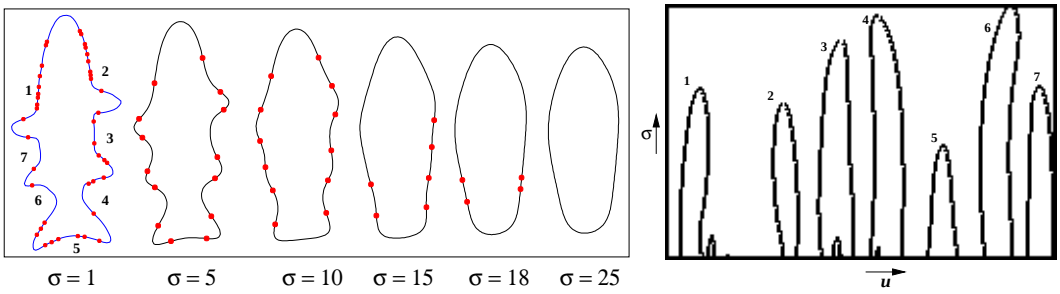
The results from this research are tested using objects viewed from arbitrary directions. The experiments are designed so that general conclusions are drawn about the investigated shape representations and object recognition system. The suitability of the system to handle objects with, not only different shapes but also belonging to different categories of shape, is also studied (i.e. examples of categories of shape include polyhedral versus curved objects or man-made versus natural objects).

The thesis is organized into seven major chapters that include the current introductory chapter. Chapter 2 discusses the background and motivation of this research. Chapter 3 reviews related work and survey the literature. Chapter 4 describes relevant theory from differential geometry, filtering and evolution properties of free-form 3-D surfaces. It explains how a multi-scale shape description can be computed for a surface. This chapter also covers the computation of Gaussian and mean curvatures as well as their zero crossing contours and local maxima of absolute values, torsion, geometric hashing and global verification. Chapter 5 discusses the implementation that includes parametrization of arbitrary 3-D surfaces, surface smoothing, curvature estimation, object recognition and verification. Chapter 6 presents experimental results. Chapter 7 reviews the work and discusses some interesting avenues for possible future work.

Chapter 2

Background and Motivation

In the early 1980's, the concept of scale space representations for one-dimensional (1-D) signals was introduced [198, 171]. These representations combined invariant features of the underlying signal extracted from a continuum of scale levels. They therefore achieved robustness with respect to considerable noise on the signal as well as invariance with respect to its shape-preserving transformations.



(a) fish (b) CSSI

Figure 2.1: *A contour during smoothing and its* CSS *image.* σ *is the width of a* 1-D *Gaussian kernel and* u *is the normalized arc length parameter.*

2-D contours [116, 117, 1, 31, 37] or planar curves [3] have been studied extensively in computer vision since the bounding contours of projections of 3-D objects are planar curves $\Gamma = \{(x(u), y(u)) \,|u \in [0,1]\}$ (see Figure 2.1(a)) [119]. If each coordinate function of Γ is convolved with the width of a 1-D Gaussian kernel σ (σ

is also referred to as scale), the resulting curve Γ_σ will be smoother than Γ. The locations of curvature zero crossings of Γ can then be found. As σ increases, Γ_σ becomes smoother and the number of zero crossings on it decreases. When σ becomes sufficiently high, Γ_σ will be convex curve with no curvature zero crossings (see Figure 2.1(a), $\sigma = 25$). The process can be terminated at this stage and the resulting points can be mapped to the (u, σ) plane. The result of this process will be a binary image called curvature scale space image (CSSI) (see Figure 2.1(b)).

This scale space concept was generalized and applied to 2-D contours [104, 127, 126]. Arc length was used to obtain a global parametrization of the contour. Curvature zero crossing points from multiple scales were used as feature points since they had the desired invariance properties. The CSSI was applied successfully to the problem of matching a Landsat satellite image [127, 35] to a map of the lower mainland of British Columbia, Canada.

Alternative ways of computing the representation were investigated in [105, 128]. As a result, two new variations (renormalize and resample) of the CSSI were discovered which offered more stability under conditions of non-homogeneous noise or local differences in shape. The existing representation types were thought of as being applicable to differing conditions of noise and local shape differences.

In order to properly evaluate the usefulness of the new representations for recognition tasks, the theoretical properties of those representations were investigated. Hence, global as well as local properties were studied. For example, it was shown that the underlying contour could be reconstructed from its CSSI representation (which showed that the CSSI was information-preserving) and that the computation of the CSSI converged to a stable solution. It was also shown that the physical interpretation of 2-D contours as object boundaries was not altered during CSSI computation. The results demonstrated that the CSSI possessed properties which were widely considered desirable for a shape representation technique in computer vision [105, 112, 128, 130].

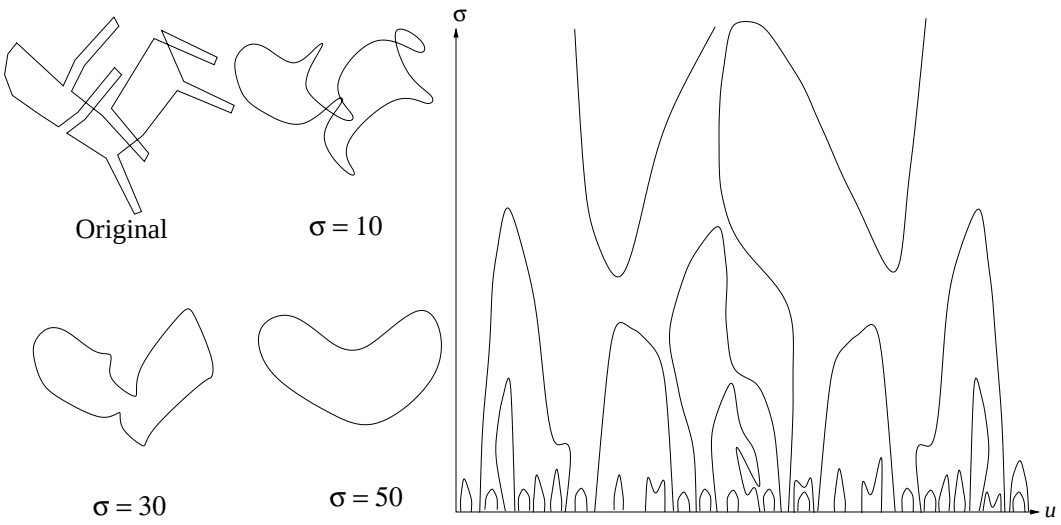
Further experiments were carried out to determine the utility of the CSSI for object recognition tasks [119]. An industrially oriented isolated object recognition system

based on the CSSI was developed which shown to be very robust and fast. No restriction was placed on object shapes or types. The system distinguished between objects with small differences in shapes [129, 115]. That system was followed up with another that was also industrially oriented but allowed occluding objects. The second system was reliable and fast (considering the complexity of the task). No modification to the system or its parameters was required in order for it to produce the correct interpretation on several complex scenes [116, 117].

Mokhtarian [119] showed the application of CSSI to the task of image database retrieval by shape content. There were approximately 1100 images of marine creatures in the chosen database. Each image showed one distinct fish species on a uniform background. Each image was processed to recover the boundary contour, which was then represented by three global shape parameters and the maxima of the curvature zero crossing contours in its CSSI.

Based on its properties, the CSSI based shape descriptor has been adopted as part of the MPEG-7 standard in September 2001 [201]. This illustrates the importance of CSSI. Many of the results obtained concerning the CSSI representation for 2-D contours have been further generalized to 3-D contours or space curves [111]. Space curves are important since they can be used to represent 3-D objects compactly and efficiently. In general, curvature zero crossing contours and orientation discontinuities on 3-D surfaces are invariant space curves. Figure 2.2(a) shows an original view of a space curve depicting an armchair. It also shows the armchair during evolution at $\sigma = 10$, $\sigma = 30$ and $\sigma = 50$. As a space curve evolves, more and more high-level descriptions of its shape are obtained. Thus the evolution process results in a continuous fine-to-coarse description of the shape of a space curve. These resulting representations are referred to as the torsion scale space image (TSSI) (see Figure 2.2(b)).

The theoretical properties of the TSSI representations have also been investigated in a systematic way. Two additional variants (renormalize and resample) have been developed, which were appropriate for varying conditions of noise and local distortions of shape [113]. The results were closely comparable to the properties of the



(a) Armchair (b) TSSI

Figure 2.2: *A space curve during smoothing and its TSS image. σ is the scale and u is the arc length measured along the space curve.*

CSSI representations and showed that TSSIs have properties generally considered useful for shape representation methods in computer vision [110, 112, 130, 114]. The TSSIs were also studied in order to determine their suitability for object recognition tasks. It was determined that the TSSIs were in fact very suitable for recognition of arbitrarily shaped space curves affected by significant noise [113].

Effective shape representation schemes are crucial to reliable and robust object recognition that in turn is a central issue in computer vision. Much work has been carried out on shape representation schemes for 3-D objects but frequently simplifying assumptions are made about the shape of objects. This gives rise to two types of problem. Firstly, the class of objects which are acceptable to the system became severely restricted since the shapes of many objects do not satisfy the simplifying assumptions. Secondly, when the class of acceptable objects is enlarged, the system became unreliable since the shapes of some objects do not conform to the built-in mathematical object models.

This research was timely since projective invariants [152, 102] and multi-scale representations based on geometric invariants have recently become the focus of much

attention. However, multi-scale representations have so far been developed only for 1-D and 2-D signals as well as 2-D and 3-D contours. The full generalization of the concept to 3-D surfaces was not yet accomplished. Equally, there was very little use of inherent geometric invariants of 3-D surfaces. Mostly the properties of artificially constructed models or primitives are used but these may not fit the 3-D free-form real data well. Since geometric invariants result in representations reflecting the properties of the data from which it is computed, so the type of representation used in this research will be invariant with respect to the transformations that do not alter the shape of the underlying object. Furthermore, the multi-scale structure of the representation promotes stability with respect to noise and local deformations of shape that are always present in real data. Noises as well as very small features of the surface are represented on the fine scales of the representation whereas the main structural elements of the surface exist on the coarser scales. This agrees well with biological vision systems [45] that organize visual data at multiple levels of scale or detail. In summary, the prime motivations for this research are:

- Generalizing the existing scale space representations to arbitrarily shaped 3-D surfaces.
- Studying the properties of these representations, developing methodology for multi-scale representations of 3-D surfaces and complex objects based on intrinsic geometric invariants.
- Using the above representations for pose estimation and recognition of 3-D objects from arbitrary viewpoints, and demonstrating their usefulness for challenging object recognition tasks.

Finally, its results should have a considerable impact on other researchers in the field and it should also benefit the computer vision community in general.

Chapter 3

Related Work and Literature Survey

Many object representation and recognition systems rely on restrictions imposed on the geometry of the object. However, complex free-form objects may not be modelled easily using such restrictions. On a free-form object, the surface normal is defined and continuous everywhere, except at sharp corners and edges. Discontinuities in the surface normal or curvature may also be present anywhere on the object. The curves that connect these points of discontinuity may meet or diverge. Therefore, representation and recognition of such an object can be difficult, especially for a free-form object where inspection of arbitrary curved surfaces and path planning for robot navigation is essential. Nevertheless, the following sections highlight related work that attempts to solve these problems in object representation and recognition [8]. At the end of this chapter, a general overview summarizing the related work and literature survey is given.

3.1 3-D Surfaces Construction and Representation

3.1.1 Automatic Registration

Neurosurgeons need reliable and accurate 3-D image recognition. Many surgical procedures require Magnetic Resonance Imaging (MRI) and Computed Tomography (CT) data to be matched while patients are on the operating table. The common approach is to use a 3-D laser range scanner for obtaining depth data [103] from the patient's skin surface where the surgery is performed. When the landmark [54] features are obtained, the surgeon can see inside the patient with MRI or CT registration. Therefore, an automatic registration method [47] is introduced which must match and reconstruct a live view of a patient. This enables a surgeon to visualize the internal structure of the patient before executing the guided neurosurgical procedures. Other applications for automatic registration are image guided biopsies and focused therapeutic procedures. These applications use a multi-stage and multi-resolution [177, 176, 170] algorithm. Its accuracy ranges from 10 microns to 1.5 millimetres.

3.1.2 Deformable Surfaces

Deformable surfaces [156, 135] can be used to fit the smooth varying objects efficiently. It reconstructs a surface model of an object from range data. Parametrization of surface is often achieved using B-splines. However, they have limitations when modelling general closed surfaces or teacup like surfaces with handles. These surfaces with a fixed mesh topology are similar to the flexible plates or membranes. They may deform but they cannot restructure their topology. In fact it is impossible to use B-splines to describe arbitrary topology surfaces because B-splines maintain C^1 (first order) continuity. Since triangulated meshes are used to represent 3-D objects, these representations are not efficient when describing curved surfaces. However, the meshes based on B-spline deformable surfaces are more efficient. The only drawback is that they connect control points in a predefined way, i.e. they cannot describe surfaces of arbitrary topology. Therefore, generalized biquadratic

B-splines or slime based on geometric continuity G^1 (i.e. the directions of two segments' tangent vectors are equal at a join point of a curve), the drag force, the elastic force and the data force, can be chosen to represent an arbitrary topology.

3.2 Object Recognition

3.2.1 Aspect Matching

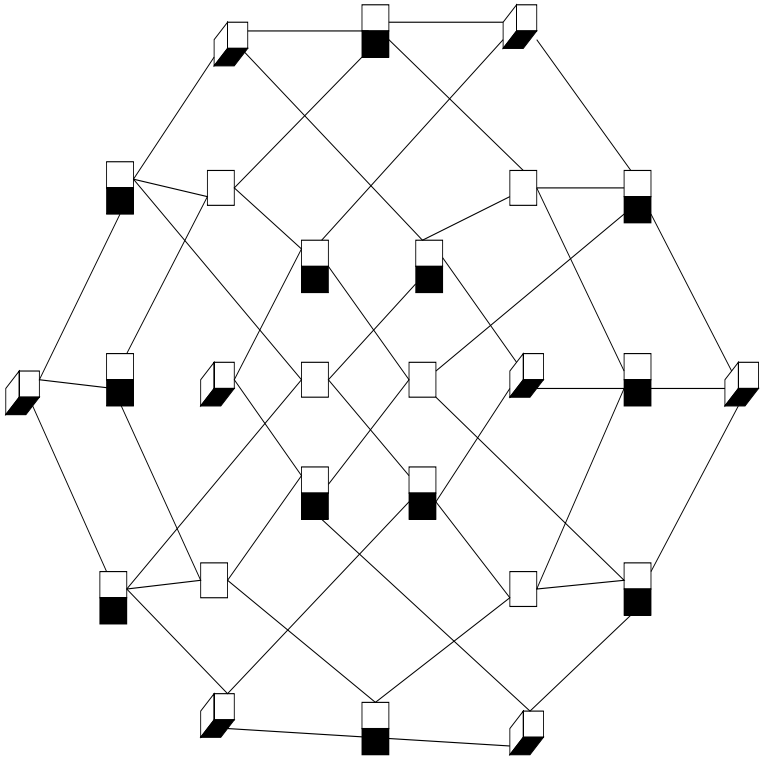


Figure 3.1: *A cuboid object and its aspect graph. Each node in the aspect graph represents a stable view. The branches show how one can go from one stable view to other stable views.*

A 3-D object can be represented using its aspect graph. An aspect graph represents all stable views of an object. An aspect graph is obtained by partitioning the view-space into areas in which the object has stable views. The aspect graph for an object represents a relationship among all the stable view. A cuboid object and its aspect graph are shown in Figure 3.1.

Since 3-D shapes can be recognized using aspect graphs, Dickinson [29, 28] applies this approach to develop an object recognition system. 3-D object are represented using several views obtained either from regularly spaced viewpoints in space or from some strategically selected viewpoints [75]. For a limited set of objects, we may consider many views of the object and then represent each view in an observer-centred representation.

On the other hand, 3-D shape recovery using distributed aspect matching [29] is an approach to the recognition of 3-D objects. It can either be described by the mathematical object representation [136, 20] or represented by the image structure graph [32]. The result of object recognition is based on the small and compact non-occluded image database [157]. For partial objects, an aspect hierarchical approach is made to overcome the problems of occlusion. Given an image of an input scene with multiple occluded objects, this image can be segmented into regions. Then the aspect hierarchy can be analysed for local matching. If there is an exact match, a hypothesis will be generated [43]. These aspect hypotheses can be ranked and used for verification [49].

3.2.2 Curvedness Orientation Shape Map

Curvedness orientation shape map on sphere (COSMOS) is a system of automated 3-D free-form rigid object representation and recognition [73]. It uses dense surface range data to recognize arbitrarily curved 3-D objects from different views. Its viewpoint can be arbitrary. The object may vary in shape and complexity.

COSMOS has a multi-level matching strategy which includes at the first level model selection and at the second level model feature identification. In the first level, the collection of views obtained from each object is used for a shape spectrum based model selection scheme [38]. This stores the object label and its pose vector. A moment-based shape spectrum can be calculated and compared with a feature histogram. A typical example of the COSMOS first level experiment includes 6400 views of 20 free-form objects. The test result with 2000 independent views shows on average 20% of the database was correctly classified.

For the second level COSMOS based view matching, the constant shape maximal patch (CSMP) method uses spherical convex shape cylinders to construct and represent a local area of a 3-D object. From the selected model view hypotheses, CSMP can match the scene object reasonably. The whole object recognition system, including the first and second levels, is tested using 50 range images obtained from ten different free-form objects, and COSMOS is able to identify 82% of objects with high matching score.

3.2.3 Edge Map Recognition

3-D free-form surface recognition by using an edge map [17] illustrates how to use a 2-D view approach to recognize free-form 3-D surfaces. This includes the basic modelling, indexing and matching of a 3-D surface. The prime objective of modelling is to construct the edge map of 3-D surfaces from any viewpoint. This edge map can be indexed and stored in a database. Then hypotheses are generated by matching the 3-D surface to the database.

At first, 2.5D models are used to construct the overlapped 2-D images from 5 different viewpoints. At each view, an edge map is presented by approximating the edges. Then the estimated edges are segmented according to the edge contour extrema [55, 33, 192, 7, 23]. Now these newly generated invariant segment features are encoded and indexed by using hash table. Afterwards, the models are used for pose clustering. The correct candidates are ranked. Then the most likely hypotheses are verified at coarse, middle and fine sampling levels.

3.2.4 Extended Gaussian Image

The extended Gaussian image (EGI) [63] represents the shapes of surfaces. It can be computed from needle maps and depth maps. It is useful for the task of 3-D surface recognition because no two convex objects have the same extended Gaussian image $G(\xi, \eta)$, where (ξ, η) is a point on the Gaussian sphere, which has the same normal as the point (u, v) on the original surface. ξ is the longitude and η is the latitude on the Gaussian sphere. An orientation histogram is used to approximate

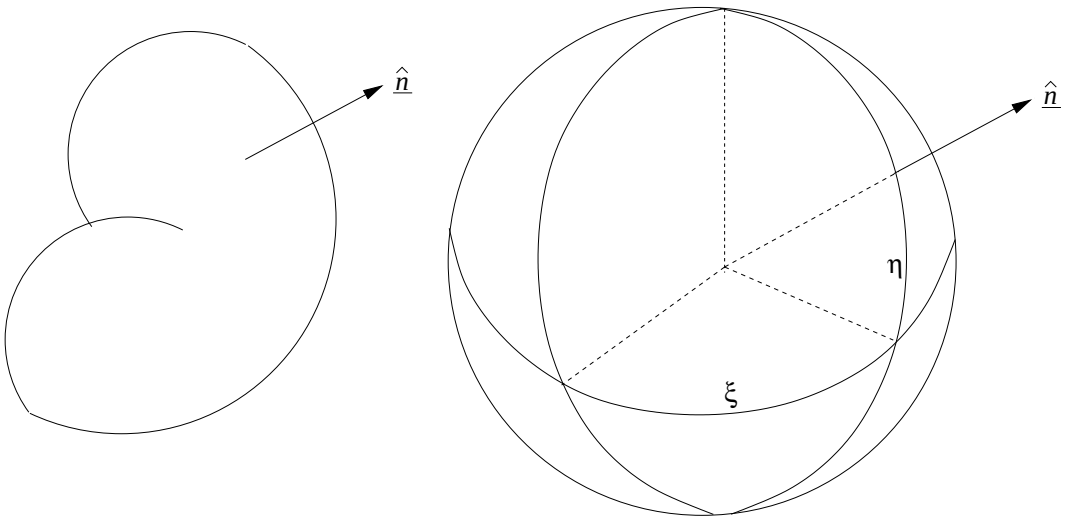


Figure 3.2: *The image of a point on the surface under the Gaussian map is the point on the unit sphere that has the same surface orientation and normal vector* $\underline{\hat{n}}$.

the extended Gaussian image. The applications of extended Gaussian images include object recognition, and to find the attitude in space of an object.

However, when part of the object is occluded, EGI cannot reliably extract the representation. Hence, an alternative approach, simplex angle image (SAI) [26], base on the idea of fitting a bounded algebraic surface of fixed degree of a set of data points can be used. It is a combination of the point set matching and of the original EGI approach. As mesh deformation is applied to produce two important properties, these firstly include the mesh nodes that must be close to the original surface. Secondly, the mesh nodes must satisfy the normal constraints. Therefore, the update rule can be obtained. A SAI image is then retrieved from this mesh. The next step is to compute the full transformation and to match the objects that include the partial views and occlusion. SAI can be used as representation for 3-D object recognition. It has invariant and coarse-to-fine properties. It can also be used to measure curvature.

3.2.5 Genetic Algorithm

To recognize a scene object from an image database requires some knowledge of object features [191]. Genetic algorithm (GA) represents an approach to optimization that uses natural evolution mechanisms to search for the maximum of an objective function [167]. As with any optimization technique, they can be used in recognition and machine learning. GA is defined as follows:

1. Create a starting population of code strings, and find the value of their objective functions.
2. Probabilistically reproduce high fitness strings in the new population, remove poor fitness strings. It is called reproduction.
3. Construct new strings combining reproduced code strings from the previous population (crossover).
4. From time to time change one character of some string at random (mutation).
5. Order code strings of the current population according to the value of their objective functions (fitness).
6. If the maximum achieved string fitness does not increase over several steps, stop. The desired optimum is represented by the current string of maximum fitness. Otherwise, repeat the sequence of steps starting at 2.

This technique is useful for object recognition [134]. For example, using a GA to perform a heuristic search of the Hough transform parameter space [5, 72, 164], rather than an exhaustive search, it overcomes the problems of computation time and memory requirements.

3.2.6 Level Set Methods

The application of level set methods in shape detection and recognition is shown in [162]. For example, a medical scan is given with the goal of isolating and identifying

tumours. It might isolate these shapes by propagating a front to the boundary with criteria built from the image I gradient. The studies of this moving front depend on geometric properties such as normal direction and curvature. Its speed function F is specified, in term of the curvature κ [153, 154]. The image $I(x, y)$ is convolved with a Gaussian smoothing filter G_σ whose characteristic width is σ. The term $\nabla G_\sigma \otimes I(x, y)$ is essentially zero except where the image gradient changes rapidly, in which case the value becomes large. Thus, the filter $k_I(x, y) = \frac{1}{1+\nabla G_\sigma \otimes I(x,y)}$ is close to unity away from boundaries, and drops to zero near sharp changes in the image gradient. This information can be used for shape detection and recognition. There are several desirable aspects of this approach:

- The initial front can consist of many fronts; due to the topological capabilities of the level set method, these fronts will merge into a single front as it grows into a particular shape.
- The front can follow intricate twists and turns in the desired boundary.
- The technique can be used to extract 3-D shapes as well by initializing as a ball inside the desired region.
- Small isolated spots of noise where the image gradient changes substantially are ignored; the front propagates around these points and closes back in on itself and then disappears.

3.2.7 Multi-view Methods

By just looking at the planar surfaces of a 3-D model, a set of linear invariant features can be extracted from these flat surfaces using the orthogonal transformation. This technique allows the occluded 3-D surface to be recognized [133, 50, 88]. As the planar surfaces of a 3-D model are examined using the 2-D views, the affine transformation parameters can be determined from three feature points. They define the scaling factor, translation and rotation for object recognition. In addition, it also satisfies the definition of linear transformation. Furthermore, the centroid of its projection can be transformed by the same transformation.

Since affine parameters are determined from three points, a group of such features clustered in a small concentrated area can be applied to a partial occluded surface. Therefore, the following algorithm is defined. Firstly, from a feature set of 2-D views of a 3-D model, at least three clusters are generated. For each cluster, its centroid is computed off-line. Secondly, the affine transformation for each of the possible combinations of triples of the cluster centroids in the original model and input object from the scene are computed. Finally, the best-fit affine transformation is applied to align the original coordinates from 3-D model to the scene.

The experiment shows that this algorithm recovers 3-D objects with more than 50% of its surface occluded. It takes 100ms for recognizing an object with more than 300 features on a Sun Sparc workstation.

3.2.8 Neural Networks

Recently, neural networks have become popular in image processing [196, 207]. This state machine technique can be used in image coding, object recognition, motion, etc [6]. A neural network using constructive solid geometry based 3-D object recognition [18] is applied to recognize input scene in term of primitives such as block, spheres, cylinders, cones and tori. Each primitive is described by concavity and convexity. Then precedence graphs are used to define the order according to subtracting arc, appending arc, gluing arc and adjacency arc [60].

After the input objects are represented, mean field annealing neural networks are used for object recognition. A 2-D array of neurones is applied to index the model primitives and the scene primitives. An energy constraints function is used to calculate the matching support of primitives. A final network output is produced for matching with optimization and eliminating the undeterminants.

When the optimal match between the scene and the model objects has been found, 3-D transformation and quaternion [173, 194, 97, 25] technique for rotation angle estimation are applied to remove the difference between the scene and the model objects. Finally, a models database with six objects is tested and successfully produces a network state symmetric matrices for object recognition.

3.2.9 Weighted Average Smoothing

A million vertices are not unusual in 3-D meshes [131]. Normally, such a surface is smoothed by Fourier Transform [144, 99]. A large surface may be simplified during the smoothing process, and re-meshed using far fewer faces. However, its curvature value must be maintained at the same time. Hence, a surface subdivision approach is required [185]. As the surfaces are represented by the vertices and polygonal faces [151], the surface subdivision causes the next face in the sequence to be constructed from the previous face by a refinement/decimation process [62, 61, 19]. Once the faces are smaller than the necessary resolution, this process stops and a new surface is created. However, complications may occur when the iterative process is applied. Therefore, a simple surface signal low-pass filter algorithm is introduced in here [183], which uses the weighted average method with neighbouring vertices, positive scale factors λ and negative scale factor μ. This method smoothes the large 3-D surface without shrinkage. The smoothing process is done in terms of the discrete surface signal. For object recognition, Taubin [184] extends the use of polyhedral approximation to estimate the curvature of surfaces.

3.3 Overview

This section presents a summary of previous work in representation and recognition of 3-D surfaces. As Sinha and Jain [165] provide an overview of geometry based representations derived from range data of objects, comprehensive surveys of 3-D object recognition systems are presented by Besl and Jain [9], Chin and Dyer [21] and Suetens [180]. Some representation schemes for 3-D objects have adopted some form of volumetric or surface parametric models to characterize the shape of the objects. Current volumetric representations rely on representing objects in terms of general cylinders, superquadrics, set-theoretical combinations of volume primitives as in constructive solid geometry or spatial occupancy [14, 143, 166, 18, 155]. However, it may not be possible to express objects with free-form surfaces using for example, superquadric primitives. Surface-based representations describe an object in terms

of the surfaces bounding the object and their properties [36, 74, 34, 38], and are employed for recognition. Although there are several methods available to model a surface, the common type is polygonal mesh that includes the triangulated mesh [59, 58] and the four sided spline patches. Since triangulated meshes are the simplest and most effective form of polygons for covering a free-form surface, they will be used in this research.

Polyhedral approximations [36] fit a polyhedral object with vertices and relatively large flat faces to a 3-D object. Their disadvantage is that the choice of vertices can be quite arbitrary which renders the representation not robust. It is important to have a robust representation with minimum noise. Noise can affect the result of object recognition. Smoothed 3-D splines [175] can also be fitted to 3-D objects. Their shortcomings are that the choice of knot points is again arbitrary and that the spline parameters are not invariant. Generalized cones or cylinders [168] as well as geons [145] approximate a 3-D object using globally parametrized mathematical models [182], but they are not applicable to detailed free-form objects. Multi-view representations [161] are based on a large number of views of a 3-D object obtained from different viewpoints, but difficulties can arise when a non-standard view is encountered. In volumetric diffusion [86] or level set methods [162], an object is treated as a filled area or volume. The object is then blurred by subjecting it to the diffusion equation. The boundary of each blurred object can then be defined by applying the Laplacian operator to the smoothed area or volume. The major shortcoming of these approaches is a lack of local support. In other words, the entire object data must be available. This problem makes them unsuitable for object recognition in the presence of occlusion. A form of 3-D surface smoothing has been carried out in [183, 186] but this method has drawbacks since it is based on weighted averaging using neighbouring vertices and is therefore dependent on the underlying triangulated mesh. Now for other areas of research, the smoothing of 3-D surfaces is a result of the diffusion process [187] that has been examined carefully. For parametrization of a 3-D surface, other methods have also been studied, such as the asymptotic coordinates [90], isothermic coordinates [44] and global coordinates [13] used for closed, simply connected objects.

Global representations such as the extended Gaussian images [63, 80, 98] describe 3-D objects in terms of their surface normal distributions on the unit sphere with appropriate support functions. However, arbitrary curved objects have to be either approximated by planar patches or divided into regions based on the Gaussian curvature. Another approach for specifying a 3-D object is the view-centred representation. The graph approach [87] attempts to group a set of infinite 2-D views of a 3-D object into a set of meaningful cluster of appearances. Murase and Nayar [132] and Swets [181] also exploit photometric information to describe and recognize objects. A major drawback of view-centred representations is a lack of complete information. Part based representations capture structure in object descriptions [150, 29], but there is a lack of agreement in deciding the general set of part primitives that need to be used in order to be sufficient and appropriate. Furthermore computation of parts from a single view of an object is difficult. Recent approaches using point set based registration [11], splash and super polygonal segments [172] and algebraic polynomials [81, 147] have addressed the issue of representing complex curved free-form surfaces. However, there are limitations relating to object segmentation issues, surface fitting convergence, restricting objects to be topologically equivalent to a sphere and sensitivity to noise when low-level surface features are used. For curvature estimation on 3-D surfaces, [27] shows an implicit fairing of irregular meshes using diffusion and curvature flow. Problem with this approach is that it creates a so-called chicken and egg issue. Since curvature estimation on a noisy mesh is unreliable, to improve the estimates, the data must be smoothed first. However, Deshrun's method requires curvature estimates before smoothing can be carried out. In contrast, the technique used in this research avoids such problem by combining smoothing and curvature estimation in a unified framework.

A number of matching topics have been acknowledged by researchers as important in 3-D object recognition [107, 193, 39]. These are related to object shape complexity, rigid and flexible objects and occlusion. The success of existing object recognition system results from the restrictions they impose on the classes of geometric objects. However, few systems can handle arbitrary surfaces with very few restrictive assumptions about their geometric shapes. Object recognition is achieved by match-

ing features derived from the scene with stored object model representations. From the related work in the previous sections, they show that an efficient algorithm has been developed for the recognition of 2-D views or flat rigid objects based on the geometric hashing technique in [199, 190, 51, 91, 93]. This technique is also extended to the recognition of arbitrary rigid 3-D objects from single 2-D images [92]. Stein and Medioni [172] and Flynn and Jain [39] have also employed geometric hashing for 3-D object recognition. In a geometric hashing technique the model information is indexed into a hash table using minimal transformation feature points. This technique determines a minimal feature set from a given scene and a corresponding feature set on one of the models, by considering only the scene features that vote for the correct interpretation. Although other model based object recognition techniques such as Hough (pose) clustering [5, 164, 174, 142], the alignment technique [70, 71], relational structures [14, 34, 200], superquadrics [53, 166, 52] with local and global 3-D deformations are available, the advantages of the geometric hashing technique over the others is the independent processing of the model and scene information as well as its quick recognition time, and its ability to process all the model objects simultaneously and to allow occlusion. Recent patch-based techniques including point signatures [22] and spin images [79] perform well on scenes containing clutter and occlusion. However, these systems have been designed for single range images, and do not generalize to more general 3-D surfaces that can be obtained by merging two or more range images. In other words, their effectiveness is limited by the use of information in only one range image.

Chapter 4

Theory and Design Analysis

This chapter introduces a technique for multi-scale representation and recognition of 3-D surfaces using geometric invariants [205, 122, 124]. The technique considered here is a generalization of earlier multi-scale representation theories proposed for 2-D contours [128] and space curves [118]. More details of these techniques appear in [120, 121, 203, 204, 82, 83].

After a 3-D object model is constructed, diffusion smoothing of its surface can be achieved through convolution of local parametrization of the surface with a 2-D Gaussian filter. Semigeodesic coordinates [44] are utilized as a natural and efficient way of locally parametrizing surface shape. They have local support and are therefore applicable to partial data corresponding to surface-segments.

Curvature estimation is one of the important tasks in 3-D object description and recognition. Surface curvature provides a unique viewpoint invariant description of local surface shape. Differential geometry [90, 89, 77, 96] gives several measures of curvature that include Gaussian and mean curvatures. Combination of these curvature values enables the local surface type to be categorized.

Once surface curvatures are estimated, then curvature zero crossing contours are recovered on the surface. Local maxima of absolute values of Gaussian and mean curvatures as well as the local maxima of absolute values of torsion of zero crossing contours of Gaussian and mean curvatures are also located on the surface. Together

they present sets of interest features for recognition.

At the end of chapter, the recognition of free-form 3-D objects based on using geometric hashing is addressed. This technique is useful for partially occluded objects. Model information is indexed into a hash-table using minimal transformation invariant features which are local maxima of absolute values of Gaussian curvature, mean curvature and torsion, so that both the model object and the observed scene can be matched. The matching result is verified by using the global parameters from the scene and models to finally accomplish the recognition of the free-form 3-D object.

This chapter includes six major sections that are the theories behind this research. Section 4.1 describes 3-D parametrization, geodesic lines, semigeodesic coordinates and geodesic polar coordinates. Section 4.2 details a 3-D free-form surface smoothing technique and its multi-scale evolution properties. Section 4.3 covers the computation of Gaussian and mean curvatures. Section 4.4 shows torsion estimation. Section 4.5 presents the geometric hashing algorithm and Section 4.6 describes the global parameter transformation process for 3-D free-form surfaces verification.

4.1 Parametrization

A crucial property of 2-D contours and space curves (or 3-D contours) is that they can be parametrized globally using the arc length parameter. However, free-form 3-D surfaces are more complex. As a result, no global coordinate system exists on a free-form 3-D surface that could yield a natural parametrization of that surface. Indeed, studies of local properties of 3-D surfaces are carried out in differential geometry using local coordinate systems called curvilinear coordinates or Gaussian coordinates. Each system of curvilinear coordinates is introduced on a patch of a regular surface referred to as a simple sheet. A simple sheet of a surface is obtained from a rectangle by stretching, squeezing, and bending but without tearing or gluing together. Given a parametric representation

$$\mathbf{r} = \mathbf{r}(u, v) \tag{4.1}$$

on a local patch, the values of the parameters u and v determine the position of each point on that patch.

4.1.1 Geodesic Lines

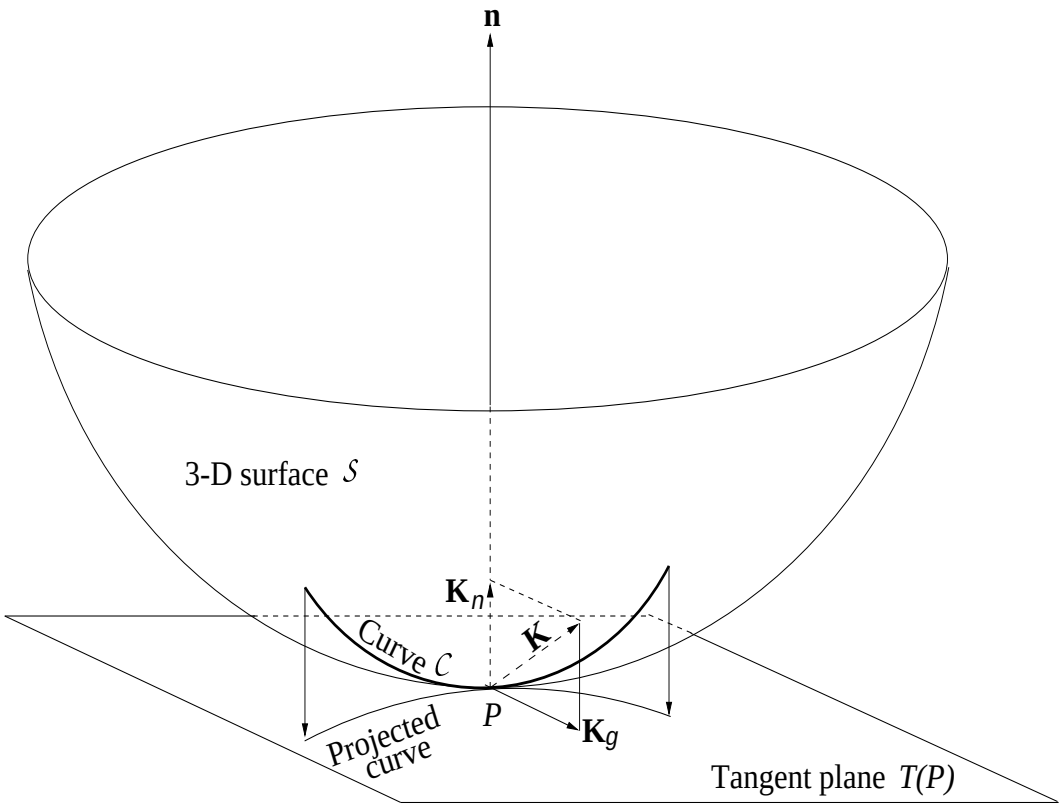


Figure 4.1: *The geodesic curvature*

Before any forms of curvilinear coordinates or Gaussian coordinates are described, it is necessary to define geodesic lines on a regular 3-D surface. A geodesic line is defined as a contour that locally represents the shortest distance on the 3-D surface between any two points on that contour [46, 139, 16]. Initially a geodesic line is drawn arbitrarily through the origin at the local area. This geodesic line is then sampled at equal-sized intervals. The second family of lines are also geodesic lines and they are drawn orthogonally at each sampled point. All the lines together produce semigeodesic coordinates that are a form of curvilinear coordinates and its details will be shown in the next Section 4.1.2. Now the following definitions are

made:

Definition 1 A geodesic line or a geodesic of a surface is a curve whose geodesic curvature is zero at every point. Geodesic curvature is the magnitude of the vector of geodesic curvature $\mathbf{K}_g$ (see Figure 4.1).

Definition 2 The vector of geodesic curvature $\mathbf{K}_g$ of a curve $\mathcal{C}$ lying on a surface $\mathcal{S}$ at a point P on $\mathcal{C}$ is obtained by projecting the curvature vector $\mathbf{K}$ of $\mathcal{C}$ at P on the tangent plane $T(P)$ to $\mathcal{S}$ at P.

Definition 3 The vector $\mathbf{K}_n$ of normal curvature is obtained by projecting the curvature vector $\mathbf{K}$ on the surface normal vector $\mathbf{n}$.

Definition 4 The curvature vector $\mathbf{K}$ of a curve $\mathcal{C}$ at point P is of the same direction as the principal normal vector at P and of length equal to the curvature of the curve at P.

Definition 5 The plane with the highest possible order of contact with the curve $\mathcal{C}$ at point P is called the osculating plane at P. [44]

NB Let P be a common point of the curve $\mathcal{C}$ with a surface $\mathcal{S}$, and let p be a variable point of the curve such that the signed arc length between P and p is h. Denote by d_h the distance of p from the surface. We say that the surface and curve have a contact of order at least n at P if $d_h = 0(h^n)$.

Definition 6 The principal normal vector of curve $\mathcal{C}$ at point P is perpendicular to $\mathcal{C}$ at P and lies in the osculating plane at P.

The following crucial minimal property of geodesic lines is actually utilized to construct geodesics on 3-D surfaces:

Definition 7 An arc of a geodesic line $\mathcal{C}$ passing through point P and lying entirely in a sufficiently small neighbourhood of point P of surface $\mathcal{S}$ of class C^2 is the shortest join of P with any other point of $\mathcal{C}$ by a curve lying in the neighbourhood.

4.1.2 Semigeodesic Coordinates

Semigeodesic coordinates are one form of curvilinear coordinates. They consist of an arbitrary geodesic line $\mathcal{C}$ and a second family of geodesic lines $\mathcal{L}$. The second family of geodesic lines must be perpendicular to the arbitrary geodesic line. They are constructed in the following way at point P on surface $\mathcal{S}$ of class C^2 (i.e. a manifold or a surface is differentiable of order 2):

- Choose a geodesic line $\mathcal{C}$ through point P in an arbitrary direction.
- Denote by u the arc length parameter on $\mathcal{C}$, such that P corresponds to the value $u = 0$.
- Take further through every point of $\mathcal{C}$, the geodesic line $\mathcal{L}$ which is perpendicular to $\mathcal{C}$ at the corresponding point.
- Denote by v the arc length parameter on $\mathcal{L}$.

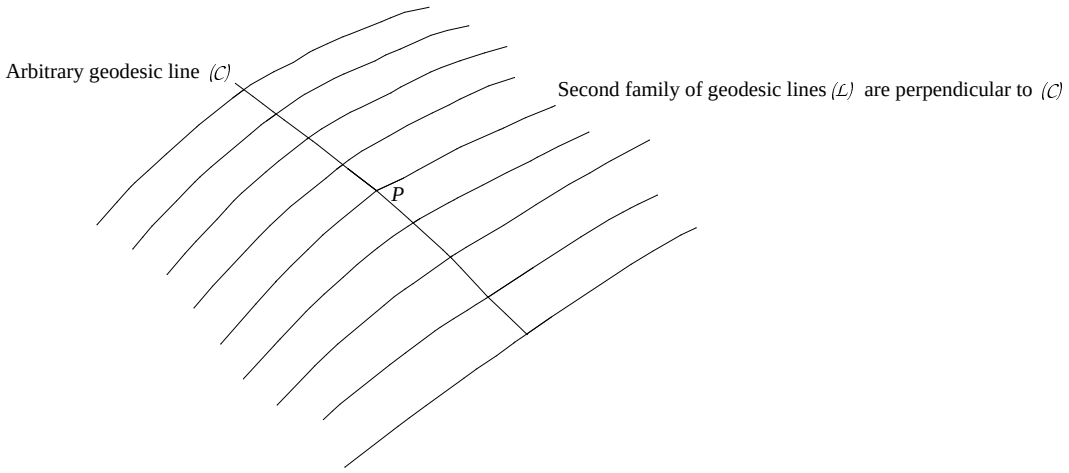


Figure 4.2: *Two families of geodesic lines $\mathcal{C}$ and $\mathcal{L}$ at point P on an arbitrary surface $\mathcal{S}$.*

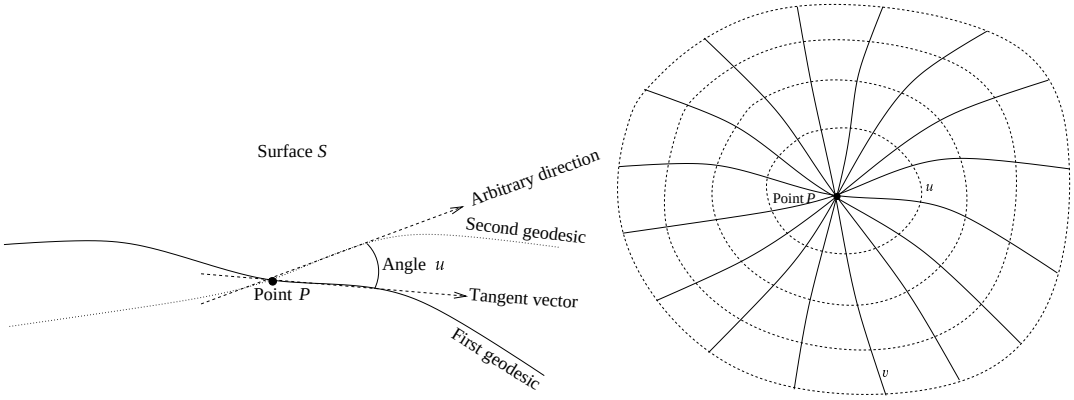
The two parameters u and v determine the position of each point in the domain swept out by these geodesic lines. The line ($v = 0$ with variable u) and the lines (u = constant with variable v) form two families of lines which give rise to the semigeodesic coordinates on the surface (see Figure 4.2). It is also shown that

in a sufficiently small neighbourhood of point P, semigeodesic coordinates always serve as curvilinear coordinates in a regular parametric representation of $\mathcal{S}$, and the orthogonal Cartesian coordinates in the plane are a special case of semigeodesic coordinates on a flat surface.

4.1.3 Geodesic Polar Coordinates

Geodesic polar coordinates are another type of geodesic surface coordinate. They are constructed at point P on surface $\mathcal{S}$ of class C^2 in the following way:

1. Choose an arbitrary direction w on $\mathcal{S}$ at point P.
2. Take all geodesic lines emanating from point P.
3. Denote by v the arc length parameter on each geodesic in previous step.
4. Denote by u the angle between w and the tangent vector of each geodesic in step 2 at point P.



(a) Angle u (b) Polar coordinates

Figure 4.3: *Geodesic polar coordinates*

Again the two parameters u and v determine the position of each point in the domain swept out by these geodesic lines. Figure 4.3 shows the geodesic polar coordinates in a sufficiently small neighbourhood of point P.

4.2 Surface Smoothing

The procedures outlined at Section 4.1.2 are followed to construct semigeodesic coordinates at every point of a 3-D surface $\mathcal{S}$, and local parametrization yields at each point P

$$\mathbf{r}(u,v) = (x(u,v),\ y(u,v),\ z(u,v)) \tag{4.2}$$

If this surface is now smoothed using a 2-D Gaussian filter, then the new location of point P is given by

$$\mathbf{R}(u,v,\sigma) = (\mathcal{X}(u,v,\sigma),\ \mathcal{Y}(u,v,\sigma),\ \mathcal{Z}(u,v,\sigma)) \tag{4.3}$$

where

$$\begin{aligned} \mathcal{X}(u,v,\sigma) &= x(u,v) \otimes G(u,v,\sigma) \\ \mathcal{Y}(u,v,\sigma) &= y(u,v) \otimes G(u,v,\sigma) \\ \mathcal{Z}(u,v,\sigma) &= z(u,v) \otimes G(u,v,\sigma) \end{aligned} \tag{4.4}$$

in which $\otimes$ denotes convolution and the 2-D Gaussian filter is

$$G(u,v,\sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{(u^2+v^2)}{2\sigma^2}} \tag{4.5}$$

In the case of geodesic polar coordinates, the Gaussian function becomes 1-D. As a result, each of the 2-D convolutions above can be expressed as a series of 1-D convolutions. In both cases, semigeodesic and geodesic polar coordinates are valid locally and the 2-D Gaussian filters always have $\sigma = 1$. The smoothing process is repeated at each point of $\mathcal{S}$. The new point positions after filtering $\mathbf{R}(u,v,\sigma)$ become the smoothed surface. Since geodesic polar coordinates can be interpreted as the semigeodesic coordinates [44, 56, 108, 90, 46, 139], semigeodesic coordinates are therefore needed for curvature estimation.

4.2.1 Evolution Properties

In order to achieve multi-scale descriptions of a 3-D surface $\mathcal{S}$, it is required to smooth the surface according to the process described previously. The smoothed surface is

then considered as the input to the next stage of smoothing. This procedure is then iterated many times to obtain multi-scale descriptions of $\mathcal{S}$ and it is called the evolution of 3-D surfaces. The following theorems show the fundamental properties of such an evolution.

Theorem 1 The order of application of evolution and a shape preserving transformation to a surface does not change the final result.

Proof Suppose surface $\mathcal{S}$ is evolved into $\mathcal{S}_\sigma$, every point of $\mathcal{S}_\sigma$ is a weighted average of a subset of points of $\mathcal{S}$. Therefore, evolution at each point Q of $\mathcal{S}$ can be expressed as the convolution of a neighbourhood of Q with a 2-D function with unknown values

$$P(X,Y,Z) = (x(u,v) \otimes f(u,v),\ y(u,v) \otimes f(u,v),\ z(u,v) \otimes f(u,v))$$

Now applying an affine transform to point P to get $P_1(X_1, Y_1, Z_1)$ where

$$\begin{aligned} X_1 &= a_1X + b_1Y + c_1Z + d_1 \\ Y_1 &= a_2X + b_2Y + c_2Z + d_2 \\ Z_1 &= a_3X + b_3Y + c_3Z + d_3 \end{aligned}$$

Alternatively, to apply an affine transform to point Q first; and then evolve

$$\begin{aligned} X_2 &= (a_1x(u,v) + b_1y(u,v) + c_1z(u,v) + d_1) \otimes f(u,v) \\ Y_2 &= (a_2x(u,v) + b_2y(u,v) + c_2z(u,v) + d_2) \otimes f(u,v) \\ Z_2 &= (a_3x(u,v) + b_3y(u,v) + c_3z(u,v) + d_3) \otimes f(u,v) \end{aligned}$$

so $X_2 = X_1$, $Y_2 = Y_1$ and $Z_2 = Z_1$. Affine also includes shape preserving transform.

Theorem 2 Let $\mathcal{S}$ be a closed surface and let $\mathcal{H}$ be its convex hull. $\mathcal{S}$ remains inside $\mathcal{H}$ during evolution.

Proof Since $\mathcal{H}$ is a convex surface, every plane T tangent to $\mathcal{H}$ contains that surface in the left (or right) half-space it creates. Since $\mathcal{S}$ is inside $\mathcal{H}$, $\mathcal{S}$ is also contained in the same half-space. Now rotate T and $\mathcal{S}$ so that T becomes parallel to the xy plane. T is now described by the equation $z = c$. Since T does not intersect $\mathcal{S}$,

it follows that $Q_z \geq c$ for every point Q on $\mathcal{S}$. Let $\mathcal{S}_\sigma$ be an evolved version of $\mathcal{S}$. Every point of $\mathcal{S}_\sigma$ is a weighted average of a subset of points of $\mathcal{S}$. Therefore, $R_z \geq c$ for every point R on $\mathcal{S}_\sigma$, and $\mathcal{S}_\sigma$ is also contained in the same half-space. This result holds for every plane tangent to $\mathcal{H}$; therefore $\mathcal{S}_\sigma$ is contained inside the intersection of all the left (or right) half-spaces created by the tangent planes of $\mathcal{H}$. It follows that $\mathcal{S}_\sigma$ is also inside $\mathcal{H}$.

Theorem 3 Iterative Gaussian filtering of a surface converges to the solution of the heat diffusion equation.

Proof Let ϵ be the maximum error in the location of any point of surface $\mathcal{S}$ when the heat diffusion [138] of $\mathcal{S}$ is approximated through Gaussian filtering with standard deviation $\Delta\sigma$. To observe that at a point P of $\mathcal{S}$

$$\epsilon = |(\mathbf{r} + H\mathbf{n}) - (\mathbf{r} + \Delta\mathbf{r}_g)| = |H\mathbf{n} - \Delta\mathbf{r}_g|$$

where H is mean curvature, $\mathbf{n}$ is the normal vector at P, $\mathbf{r}$ is the position vector of P and $\Delta\mathbf{r}_g$ is the amount of change in the position vector of P after Gaussian filtering. According to heat diffusion equation

$$\frac{\partial \mathbf{r}}{\partial t} = H\mathbf{n} \tag{4.6}$$

where t is time. Let

$$\Delta\mathbf{r}_g = H_g\mathbf{n}_g$$

where $\mathbf{n}_g$ is a unit vector with the same direction as that of $\Delta\mathbf{r}_g$, and H_g is equal to length of $\Delta\mathbf{r}_g$. Let k_1 and k_2 be the principal curvatures at P. Assume that k_1 and k_2 are constant in a small neighbourhood of P. The following cases can be distinguished:

- k_1 and k_2 are both zero: the surface is locally planar.
- One of k_1 and k_2 is zero: the surface is locally cylindrical.
- k_1 and k_2 are either both positive or both negative: the surface is locally ellipsoidal.

- One of k_1 and k_2 is positive and the other is negative: the surface is locally saddle-shaped.

In each case, it can be confirmed that Gaussian filtering is equivalent to diffusion smoothing of the surface. It follows that for a small $\Delta\sigma$, $H_g \to H$ and $\mathbf{n}_g \to \mathbf{n}$, and therefore $\epsilon \to 0$. After i iterations of smoothing, total error is given by $i\epsilon$ which is also small.

Theorem 4 Let $\mathcal{S}$ be a 3-D surface in C^2. Let $\mathcal{S}_\sigma$ be an evolved version of $\mathcal{S}$ with a cusp point at P. There is a $\delta > 0$ such that $\mathcal{S}_{\sigma-\delta}$ intersects itself in a neighbourhood of point P.

Proof It follows from the Equation (4.6) for heat diffusion that for two points with infinitesimal distance on $\mathcal{S}$ [203], application of infinitesimal diffusion will result in two new points also with infinitesimal distance. This is because at two nearby points P_1 and P_2, $H_1 \approx H_2$ and $\mathbf{n}_1 \approx \mathbf{n}_2$ so

$$\frac{\partial \mathbf{r}_1}{\partial t} \approx \frac{\partial \mathbf{r}_2}{\partial t}$$

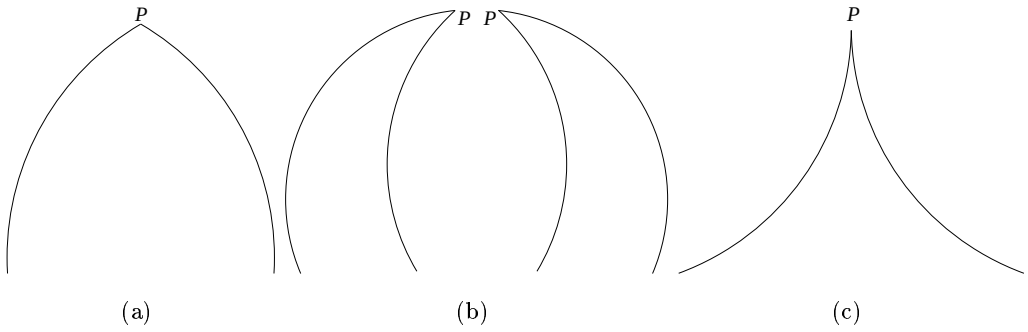


Figure 4.4: *Cross-section of the surface*

It follows further that the tangent planes $T(P_1)$ and $T(P_2)$ will also be at infinitesimal distance. Now suppose there is a cusp point on $\mathcal{S}_\sigma$ at point P. This cusp point does not exist on the original surface. Consider two points P_1 and P_2 on $\mathcal{S}_{\sigma-\delta}$ in a small neighbourhood of P. The cusp point at P on $\mathcal{S}_\sigma$ cannot be of the forms shown in

Figures 4.4(a) and (b), since the difference between the tangent planes at P_1 and P_2 on $\mathcal{S}_\sigma$ would be large (which is not possible). It follows that only cusp points of the form shown in Figure 4.4(c) are possible since only in this case, is the difference between tangent planes at P_1 and P_2 near P small. Applying reverse diffusion to this object in a neighbourhood of P results in a surface that intersects itself near P. It follows that $\mathcal{S}_{\sigma-\delta}$ is self-intersecting in a neighbourhood of the cusp point.

Theorem 5 Simple (not self-intersecting) surfaces remain simple during evolution.

Proof Assume by contradiction that $\mathcal{S}$ is a simple surface that intersects itself during evolution. The location vector of each point of $\mathcal{S}$ is a continuous function of σ during evolution; therefore $\mathcal{S}$ must touch itself at point P before self-intersection. Let $\mathcal{S}_{\sigma_0}$ be such a surface. Consider two neighbourhoods $\mathcal{S}_1$ and $\mathcal{S}_2$ of $\mathcal{S}_{\sigma_0}$ that only have point P in common. Hence $\mathcal{S}_1$ and $\mathcal{S}_2$ are non-overlapping. Note that $\mathcal{S}_1$ and $\mathcal{S}_2$ have the same tangent plane at P. Denote this tangent plane by $T(P)$. The tangent plane exists since it follows from Theorem 4 that P cannot be a cusp point on either $\mathcal{S}_1$ or $\mathcal{S}_2$ since $\mathcal{S}_\sigma$ does not self-intersect for $\sigma \leq \sigma_0$. Recall that the infinitesimal movement during arc length evolution of each point of $\mathcal{S}_1$ and $\mathcal{S}_2$ is determined by the heat diffusion Equation (4.6). Therefore, during arc length evolution, every point will move in the direction of the normal vector by an amount equal to the curvature at that point. Similarly, during reverse arc length evolution, every point will move in the opposite direction of the normal vector by an amount equal to the curvature at that point. It follows that if $\mathcal{S}_1$ and $\mathcal{S}_2$ are on opposite sides of $T(P)$, after an infinitesimal amount of reverse arc length evolution they will intersect. This is a contradiction of the assumption that $\mathcal{S}$ is simple before touching itself. Assume then that $\mathcal{S}_1$ and $\mathcal{S}_2$ are on the same side of $T(P)$. Note that $\mathcal{S}_1$ and $\mathcal{S}_2$ cannot be overlapping. Since they would still be overlapping after an infinitesimal amount of reverse arc length evolution, which is also a contradiction of the assumption that $\mathcal{S}$ is simple before touching itself. Let $\mathcal{S}_1$ be the segment inside $\mathcal{S}_2$, i.e., the tangent to $\mathcal{S}_2$ always has $\mathcal{S}_1$ to the same side. It can be seen that $\mathcal{S}_1$ has a larger curvature at P than $\mathcal{S}_2$. Therefore, after an infinitesimal amount of reverse arc length evolution, point P on $\mathcal{S}_1$ and point P on $\mathcal{S}_2$ will move in the same direction, but point P on

$\mathcal{S}_1$ will move by a larger amount. It follows that after an infinitesimal amount of reverse arc length evolution, $\mathcal{S}_1$ and $\mathcal{S}_2$ will intersect, which is again a contradiction. It follows that $\mathcal{S}$ remains simple during arc length evolution [203].

4.3 Curvature Estimation

This section presents techniques for accurate estimation of Gaussian and mean curvatures at multiple scales on smoothed free-form 3-D surfaces. Differential geometry provides several measures of curvature, which include Gaussian and mean curvatures [123, 44]. Initiated from the principal curvatures (see Appendix A.2), k_1 and k_2 at a point P of a surface are the largest and smallest values of normal curvatures for all directions at P. The product of the principal curvatures at a point P of a surface equals the Gaussian curvature of the surface at that point

$$K = k_1 k_2 \tag{4.7}$$

Similarly, the arithmetic mean of the principal curvatures at a point P of a surface equals the mean curvature of the surface at that point

$$H = \frac{k_1 + k_2}{2} \tag{4.8}$$

Consider a local parametric representation of a 3-D surface with coordinates u and v at Equations (4.1) and (4.2), Gaussian curvature K exists at regular points of a surface of class C^2. When $\mathbf{r}(u, v)$ corresponds to semigeodesic coordinates, K is given by

$$K = \frac{b_{uu} b_{vv} - {b_{uv}}^2}{x_v^2 + y_v^2 + z_v^2} \tag{4.9}$$

where subscripts denote partial derivatives, and

$$\begin{aligned} b_{uu} &= \frac{A x_{uu} + B y_{uu} + C z_{uu}}{\sqrt{A^2 + B^2 + C^2}} \\ b_{vv} &= \frac{A x_{vv} + B y_{vv} + C z_{vv}}{\sqrt{A^2 + B^2 + C^2}} \\ b_{uv} &= \frac{A x_{uv} + B y_{uv} + C z_{uv}}{\sqrt{A^2 + B^2 + C^2}} \end{aligned}$$

in which

$$A = y_u z_v - z_u y_v$$

$$B = x_v z_u - z_v x_u$$

$$C = x_u y_v - y_u x_v$$

Mean curvature H also exists at regular points of a surface of class C^2. Again, when $\mathbf{r}(u,v)$ corresponds to semigeodesic coordinates, H is given by

$$H = \frac{b_{vv} + (x_v^2 + y_v^2 + z_v^2) b_{uu}}{2(x_v^2 + y_v^2 + z_v^2)} \tag{4.10}$$

The mathematical properties of this two surface curvature functions are now discussed in more detail. Both Gaussian and mean curvature values are direction-free quantities. They are invariant to arbitrary transformation of the (u,v) parameters as well as rotations and translations of a surface. Combination of these curvature measures enables the local surface type to be categorized. On smoothed surfaces of 3-D objects, the variables for estimating the Gaussian and mean curvatures for each point (i.e. $P(x(u,v), y(u,v), z(u,v))$) in Equation (4.2) of the surface are defined as follows:

$$\begin{aligned}
x_u &= x(u,v) \otimes \frac{\partial G(u,v,\sigma)}{\partial u}, & x_{uu} &= x(u,v) \otimes \frac{\partial^2 G(u,v,\sigma)}{\partial u^2}, \\
y_u &= y(u,v) \otimes \frac{\partial G(u,v,\sigma)}{\partial u}, & y_{uu} &= y(u,v) \otimes \frac{\partial^2 G(u,v,\sigma)}{\partial u^2}, \\
z_u &= z(u,v) \otimes \frac{\partial G(u,v,\sigma)}{\partial u}, & z_{uu} &= z(u,v) \otimes \frac{\partial^2 G(u,v,\sigma)}{\partial u^2}, \\
x_v &= x(u,v) \otimes \frac{\partial G(u,v,\sigma)}{\partial v}, & x_{vv} &= x(u,v) \otimes \frac{\partial^2 G(u,v,\sigma)}{\partial v^2}, \\
y_v &= y(u,v) \otimes \frac{\partial G(u,v,\sigma)}{\partial v}, & y_{vv} &= y(u,v) \otimes \frac{\partial^2 G(u,v,\sigma)}{\partial v^2}, \\
z_v &= z(u,v) \otimes \frac{\partial G(u,v,\sigma)}{\partial v}, & z_{vv} &= z(u,v) \otimes \frac{\partial^2 G(u,v,\sigma)}{\partial v^2}, \\
& & x_{uv} &= x(u,v) \otimes \frac{\partial^2 G(u,v,\sigma)}{\partial u \partial v}, \\
& & y_{uv} &= y(u,v) \otimes \frac{\partial^2 G(u,v,\sigma)}{\partial u \partial v}, \\
& & z_{uv} &= z(u,v) \otimes \frac{\partial^2 G(u,v,\sigma)}{\partial u \partial v}
\end{aligned} \tag{4.11}$$

where the corresponding local neighbourhood data $x(u,v)$, $y(u,v)$ and $z(u,v)$ at each point are convolved with the partial derivatives of the 2-D Gaussian function

$G(u, v, \sigma)$. Finally, curvature values on a 3-D surface are estimated by substituting these values into Equations (4.9) and (4.10) respectively.

4.4 Torsion Estimation

Generally, 3-D surfaces are represented by 3-D meshes that may have a large number of points. The process of storage and computation of such a surface could cause enormous problems in term of large memory size and long processing time. Many 3-D objects have flat and curved surfaces; the zero crossing curvature contours of these objects are space curves that are geometric invariant and stable. This important local feature can be used to recognize any surface which is either complete or occluded. Therefore, the multi-scale torsion-based shape features for space curves are considered here.

Torsion is the instantaneous rate of change of the osculating plane with respect to the arc length parameter. The osculating plane at a point P is defined to be the plane with the highest order of contact with the curve at P (see Definition 5 in Section 4.1.1). Applying the Frenet's trihedron for this space curve (also see Appendix A.1), torsion τ is a local measure of the non-planarity of a space curve. The set of points of a space curve are the values of a continuous, vector-valued, locally one-to-one function

$$\Gamma = \Gamma(u) = (x(u), y(u), z(u)) \tag{4.12}$$

where $x(u)$, $y(u)$ and $z(u)$ are the components of $\mathbf{r}(u)$, and u is a function of arc length of the curve. The smoothed curve is defined as

$$\Gamma_\sigma = \Gamma_\sigma(u) = (\mathcal{X}(u, \sigma), \mathcal{Y}(u, \sigma), \mathcal{Z}(u, \sigma))$$

in which

$$\begin{aligned} \mathcal{X}(u, \sigma) &= x(u) \otimes g(u, \sigma) \\ \mathcal{Y}(u, \sigma) &= y(u) \otimes g(u, \sigma) \\ \mathcal{Z}(u, \sigma) &= z(u) \otimes g(u, \sigma) \end{aligned}$$

and the 1-D Gaussian function is

$$g(u,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-u^2}{2\sigma^2}} \tag{4.13}$$

In order to compute torsion τ at each point of the curve, it is necessary to express this in terms of the derivatives of $x(u)$, $y(u)$ and $z(u)$ [118]. In the case of an arbitrary parametrization, torsion can be given by

$$\tau(u) = \frac{\dot{x}(\ddot{y}\breve{z} - \ddot{z}\breve{y}) - \dot{y}(\ddot{x}\breve{z} - \ddot{z}\breve{x}) + \dot{z}(\ddot{x}\breve{y} - \ddot{y}\breve{x})}{(\dot{y}\ddot{z} - \dot{z}\ddot{y})^2 + (\dot{z}\ddot{x} - \dot{x}\ddot{z})^2 + (\dot{x}\ddot{y} - \dot{y}\ddot{x})^2} \tag{4.14}$$

where $\dot{x}$, $\dot{y}$ and $\dot{z}$ are the convolutions of $x(u)$, $y(u)$ and $z(u)$ in Equation (4.12) with the first derivative of a 1-D Gaussian function (4.13).

$$\begin{aligned}
\dot{x} &= x(u) \otimes \frac{\partial g(u,\sigma)}{\partial u}, \quad & \ddot{x} &= x(u) \otimes \frac{\partial^2 g(u,\sigma)}{\partial u^2}, \quad & \breve{x} &= x(u) \otimes \frac{\partial^3 g(u,\sigma)}{\partial u^3}, \\
\dot{y} &= y(u) \otimes \frac{\partial g(u,\sigma)}{\partial u}, \quad & \ddot{y} &= y(u) \otimes \frac{\partial^2 g(u,\sigma)}{\partial u^2}, \quad & \breve{y} &= y(u) \otimes \frac{\partial^3 g(u,\sigma)}{\partial u^3}, \\
\dot{z} &= z(u) \otimes \frac{\partial g(u,\sigma)}{\partial u}, \quad & \ddot{z} &= z(u) \otimes \frac{\partial^2 g(u,\sigma)}{\partial u^2}, \quad & \breve{z} &= z(u) \otimes \frac{\partial^3 g(u,\sigma)}{\partial u^3}
\end{aligned} \tag{4.15}$$

$\otimes$ denotes convolution. Note that $\ddot{x}$, $\ddot{y}$, $\ddot{z}$, $\breve{x}$, $\breve{y}$ and $\breve{z}$ represent convolutions with the second and third derivatives of Equation (4.13), respectively. While derivative estimation can be sensitive to noise, thus torsion estimation takes place only after sufficient smoothing has been applied to the data, and is therefore a robust process. This also helps to reduce the number of feature points used for matching later on.

Once Gaussian and mean curvatures have been determined at each point of a 3-D surface, zero crossing contours of those curvatures are also recovered from that surface. In general, these contours are space curves. Torsion is computed at each point of the contours using Equation (4.14), and now the local maxima of absolute values of torsion are then recovered. These points are added to the set of feature points extracted from the surface for object recognition. Since torsion has invariance, stability, local support, efficiency and ease of implementation, the surfaces represented by these characteristics could therefore benefit from such compact and robust information.

4.5 Geometric Hashing

The geometric hashing technique for model based object recognition was introduced by Lamdan and Wolfson [93, 92]. Stein and Medioni [172] and Flynn and Jain [39] have also employed geometric hashing for 3-D object recognition. In a model based object recognition system one has to address representation and matching problems. The representation should be rich enough to allow reliable distinction between the different object models in the database as well as for efficient matching. The information in the database includes the number of models, model name, number of features, feature magnitudes and positions. A major factor in a reliable representation scheme is its ability to deal with partial occlusion. The objects are represented as sets of geometric features such as points, and their geometric relations are encoded using minimal sets of such features under the allowed object transformations.

4.5.1 Hash Table

After getting the invariant features from a 3-D object, the next task is to recognize this object. However, it is not that simple to recognize a free-form 3-D object from a complex scene. For example, m interest features from an object model and n interest features from a scene will require $n \times m$ ways to match. This is unacceptable for object recognition when the number of object models in a database is large. Instead, geometric hashing, an efficient and cost effective approach is used for the model-base recognition of 3-D objects. It uses the invariant features including the local maxima of absolute values of Gaussian curvature, mean curvature and torsion. It is also based upon an off-line model learning pre-processing technique, where model information is indexed into a hash table using the invariant features. Geometric hashing involves voting of observed features and this enables occluded or partial 3-D surfaces to be recognized. The basic elements of geometric hashing are the hash table and hash addressing algorithm (hashing function/hash code/multi-words key transformation) [141, 68]. Let there be $\mathcal{N}$ hash table entries. They contain $(\mathcal{K}_i, \mathcal{D}_i)$ where $\mathcal{K}_i$ represents the triplet curvatures k_a, k_b, k_c and $\mathcal{D}_i$ represents the triplet distance ratios among d_1, d_2, d_3 in Figure 4.5. They are the invariant feature data

for the i^{th} entry ($i = 1, 2, 3, \ldots, \mathcal{N}$). Each pair of $\mathcal{K}_i$ and $\mathcal{D}_i$ is used to produce an address $\mathcal{A}_i = f(\mathcal{K}_i, \mathcal{D}_i)$. The hashing function $f(\mathcal{K}_i, \mathcal{D}_i)$ is given as follows:

$$f(\mathcal{K}_i, \mathcal{D}_i) = a\mathcal{K}_i + b\mathcal{D}_i \tag{4.16}$$

where a and b are constants. The development of such a multi-word key transformation has greatly improved the speed of searching and matching in the database. It also makes geometric hashing an extremely fast method for object recognition.

4.5.2 Matching

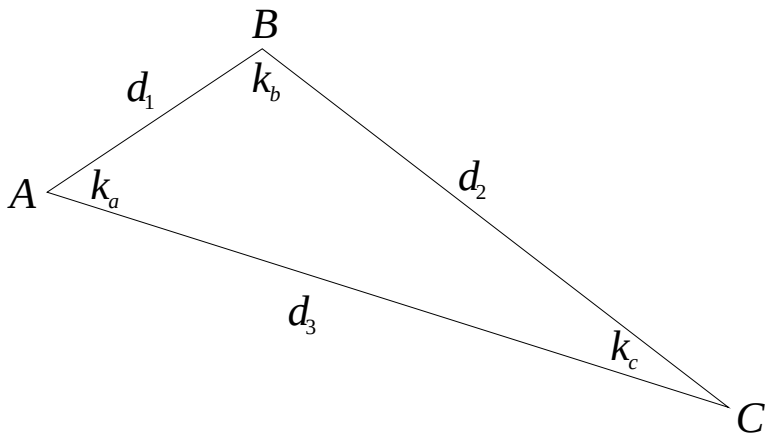


Figure 4.5: *Triplet of non-collinear points A, B and C*

The matching algorithm uses the hash table prepared in the off-line stage. Given a scene of feature points, one tries to match the measurements taken at scene points to those stored in the hash table [4]. On smoothed surfaces of 3-D objects, the procedures for indexing data into the hash table are defined as follows:

1. For each 3-D object model in the database, the local maxima of absolute values of Gaussian curvature are selected as one set of feature points. The local maxima of absolute values of mean curvature are selected as another set of feature points. Similarly, the local maxima of absolute values of torsion of zero crossing contours of Gaussian and mean curvatures are also used as another two sets of feature points.

2. Choose an arbitrary ordered triplet of non-collinear points A, B and C to form a triangle ABC. Denote the curvature/torsion values of points A, B and C by k_a, k_b and k_c, and the edge lengths AB, BC and CA as d_1, d_2 and d_3, respectively (see Figure 4.5). Select the maximum curvature/torsion value and maximum edge length to generate an indexed value V_i for the hash table. For example, if k_a and d_3 are the maximum curvature/torsion value and maximum edge length, then calculate the indexed value V_i as

$$V_i = \frac{k_b + k_c}{k_a} \times \frac{d_1 + d_2}{d_3} \tag{4.17}$$

and using the hashing function from Equation (4.16) for generating multi-words key $\mathcal{A}_i$ to create an evenly spread hash table [141],

$$\mathcal{A}_i = 10 \times \frac{k_b + k_c}{k_a} + \frac{d_1 + d_2}{d_3} \tag{4.18}$$

with $a = 10$ and $b = 1$.

3. Go back to step 2 and repeat the procedure for different triplets of feature points until all triplets are visited. Note that some of these newly selected points may have already been chosen in previous selections. Now a hash table is produced with all the data indexed into its memory from a database. Given a scene of features from a 3-D object, the indexed value V_i as well as its individual features are matched to those memorized in the hash table. Notice that the input 3-D object can either be complete or incomplete.

Thus, given a 3-D object in a scene, the following steps are defined for matching that object to the models database:

4. Repeat steps 1 to 3 above with the scene object, and then for each multi-words key $\mathcal{A}_i$ and indexed value V_i, check the appropriate entry data and features in the hash table and database. Tally a vote for each model that appears.

5. If several object models score large number of votes close to each other, then the most likely candidates will be chosen using global verification applied at next stage.

4.6 Global Verification

In general the voting scheme may give more than one solution with very close scores. In this case, a number of models from the database with close high scores (i.e. a threshold is used to select the most likely models) are selected for global verification. Global verification requires the estimation of 3-D transformation parameters for the surviving models. It is possible to make use of closed form solution techniques [65, 66, 188, 36] to obtain these parameters. However, these techniques are quite complex to implement and relatively inefficient. Considering that the estimation procedure must be repeated many times, it is advantageous to use a method which is as efficient as possible. A relatively simple and efficient technique that generates approximate solutions has been developed. It is found that this is quite satisfactory for this research [122].

For each of selected object models, seven global transform parameters including scaling ($\mathcal{D}$), translation (d, h, q) and rotation (γ, β, α) are estimated and compared [41, 163, 195, 94, 15, 202]. These 3-D coordinate transformations are formulated in terms of the feature coordinates $PT = Q$ where P is the coordinates for the object model feature points, T is the transformation matrix and Q is the coordinates for the scene object feature points. This general transformation matrix consists of arbitrary amounts of scaling, translation and rotation.

From three points of the object as described in Figure 4.5, another point can be obtained as the centroid [133] of these 3 points in the space. Let $P_1(x_{1p}, y_{1p}, z_{1p})$, $P_2(x_{2p}, y_{2p}, z_{2p})$ and $P_3(x_{3p}, y_{3p}, z_{3p})$ be the 3 non-collinear points selected from the model object and $P_4(x_{4p}, y_{4p}, z_{4p})$ be a point in the 3-D space which is the centroid of P_1, P_2 and P_3. A plane P in the space from points P_1, P_2 and P_3 is formed. Now the same procedure is applied to the object in the scene. Let $Q_1(x_{1q}, y_{1q}, z_{1q})$, $Q_2(x_{2q}, y_{2q}, z_{2q})$, $Q_3(x_{3q}, y_{3q}, z_{3q})$ and $Q_4(x_{4q}, y_{4q}, z_{4q})$ be the points in the 3-D space. Point Q_4 is the centroid of points Q_1, Q_2 and Q_3, thus a plane Q is also formed. Hence, the linear equations [118] for the transformation, mapping model points to

scene points are given by

$$\begin{bmatrix} x_{1p} & y_{1p} & z_{1p} & 1 \\ x_{2p} & y_{2p} & z_{2p} & 1 \\ x_{3p} & y_{3p} & z_{3p} & 1 \\ x_{4p} & y_{4p} & z_{4p} & 1 \end{bmatrix} \begin{bmatrix} a & e & m \\ b & f & n \\ c & g & p \\ d & h & q \end{bmatrix} = \begin{bmatrix} x_{1q} & y_{1q} & z_{1q} \\ x_{2q} & y_{2q} & z_{2q} \\ x_{3q} & y_{3q} & z_{3q} \\ x_{4q} & y_{4q} & z_{4q} \end{bmatrix} \tag{4.19}$$

Note that this approach is employed in order to obtain a quick and approximate solution which is sufficient for verification. From the set of linear Equations (4.19), one can solve for the twelve parameters a, b, c, d, e, f, g, h, m, n, p and q. In order to verify a match, 3-D scaling, translation and rotation will be used to determine global consistency. For the scaling factor, the distances from the centroid points P_4 and Q_4 to their corresponding 3 points are measured and the longest distances for each triplet are selected. Let D_1 and D_2 be the longest distances selected from the model object and the scene object, then their ratio is the scaling factor $\mathcal{D}$

$$\mathcal{D} = \frac{D_1}{D_2} \tag{4.20}$$

The translation parameters d, h and q can be measured from the distance between two centroids P_4 and Q_4. Now for rotation, let γ, β and α be the angles in the x, y and z directions for the rotation of the plane P to the plane Q in 3-D space. The 3-D rotation matrices about x-axis, y-axis and z-axis denoted $R_x(\gamma)$, $R_y(\beta)$ and $R_z(\alpha)$, respectively, are given by

$$R_x(\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\gamma & -sin\gamma \\ 0 & sin\gamma & cos\gamma \end{bmatrix}$$

$$R_y(\beta) = \begin{bmatrix} cos\beta & 0 & sin\beta \\ 0 & 1 & 0 \\ -sin\beta & 0 & cos\beta \end{bmatrix}$$

$$R_z(\alpha) = \begin{bmatrix} cos\alpha & -sin\alpha & 0 \\ sin\alpha & cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The columns (and the rows) of matrices $R_x(\gamma)$, $R_y(\beta)$ and $R_z(\alpha)$ are mutually perpendicular unit vectors and they have determinant of 1, so they are orthogonal. Therefore, the rotation parameters (γ, β, α) can be obtained from products of $R_z(\alpha)$ $R_y(\beta)$ $R_x(\gamma)$ [163, 41] and also from the solution of Equation (4.19)

$$\begin{bmatrix} a \\ e \\ m \\ n \\ p \end{bmatrix} = \begin{bmatrix} cos\alpha(cos\beta) \\ sin\alpha(cos\beta) \\ -sin\beta \\ cos\beta(sin\gamma) \\ cos\beta(cos\gamma) \end{bmatrix} \tag{4.21}$$

Since a number of model objects with close high scores are selected for the global verification stage, the hash table yields many candidate matches for each selected model. For each of these candidates, seven global transform parameters are estimated using the method described earlier in this section. The candidates are compared and if their corresponding parameters are compatible, they are clustered together. The largest cluster then indicates the largest group of globally consistent matches for each model. The model object with the largest cluster is then chosen as the most likely object present in the scene. The clustering algorithm is quite efficient since it avoids the creation of an explicit high-dimensional parameter space. The following is a step-by-step description of the clustering algorithm:

1. Create a cluster for one of the values in the multi-dimensional parameter space. Consider that value as the centre of the cluster.

2. Find another value which is closer than a threshold to the centre of the cluster, and add that value to the cluster. If no values are added, go to step 4.

3. Compute the new centre of the cluster as the centre of mass of the values already in the cluster. Go to step 2.

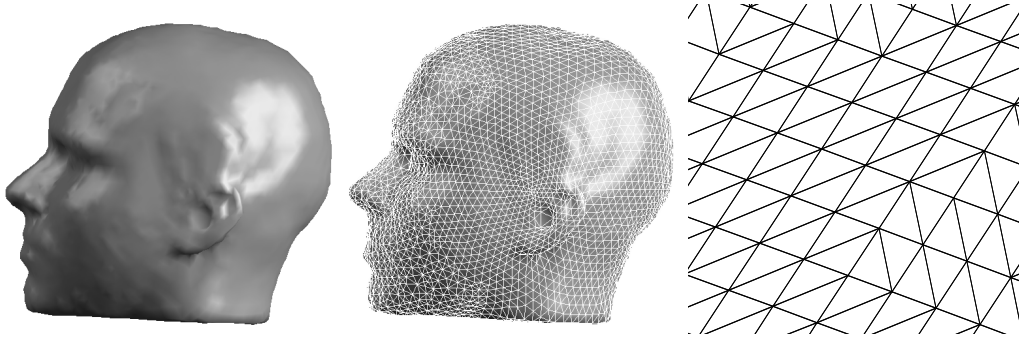
4. Repeat this procedure for all values which are not already in a cluster.

Chapter 5

Implementation

Multi-scale representation and recognition of 3-D surfaces using geometric invariants is a complex process. It requires a methodological approach to perform such a task which includes parametrization of 3-D surface [46, 108, 44, 56, 146], the application of the smoothing process to a noisy surface at different scales, feature extraction and object recognition. This chapter starts by explaining the basic structure of 3-D surfaces and data preparation. It then shows the implementation of local parametrization of 3-D surfaces, a novel step-by-step approach of arbitrary geodesic line construction is outlined. It is followed by the creation of the second family of lines that involves either using orthogonal trajectories to the arbitrary geodesic line or applying the modulus and amplitude at a local patch. Such a procedure leads to the construction of semigeodesic coordinates or geodesic polar coordinates [120, 44] on a free-form 3-D surface. Then a powerful and robust multi-scale smoothing technique [121, 204, 82, 128] is used to eliminate the unwanted noise on the surface and at the same time to produce the useful features such as the local maxima of absolute values of Gaussian curvature, mean curvature and torsion [203, 205, 83]. After the features are extracted, this chapter will then present the recognition of 3-D surfaces using geometric hashing and global verification [122]. The thresholds are implemented automatically by software.

5.1 Surface Structure



(a) This human head is represented by a 3-D surface

(b) Its corresponding triangulated mesh

(c) A magnified patch of the triangulated mesh which is shown in (b)

Figure 5.1: *A free-form 3-D surface and its triangulated mesh*

As mentioned in previous chapters, the technique for describing a 3-D multi-scale surface is independent of its underlying structure. In other words, the underlying structure can be arbitrary. Although several methods are available to model 3-D surfaces, triangulated meshes are the most popular approach because triangles are simple from the implementation point of view and they are also the most effective form of polygons for covering a free-form 3-D surface [86, 140, 149, 148]. Increasingly with the advantage of this type of representation, the technique to visualize triangulated meshes is embedded in the computer graphic hardware. Figures 5.1(b) and (c) show an example of a triangulated mesh coming from the human head in Figure 5.1(a), with each triangle defined by three vertices.

Initially, data information about the triangulated mesh is stored internally in an array. The basic data structure includes the total number of triangular vertices, the first vertex and the subsequent vertices. This is followed by the number of neighbours at each vertex and its neighbours (i.e. minimum three neighbours for triangulated mesh). These neighbouring vertices are always arranged in an anti-clockwise direction. For simplicity, the local parametrization is started from the first vertex. Hence, each vertex and its neighbourhood form a local patch that will

be eventually parametrized into the semigeodesic coordinates.

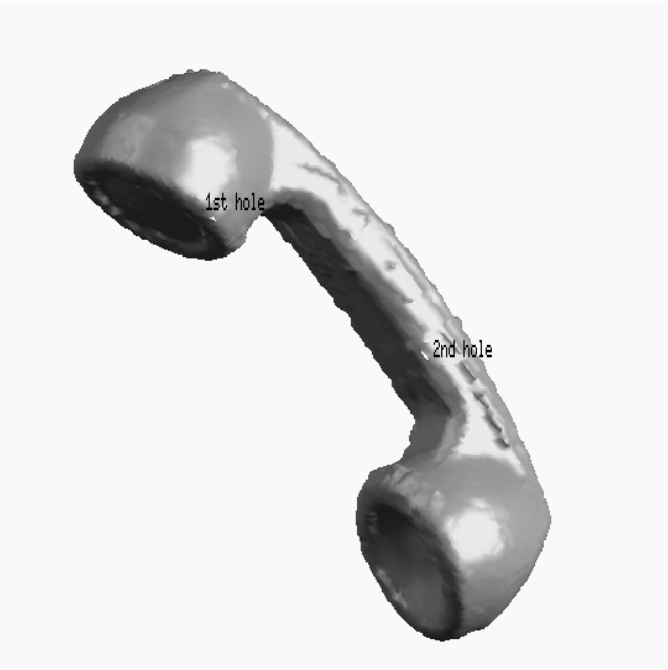
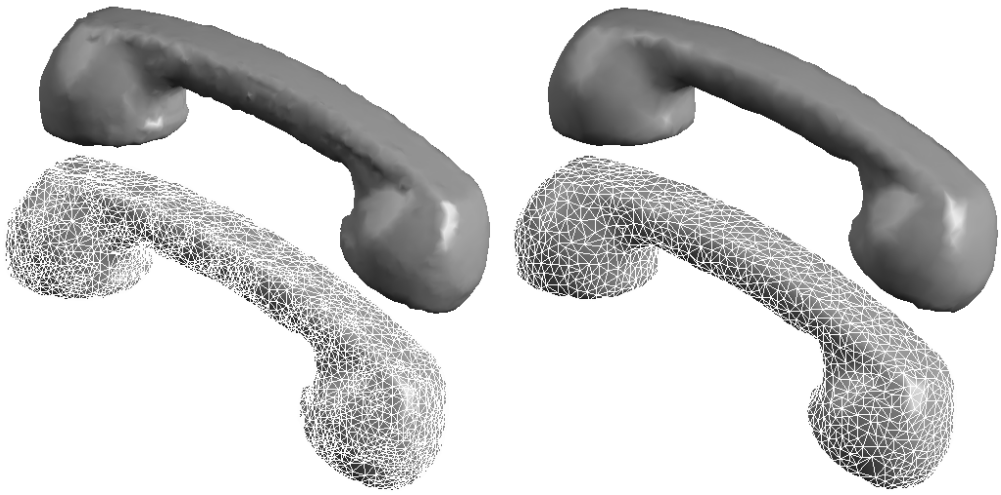


Figure 5.2: *A phone with two small holes*

During the construction and fusion [58, 169] of triangulated meshes, noise may occur within such a process. Consequently, small unwanted holes very often appear on the surface (see Figure 5.2). When these holes are reached in the surface smoothing, the area nearby cannot be filtered due to the incomplete local patch. Hence, the curvature and torsion cannot be estimated. Therefore, it is logical to cover the small holes in order to minimize such problems. The procedure for filling a hole is described as follows:

- Split a hole in half and create two sub-loops.

- Validate the sub-loops by checking each sub-loop lies on opposite sides of other sub-loop.
- Check the aspect ratio of each sub-loop.
- The subdivision of each sub-loop continues recursively until all sub-loops consist of three edges.



(a) 3-D surface of a phone and its triangulated mesh with 5564 vertices

(b) Decimated surface from (a) and its triangulated mesh with 1993 vertices

Figure 5.3: *Decimation of a 3-D surface*

Before the process of local parametrization is applied to a 3-D surface, the surface needs to be prepared in advance [78]. For example, a flat surface could be represented by a few large triangles instead of a large number of small triangles (including needle-like triangles) that occupy the same surface area. A common approach is to reduce the number of triangles on the surface by using a decimation technique [159, 62]. For example, Figure 5.3(a) shows a phone which is represented by 5564 vertices. After decimation is applied, the number of vertices for the same phone is reduced to 1993 (see Figure 5.3(b)). Obviously, this data preparation can improve the efficiency of the surface representation. The decimation algorithm used in our work on a 3-D mesh is as follows:

1. In a local region surrounding a point - the centroid of region, distance d between the region and the point is measured.

2. If the above distance d is less than a predefined threshold, all triangles within the region are deleted, leaving a hole in the 3-D mesh.

3. The newly created hole is patched by a small amount of flat triangles (see Figure 5.2) and its procedure for filling a hole.

4. The process terminates when all regions and points are visited. Otherwise, the next region and point are selected, and the algorithm returns to step 1.

When a smoothing process is applied to a 3-D surface iteratively, the size of surface will be changed because of the different normalization factors of the 2-D Gaussian filter in Equation (4.5) [167], and this leads to shrinkage [183, 137, 206]. The shrinkage is caused by the process of averaging. The simplest way to minimize these problems is to rescale the surface mesh according to its average edge length, where the scale factor is calculated as

$$\text{scale factor} = \frac{\text{original average edge length}}{\text{current average edge length}}$$

Therefore, the surface can be approximately maintained at the same overall size in metres.

5.2 Construction of Geodesic Line

In Section 4.1.1, Definition 7, the minimal property of geodesics provides the basic structure of semigeodesic coordinates at each local patch. Such a local parametrization can be constructed at each vertex of the mesh and this becomes the local origin. Starting with the construction of an arbitrary geodesic line, the edge connecting the local origin vertex and one of its neighbouring vertices is selected as the arbitrary direction. Once this direction is determined, the next step is to construct a geodesic line. This line is constructed on the local 3-D surface by following the geodesic path in a straight line until an edge or vertex is reached. The new edge point or vertex

becomes a starting point for the next extension. To continue a geodesic line into the next triangle, firstly the angle between the path and the common edge that the path has intersected is measured. Then the path is extended to the next triangle using the same angle. The construction of this geodesic line continues until the last edge or vertex of the local area is reached. The same process is repeated to construct the reverse direction of the geodesic line.

Quite often, due to occlusion, it is not possible to construct complete and closed surfaces. As a result, some local patches may be at a boundary and so the curvature and torsion cannot be estimated. Therefore, it is necessary to stop the local parametrization process when a geodesic line reaches an occluded area. It then moves onto the next vertex for next local parametrization process.

5.2.1 Arbitrary Direction of Geodesic Line

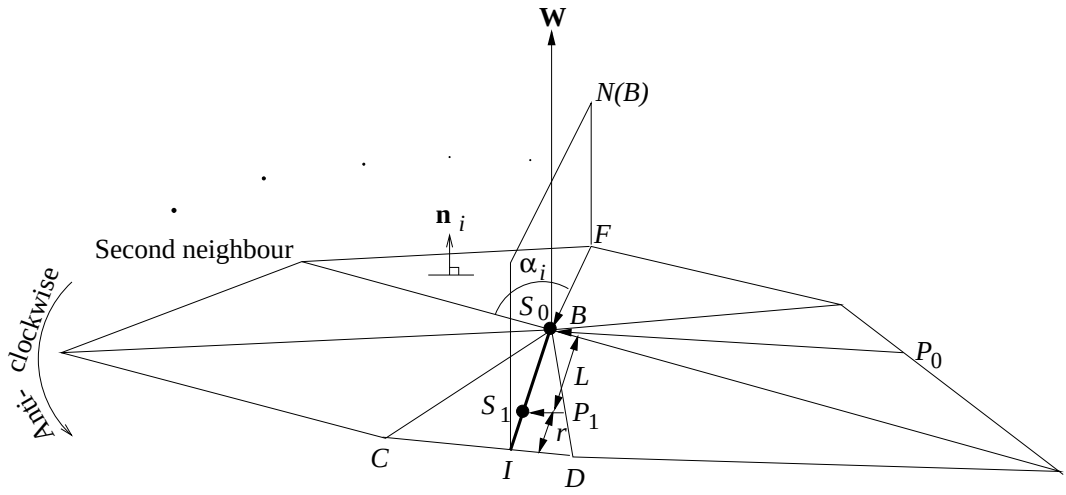


Figure 5.4: *The first segment of the arbitrary geodesic line*

Since the direction of the arbitrary geodesic line can be randomly selected, the edge FB between vertex B and the first neighbouring vertex F is chosen to create the arbitrary direction, as shown in Figure 5.4. The next step is to create the first segment of arbitrary geodesic line from the vertex B. Then the construction of the positive portion $(+u)$ of the geodesic line will follow.

At first, a weighted average normal vector $\mathbf{W}$ at vertex B is generated from the unit normal $\mathbf{n}_i$ and the vertex angle α_i of neighbouring triangles.

$$\mathbf{W} = \frac{\sum_{i=1}^{m} \alpha_i \mathbf{n}_i}{\sum_{i=1}^{m} \alpha_i} \tag{5.1}$$

where m is the number of neighbouring triangles. Therefore, a normal plane $N(B)$ is defined by the weighted average normal unit vector $\mathbf{W}$ and the arbitrary direction vector $\mathbf{FB}$. Now the intersection I between the normal plane and the opposite edge CD produces the first segment BI of the geodesic line. This segment is then sampled at equal sized intervals. The number of sample points n in this segment depends on the choice of sample step size and the length of the segment. For a filter size of 9×9, n ranges from -4 to $+4$, where n is integer and the size of sample steps is equal to the average edge length of the whole triangulated mesh. At the same time, the closest vertex at each sample point S_n and the perpendicular direction vector $\mathbf{P}_n\mathbf{S}_n$ are computed and stored for the construction of the second family of lines later on. Figure 5.4 shows that r is the remaining step for the next segment. Furthermore, the negative portion $(-u)$ of the geodesic line can be extended from edge BF by just applying the same sampling procedure. Eventually, these two portions ($+u$ and $-u$) are joined together to form a complete geodesic line.

5.2.2 Geodesic Line Extension

Figure 5.5 shows that the segment of a geodesic line which lies on any given triangle is a straight line. Thus, two situations are considered:

- Extension of a geodesic line when it intersects a triangle edge
- Extension of a geodesic line when it intersects a triangle vertex

Now the following theorems address two different situations:

Theorem 6 Suppose a geodesic line intersects an edge e shared by triangles T_1 and T_2. The extension of this geodesic line beyond e is obtained by rotating T_2 about e

so that it becomes coplanar with T_1, extending the geodesic line in a straight line on T_2, and rotating T_2 about e back to its original position.

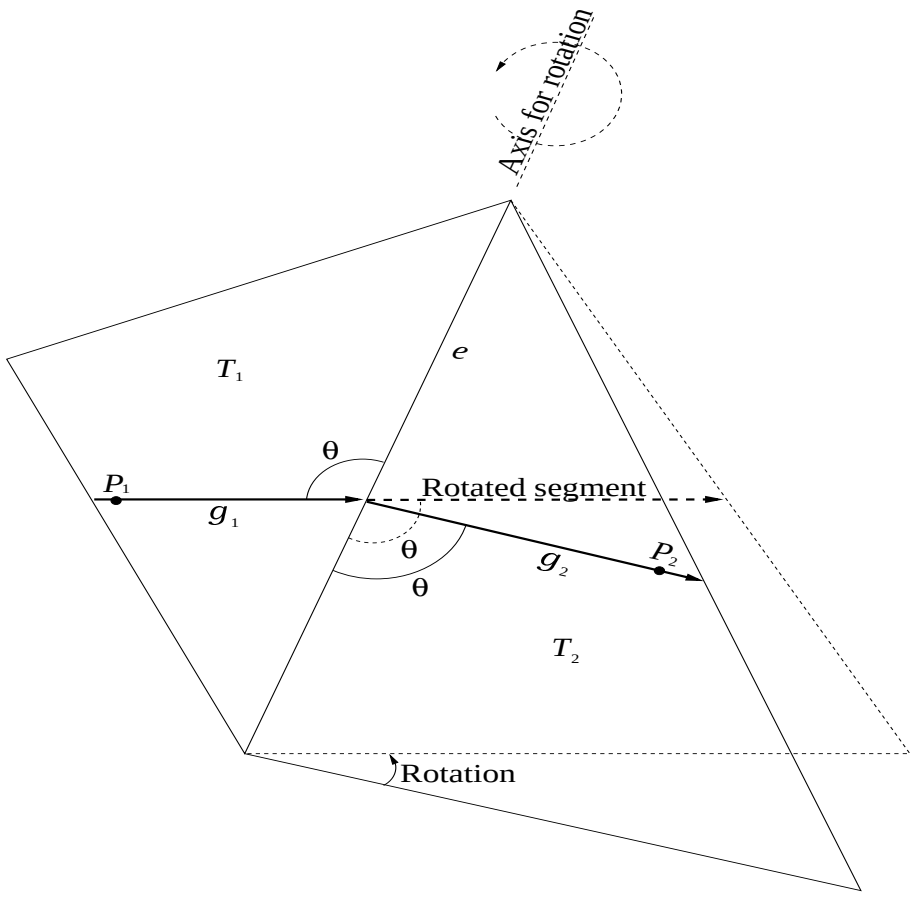


Figure 5.5: *Geodesic line on a triangulated mesh*

Proof Assuming by contradiction, the above theorem does not construct a geodesic line. Let g_1 be the segment of the geodesic line on T_1 and let g_2 be the segment of the geodesic line on T_2. Rotate T_2 about e so that it becomes coplanar with T_1. By assumption, g_1 and g_2 will not be collinear. Hence, for a point P_1 on g_1 and a point P_2 on g_2, there will be a shorter path from P_1 to P_2. This is the straight line joining P_1 to P_2. Now rotate T_2 back to its original position. The length of the path just constructed remains the same, so it will still be shorter than the geodesic line from P_1 to P_2. A contradiction has been reached. Therefore, the theorem described correctly constructs a geodesic line. Note that the angle θ between the path and the

common edge e which the path has intersected is measured. Then the path to the next triangle using the same angle is extended. The same construction technique is used to extend to several triangles as long as they remain in a local neighbourhood.

Theorem 7 Suppose that a geodesic line arrives at a vertex V of the mesh. We define the normal vector $\mathbf{W}$ at V as the average of the surface normals of all the triangles incident on V weighted by the incident angle. Let $N(V)$ be the plane formed by the geodesic incident on V and $\mathbf{W}$. The extension of this geodesic line beyond V is found by intersecting $N(V)$ with the mesh.

Proof The curvature vector $\mathbf{K}$ of the path lies in the plane $N(V)$. $\mathbf{K}$ is perpendicular to tangent plane $T(V)$ which is also defined as perpendicular to $\mathbf{W}$ at V. Since the vector of geodesic curvature $\mathbf{K}_g$ of the path is obtained by projecting $\mathbf{K}$ on the tangent plane $T(V)$, the geodesic curvature of the path must be zero. Hence, the path is a geodesic line as stated at Definition 1 in Section 4.1.1.

5.2.3 Adjustment of Arbitrary Geodesic Line

It is possible that an arbitrary geodesic line could travel near and almost parallel to an edge. It is also possible for a sample point to be very close to a vertex or edge. In these cases, the failure to find the intersection of the parallel lines or the failure to calculate the distance between very close points, may result in a computational fault during the construction of a perpendicular direction for the second family of lines. Therefore, a fine adjustment or alignment [4] to the direction of the arbitrary geodesic line or the position of a sample point is required. The criterion for these adjustments is based on the average edge length L of the entire surface. Figures 5.6(a) and (b) show that if the length of IU is very small compared with L, the segment VI and the edge VU are almost parallel. Thus the segment VI is adjusted and placed over the edge VU. Similarly, if the sample point S_1 is relatively close to the edge MK with respect to L or the sample point S_2 is very near to the vertex V, the sample point S_1 is moved towards the edge MK or the sample point S_2 is moved to the vertex V.

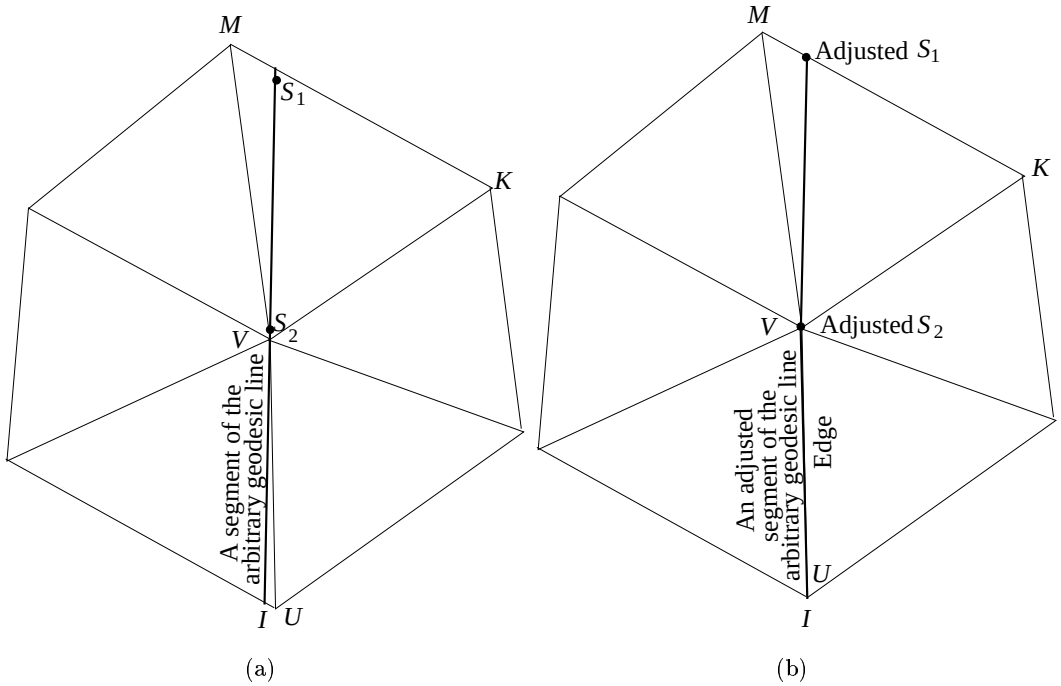


Figure 5.6: *The adjustment of the arbitrary geodesic line and the sample points*

5.2.4 To Generate the Perpendicular Direction

For generating the perpendicular direction to arbitrary geodesic line, many different conditions have to be considered. If a sample point is placed on a vertex S, a normal plane $N(S)$ can be created by the weighted average normal vector **W** at the vertex, and a segment vector **SI** of the arbitrary geodesic line, as shown in Figure 5.7. Then the perpendicular direction vector **PS** on the surface is obtained by rotating the normal plane 90° anti-clockwise from **SI**. To eliminate the unwanted intersection, the rotated normal plane must only intersect with the edge of one of the neighbouring triangles from anti-clockwise.

When a sample point of the arbitrary geodesic line resides on the edge of a triangle, three different cases arise during constructing the perpendicular direction:

- If the segment IS of the arbitrary geodesic line and the edge CD are orthogonal, a part of this edge becomes the perpendicular direction vector **PS** (see Figure 5.8(a)).

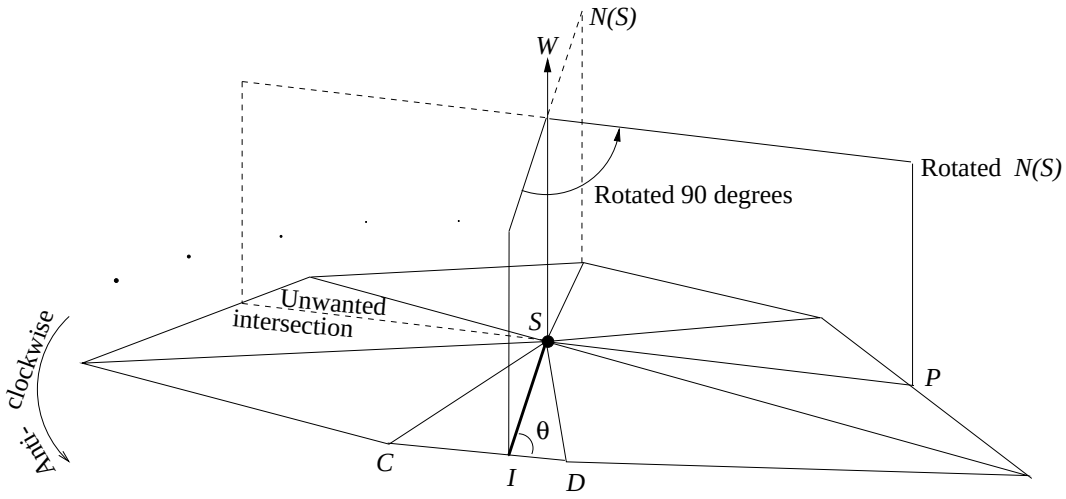


Figure 5.7: *A perpendicular direction of an arbitrary geodesic line*

- If the angle ϕ between the segment IS of the arbitrary geodesic line and the edge CD is less than 90° as shown in Figure 5.8(b), the perpendicular direction vector **PS** is produced by rotating the segment IS by 90° anti-clockwise. Then the rotated segment intersects with an edge on the same triangle CDE.

- If the angle ϕ is greater than 90°, the segment IS is rotated 90° clockwise. Then the rotated segment intersects with an edge on the same triangle CDE, it produces a segment $P'S$ of new geodesic line (see Figure 5.8(c)). Now this segment $P'S$ of the geodesic line is extended from the triangle CDE towards the triangle GDC, by using the same construction technique mentioned in Theorem 6. This newly created segment becomes the perpendicular direction vector **PS** of the arbitrary geodesic line. This method has the advantage of simplifying the construction of the perpendicular direction vector. It also produces the correct second family of lines in such a case.

Furthermore, when the segment II' of the arbitrary geodesic line lies on the common edge CD, the perpendicular direction vector **PS** at the sample point S is defined by rotating the common edge CD on the right hand side triangle GDC, 90° anti-clockwise at the sample point S (see Figure 5.9(a)). Finally, if a sample point S is inside a triangle, the perpendicular direction vector **PS** is defined by simply rotating

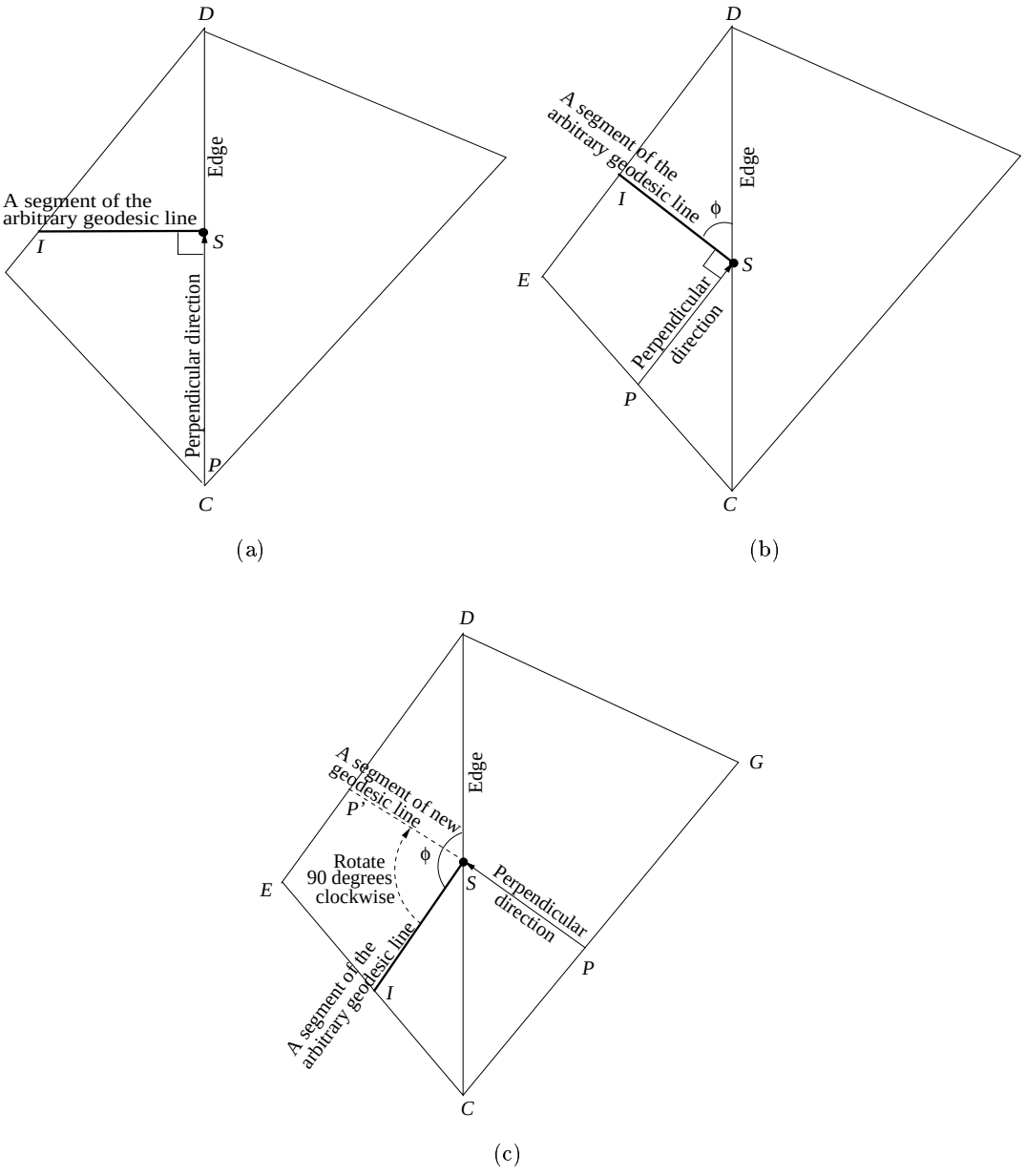
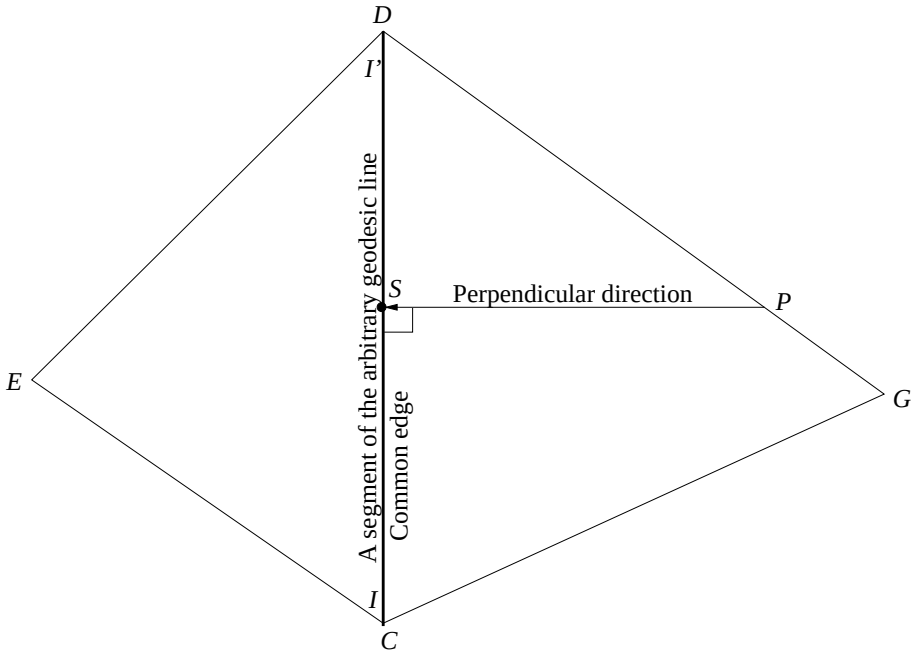
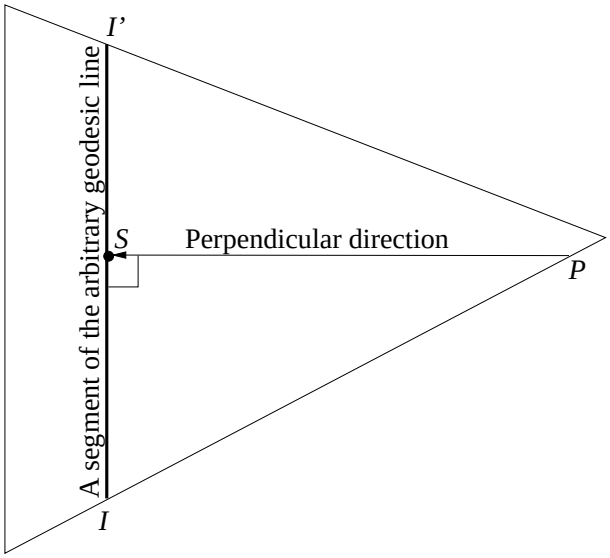


Figure 5.8: *Creation of a perpendicular direction of the arbitrary geodesic line when a sample point resides on the edge of a triangle*



(a) A segment of arbitrary geodesic line lies on the common edge of the triangles



(b) A segment of arbitrary geodesic line lies inside a triangle

Figure 5.9: *Creation of perpendicular direction from the segment of the arbitrary geodesic line*

the segment II' of the arbitrary geodesic line 90° anti-clockwise at the sample point S as shown in Figure 5.9(b).

5.3 Semigeodesic Coordinates

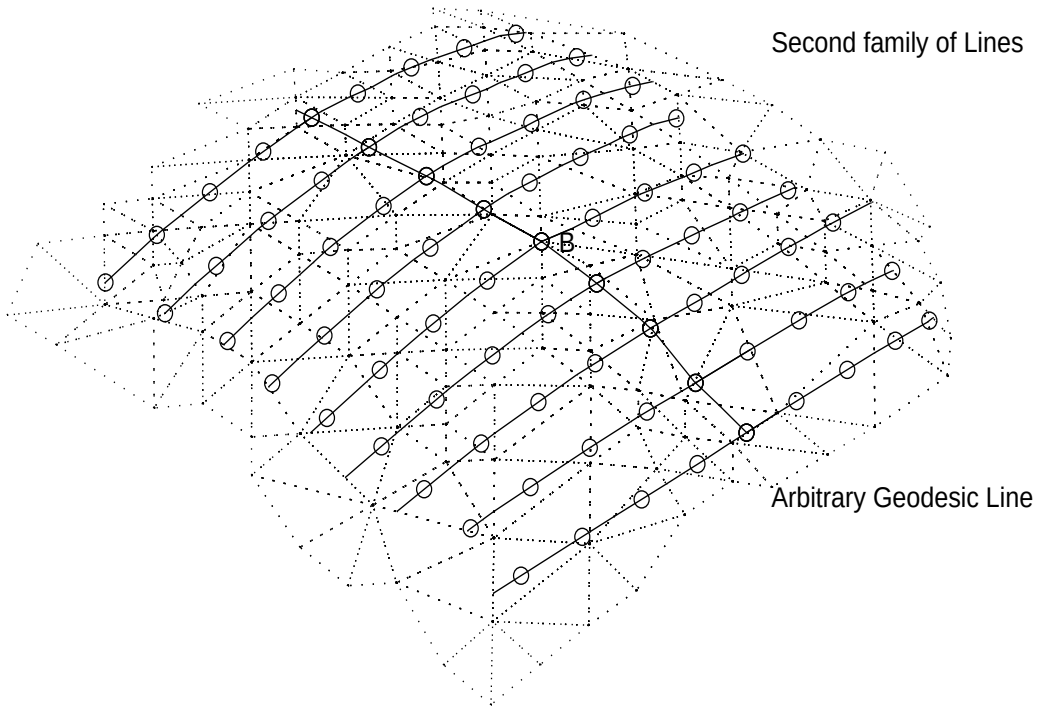


Figure 5.10: *A completed semigeodesic coordinates on a triangulated mesh*

The semigeodesic coordinates are produced by sampling the second family of geodesic lines at regular intervals. This second family of lines are constructed perpendicularly to an arbitrary geodesic line. Firstly, each sample point of the arbitrary geodesic line is used as a reference for constructing the second family of geodesic lines. Secondly, the second family of lines must start at the perpendicular direction of sample points of arbitrary geodesic line. Then, using a similar technique as described in previous Section 5.2.2, the perpendicular geodesic lines are constructed in the forward direction, as well as the backward direction. Now the following algorithm shows the complete parametrization procedure at each local patch:

1. Construct a geodesic line from the local origin in an arbitrary direction such as the direction of one of the incident edges.

2. Construct the other half of the geodesic line by extending it through the origin in the reverse direction.

3. Parametrize that geodesic line by the arc length parameter at regular intervals to obtain a sequence of sample points. The sampling interval should be proportional to the average edge length of entire surface so that the over sampling or under sampling could be minimized.

4. At each sample point on the arbitrary geodesic line, construct a perpendicular geodesic line (i.e. the second family of lines) and extend it in both directions.

5. Parametrize each of the geodesic lines constructed in the previous step by the arc length parameter at regular intervals. The sampling interval should be equal to what was used in step 3.

Figure 5.10 shows for a triangulated mesh example, the complete semigeodesic coordinates at a local origin (vertex B). Once the construction of semigeodesic coordinates for that local patch is accomplished, the algorithm moves onto the next vertex of the triangulated surface.

5.4 Geodesic Polar Coordinates

Geodesic polar coordinates can be constructed at each vertex of the mesh. The following procedure is used:

- Construct a geodesic line from the local origin in an arbitrary direction such as the direction of one of the incident edges.

- Let N be the normal plane at the origin defined by the geodesic line constructed in the previous step and the weighted average normal vector $\mathbf{W}$.

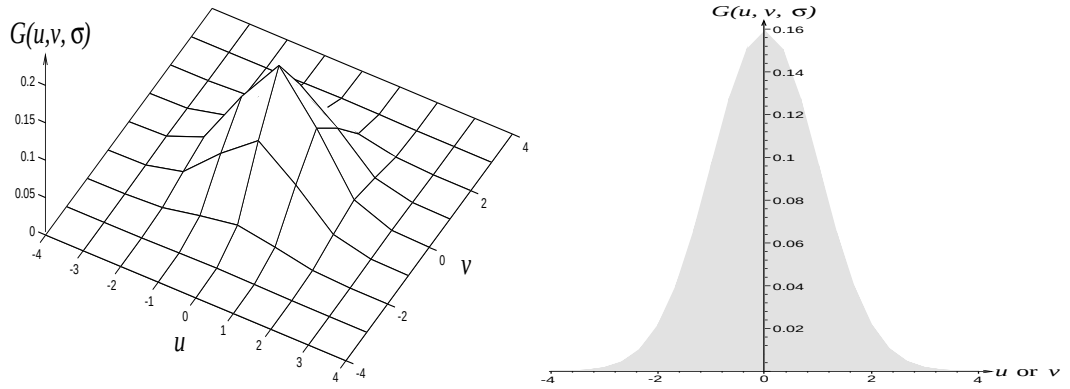
- Rotate N about $\mathbf{W}$ by an angle α and intersect it with the mesh to obtain the next geodesic line emanating from the origin.

- Repeat the previous step until N is back in its original position.
- Parametrize each of the constructed geodesic lines by the arc length parameter at regular intervals to obtain a sequence of sample points on each geodesic line. The sampling interval should be proportional to the average edge length of the whole surface.

5.5 Multi-Scale Gaussian Convolution

After the semigeodesic coordinates for the local patch are determined, the next step is to introduce a smoothing process at each local patch to eliminate noise and to reduce surface details as well as at the same extracting features. This approach is done in using a 2-D Gaussian convolution. The position vector at each semigeodesic coordinate is weighted and averaged by a 2-D Gaussian filter (see Figure 5.11). The formula for this filter is

$$G(u, v, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{(u^2+v^2)}{2\sigma^2}}$$



(a) Top view (b) Side view

Figure 5.11: *2-D Gaussian filter with size 9×9*

Taking into account the range of u and v, the extent of each side of the 2-D Gaussian filter is $8\sigma + 1$ a normalization constant. The filter needs to have sufficient sample

points in order to provide an accurate result. However, if the size of neighbourhood is too large, the computation becomes inefficient. Furthermore, the standard deviation σ of the Gaussian distribution is directly proportional to the size of local patches. A suitable compromise between size and efficiency is considered. Hence, a value of 1 is chosen for the fixed standard deviation σ and this results in a 9 by 9 filter.

After the 2-D Gaussian filter is defined as above, the next step is to convolve every point $\mathbf{r}(u, v)$ on the local patch with the filter. Hence, a smoothed point $\mathbf{R}(u, v, \sigma)$ on the surface is obtained (see Equations (4.2), (4.3) and (4.4)). This process is repeated at every local patch. Eventually, a complete new smoothed surface is produced. Then the next further step is to produce the multi-scale surface smoothing. Multi-scale surface smoothing is an iterative process that has the evolution properties as mentioned in Section 4.2.1. Hence, this smoothing process can reduce the noise on the surface and can also provide the surfaces at different scales.

Due to the displacement of vertices which occurs as a result of smoothing, either very small or very thin/needle-like triangles can be generated. Actually, some parts of the surface become very thin, e.g. dumbbell shape. These areas are composed of either very small or very thin/needle-like triangles before they collapse. Hence, such areas are no longer a simple surface. Detection of this type of surface is based on the length of the shortest side of triangle becoming less than a small threshold. This threshold is chosen as a small fraction of the average edge length over the entire surface. Since the odd triangles cause computational problems at each iteration, they are either removed or merged with neighbouring triangles using segmentation/decimation [160]. However, the number of triangles on the surface decreases after segmentation/decimation. As a result, refinement [62] is applied in order to maintain the number of triangles on the surface. Now smoothing continues either for the whole surface or for each part independently after segmentation. The results of this segmentation are shown in Figures 6.6 and 6.7 at Section 6.1.1.

5.6 Surface Curvature

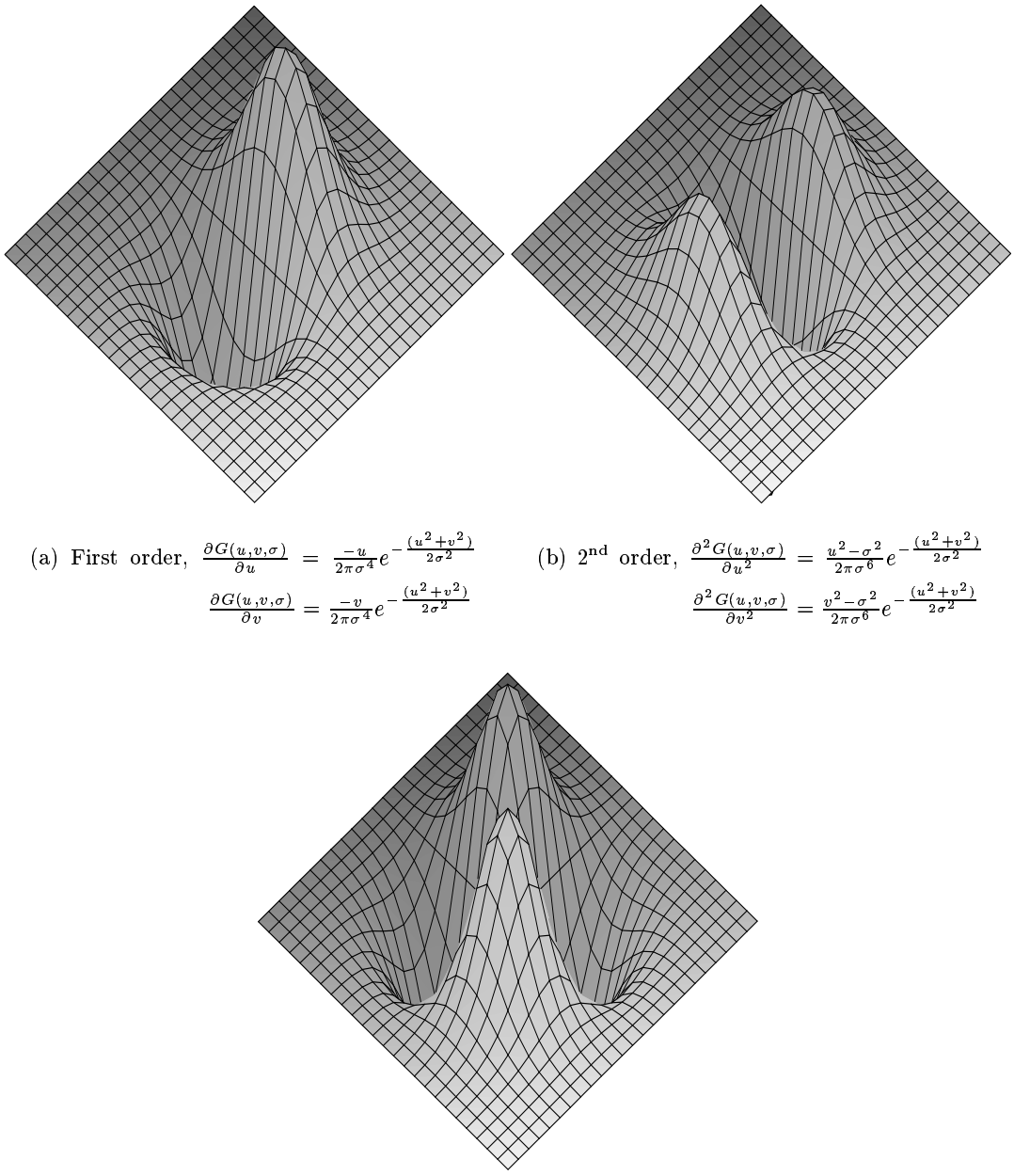
So far a technique for the diffusion of a 3-D surface is demonstrated. The local surface type can be categorized by estimating surface curvatures. This section presents a review of techniques for estimating Gaussian and mean curvatures of surface of a 3-D object. These are then modified to estimate curvature at multiple scales on a free-form 3-D surface. Differential geometry provides several measures of curvature, which include Gaussian and mean curvatures. Consider a local parametric representation of $\mathbb{R}^2$ of 3-D surface. The general parametric representation is shown in Equations (4.1) and (4.2).

The Gaussian curvature exists at regular points of a surface of class C^2. The Gaussian curvature function K is defined by Equation (4.9). Gaussian curvature is positive at elliptic points, negative at hyperbolic points and zero at parabolic points. The mean curvature function H is given by Equation (4.10). With the change of gradient of the surface the mean curvature changes its sign. Both Gaussian and mean curvatures values are direction-free quantities, i.e. magnitude only. Gaussian and mean curvatures are invariant to arbitrary transformation of the (u, v) parameters, to scaling, to translation and to rotation of a surface. A combination of curvature measures enables the local surface type to be categorized. On smoothed surfaces of 3-D objects, the following procedures for estimating the Gaussian and mean curvatures are used. The first and second partial derivatives of the Gaussian function, Equation (4.5), with respect to u and v are obtained (see Figure 5.12).

Then for each point $P(x(u,v), y(u,v), z(u,v))$ of the surface, the corresponding local neighbourhood data is convolved with the partial derivatives of the Gaussian function (see Equations (4.11)).

5.6.1 Local Maxima of Absolute Values of Curvature

Local maxima of absolute values of Gaussian and mean curvatures are significant and robust feature points on smoothed surfaces since noise has been eliminated from those surfaces. The process of recovery of the local maxima of absolute values



(a) First order, $\frac{\partial G(u,v,\sigma)}{\partial u} = \frac{-u}{2\pi\sigma^4} e^{-\frac{(u^2+v^2)}{2\sigma^2}}$ $\frac{\partial G(u,v,\sigma)}{\partial v} = \frac{-v}{2\pi\sigma^4} e^{-\frac{(u^2+v^2)}{2\sigma^2}}$

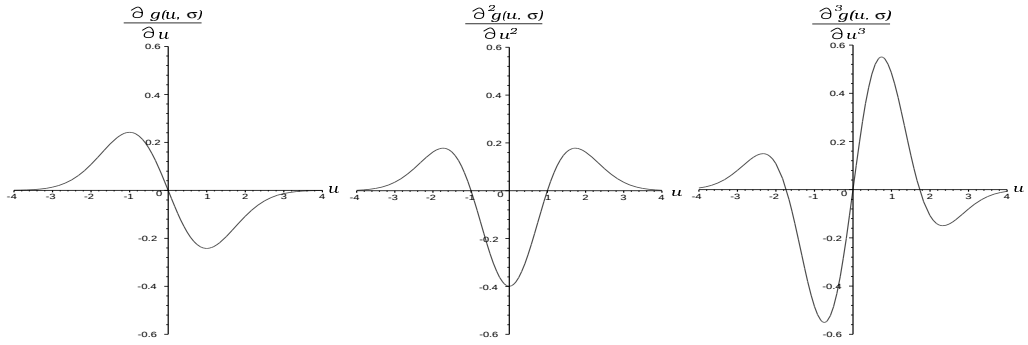
(b) 2nd order, $\frac{\partial^2 G(u,v,\sigma)}{\partial u^2} = \frac{u^2-\sigma^2}{2\pi\sigma^6} e^{-\frac{(u^2+v^2)}{2\sigma^2}}$ $\frac{\partial^2 G(u,v,\sigma)}{\partial v^2} = \frac{v^2-\sigma^2}{2\pi\sigma^6} e^{-\frac{(u^2+v^2)}{2\sigma^2}}$

(c) $\frac{\partial^2 G(u,v,\sigma)}{\partial v \partial u} = \frac{uv}{2\pi\sigma^6} e^{-\frac{(u^2+v^2)}{2\sigma^2}}$

Figure 5.12: *Partial derivatives of the Gaussian function*

is identical for Gaussian and mean curvatures. Every vertex V of the smoothed surface is examined in turn. The neighbours of V are defined as vertices which are connected to V by an edge. If the absolute curvature value of V is higher than the absolute curvature values of all its neighbours, then V is marked as a local maxima of absolute values of curvature. Such maxima can be utilized by later processes for robust surface matching and object recognition with occlusion.

5.7 Curvature Zero Crossing Contours and Torsion



(a) 1st order (b) 2nd order (c) 3rd order

Figure 5.13: *Derivatives of 1-D Gaussian function*

Having computed curvature values at each vertex of a smoothed 3-D surface, one can locate curvature zero crossing contours where curvature values K or H of neighbours change sign. Their curvature functions must pass through zero between the neighbours. Curvature zero crossing contours can be useful for calculating torsion local maxima of absolute values. The process of recovery of the curvature zero crossing contours is identical for Gaussian and mean curvatures. Every edge e of the smoothed surface is examined in turn. If the vertices of e have the same signs of curvature, then there is no curvature zero crossing point on e. However, if the vertices of e have different signs of curvature, then there exists a point on e at which curvature goes to zero. The zero crossing point lies on the edge e. The other two edges of the triangle to which e belongs will then be checked since there will be another zero crossing point on one of those edges. When that zero crossing is found,

it is connected to the previously found zero crossing. The curvature zero crossing contour is tracked in this fashion. Then the contour is sampled and convolved with the 1-D Gaussian filter using Equations (4.14) and (4.15). The first and second and third derivatives of the filter are shown in Figure 5.13 and their equations are

$$\frac{\partial g(u,\sigma)}{\partial u} = \frac{-u}{\sigma^3\sqrt{2\pi}} e^{-\frac{u^2}{2\sigma^2}}$$
$$\frac{\partial^2 g(u,\sigma)}{\partial u^2} = \frac{u^2-\sigma^2}{\sigma^5\sqrt{2\pi}} e^{-\frac{u^2}{2\sigma^2}}$$
$$\frac{\partial^3 g(u,\sigma)}{\partial u^3} = \frac{3u\sigma^2-u^3}{\sigma^7\sqrt{2\pi}} e^{-\frac{u^2}{2\sigma^2}}$$

Now torsion of the contour is calculated. Then the local maxima of absolute values of torsion are used for matching.

5.8 Optimal Scale

After the local maxima of absolute values of curvature and torsion have been computed and stored, geometric hashing and object recognition are now considered. Section 4.5 shows that the voting is done simultaneously for all models in the hash table. The overall recognition time is dependent on the number of feature points in the scene. The aim is to reduce the noise and number of small feature points. This is achieved through the multi-scale smoothing process. In order to find the optimal scale (i.e. minimum noise, minimum number of small features and maximum number of significant features) for the scene object, the following procedure is used:

1. Apply one iteration of smoothing and count the number of feature points recovered from the object.

2. Repeat step 1 until several iterations of smoothing have been carried out. Construct a graph of the number of feature points versus the number of iterations (see Figure 5.14).

3. Smooth this graph, and find the point where the slope is minimum. This indicates that the features have become stable at the corresponding scale which

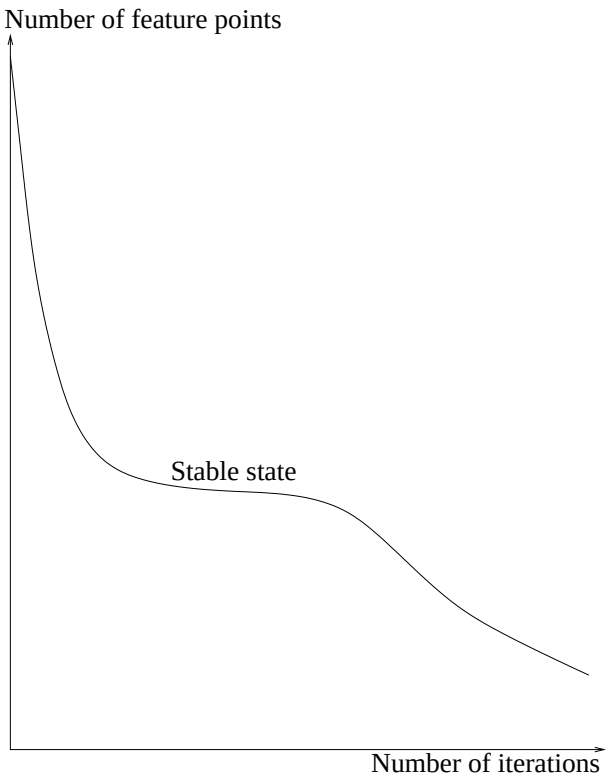


Figure 5.14: *Optimal smoothing scale*

is then used as the optimal scale for object recognition.

5.9 Geometric Hashing

The most important parts of geometric hashing are hash table and hash addressing algorithm (hashing function/hash code/multi-words key transformation). A good hash addressing algorithm can improve surface matching efficiency. Hence, it is logical to make use of the curvature/torsion features and edge length of triplet (see Figure 4.5 in Section 4.5.2); which are then encoded to produce an indexed value V_i in Equation (4.17). Such an indexed value is a unique real number that is generated by the hashing function. It is also distributed uniformly across the whole hash space. This indexed value is ideal for representing the triplet and it is also extremely valuable for creating the multi-words key $\mathcal{A}_i$ in Equation (4.18). This

multi-words key can quickly address an evenly spread hash table with minimum collisions (see Figure 5.15). Such hash table and hash addressing algorithm apply to all the combination of triplets from 3-D surface.

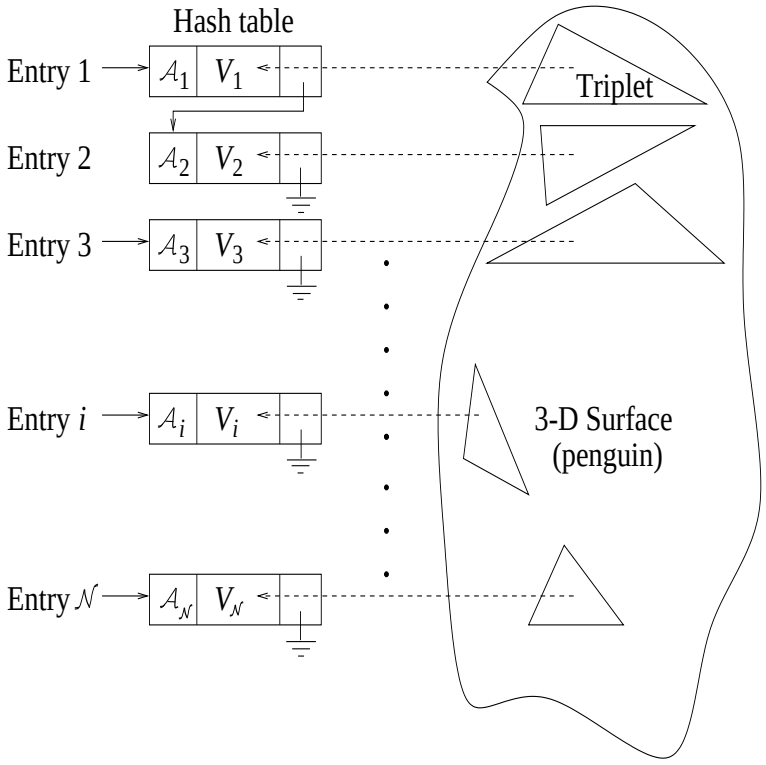


Figure 5.15: *Geometric hashing technique using linked lists and linked stacks [85] on a 3-D surface.*

Now four features sets of the models in database and four corresponding features sets of the scene are matched. They include the local maxima of absolute values of

- Gaussian curvature,
- mean curvature,
- torsion of zero crossing contours of Gaussian curvature and
- torsion of zero crossing contours of mean curvature.

Firstly, four individual hash tables are created off-line from the given models database. Secondly, the feature triplets from the scene are matched with the data

in these hash tables. Each matched triplet gives one vote for the corresponding model. This procedure continues until all permutations of triplets from the scene are computed. Therefore, the votes from each object model can be converted to the following score

$$\text{score} = \frac{m}{{}_nC_3} \times 100$$

where m is number of votes from the features set of an object model, n is the number of features of that object model from the corresponding features set of the database, and ${}_nC_3 = \frac{n!}{3!(n-3)!}$ is the combination of triplet. Then the total score for each model is calculated using the average value of above four features sets score. Hence, the highest score model is now obtained.

5.10 Global Verification

It may be possible to get more than one candidate solution with very close high scores from the geometric hashing stage. In this case, a threshold is used to select the most likely models so that global verification can work with those models. For each of selected object models, seven global transform parameters (including scaling $\mathcal{D}$, translation $d\ h\ q$ and rotation $\gamma\ \beta\ \alpha$) of every triplet are estimated, and then these candidates are compared with the parameters of scene. The successful candidates are clustered together. The model with the largest cluster is chosen as the most likely object. The details of this clustering algorithm can be found in Section 4.6.

To simplify the estimation of seven global transform parameters of triplet in Equation (4.19), the scale parameter is firstly expressed by $\mathcal{D}$ in Equation (4.20), and the translation parameters (d, h, q) are the displacement between the centroid $P_4(x_{4p}, y_{4p}, z_{4p})$ of model triplet and the centroid $Q_4(x_{4q}, y_{4q}, z_{4q})$ of scene triplet. Since the position vector of all triplet vertices have already been found at the geometric hashing stage, the centroids can be computed using the section formulae, and the translation parameters are $d = x_{4q} - x_{4p}$, $h = y_{4q} - y_{4p}$ and $q = z_{4q} - z_{4p}$.

Furthermore, Equation (4.19) can be reduced and expressed as

$$x_{1p}a + y_{1p}b + z_{1p}c = x_{1q} - d, \quad x_{1p}e + y_{1p}f + z_{1p}g = y_{1q} - h,$$
$$x_{2p}a + y_{2p}b + z_{2p}c = x_{2q} - d, \quad x_{2p}e + y_{2p}f + z_{2p}g = y_{2q} - h,$$
$$x_{3p}a + y_{3p}b + z_{3p}c = x_{3q} - d, \quad x_{3p}e + y_{3p}f + z_{3p}g = y_{3q} - h,$$
$$x_{1p}m + y_{1p}n + z_{1p}p = z_{1q} - q,$$
$$x_{2p}m + y_{2p}n + z_{2p}p = z_{2q} - q,$$
$$x_{3p}m + y_{3p}n + z_{3p}p = z_{3q} - q$$

Now the Cramer's rule [96] is applied to solve above equations. Given Equation (4.21) and

$$|D| = \begin{vmatrix} x_{1p} & y_{1p} & z_{1p} \\ x_{2p} & y_{2p} & z_{2p} \\ x_{3p} & y_{3p} & z_{3p} \end{vmatrix}, \quad \text{then} \quad a = \frac{\begin{vmatrix} x_{1q} - d & y_{1p} & z_{1p} \\ x_{2q} - d & y_{2p} & z_{2p} \\ x_{3q} - d & y_{3p} & z_{3p} \end{vmatrix}}{|D|} = cos\alpha(cos\beta),$$

$$e = \frac{\begin{vmatrix} y_{1q} - h & y_{1p} & z_{1p} \\ y_{2q} - h & y_{2p} & z_{2p} \\ y_{3q} - h & y_{3p} & z_{3p} \end{vmatrix}}{|D|} = sin\alpha(cos\beta), \quad m = \frac{\begin{vmatrix} z_{1q} - q & y_{1p} & z_{1p} \\ z_{2q} - q & y_{2p} & z_{2p} \\ z_{3q} - q & y_{3p} & z_{3p} \end{vmatrix}}{|D|} = -sin\beta,$$

$$n = \frac{\begin{vmatrix} x_{1p} & z_{1q} - q & z_{1p} \\ x_{2p} & z_{2q} - q & z_{2p} \\ x_{3p} & z_{3q} - q & z_{3p} \end{vmatrix}}{|D|} = cos\beta(sin\gamma), \quad p = \frac{\begin{vmatrix} x_{1p} & y_{1p} & z_{1q} - q \\ x_{2p} & y_{2p} & z_{2q} - q \\ x_{3p} & y_{3p} & z_{3q} - q \end{vmatrix}}{|D|} = cos\beta(cos\gamma)$$

Therefore, the rotation parameters are

$$\gamma = cos^{-1}\left(\frac{p}{\pm\sqrt{n^2 + p^2}}\right) \text{ or } sin^{-1}\left(\frac{n}{\pm\sqrt{n^2 + p^2}}\right)$$
$$\beta = cos^{-1}\left(\pm\sqrt{a^2 + e^2}\right) \text{ or } sin^{-1}(-m)$$
$$\alpha = cos^{-1}\left(\frac{a}{\pm\sqrt{a^2 + e^2}}\right) \text{ or } sin^{-1}\left(\frac{e}{\pm\sqrt{a^2 + e^2}}\right)$$

Now 3-D scaling, translation and rotation parameters are used for global verification and the clustering algorithm is used to obtain the most likely object as described in

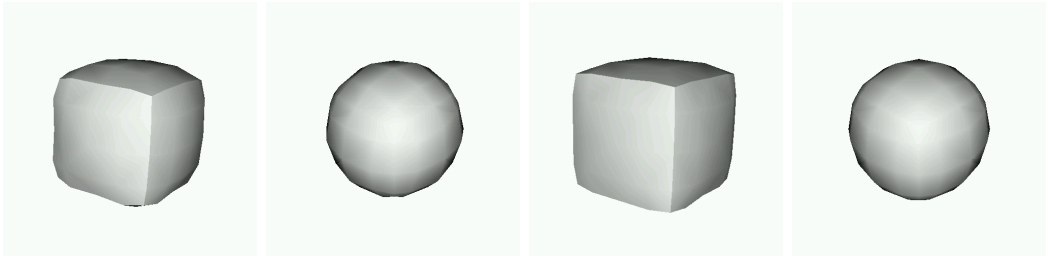
Section 4.6. The results indicated that the technique is invariant to the transformations.

Chapter 6

Results and Discussion

The programs were implemented entirely in C++ [179, 57, 69, 158] and the results were displayed using Visualisation Toolkit (VTK) [160]. Complete triangulated models of 3-D objects, and scene objects used for the experiments were constructed using 3-D laser scanner ModelMaker. This chapter presents research results on free-form surface smoothing, curvature estimation as well as object recognition.

6.1 Diffusion

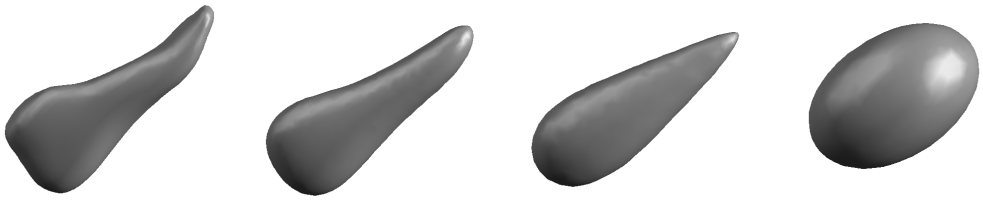


(a) 1^{st} iteration (b) 5^{th} iteration (c) 1^{st} iteration (d) 5^{th} iteration

Semigeodesic coordinates Geodesic polar coordinates

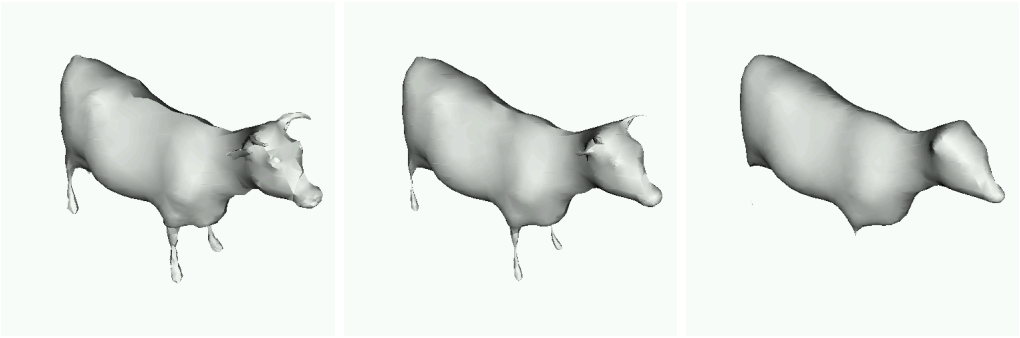
Figure 6.1: *Diffusion of the cube*

In order to experiment with the research techniques, both simple and complex 3-D objects with different numbers of triangles are used. Each iteration of smoothing



(a) Original (b) 10^{th} iteration (c) 20^{th} iteration (d) 100^{th} iteration

Figure 6.2: *Diffusion of the foot*



(a) Original (b) 1^{st} iteration (c) 3^{rd} iteration

(d) 8^{th} iteration (e) 12^{th} iteration (f) 21^{st} iteration

Figure 6.3: *Diffusion of the bull*

of a surface with 1000 vertices takes about 0.5 second of CPU time on a Sun Ultra Sparc workstation. The diffusion results for 3-D surfaces are given in [120]. It should be noted that Gaussian smoothing causes shrinkage of objects. In general, this is not a problem. In fact, objects after each iteration of smoothing are rescaled. This procedure cancels out the resulting shrinkage.

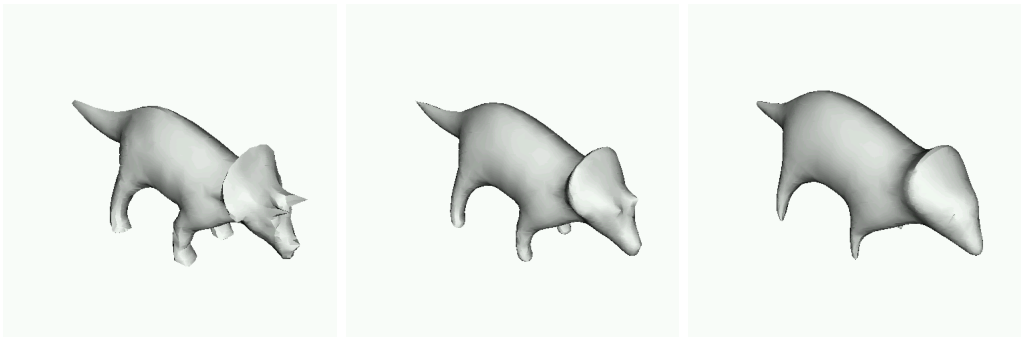
The first test object is a cube with 98 vertices and 192 triangles. The smoothing results using semigeodesic coordinates (with filter size equal to 9) are shown in Figures 6.1(a) and (b). The original cube is changed to a sphere after five iterations. The experiment is also repeated using geodesic polar coordinates, and the smoothing results are shown in Figures 6.1(c) and (d). These results indicate that smoothing using semigeodesic coordinates and geodesic polar coordinates produce similar results. Therefore, smoothing technique using semigeodesic coordinates with filter size equal to 9 is applied for the following 3-D surfaces.

The second test object is a foot with 996 triangles and 500 vertices. The smoothing results are shown in Figure 6.2. The foot also becomes rounded iteratively and evolves into an ellipsoidal shape after 100 iterations. This technique is examined further with more complex 3-D objects.

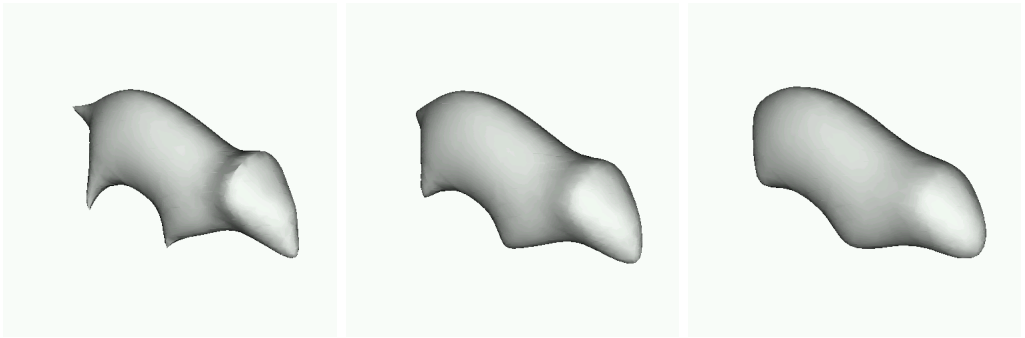
The third test object is a bull with 3348 triangles and 1676 vertices as shown in Figure 6.3. The surface noise is eliminated iteratively with the object becoming gradually smoother and after 8 iterations the legs, ears, horns and tail are removed.

The fourth test object is a dinosaur with 2996 triangles and 1500 vertices as shown in Figure 6.4. The object becomes gradually smoother and the legs, tail, horns and ears are removed after 12 iterations.

The fifth test object is a rabbit with 1996 triangles and 1000 vertices as shown in Figure 6.5. The ears disappear after 10 iterations and the object becomes a smooth and rounded shape.

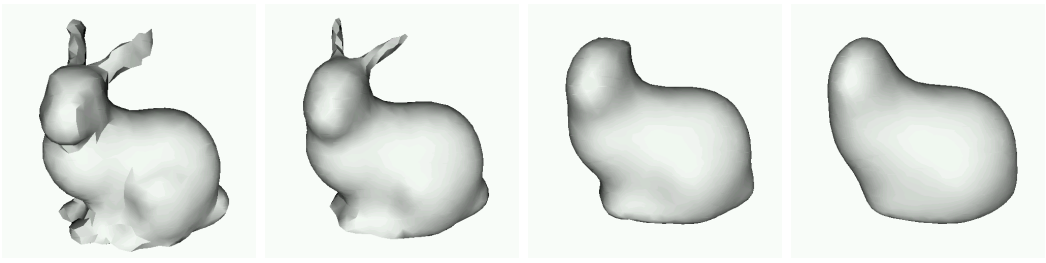


(a) Original (b) 2^{nd} iteration (c) 5^{th} iteration



(d) 8^{th} iteration (e) 12^{th} iteration (f) 20^{th} iteration

Figure 6.4: *Diffusion of the dinosaur*



(a) Original (b) 5^{th} iteration (c) 10^{th} iteration (d) 24^{th} iteration

Figure 6.5: *Diffusion of the rabbit*

6.1.1 Segmentation

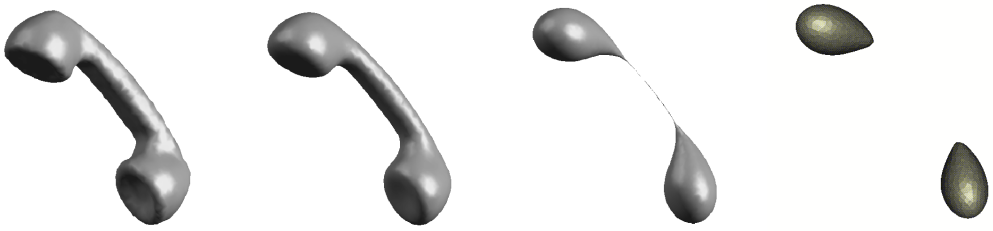
The smoothing process can cause disconnection of object parts. This is a natural consequence of the heat diffusion of objects. A decimation procedure is applied after each iteration of smoothing to remove odd-shaped triangles. The procedure can also segment objects into parts when they become very thin. This is due to the fact that very thin regions give rise to odd-shaped (elongated) triangles.

Figure 6.6 shows a phone represented by 1996 triangles and 1000 vertices. Notice that the surface noise is eliminated iteratively with the object becoming gradually smoother and after 15 iterations the object becomes very thin in the middle [42]. Decimation then removes the thin triangles and segments the object into two parts. Smoothing then continues for each part as shown in Figure 6.6(d).

The next test object is a chair with 2000 triangles and 1000 vertices as shown in Figure 6.7, where again the legs of the chair, the joints between the chair and its back become thin after 4 iterations as previously seen for the phone. The joints are then removed and the object is segmented into two parts after smoothing, and the result is shown in Figure 6.7(d).

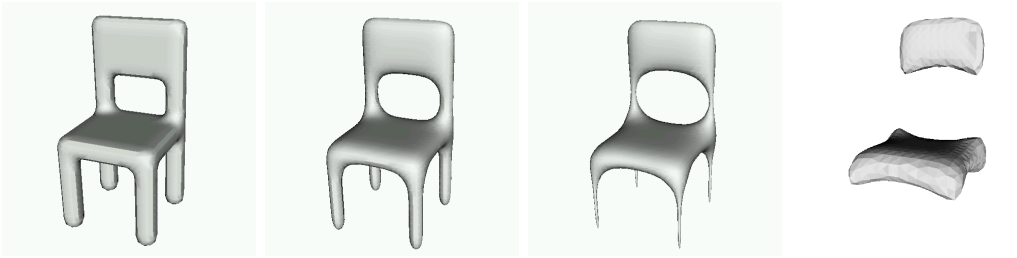
6.1.2 Open or Incomplete Surface

The smoothing technique is applied to a number of open/incomplete surfaces. Figure 6.8 shows the smoothing results obtained on a part of the foot object shown in Figure 6.2(a). Figure 6.9 shows the results obtained on a part of the phone shown in Figure 6.6(a). This object also has a triangle removed in order to generate an internal hole. Figure 6.10(b) shows the smoothing and segmentation results obtained on the lower part of the chair object shown in Figure 6.7(a), and then the small segment is removed by decimation at the third iteration (see Figure 6.10(c)). Figure 6.11 shows smoothing results obtained on a partial rabbit. The object is smoothed iteratively and the ears disappear as well.



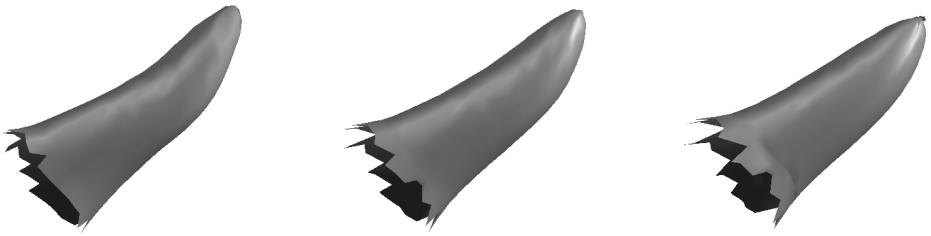
(a) Original (b) 3rd iteration (c) 15th iteration (d) 25th iteration

Figure 6.6: *Diffusion and segmentation of the phone*



(a) Original (b) 2nd iteration (c) 4th iteration (d) 14th iteration

Figure 6.7: *Diffusion and segmentation of the chair*

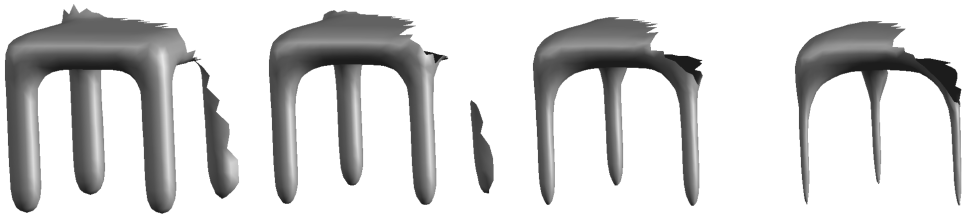


(a) Original (b) 2nd iteration (c) 5th iteration

Figure 6.8: *Diffusion of the partial foot*

(a) Original (b) 3^{rd} iteration (c) 5^{th} iteration

Figure 6.9: *Diffusion of the partial phone*



(a) Original (b) 2^{nd} iteration (c) 3^{rd} iteration (d) 4^{th} iteration

Figure 6.10: *Diffusion, segmentation and decimation of the partial chair*

(a) Original (b) 5^{th} iteration (c) 9^{th} iteration (d) 15^{th} iteration

Figure 6.11: *Diffusion of the rabbit head*

6.1.3 General View of Diffusion

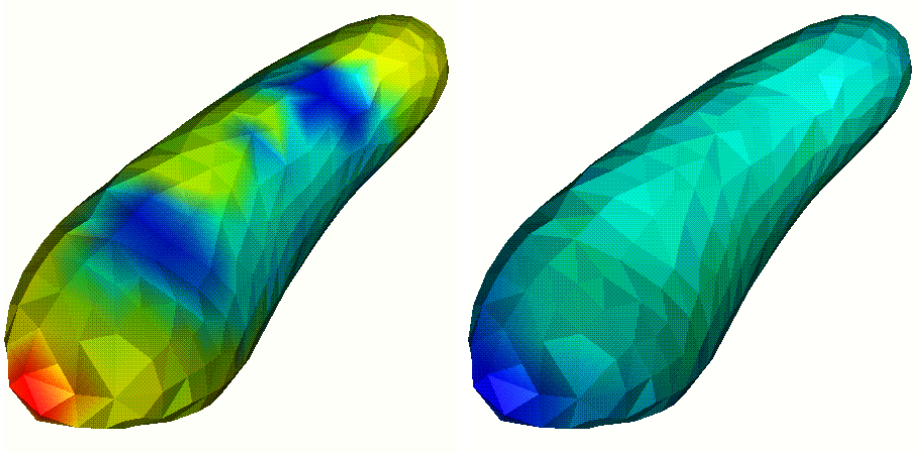
From the results in this section, one can conclude that the proposed technique is effective in eliminating surface noise as well as removing surface detail and it is also independent of the underlying triangulation. The result is gradual simplification of the object shape. It is also true that the technique can apply to segmented, open and incomplete 3-D surfaces.

6.2 Curvature Estimation

This section presents the results of application of curvature estimation techniques to 3-D objects using methods described in Sections 4.3 and 5.6. These curvature estimation results for 3-D surfaces are reported in [203]. The first example is a foot. After smoothing the object, the Gaussian curvatures of all vertices are estimated using Equation (4.9). To visualize these curvature values on the surface, they are then mapped to colours [84] using the Visualisation Toolkit (VTK) [160], and the results are shown in Figure 6.13(a). Surface curvature colours are coded as follows:

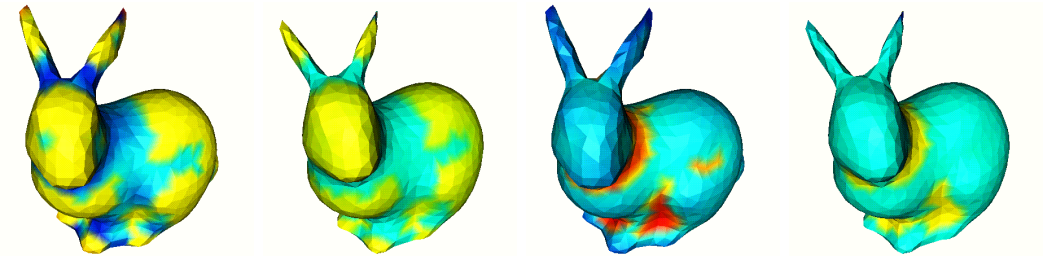
Figure 6.12: *red=high, blue=low*

and other colours designate in-between values. The flat areas are green since their curvature values are close to zero. The same experiment is repeated to estimate the mean curvatures of the foot using Equation (4.10) and the results are shown in Figure 6.13(b). This indicates that mean curvature values for the toes are different



(a) Gaussian (b) Mean

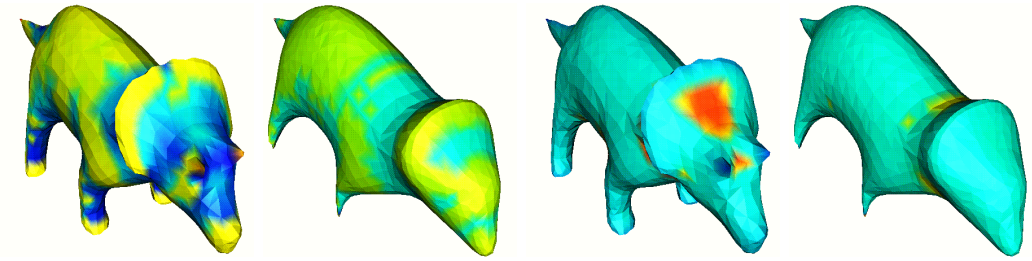
Figure 6.13: *Gaussian and mean curvatures on the foot*



(a) 1st iteration (b) 2nd iteration (c) 1st iteration (d) 2nd iteration

Gaussian mean

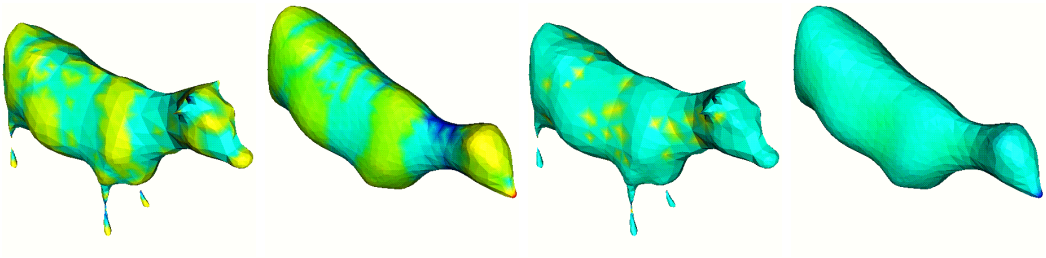
Figure 6.14: *Curvatures on the rabbit*



(a) 1st iteration (b) 6th iteration (c) 1st iteration (d) 6th iteration

Gaussian mean

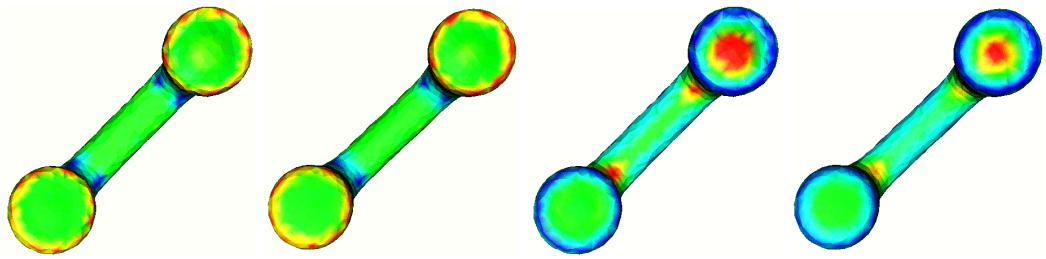
Figure 6.15: *Curvatures on the dinosaur*



(a) 1st iteration (b) 6th iteration (c) 1st iteration (d) 6th iteration

Gaussian mean

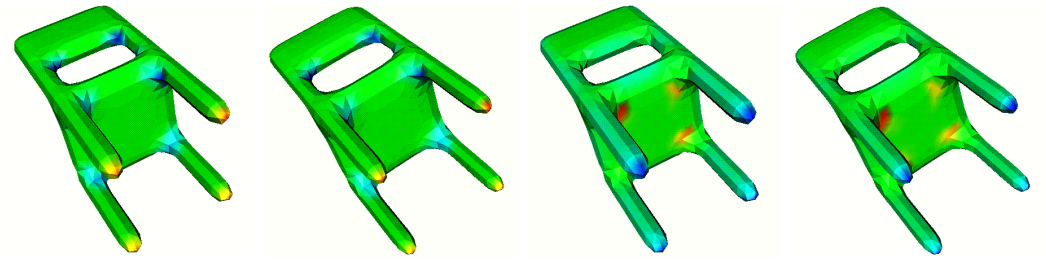
Figure 6.16: *Curvatures on the bull*



(a) 1st iteration (b) 3rd iteration (c) 1st iteration (d) 3rd iteration

Gaussian mean

Figure 6.17: *Curvatures on the phone*



(a) 1st iteration (b) 2nd iteration (c) 1st iteration (d) 2nd iteration

Gaussian mean

Figure 6.18: *Curvatures on the chair*

than those for other areas, as expected.

The next object is a rabbit. Its Gaussian curvature values are estimated and results are shown in Figures 6.14(a) and (b). These results again confirm that the curvature values are high and low at convex and concave areas, respectively. The mean curvatures of the rabbit are also estimated and the results are shown in Figures 6.14(c) and (d). Moreover, Gaussian and mean curvatures are also estimated for other objects. Figures 6.15, 6.16, 6.17 and 6.18 show the results for a dinosaur, a bull, a phone and a chair respectively.

6.2.1 Curvature Error

Two principal curvatures k_1 and k_2 are defined at each point of a 3-D surface of a sphere. The Gaussian and mean curvatures are calculated as in Equations (4.7) and (4.8). O'Neill [139] shows the principal curvatures of the sphere, $k_1 = k_2 = -\frac{1}{r}$, where r is the radius of the sphere. Hence, the Gaussian curvature $K = k_1 k_2 = \frac{1}{r^2}$ and the mean curvature $H = \frac{k_1+k_2}{2} = -\frac{1}{r}$. Given $r = 2430$ units, we obtain $K = 1.7 \times 10^{-7}$ and $H = -4.12 \times 10^{-4}$. By comparing these calculations with the experimental results in Figure 6.19, we confirmed that curvature estimation using our technique is quite accurate.

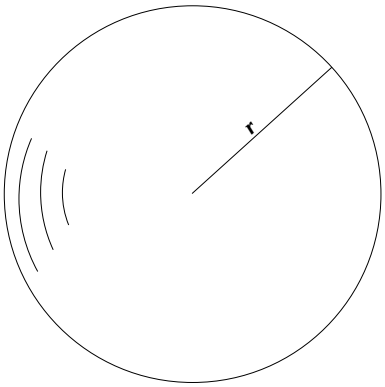
Furthermore, if a 3-D surface of class C^2 is given, $\tilde{k_1}$ and $\tilde{k_2}$ are the measured values of k_1 and k_2. It follows that if $\tilde{k_1} = k_1 + \varepsilon$ and $\tilde{k_2} = k_2 + \varepsilon$, where ε represents error ($\varepsilon \ll 1$), then the estimated Gaussian curvature is given by

$$K = \tilde{k_1}\tilde{k_2} = k_1 k_2 + \varepsilon(k_1 + k_2) + \varepsilon^2$$

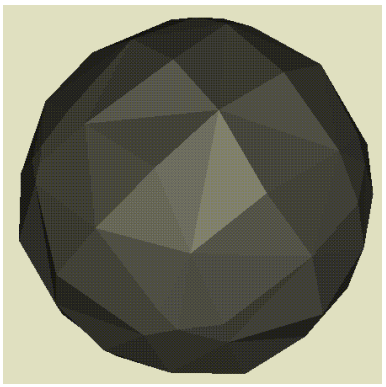
Similarly, the mean curvature H is given by

$$H = \frac{\tilde{k_1} + \tilde{k_2}}{2} = \frac{k_1 + k_2}{2} + \varepsilon$$

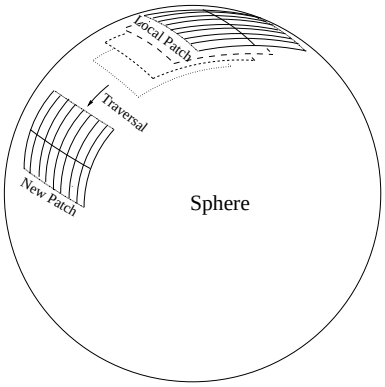
Since the minimal surface (a surface whose mean curvature H vanishes identically; a surface for which the first variation of the area integral vanishes. A minimal surface does not necessarily minimize the area spanned by a given contour; but if a smooth



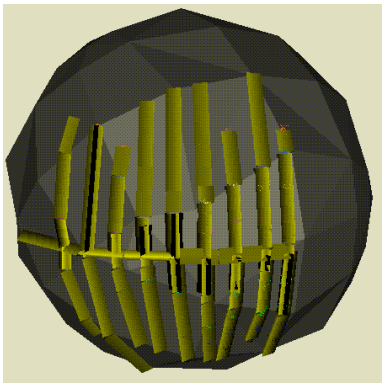
(a) Sphere radius $r = 2430$ units



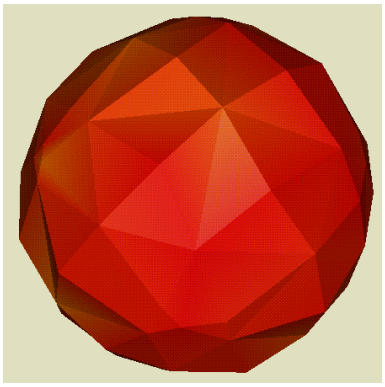
(b) Sphere with 98 vertices



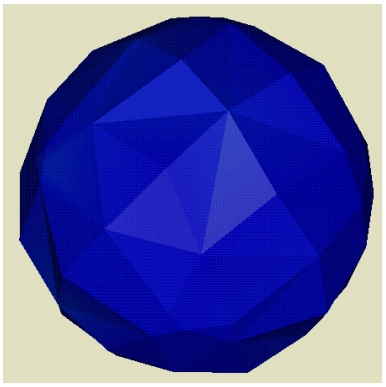
(c) Local patch



(d) Semigeodesic Coordinates

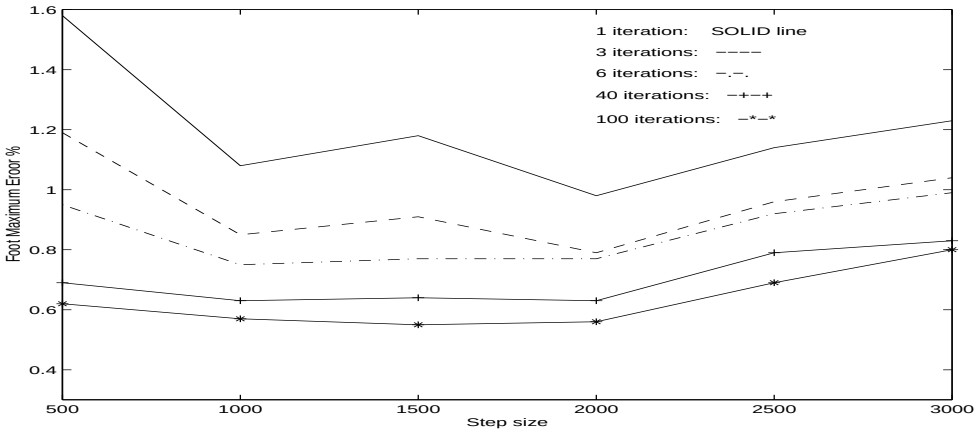


(e) Gaussian curvature $K = 1.79 \times 10^{-7}$, whole sphere becomes red

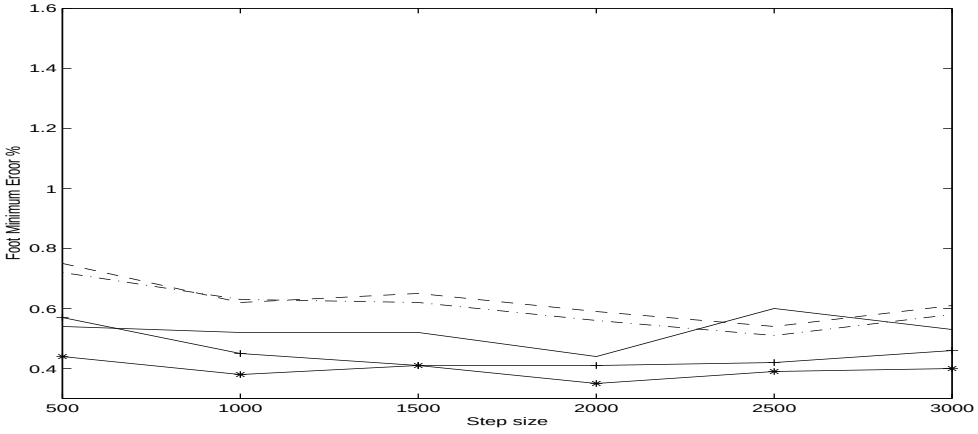


(f) Mean curvatures $H = -4.24 \times 10^{-4}$, whole sphere becomes blue

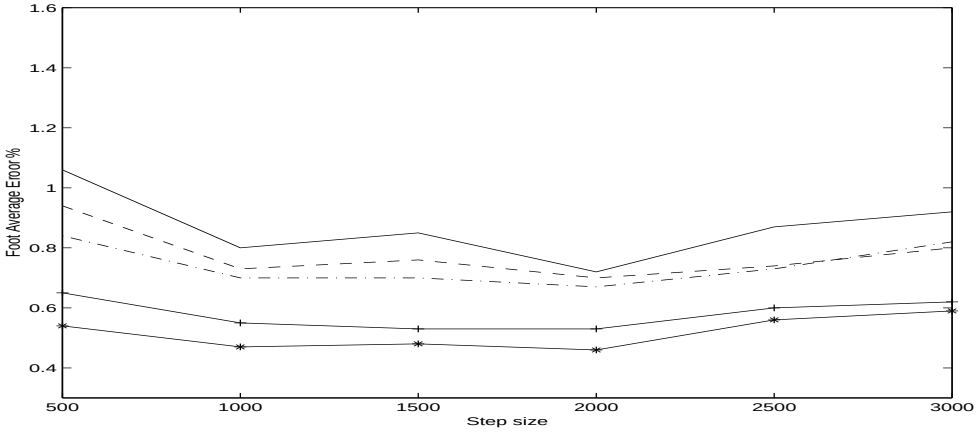
Figure 6.19: *The Gaussian and mean curvatures of a sphere*



(a) Maximum error

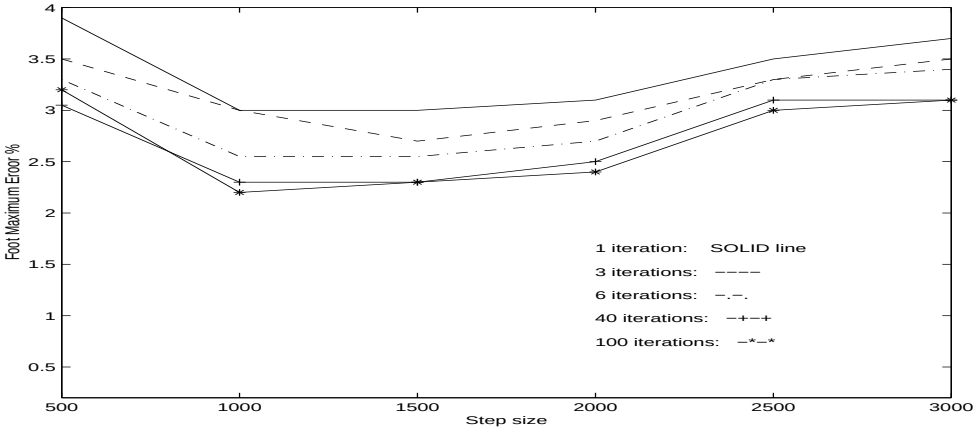


(b) Minimum error

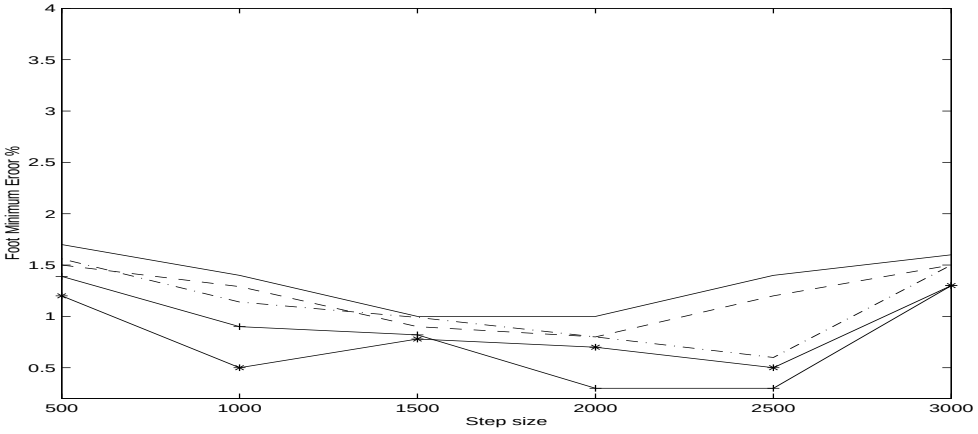


(c) Average error

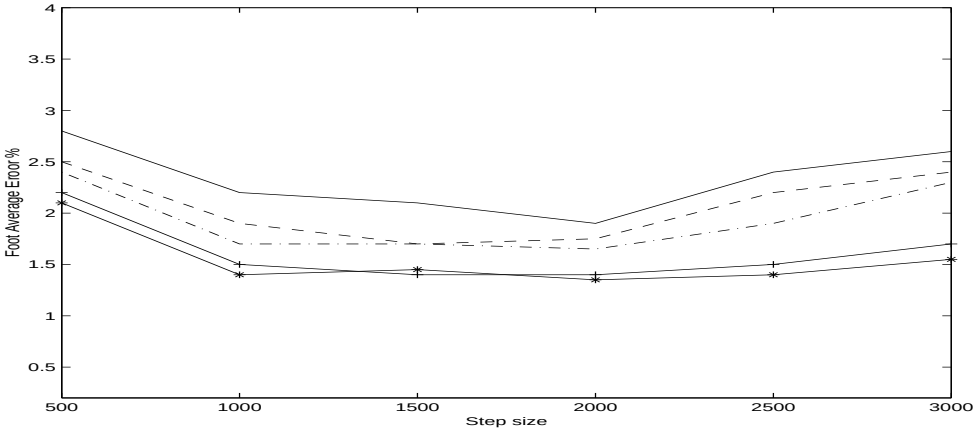
Figure 6.20: Gaussian curvature error distribution of the foot



(a) Maximum error



(b) Minimum error



(c) Average error

Figure 6.21: *Mean curvature error distribution of the foot*

surface $\mathcal{S}$ minimizes the area, then $\mathcal{S}$ is a minimal surface) is applied in every local patch, $\varepsilon(k_1 + k_2) + \varepsilon^2 \ll \varepsilon$. It follows that for an error in the values of principal curvatures k_1 and k_2, the error in the estimation of Gaussian curvature is expected to be smaller than that of the mean curvature. On smoothed surfaces of 3-D objects, the following procedure for curvature estimation error is defined.

Estimation of error in curvature computation is important. As mentioned before, the direction of the first geodesic line is randomly selected at each vertex. Its curvature value is k_i. Therefore, it is logical to examine the curvature estimation error in k_i against the average curvature value $\bar{k}$ for all possible directions. Then, the percentage error in direction i at each vertex is given by

$$\varepsilon_i = \frac{|\bar{k} - k_i|}{|\bar{k}|} \times 100\%$$

$$\text{Maximum error} = \max_{i=1}^{n} \varepsilon_i$$

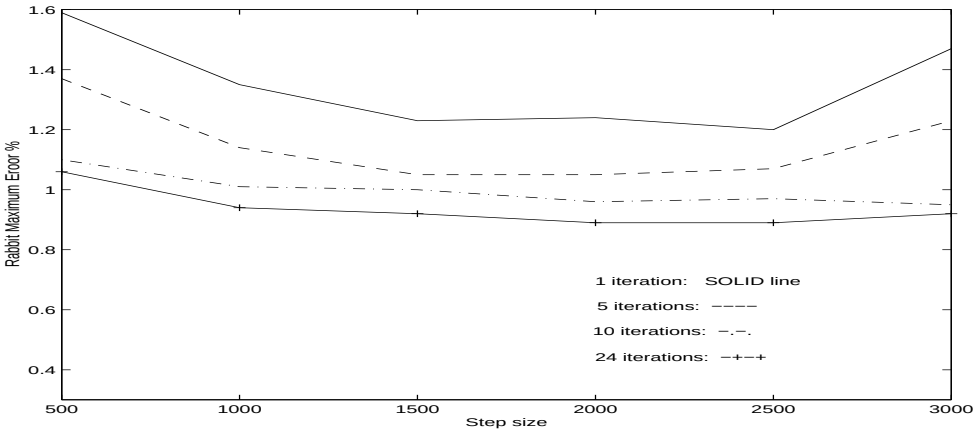
$$\text{Minimum error} = \min_{i=1}^{n} \varepsilon_i$$

$$\text{Average error} = \frac{\sum_{i=1}^{n} \varepsilon_i}{n}$$

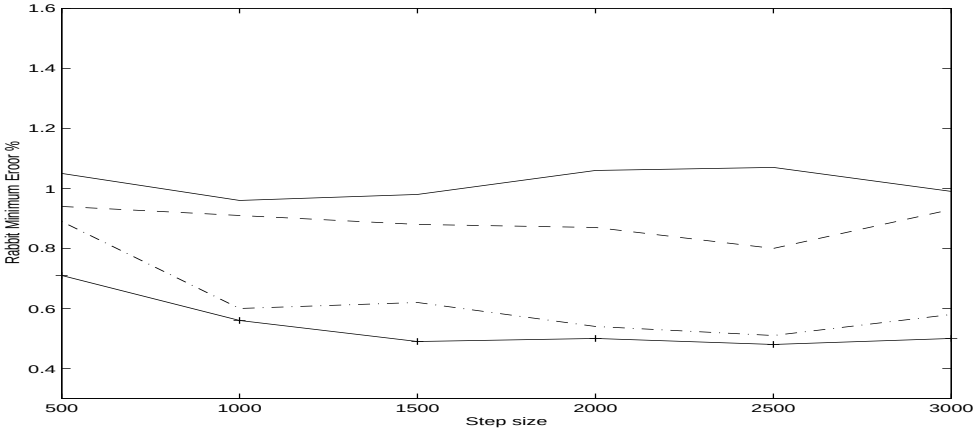
where n is the number of incident edges at a vertex.

Specifically, the direction of each incident edge is defined as a possible direction. The error estimation procedure is repeated at each vertex. Similarly, the different step sizes are also applied to this error estimation procedure. The maximum, minimum and average values of the curvature errors are computed and plotted in the graphs.

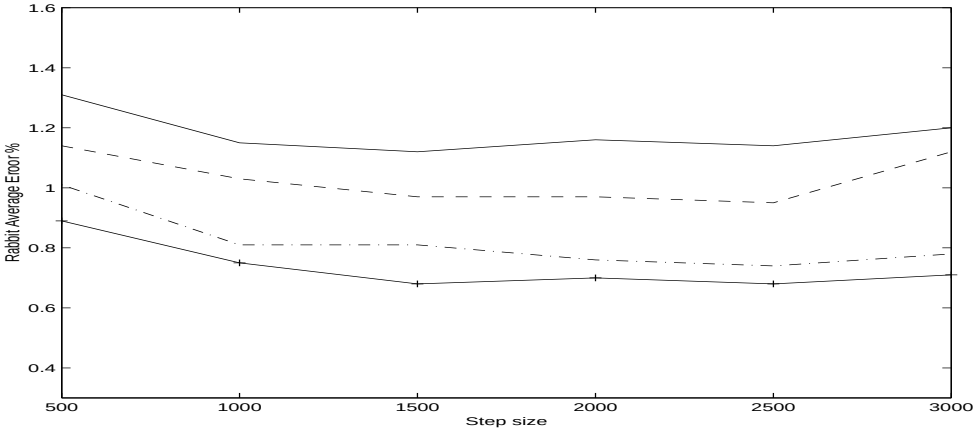
Figure 6.20(a) shows the error distribution for estimating maximum Gaussian curvature with the step size varying from 500 units to 3000 units. These results indicate that for the step size between 1000 units to 2000 units the error is reduced to about 1.2% after one iteration. Then the experiment is repeated for each vertex to compute the minimum error in curvature values of all possible directions. The results are shown in Figure 6.20(b). These indicate that for the step size between 1000



(a) Maximum error

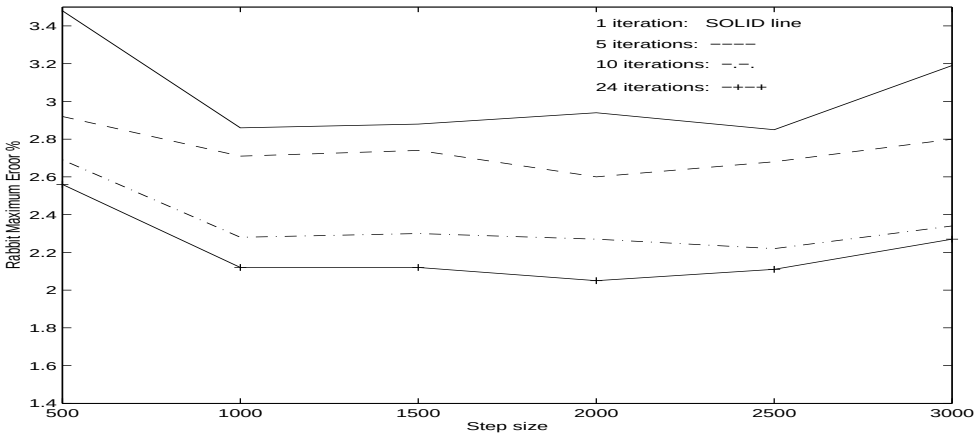


(b) Minimum error

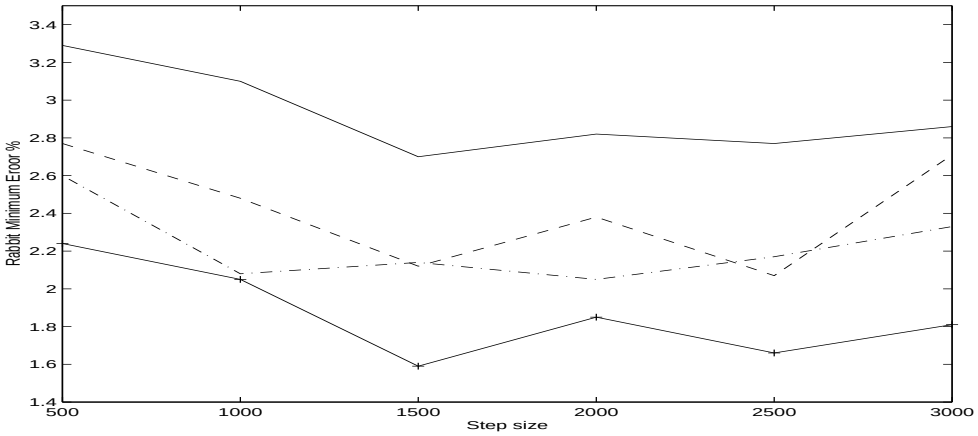


(c) Average error

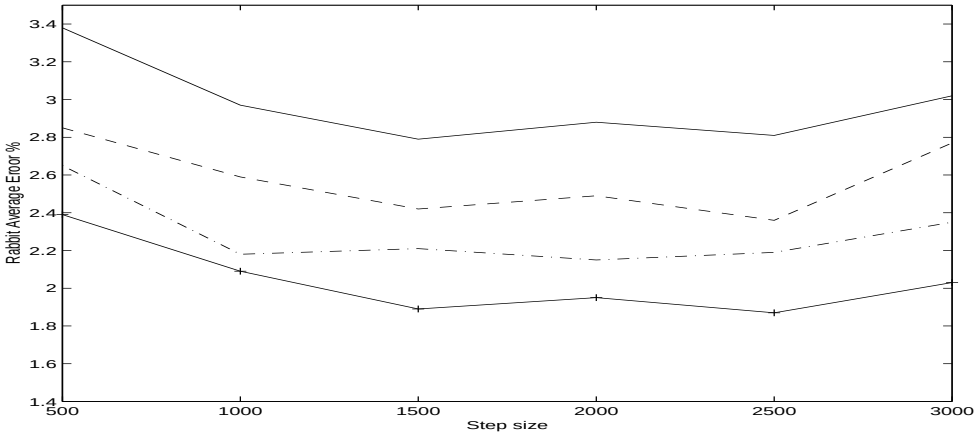
Figure 6.22: *Gaussian curvature error distribution of the rabbit*



(a) Maximum error



(b) Minimum error



(c) Average error

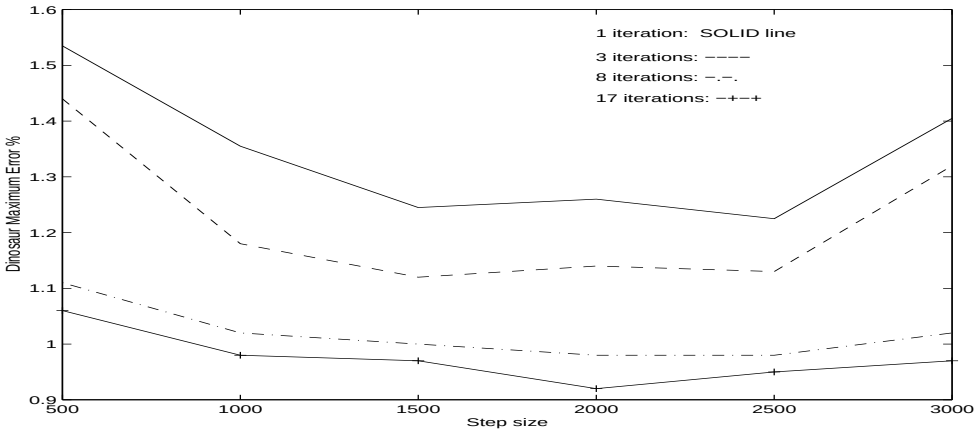
Figure 6.23: *Mean curvature error distribution of the rabbit*

units to 2000 units the error is reduced to about 0.5% after one iteration. When the average curvature value of all possible directions is calculated, the average curvature error is about 0.8% after one iteration, as shown in Figure 6.20(c). However, as the surface becomes iteratively smoother, the errors reduce as shown in Figure 6.20. After 100 iterations the errors in maximum, minimum and average curvature values are reduced to 0.6%, 0.4% and 0.5%, respectively. These experiments show they are not significantly different from the 0.5% quoted for a single iteration. It means the whole smoothing process is stable and has very small error.

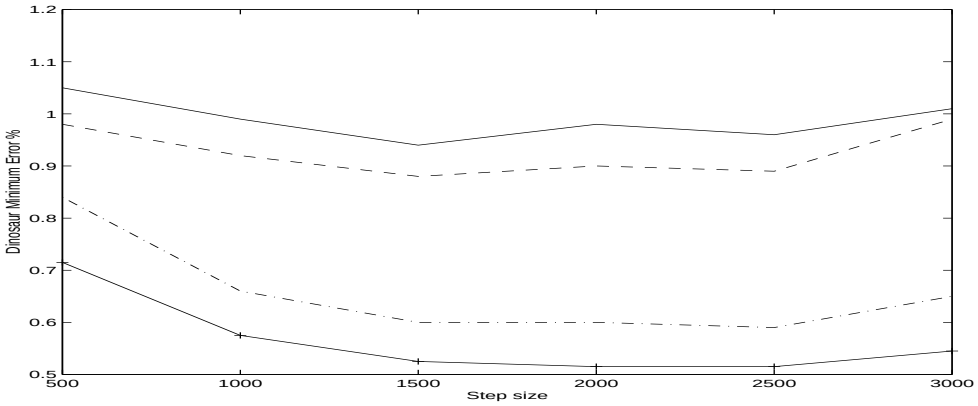
The above procedures are also repeated for estimation of mean curvatures. Figures 6.21(a) and (b) show the error distributions for the estimation of mean curvatures, and the error is also reduced for the step sizes between 1000 units to 2000 units, which are about 3.0% and 1.0% for the maximum and minimum mean curvatures, respectively. For the average value of mean curvature, the error is about 2.0% as shown in Figure 6.21(c). As the surface becomes smooth iteratively, the errors are reduced as shown in Figure 6.21 and after 100 iterations the errors in maximum, minimum and average curvature values drop to about 2.2%, 0.5% and 1.4%, respectively. Notice that the error for the estimation of Gaussian curvatures is lower than that of the mean curvature as is discussed in the previous section.

Figure 6.22 shows the error distribution for estimating Gaussian curvatures of the rabbit when all possible directions are selected. Again the errors are reduced for step sizes between 1000 units to 2000 units and for one iteration, the errors for maximum, minimum and average Gaussian curvatures are about 1.23%, 0.96% and 1.15%, respectively. After 24 iterations these errors reduce to about 0.92%, 0.5% and 0.7%, respectively. Figure 6.23 also shows the error distribution for estimating mean curvature of the rabbit and these results also indicate that the errors for maximum, minimum and average curvature values are about 2.85%, 2.7% and 2.8%, respectively. After 24 iterations these errors reduce to about 2.1%, 1.6% and 1.9%, respectively.

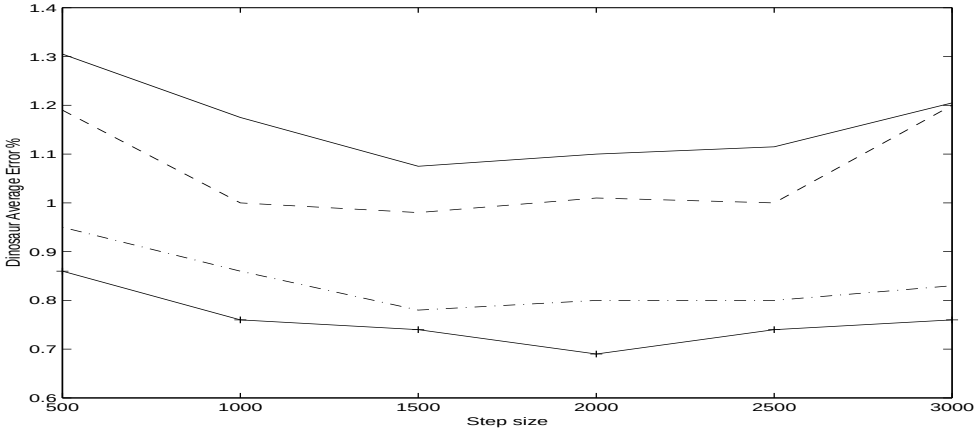
Figure 6.24 shows the error distribution for estimating Gaussian curvatures of the dinosaur when all possible directions are selected. The errors are reduced for step



(a) Maximum error

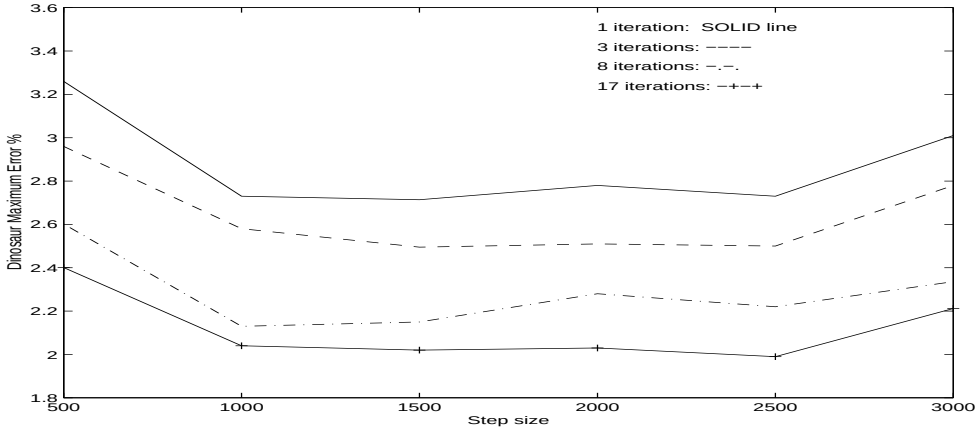


(b) Minimum error

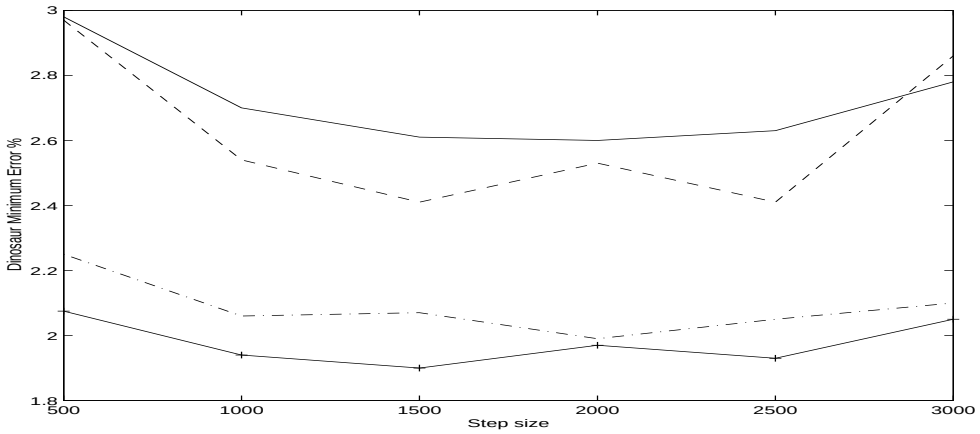


(c) Average error

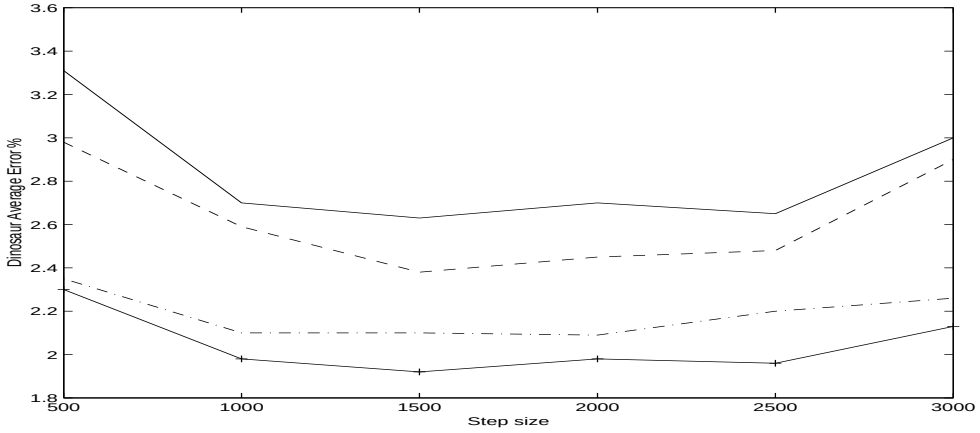
Figure 6.24: *Gaussian curvature error distribution of the dinosaur*



(a) Maximum error

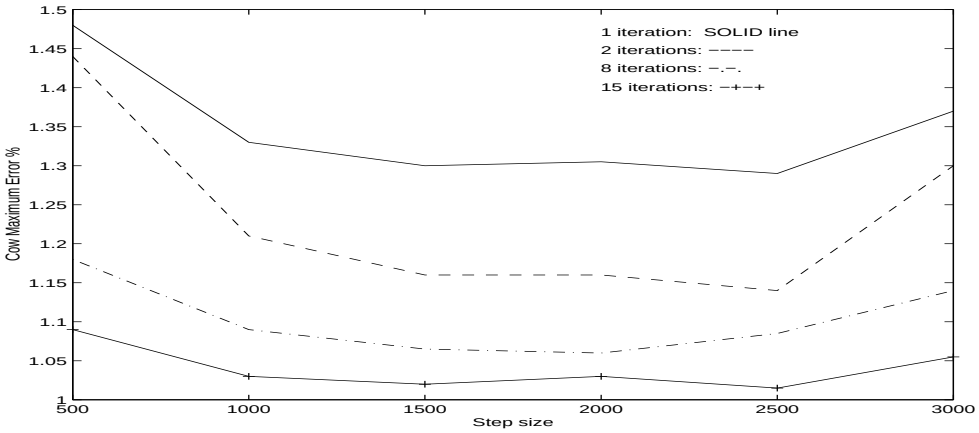


(b) Minimum error

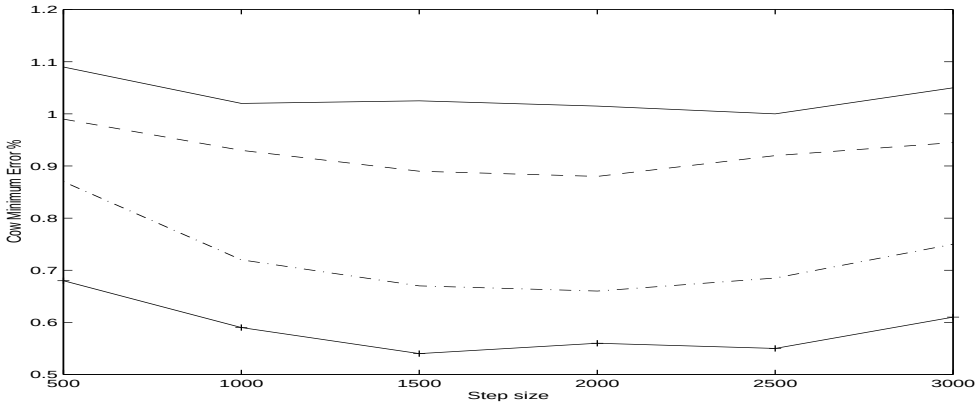


(c) Average error

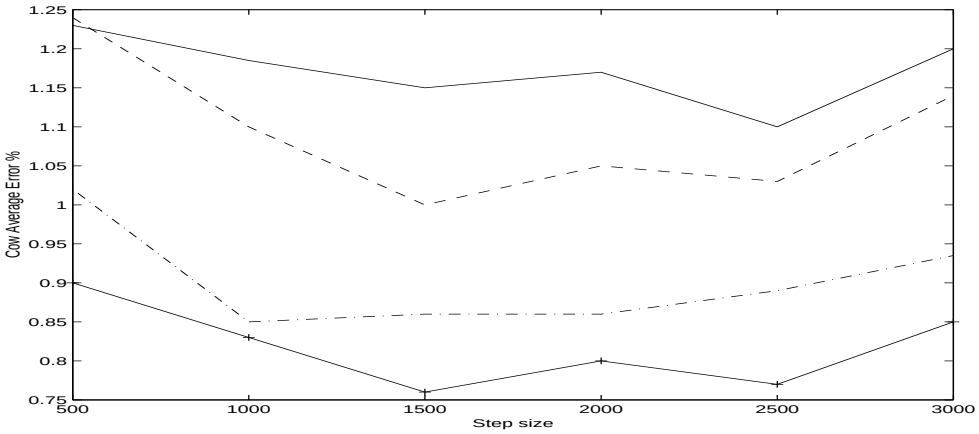
Figure 6.25: *Mean curvature error distribution of the dinosaur*



(a) Maximum error

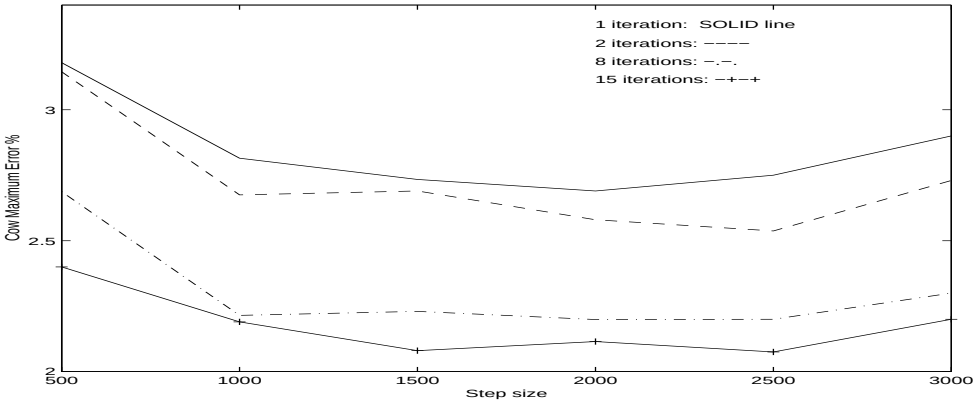


(b) Minimum error

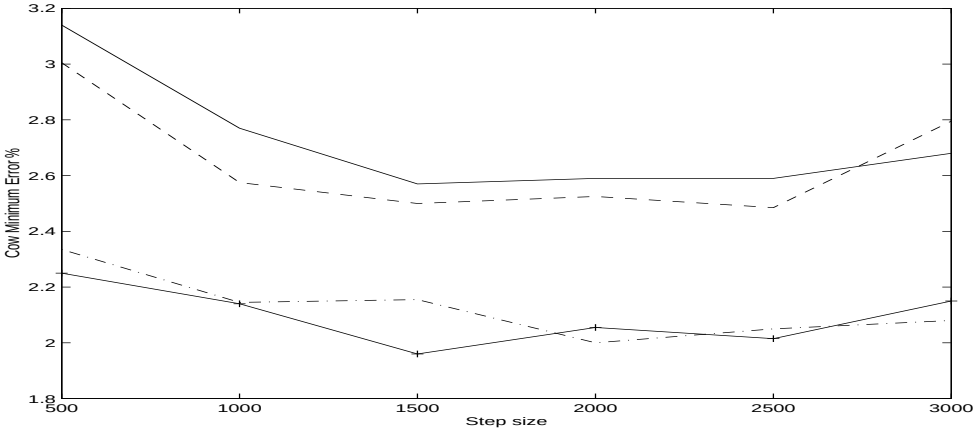


(c) Average error

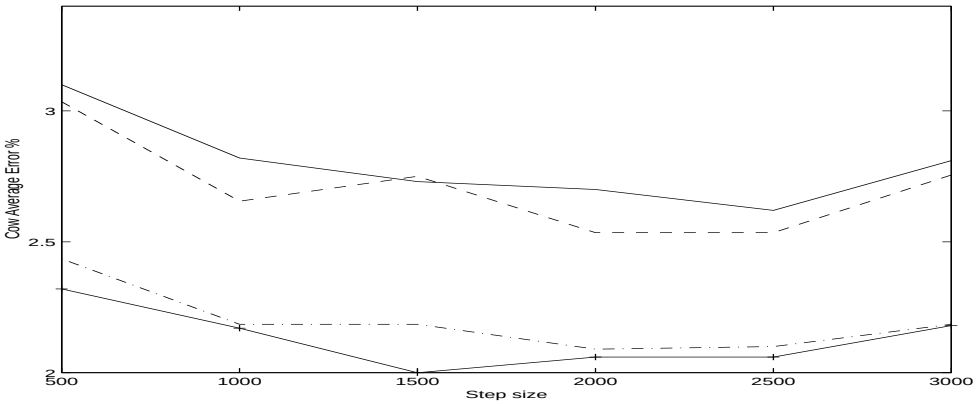
Figure 6.26: *Gaussian curvature error distribution of the bull*



(a) Maximum error



(b) Minimum error



(c) Average error

Figure 6.27: *Mean curvature error distribution of the bull*

sizes between 1000 units to 2000 units and for one iteration, the errors for maximum, minimum and average Gaussian curvatures are about 1.23%, 0.94% and 1.10%, respectively. After 17 iterations these errors reduce to about 0.92%, 0.52% and 0.7%, respectively. Figure 6.25 also shows the error distribution for estimating mean curvature of the dinosaur and these results also indicate that the errors for maximum, minimum and average curvature values are about 2.72%, 2.60% and 2.65%, respectively. After 17 iterations these errors reduce to about 2.0%, 1.9% and 1.92%, respectively.

Figure 6.26 shows the error distribution for estimating Gaussian curvatures of the bull when all possible directions are selected. Again the errors are reduced for step sizes between 1000 units to 2000 units and for one iteration, the errors for maximum, minimum and average Gaussian curvatures are about 1.29%, 1.0% and 1.1%, respectively. After 15 iterations these errors reduce to about 1.0%, 0.54% and 0.76%, respectively. Figure 6.27 further shows the error distribution for estimating mean curvature of the bull and these results also indicate that the errors for maximum, minimum and average curvature values are about 2.70%, 2.57% and 2.62%, respectively. After 15 iterations these errors reduce to about 2.07%, 1.96% and 2.0%, respectively.

The experiments indicate that estimation of Gaussian and mean curvatures on smoothed surfaces are quite accurate and not affected by the arbitrary direction of the first geodesic line when constructing semigeodesic coordinates. This technique is also applied to a number of incomplete surfaces [204].

6.2.2 Curvature Zero Crossing Contours

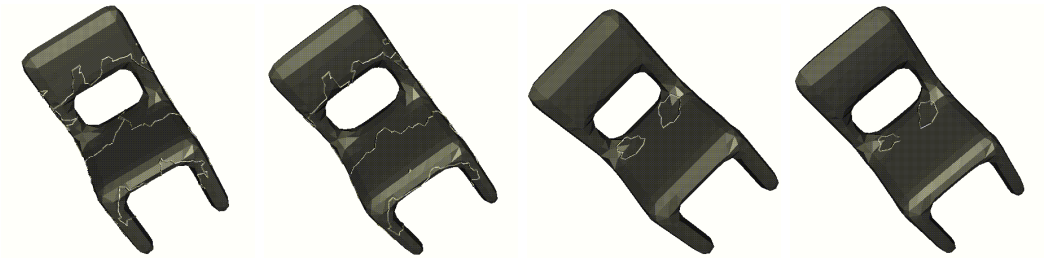
Next, the curvature zero crossing contours of these surfaces are found and displayed on the surface using VTK. Curvature zero crossing contours can be used for calculating torsion local maxima of absolute values. Figures 6.28(a) and (b) show Gaussian curvature zero crossing contours for the smoothed phone. Figures 6.28(c) and (d) show mean curvature zero crossing contours for the same object.

Figures 6.29(a) and (b) show the Gaussian curvature zero crossing contours for the

(a) 1st iteration (b) 3rd iteration (c) 1st iteration (d) 3rd iteration

Gaussian mean

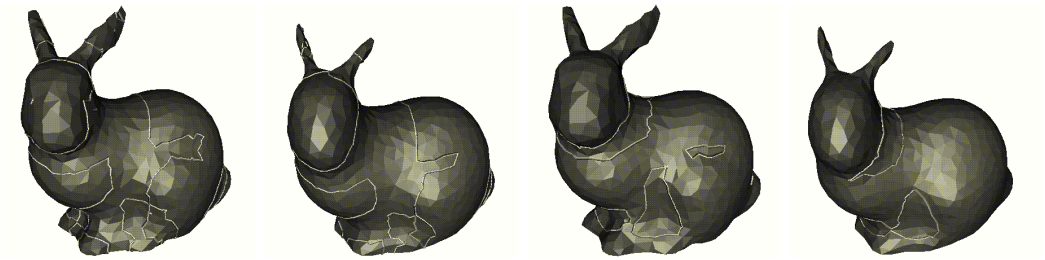
Figure 6.28: *Curvature zero crossing contours on the phone*



(a) 1st iteration (b) 2nd iteration (c) 1st iteration (d) 2nd iteration

Gaussian mean

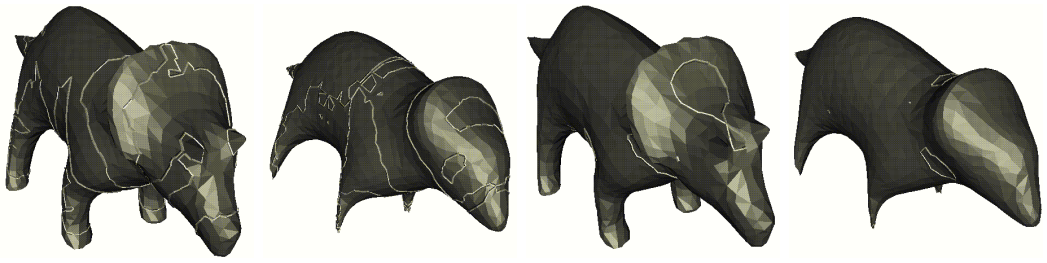
Figure 6.29: *Curvature zero crossing contours on the chair*



(a) 1st iteration (b) 6th iteration (c) 1st iteration (d) 6th iteration

Gaussian mean

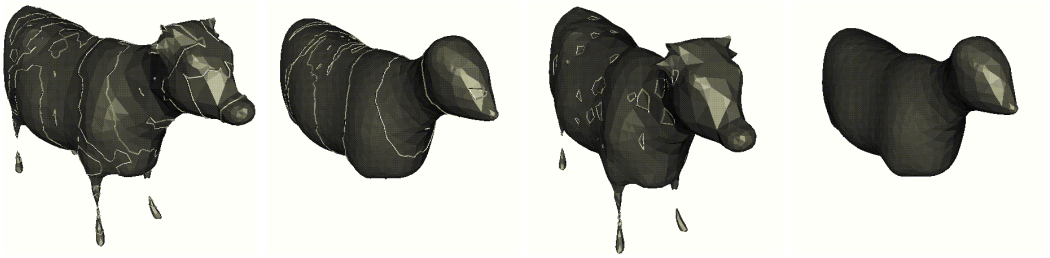
Figure 6.30: *Curvature zero crossing contours on the rabbit*



(a) 1st iteration (b) 6th iteration (c) 1st iteration (d) 6th iteration

Gaussian mean

Figure 6.31: *Curvature zero crossing contours on the dinosaur*



(a) 1st iteration (b) 6th iteration (c) 1st iteration (d) 6th iteration

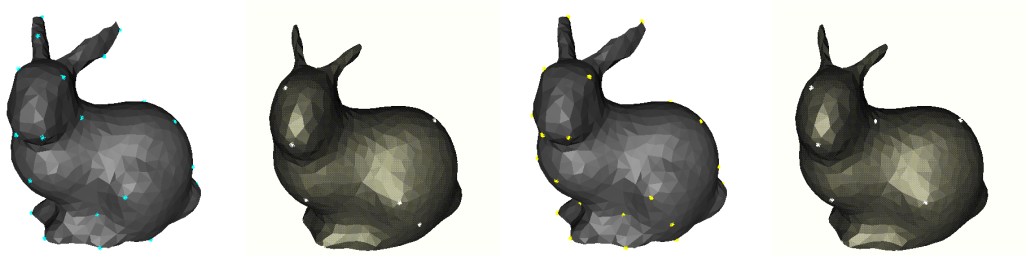
Gaussian mean

Figure 6.32: *Curvature zero crossing contours on the bull*

(a) phone (b) chair (c) phone (d) chair

Gaussian mean

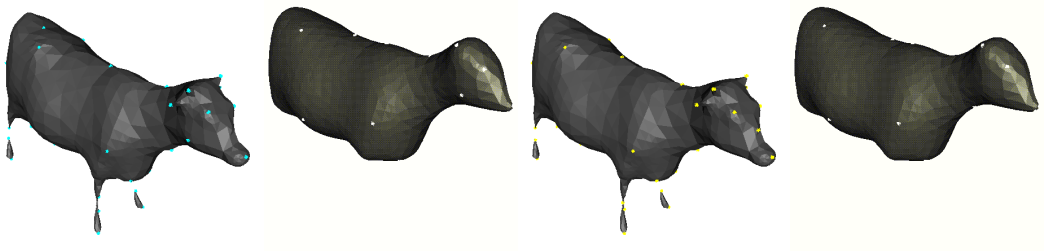
Figure 6.33: *Local maxima of absolute values of curvature*



(a) 1st iteration (b) 6th iteration (c) 1st iteration (d) 6th iteration

Gaussian Mean

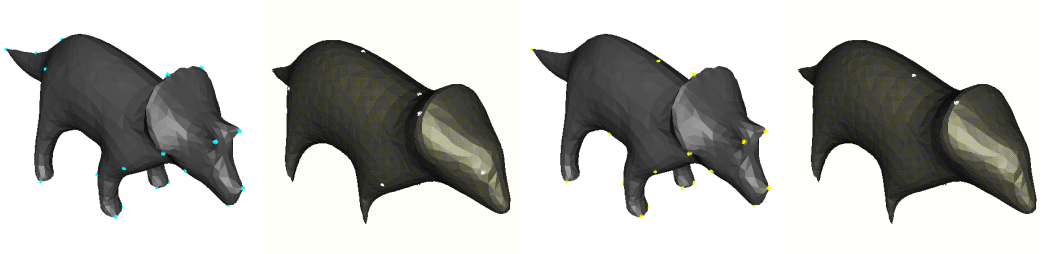
Figure 6.34: *Local maxima of absolute values of curvature on the rabbit*



(a) 1st iteration (b) 6th iteration (c) 1st iteration (d) 6th iteration

Gaussian Mean

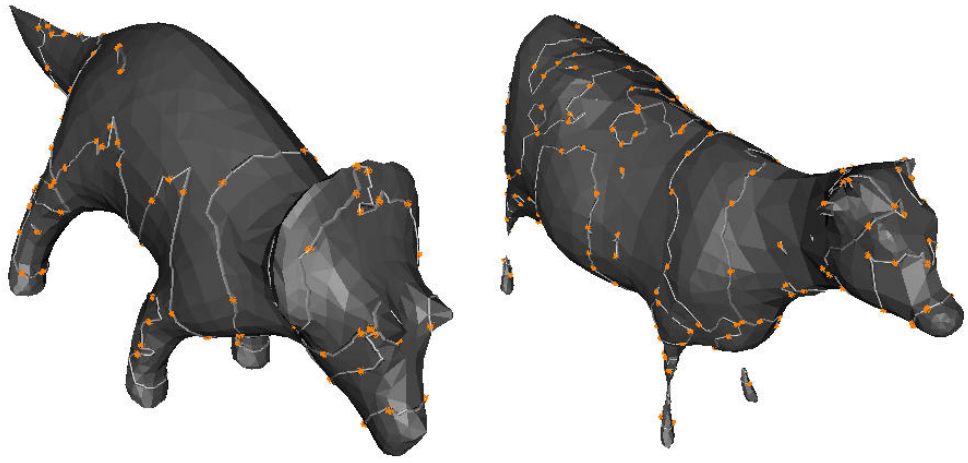
Figure 6.35: *Local maxima of absolute values of curvature on the bull*



(a) 1st iteration (b) 6th iteration (c) 1st iteration (d) 6th iteration

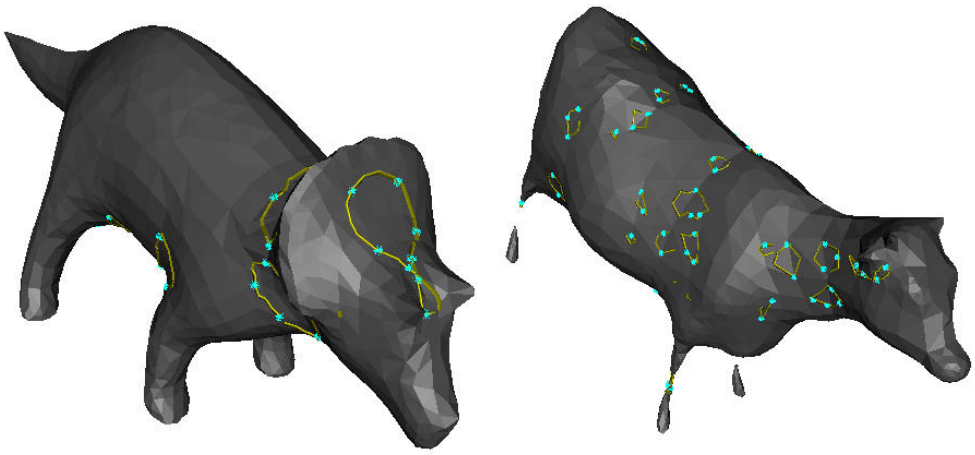
Gaussian Mean

Figure 6.36: *Local maxima of absolute values of curvature on the dinosaur*



(a) dinosaur (b) bull

Gaussian



(c) dinosaur (d) bull

mean

Figure 6.37: *Torsion of curvature zero crossing contours and the local maxima of absolute values*

smoothed chair, and Figures 6.29(c) and (d) show the mean curvature zero crossing contours for the same object. Figure 6.30 shows Gaussian and mean curvature zero crossing contours for the smoothed rabbit. The same experiments are also repeated for the dinosaur and bull, and these results are shown in Figure 6.31 and Figure 6.32 respectively. Notice that the number of curvature zero crossing contours is reduced, as the object is smoothed iteratively.

6.2.3 Local Maxima of Absolute Values of Curvature

The local maxima of absolute values of curvature for the smoothed phone are computed. The local maxima of absolute values of Gaussian curvature are displayed on the surface as shown in Figure 6.33(a). Figure 6.33(c) shows the local maxima of absolute values of mean curvature for the same object. Figure 6.33(b) shows the local maxima of absolute values of Gaussian curvature for the smoothed chair, and Figure 6.33(d) shows the local maxima of absolute values of mean curvature for the same object.

Figure 6.34 shows the local maxima of absolute values of Gaussian and mean curvatures for the rabbit. The local maxima of absolute values of Gaussian and mean curvatures for the dinosaur and bull are also shown in Figure 6.35 and Figure 6.36 respectively. All local maxima of absolute values of curvature are shown after smoothing. These features can be utilized by later processes for robust object recognition with occlusion.

6.3 Local Maxima of Absolute Values of Torsion

Local maxima of torsion absolute values of curvature zero crossing contours are alternative features that can be used for matching. Figures 6.37(a) and (c) show the local maxima of torsion absolute values of curvature zero crossing contours of the dinosaur for Gaussian and mean curvatures, respectively. Figures 6.37(b) and (d) show the results for the bull. These features are utilized by the surface matching.

Experiments show above local maxima of absolute values of curvature and torsion are stable against the transformations including rotation, translation and scaling. These features are tested under the conditions of the complete, partial and occluded 3-D meshes. Moreover, these features have been applied in the following object recognition and the results are stable.

6.4 Object Recognition

	Scale factor	Rotated in X°	Rotated in Y°	Rotated in Z°
brick[1]	5.0	30	30	30
chair	2.3	109	178	138
bull	9.9	332	258	40
cube	7.8	149	143	126
dinosaur	0.9	201	69	355
foot	0.5	45	11	23
head	0.7	309	241	206
leg[2]	6.3	34	120	275
rider[3]	0.3	11	349	189
phone	7.0	92	286	211
rabbit	0.1	6	6	6
torus[4]	1.1	74	304	132

Table 6.1: *Scale factors and rotation angles of scene objects.*

This section presents the matching results of algorithm applied to 3-D surfaces in a database. The database consists of several 3-D objects with arbitrary shapes. Given a 3-D surface in a scene at optimal scale, next step is to match the measurements taken at the scene to those stored in the hash table. In general, there are a number of different object models used for matching. It should be pointed out that most of the object models in the database correspond to real range data, which are shown previously in this chapter except those object models listed at the footnote (see

[1] [2] [3] [4] See Figures 6.40(a), (k), (n), (t).

Figures 6.40(a), (k), (n) and (t)). All these models are created by merging range images of real objects obtained from different viewpoints. The matching system is fast with matching times not exceeding 2-3 CPU seconds in each case.

					S	c	e	n	e	s			
	Score	brick	bull	chair	cube	dinosaur	foot	head	leg	phone	rabbit	rider	torus
	brick	107	0	1	68	0	0	0	0	0	0	0	1
	bull	80	**85**	76	67	54	63	56	72	96	55	48	79
	chair	83	21	68	49	17	20	21	21	37	19	16	40
M	cube	4	0	0	**100**	0	0	0	0	0	0	0	1
o	dinosaur	92	49	66	42	83	57	59	58	90	53	43	86
d	foot	0	1	1	0	1	91	2	2	5	1	1	5
e	head	62	19	33	28	21	31	87	20	50	28	17	58
l	leg	17	24	27	8	21	37	21	87	65	21	26	50
s	phone	69	29	50	59	29	46	39	50	116	0	30	97
	rabbit	71	33	47	64	32	37	44	34	65	**87**	27	66
	rider	26	41	52	28	35	45	39	55	78	37	59	72
	torus	47	2	8	64	3	9	8	7	25	6	4	113

Table 6.2: *Matching result of rotated and scaled object from the scenes against the models within the database. At the end of this geometric hashing stage, object models $\mathcal{M}$ bull, cube and rabbit are correctly recognized.*

While some object recognition systems have employed grey scale images as their input [161, 70], others have made use of range images [22, 79] to achieve recognition. Those systems have been designed for feature extraction and matching based on range images, and cannot cope with more general 3-D surfaces. In this research, the system goes further by accepting 3-D surfaces that are more general than range images. They are formed by merging two or more range images obtained from different viewpoints. This makes it possible to obtain more information about the objects to be recognized, and achieve more reliable recognition.

Once the local maxima of absolute values of curvature and torsion of each object

are obtained, they are then indexed in the hashing table as explained in Section 4.5. However, there is no exact correspondence between model vertices in the database and input scene object vertices. This is because the model and input objects are subjected to different levels of smoothing followed by decimation that modifies the vertex structure on those objects.

		Scenes											
	Score	brick	bull	chair	cube	dinosaur	foot	head	leg	phone	rabbit	rider	torus
	brick	**280**	-	-	-	-	-	-	-	-	-	-	-
	bull	10	-	43	-	277	6	-	123	327	-	224	-
	chair	44	-	**357**	-	-	-	-	-	-	-	-	-
M	cube	-	-	-	-	-	-	-	-	-	-	-	-
o	dinosaur	6	-	22	-	**3665**	-	24	28	180	-	76	50
d	foot	-	-	-	-	-	**35**	-	-	-	-	-	-
e	head	-	-	-	-	-	-	**1309**	-	-	-	-	-
l	leg	-	-	-	-	-	-	-	**1029**	-	-	-	-
s	phone	-	-	33	-	-	-	-	-	**9395**	-	130	246
	rabbit	-	-	13	-	-	-	-	-	-	-	-	-
	rider	-	-	19	-	-	-	-	-	-	-	**2081**	-
	torus	-	-	-	-	-	-	-	-	-	-	-	**4567**

Table 6.3: *The verification results of complete objects from table 6.2.*

The first experiment consists of applying arbitrary amounts of scaling and 3-D rotation to the model objects from database, these model objects become the scene objects (see Table 6.1), and to determine whether they can be recognized correctly by the system. In the first stage, geometric hashing is applied to the input scene object. If one of the models $\mathcal{M}$ receives a vote count that is substantially higher than the vote counts for all other objects, then $\mathcal{M}$ is selected as the correct object. Otherwise, two or more models receive high vote counts that are relatively similar. In this case, the system applies global verification only to the surviving models in order to select one of them. Table 6.2 shows the results of geometric hashing for the rotated and scaled scene objects. The numbers shown are the amount of triplets

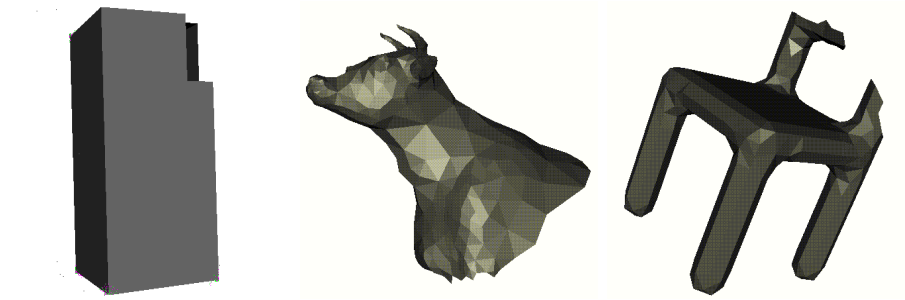
on each object model that receives votes. When the score number shown is greater than 100, some triplets receive more than one vote. It is because their matching values are near to a number of model's triplets. Table 6.3 shows the results of global verification. The numbers shown are the score in the largest clusters for each surviving object model. Blanks with dash indicate that no verification is considered necessary for the object models. The results show that all input scene objects are recognized correctly by the system.

		Scene											
	Score	brick	bull	chair	cube	dinosaur	foot	head	leg	phone	rabbit	rider	torus
	brick	100	0	2	37	0	0	0	0	0	0	0	0
	bull	80	46	56	50	40	66	56	0	89	43	50	89
	chair	95	15	70	50	13	25	19	0	38	20	19	37
M	cube	10	0	0	100	0	0	0	0	0	0	0	0
o	dinosaur	75	40	59	37	68	60	62	50	87	46	47	93
d	foot	5	1	1	0	0	38	0	0	5	1	1	6
e	head	70	18	34	50	22	28	65	0	58	25	24	49
l	leg	20	20	23	12	14	26	21	50	56	21	23	67
s	phone	75	32	51	25	24	35	38	0	118	41	34	104
	rabbit	80	25	46	75	30	40	42	0	68	52	33	62
	rider	25	29	41	12	28	49	54	0	67	30	48	84
	torus	45	3	10	75	3	4	4	0	22	8	5	112

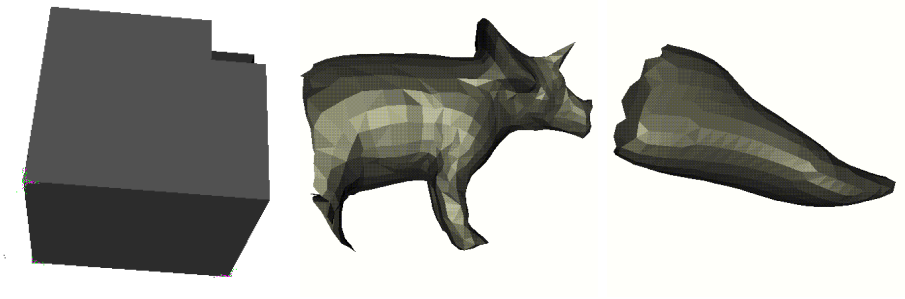
Table 6.4: *Matching results of partial objects from the scenes against the models from database.*

The second experiment makes use of incomplete or partial scene objects that are again subjected to arbitrary amounts of scaling and 3-D rotation. In order to obtain partial scene objects, up to 60% of connected vertices are removed from database model objects. The following steps are utilized for vertex removal:

- A vertex chosen randomly is removed from a given surface.
- Following that, all of its neighbouring vertices within a distance (radius) are

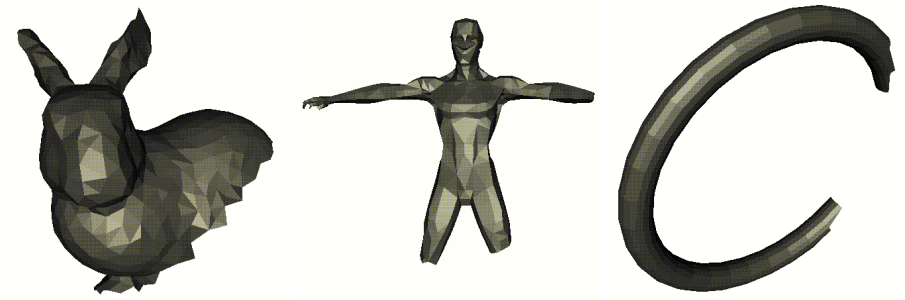


(a) brick (b) bull (c) chair



(d) cube (e) dinosaur (f) foot

(g) head (h) leg (i) phone



(j) rabbit (k) rider (l) torus

Figure 6.38: *Partial scene objects*

removed from that surface.

Figure 6.38 shows the partial scene objects used in this experiment. After the features of these objects are extracted, the system applies geometric hashing to match the scene to all database models followed by global verification to the surviving models. Again, all input scene objects are correctly recognized by the system. This experiment shows that partial scene objects can also be matched successfully to the database models by the system. Table 6.4 shows the results of geometric hashing for a number of rotated and scaled incomplete scene objects, and Table 6.5 shows the results of global verification.

		Scenes											
	Score	brick	bull	chair	cube	dinosaur	foot	head	leg	phone	rabbit	rider	torus
Models	brick	**46**	-	-	-	-	-	-	-	-	-	-	-
	bull	-	**28**	11	-	41	7	4	-	12	8	35	-
	chair	12	-	**196**	-	-	-	-	-	-	-	-	-
	cube	-	-	-	**240**	-	-	-	-	-	-	-	19
	dinosaur	-	6	9	-	**1296**	4	4	1	-	5	17	13
	foot	-	-	-	-	-	**8**	-	-	-	-	-	-
	head	-	5	-	-	-	-	**24**	-	-	5	6	-
	leg	-	6	-	-	-	-	-	**1**	-	-	15	-
	phone	-	17	19	-	-	-	6	-	**781**	12	61	86
	rabbit	-	5	6	2	-	2	2	-	-	**312**	11	-
	rider	-	8	6	-	-	4	4	-	-	13	**370**	20
	torus	-	-	-	72	-	-	-	-	-	-	-	**516**

Table 6.5: *The verification results of partial objects from table 6.4.*

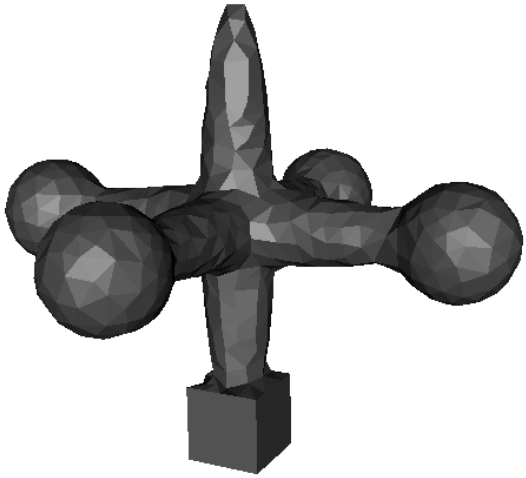
In the third experiment, three complex scenes are created each consisting of two or more objects. The scene objects have different digitizations and resolutions to their models in the database. Figure 6.39(a) shows the kitchenware scene. This scene contains a dish, a teapot, a spatula and a rolling pin. Figure 6.39(b) shows the bull-rider scene. This scene contains a bull and a rider. Figure 6.39(c) shows the

space-station scene. This scene contains a station and a spacecraft attached to it.

As in the earlier experiments, the system applies geometric hashing to all scene models. Then they match against the database as shown in Figure 6.40; followed by global verification to the surviving models (see Table 6.6). In the bull-rider scene, bull receives the highest score at global verification. In the kitchenware scene, dish has the highest score. In the space-station scene, the station scores the highest. This experiment shows that scenes depicting occlusion can also be recognized satisfactorily by the system.

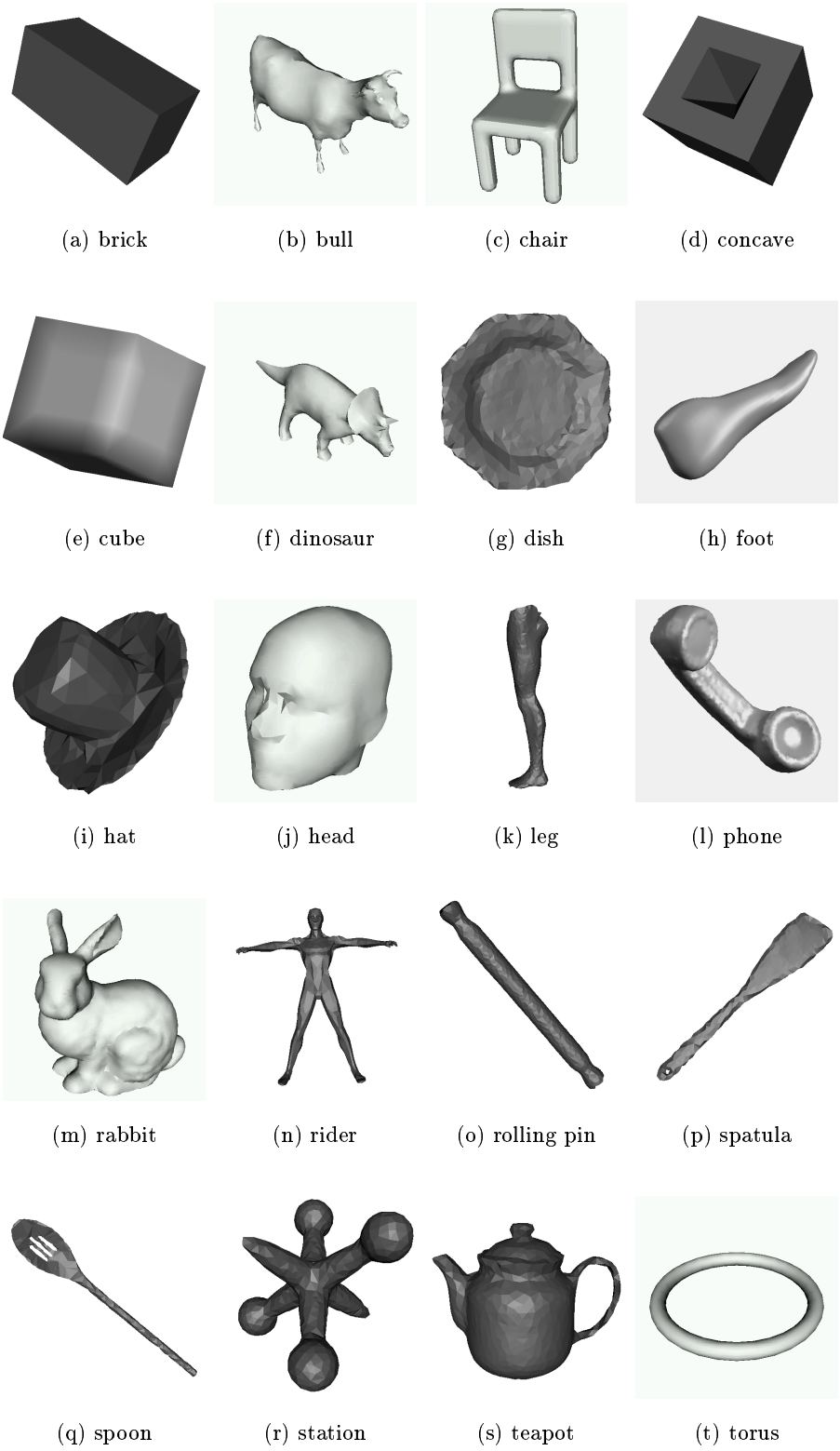
(a) Kitchenware

(b) Bull-rider



(c) Space-station

Figure 6.39: *Scenes for object recognition*



(a) brick (b) bull (c) chair (d) concave

(e) cube (f) dinosaur (g) dish (h) foot

(i) hat (j) head (k) leg (l) phone

(m) rabbit (n) rider (o) rolling pin (p) spatula

(q) spoon (r) station (s) teapot (t) torus

Figure 6.40: *3-D object models in database*

	Scenes					
	Geometric hashing score			Global verification score		
	Bull-rider	Kitchenware	Space-station	Bull-rider	Kitchenware	Space-station
brick	0	0	1	-	-	-
bull	54	47	51	**3699**	76	132
chair	17	15	26	-	-	35
concave	0	0	0	-	-	-
cube	0	0	0	-	-	-
dinosaur	41	39	53	1107	32	118
dish	56	47	41	1573	**241**	44
M foot	0	1	2	-	-	-
o hat	16	16	19	-	-	-
d head	17	17	26	-	-	14
e leg	19	18	28	-	27	75
l phone	23	24	46	-	60	400
s rabbit	27	27	36	113	20	31
rider	33	29	38	492	31	68
rolling pin	28	24	49	340	31	274
spatula	11	9	17	-	-	-
spoon	12	10	12	-	-	-
station	11	15	55	-	-	**2806**
teapot	26	28	45	100	18	27
torus	2	3	11	-	-	-

Table 6.6: *Recognition results.*

Chapter 7

Conclusions and Future Work

7.1 Conclusions

A novel technique for multi-scale representation and recognition of 3-D surface using geometric invariants was presented. The main goal of the research was to be able to recognize any of the objects from arbitrary viewpoints. The results shown demonstrate that the method based on multi-scale representations and geometric invariants was robust with respect to noise and local distortions of underlying shape as well as invariance with respect to shape-preserving transformations. Using these powerful representations, the reliable object models features were obtained for matching and the scene objects were recognized satisfactorily by the system.

One of the major strengths of this research was using the semigeodesic coordinates for local parametrization. A geodesic from the local origin was first constructed in an arbitrary direction, and then the second family of lines were sampled to create the semigeodesic coordinates. During the diffusion process, 3-D surfaces were also sampled locally using an appropriate step size. These 3-D surfaces were then smoothed iteratively using a 2-D Gaussian filter. The smoothing eliminated surface noise, small/unnecessary surface details, and resulted in simplification of object shape. Meanwhile, the surface Gaussian and mean curvatures were estimated accurately at multiple scales. All convex areas of the surface indicated high Gaussian curvature values, whereas the concave areas indicated low Gaussian curvature values

and the curvature values of flat areas were close to zero. For the mean curvature, all convex areas had low mean curvature values but the concave areas had high values. Similar to Gaussian curvature, the curvature values of flat areas were close to zero. Furthermore, Gaussian and mean curvature zero crossing contours were also recovered. Results indicated that as the surface was smoothed iteratively, the number of curvature zero crossing contours was reduced. Then, the local maxima of absolute values of Gaussian and mean curvatures as well as the torsion of their zero crossing contours were located. These features were utilized for the process of robust object recognition.

Other strengths of this research were to be able to recognize free-form objects, using 3-D models and under different viewing conditions, based on the geometric hashing algorithm and global verification. The matching features algorithm used the hash table prepared in an off-line stage. This technique was shown to be useful for partially occluded objects. To verify matching, 3-D scaling, translation and rotation parameters were used for global verification. Results indicated that the technique was robust and invariant to the transformations.

Finally, this research can have considerable impact on other researchers in the field. Since the research results have been published in several journals and conferences [82, 83, 120, 121, 122, 124, 123, 125, 203, 204, 205], it should therefore benefit the computer vision community in general.

7.2 Future Work and Applications

Chapter 6 shows that the 3-D object recognition results are good. Additional features such as colour, texture and motion can further enhance the process of object recognition. Combination of colour, texture, motion and geometric features can be used to match objects in a large 3-D digital library database. Many 3-D objects have similar shapes. For example, the previous matching results show that bull and dinosaur are close because both objects have the same common component features such as sharp horns, four legs and a large body. A content-based 3-D database

retrieval system is useful so that these common component features can be classified and indexed into a relational database. This approach can increase the speed of matching occluded objects and further improve the robustness of object recognition.

In recent years, there have been many major breakthroughs in image processing research. It is now possible to produce small low-cost cameras and 3-D range scanners that are useful for industrial automation, mobile robot navigation, guided neurosurgery, etc. Considering the above, we suggest the following applications:

- Industrial automation, automobile manufacturing, aerospace industry and architecture often deal with 3-D objects. Therefore, 3-D object recognition used in this research can be applied to those areas. For example, an intelligent industrial robot with object recognition capability can be used to perform highly critical tasks such as aircraft fuselage crack and structural inspection, reverse engineering, etc.

- Outdoor navigation of a civilian vehicle in a structured road environment such as highways and city roads requires a 3-D recognition system. The recognition of objects on the road must satisfy many challenging conditions such as unforeseeable road obstacles, road safety criteria, uncalibrated camera, hostile and noisy backgrounds. Therefore, the 3-D object recognition system used in this research can recognize vehicles and pedestrians. It can also calculate the safe distance from road obstacles and then apply an appropriate speed limit.

- Sometimes during a medical operation, a patient needs to be registered to diagnostic imagery obtained by structured light extraction of the skin surface, followed by the insertion of marker pins to be tracked by video processing. This task calls for a registration process stable with respect to noise arising from different modes of surface extraction as well as temporal changes in surface geometry exhibited by human beings. Hence, the technique of curvature estimation used in this research is ideal for such navigational systems for neurosurgical and other surgical operations.

- Finally, this research of multi-scale representation and recognition of 3-D sur-

faces using geometric invariants can be used to create a sophisticated real-time reliable mobile robot navigation system. Such system can interact with different 3-D objects. It can also work in different environments such as laboratory and industrial complex.

Appendix A

Differential Geometry

Differential Geometry is the theory of the properties of configurations in the neighbourhood of one of its general elements. It is a study, by means of differential calculus of properties of the general elements of curves and surfaces that are invariant under rigid bodies.

A.1 Space Curves

A space curve $\mathcal{C}$ is defined as a C^2 mapping $u \to \mathbf{M}(u)$ from an interval of $\mathbb{R}$ into $\mathbb{R}^3$. Parameter u is the arc length S of $\mathcal{C}$. The 3-D Frenet formulas are

$$\begin{aligned}
\frac{d\mathbf{M}}{dS} &= \mathbf{T} \\
\frac{d\mathbf{T}}{dS} &= \kappa\mathbf{N} \\
\frac{d\mathbf{N}}{dS} &= -\kappa\mathbf{T} + \tau\mathbf{B} \\
\frac{d\mathbf{B}}{dS} &= -\tau\mathbf{N}
\end{aligned}$$

where $\mathbf{T}$ is the tangent, $\mathbf{N}$ the normal and $\mathbf{B}$ the binormal unit vectors to $\mathcal{C}$ at the point under consideration, κ the curvature, and τ the torsion.

A.2 Surface Patches

A surface patch $\mathcal{S}$ is defined as a C^2 mapping $(u, v) \to \mathbf{P}(u, v)$ from an open set of $\mathbb{R}^2$ into $\mathbb{R}^3$. Such a patch is intrinsically characterized, up to a rigid motion, by two quadratic forms, called the two fundamental forms, which are defined at every point of the patch.

The first quadratic form Φ_1 defines the length of a vector in the tangent plane $T(P)$. More precisely, the two vectors $\mathbf{P}_u = \frac{\partial \mathbf{P}}{\partial u}$ and $\mathbf{P}_v = \frac{\partial \mathbf{P}}{\partial v}$ are parallel to this plane and define therein a system of coordinates. Each vector in the tangent plane can be defined as a linear combination $\lambda \mathbf{P}_u + \mu \mathbf{P}_v$. Its squared length is given by the value of the first fundamental form Φ_1

$$\Phi_1(\lambda \mathbf{P}_u + \mu \mathbf{P}_v) = \lambda^2 E + 2\lambda\mu F + \mu^2 G$$

with the following definitions for E, F and G

$$E = \|\mathbf{P}_u\|^2$$
$$F = \mathbf{P}_u \cdot \mathbf{P}_v$$
$$G = \|\mathbf{P}_v\|^2$$

Moreover, the normal $\mathbf{N}_p$ to $\mathcal{S}$ is parallel to the cross-product $\mathbf{P}_u \times \mathbf{P}_v$ whose length is the quantity $H = \sqrt{EG - F^2}$.

The second fundamental quadratic form Φ_2 is related to curvature. For a vector $\mathbf{x} = \lambda \mathbf{P}_u + \mu \mathbf{P}_v$ in the tangent plane, all curves drawn on $\mathcal{S}$ tangent to $\mathbf{x}$ at $\mathbf{P}$ are considered. All of these curves have all the same normal curvature, the ratio $\frac{\Phi_1(x)}{\Phi_2(x)}$, with the following definitions

$$\Phi_2(\lambda \mathbf{P}_u + \mu \mathbf{P}_v) = \lambda^2 L + 2\lambda\mu M + \mu^2 N$$

and

$$L = \frac{\partial^2 \mathbf{P}}{\partial u^2} \cdot \frac{\mathbf{N}_p}{\|\mathbf{N}_p\|}$$
$$M = \frac{\partial^2 \mathbf{P}}{\partial u \partial v} \cdot \frac{\mathbf{N}_p}{\|\mathbf{N}_p\|}$$
$$N = \frac{\partial^2 \mathbf{P}}{\partial v^2} \cdot \frac{\mathbf{N}_p}{\|\mathbf{N}_p\|}$$

It is important to study the invariants of Φ_2, i.e., quantities which do not depend upon the parametrization (u, v) of $\mathcal{S}$. Φ_2 defines a linear mapping $T(P) \to T(P)$ by $\Phi_2(\mathbf{x}) = \psi(\mathbf{x}) \cdot \mathbf{x}$. The invariants of Φ_2 are those of shape operator ψ.

A.2.1 Principal Directions

The principal directions are the eigenvectors of ψ. Their coordinates (λ, μ) in the coordinate system $(\mathbf{P}_u, \mathbf{P}_v)$ are solutions of the following equation

$$(FL - EM)\lambda^2 + (GL - EN)\lambda\mu + (GM - FN)\mu^2 = 0$$

This yields the following possible values for λ and μ

$$\lambda = EN - GL \pm \sqrt{(GL - EN)^2 - 4(FL - EM)(GM - FN)}$$
$$\mu = 2(FL - EM)$$

A.2.2 Principal Curvatures

The principal curvatures k_1 and k_2 are the eigenvalues of ψ. They are solutions of the following quadratic equation

$$(EG - F^2)k^2 - (LG + EN - 2FM)k + LN - M^2 = 0$$

In particular, their product K and their half-sum H are the Gaussian and mean curvatures of $\mathcal{S}$

$$K = \frac{LN - M^2}{EG - F^2}$$
$$H = \frac{LG + EN - 2FM}{2(EG - F^2)}$$

Bibliography

[1] Sadegh Abbasi, Farzin Mokhtarian, and Josef Kittler. Shape similarity retrieval using a height adjusted curvature scale space. *Proc. International Conference on Visual Information Systems*, pages 173–180, 1997.

[2] Martha L. Abell and James P. Braselton. *Differential Equations with Maple V.* Academic Press, second edition, 2000. ISBN 0120415607.

[3] Maher Al-Khaiyat and Farhad Kamangar. Planar curve representation and matching. *Proc. of the* 9^{th} *British Machine Vision Conference*, 1:174–184, September 1998. ISBN 1-901725-04-9.

[4] T. D. Alter and W. Eric L. Grimson. Fast and robust 3-D recognition by alignment. *IEEE International Conference on Computer Vision*, pages 113–120, 1993.

[5] D. H. Ballard. Generalizing the Hough transform to detect arbitrary shapes. *Pattern Recognition*, 13:111–122, 1981.

[6] Christopher F. Barnes, Syed A. Rizvi, and Nasser M. Nasrabadi. Advances in residual vector quantization: A review. *IEEE Transactions on Image Processing*, 5(2):226–262, February 1996.

[7] Bonen Basri and Shimon Ullman. The alignment of objects with smooth surfaces. *IEEE Second International Conference on Computer Vision*, pages 482–488, 1988.

[8] Paul J. Besl. The free-form surface matching problem. *In Machine Vision for Three-dimensional Scenes*, pages 25–71, 1990. H. Freeman, ed.

[9] Paul J. Besl and Ramesh C. Jain. Three dimensional object recognition. *ACM Computing Surveys*, 17:75–145, 1985.

[10] Paul J. Besl and Ramesh C. Jain. Invariant surface characteristics for 3-D object recognition in range images. *Computer Vision, Graphics and Image Processing*, 33:33–80, 1986.

[11] Paul J. Besl and N. D. McKay. A method for registration of 3-D shapes. *IEEE Trans Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

[12] M. Brady, J. Ponce, A. Yuille, and H. Asada. Describing surface. *Computer Vision, Graphics and Image Processing*, 32:1–28, 1985.

[13] C. H. Brechbuhler, G. Gerig, and O. Kubler. Parametrization of closed surfaces for 3-D shape description. *Computer Vision and Image Understanding*, 61(2):154–170, March 1995.

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

[15] R. P. Burn. *GROUPS a path to geometry.* Cambridge University Press, 1984. ISBN 0521347939.

[16] Peter Buser and Detlef Gromoll. *On the almost negatively curved 3-sphere*, volume I0288834. Springer Verlag, 1988. Edited T. Sunada.

[17] Jin-Long Chen and George C. Stockman. 3-D free-form object recognition using indexing by contour features. *Computer Vision and Image Understanding*, 71(3):334–355, September 1998.

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

[19] Xin Chen and Francis Schmitt. Vision-based construction of CAD models from range images. *IEEE International Conference on Computer Vision*, pages 129–136, April 1993.

[20] Shing-Shen Chern, Philip Hartman, and Aurel Wintner. On isothermic coordinates. *Commentaries Mathematical helvetici*, 28:301–309, 1954.

[21] R. T. Chin and C. R. Dyer. Model-based recognition in robot vision. *ACM Computing Surveys*, 18:67–108, 1986.

[22] C. S. Chua and R. Jarvis. Point signatures: A new representation for 3-D object recognition. *International Journal of Computer Vision*, 25(1):63–85, 1997.

[23] Robert T. Collins, Christopher O. Jaynes, Yong-Qing Cheng, Xiaoguang Wang, and Frank Stolle. The ascender system: Automated site modelling from multiple aerial images. *Computer Vision and Image Understanding*, 72(2):143–162, November 1998.

[24] Robert C. Crida, Andrew J. Stoddart, and John Illingworth. Using PCA to model shape for process control. *International Conference on Recent Advances in 3-D Digital Imaging and Modelling*, 1997.

[25] Tony Crilly. Arthur Cayley as Sadleirian professor: A glimpse of mathematics teaching at 19th-century Cambridge. *Historia Mathematica*, 26(Article ID hmat.1999.2233):125–160, 1999. 0315-0860/99.

[26] H. Delingette, M. Hebert, and K. Ikeuchi. A spherical representation or the recognition of curved objects. *Proc. of the International Conference on Computer Vision*, pages 103–112, 1993.

[27] Mathieu Desbrun, Mark Meyer, Peter Schroder, and Alan H. Barr. Implicit fairing of irregular meshes using diffusion and curvature flow. *Computer Graphics Proceedings*, pages 317–324, 1999.

[28] Sven J. Dickinson, Henrik I. Christensen, John K. Tsotsos, and Goran Olofsson. Active object recognition integrating attention and viewpoint control. *Computer Vision and Image Understanding*, 67(3):239–260, September 1997.

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

[30] M. P. DoCarmo. *Differential Geometry of Curves and Surfaces*. Prentice Hall, 1976.

[31] Gregory Dudek and John K. Tsotsos. Shape representation and recognition from multi-scale curvature. *Computer Vision and Image Understanding*, 68(2):170–189, November 1997.

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

[33] James H. Elder and Steven W. Zucker. Local scale control for edge detection and blur estimation. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, 20(7):699–716, July 1998.

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

[35] M. Faraklioti and M. Petrou. Recovering more classes than available bands for mixed pixels in remote sensing. *Proc. of the* 9^{th} *British Machine Vision Conference*, 2:690–699, September 1998. ISBN 1-901725-04-9.

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

[37] Marta Fidrich and Jean-Philippe Thirion. Stability of corner points in scale space: The effects of small nonrigid deformations. *Computer Vision and Image Understanding*, 72(1):72–83, October 1998.

[38] Patrick J. Flynn and Anil K. Jain. BONSAI: 3-D object recognition using constrained search. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, 13(10):1066–1074, October 1991.

[39] Patrick J. Flynn and Anil K. Jain. Three dimensional object recognition. *Handbook of Pattern Recognition and Image Processing*, 2:497–541, 1994. T. Y. Young, ed.

[40] James D. Foley and Andries van Dam. *Fundamentals of Interactive Computer Graphics*. Addison-Wesley, 1984.

[41] James D. Foley, Andries van Dam, Steven K. Feiner, and John Hughes. *Computer Graphics Principles and Practice*. Addison-Wesley, 1996. ISBN 0201848406.

[42] K. Fukaya. Hausdorff convergence of Riemannian manifolds and its applications. *Advanced Studies in Pure Mathematics, Recent Topics in Differential and Analytic Geometry*, 18-I:143–238, 1990.

[43] Donald Geman and Bruno Jedynak. Shape recognition and twenty questions. *INRIA*, 2155(Programme 4 Robotique):1–31, November 1993.

[44] Abraham Goetz. *Introduction to Differential Geometry*. Addison-Wesley, 1970.

[45] Duncan Graham-Rowe. Now you see it. *New Scientist*, 7th November 1998.

[46] Alfred Gray. *Modern Differential Geometry of Curves and Surfaces*. CRC Press, 1993. ISBN 0849378729.

[47] W. E. L. Grimson, T. Lozano-Perez, W. M. Wells, G. J. Ettinger, S. J. White, and R. Kikinis. An automatic registration method for frameless stereotaxy, image guided surgery, and enhanced reality visualization. *IEEE Transactions on Medical Imaging*, 15(2):129–140, April 1994.

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

[49] W. Eric L. Grimson and Tomas Lozano-Perez. Localizing overlapping parts by searching the interpretation tree. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, PAMI-9(4):469–482, July 1987.

[50] Patrick Gros, Olivier Bournez, and Edmond Boyer. Using local planar geometric invariants to match and model images of line segments. *Computer Vision and Image Understanding*, 69(2):135–155, 1998.

[51] Bilge Gunsel, Anil K. Jain, and Erdal Panayirci. Reconstruction and boundary detection of range and intensity images using multi-scale MRF representations. *Computer Vision and Image Understanding*, 63(2):353–366, March 1996.

[52] Alok Gupta and Ruzena Bajcsy. Surface and volumetric segmentation of range images using biquadrics and superquadrics. *Proc. Eleventh International Conference on Pattern Recognition*, 001:158–162, August 1992. 0-8186-2910-X.

[53] Alok Gupta, L. Bogoni, and Ruzena Bajcsy. Quantitative and qualitative measures for the evaluation of the superquadric models. *IEEE Workshop on Interpretation of 3-D Scenes*, pages 162–169, 1989.

[54] Tony Heap and David Hogg. Automated pivot location for the Cartesian-polar hybrid point distribution model. *Image and Vision Computing*, 14(8):97–106, 1996.

[55] Mike Heath, Sudeep Sarkar, Thomas Sanocki, and Kevin Bowyer. Comparison of edge detectors. *Computer Vision and Image Understanding*, 69(1):38–54, January 1998.

[56] Siegfried Heitz. *Coordinates in Geodesy.* Springer-Verlag, 1985. ISBN 354050088X.

[57] Mats Henricson and Erik Nyquist. *Industrial Strength C++.* Prentice Hall, 1997. ISBN 0131209655.

[58] Adrian Hilton, Andrew J. Stoddart, John Illingworth, and T. Windeatt. Marching triangles: Range image fusion for complex object modelling. *Proc of IEEE International Conference on Image Processing*, 2:381–384, 1996.

[59] Adrian Hilton, Andrew J. Stoddart, John Illingworth, and T. Windeatt. Reliable surface reconstruction from multiple range images. *Proc. European Conference on Computer Vision*, I:117–126, 1996. 3-540-61122-3.

[60] Adam Hoover, Dmitry Goldgof, and Kevin W. Bowyer. The space envelope: A representation for 3-D scenes. *Computer Vision and Image Understanding*, 69(3):310–329, March 1998.

[61] H. Hoppe, T. DeRose, T. Duchamp, J. MchDonald, and W. Stuetzle. Mesh optimization. *Computer Graphics Proceedings*, 1993.

[62] Hugues Hoppe. Progressive meshes. *SIGGRAPH 96, Computer Graphics Proceedings*, pages 99–106, August 1996.

[63] Berthold Klaus Paul Horn. Extended gaussian images. *Proceedings of The IEEE*, 72(12), December 1984.

[64] Berthold Klaus Paul Horn. *Robot Vision.* MIT Press, 1986. ISBN 0262081598.

[65] Berthold Klaus Paul Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of Optical Society of America*, 4(4):629–642, April 1987.

[66] Berthold Klaus Paul Horn, Hugh M. Hilden, and Shahriar Negahdaripour. Closed-form solution of absolute orientation using orthonormal matrices. *Journal of Optical Society of America*, 5(7):1127–1135, July 1988.

[67] Berthold Klaus Paul Horn and E. J. Weldon. Filtering closed curves. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, PAMI-8(5):665–668, September 1986.

[68] Cay S. Horstmann and Gary Cornell. *Core Java Fundamentals*, volume 1 of *The Sunsoft Press Java.* Prentice Hall, 1.1 edition, 1997. ISBN 0137669577.

[69] John Hubbard. *Programming with C++.* Schaum's Outline Series. McGraw-Hill, 1996. ISBN 0070308373.

[70] D. P. Huttenlocher and S. Ullman. Object recognition using alignment. *Proc. International Conference on Computer Vision*, pages 102–111, 1987.

[71] D. P. Huttenlocher and S. Ullman. Recognizing solid objects by alignment. *Proc. Image Understanding Workshop*, pages 1114–1124, 1988.

[72] John Illingworth, Josef Kittler, and J. Princen. Shape detection using the adaptive Hough transform. *Real-Time Object Measurement and Classification Edited by Anil K. Jain*, F42:119–142, 1988.

[73] Anil K. Jain and Chitra Dorai. 3-D object recognition: Representation and matching. *CSIRO MATHS and STATS*, 1998.

[74] Anil K. Jain and Richard Hoffman. Evidence-based recognition of 3-D objects. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, 10(6):783–801, November 1988.

[75] Ramesh Jain, Rangachar Kasturi, and Brian G. Schunck. *Machine Vision.* Computer Science. McGraw-Hill, 1995. ISBN 0-07-113407-7.

[76] Ben K. Jang and Roland T. Chin. Morphological scale space for 2-D shape smoothing. *Computer Vision and Image Understanding*, 70(2):121–141, May 1998.

[77] Alan Jeffrey. *Mathematics for Engineers and Scientists.* Nelson, 1976. ISBN 0177716045.

[78] Andrew E. Johnson and Martial Hebert. Control of polygonal mesh resolution for 3-D computer vision. *Graphical Models and Image Processing*, 60(Article no IP980474):261–285, 1998.

[79] Andrew E. Johnson and Martial Hebert. Using spin images for efficient object recognition in cluttered 3-D scenes. *IEEE Trans Pattern Analysis and Machine Intelligence*, 21(5):433–449, 1999.

[80] S. B. Kang and K. Ikeuchi. The complex EGI, a new representation for 3-D pose determination. *IEEE Trans Pattern Analysis and Machine Intelligence*, 15:707–721, 1993.

[81] D. Keren, D. Cooper, and J. Subrahmonia. Describing complicated objects by implicit polynomials. *IEEE Trans Pattern Analysis and Machine Intelligence*, 16:38–53, 1994.

[82] Nasser Khalili, Farzin Mokhtarian, and P. Yuen. Free-form surface description in multiple scales: Extension to incomplete surfaces. *Proc. International Conference on Computer Analysis of Images and Patterns*, pages 283–300, 1999.

[83] Nasser Khalili, Farzin Mokhtarian, and P. Yuen. Recovery of curvature and torsion features from free-form 3-D meshes at multiple scales. *Proc. Asian Conference on Computer Vision*, pages 1070–1075, 2000.

[84] Ron Kimmel. Intrinsic scale space for images on surfaces: The geodesic curvature flow. *Graphical Models and Image processing*, 59(5):365–372, September 1997.

[85] Donald E. Knuth. *Fundamental Algorithms*, volume 1. Addison-Wesley, second edition, 1975.

[86] Jan J. Koenderink. *Solid Shape*. MIT Press, 1990. ISBN 026211139X.

[87] Jan J. Koenderink and A. J. vanDoorn. Internal representation of solid shape with respect to vision. *Biological Cybernetics*, 32(4):211–216, 1979.

[88] Stanislav Kovacic, Ales Leonardis, and Franjo Pernus. Planning sequences of views for 3-D object recognition and pose determination. *Pattern Recognition*, 31(10):1407–1417, 1998.

[89] Erwin Kreyszig. *Advanced Engineering Mathematics*. Wiley, 1972. ISBN 0471507296.

[90] Erwin Kreyszig. *Differential Geometry*. Dover Publications, 1991. ISBN 0486667219.

[91] Yehezkel Lamdan, J. T. Schwartz, and Haim J. Wolfson. On recognition of 3-D objects from 2-D images. *Proc. IEEE International Conference on Robotics and Automation*, pages 1407–1413, 1988.

[92] Yehezkel Lamdan and Haim J. Wolfson. Geometric hashing: A general and efficient model-based recognition scheme. *IEEE International Conference on Computer Vision*, pages 238–249, 1988.

[93] Yehezkel Lamdan and Haim J. Wolfson. Object recognition by affine invariant matching. *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 335–344, 1988.

[94] Walter Ledermann and Alan J. Weir. *Introduction to group theory*. Longman mathematics series. Addison Wesley, 1996. ISBN 0582259541.

[95] Andre Lejeune and Frank P. Ferrie. Finding the parts of objects in range images. *Computer Vision and Image Understanding*, 64(2):230–247, September 1996.

[96] Stanley C. Lennox and Mary Chadwick. *Mathematics for Engineers and Applied Scientists*. Heinemann, 1970. ISBN 0435712810.

[97] Cai Heng Li. Isomorphisms of finite Cayley digraphs of bounded valency, ii. *Journal of Combinatorial Theory*, 87(Article ID jcta.1999.2966):333–346, February 1999. 0097-3165/99.

[98] P. Liang and C. H. Taubes. Orientation-based differential geometric representations for computer vision applications. *IEEE Trans Pattern Analysis and Machine Intelligence*, 16(3):249–258, 1994.

[99] Chun-Shin Lin. Invariants of three-dimensional contours. *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 286–290, 1988.

[100] T. P. Lindeberg and J. O. Eklundh. Construction of a scale space primal sketch. *Proc. British Machine Vision Conference*, 1990.

[101] P. Lindsey and A. Blake. Real-time tracking of surfaces with structured light. *Proc. of the* 5^{th} *British Machine Vision Conference*, 2:619–628, September 1994. ISBN 95218981X.

[102] Manolis I.A. Lourakis, Spyros T. Halkidis, and Stelios C. Orphanoudakis. Matching disparate views of planar surfaces using projective invariants. *Proc. of the* 9^{th} *British Machine Vision Conference*, 1:94–101, September 1998. ISBN 1-901725-04-9.

[103] David G. Lowe. Three dimensional object recognition from single two-dimensional images. *Artificial Intelligence*, 31:355–395, 1987.

[104] Alan Mackworth and Farzin Mokhtarian. Scale-based description of planar curves. *Proc. Canadian Society for Computational Studies of Intelligence*, pages 114–119, 1984.

[105] Alan Mackworth and Farzin Mokhtarian. The renormalized curvature scale space and the evolution properties of planar curves. *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 318–326, 1988.

[106] David Marr. *Vision*. W. H. Freeman, 1982. ISBN 0716715678.

[107] David Marr and H. K. Nishihara. Representation and recognition of the spatial organization of 3-D structures. *Proc. Royal Society*, 200:269–294, 1978.

[108] J. McCleary. *Geometry from a Differentiable Viewpoint*. Cambridge University Press, 1994. ISBN 0521424801.

[109] P. R. S. Mendonca and R. Cipolla. Analysis and computation of an affine trifocal tensor. *Proc. of the* 9^{th} *British Machine Vision Conference*, 1:125–133, September 1998. ISBN 1-901725-04-9.

[110] Farzin Mokhtarian. Evolution properties of space curves. *IEEE International Conference on Computer Vision*, 1988.

[111] Farzin Mokhtarian. Multi-scale description of space curves and three dimensional objects. *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 298–303, 1988.

[112] Farzin Mokhtarian. Fingerprint theorems for curvature and torsion zero-crossings. *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 269–275, 1989.

[113] Farzin Mokhtarian. Multi-scale, torsion-based shape representation for space curves. *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 660–661, 1993.

[114] Farzin Mokhtarian. Convergence properties of curvature scale space representation. *Proc. of the* 6^{th} *British Machine Vision Conference*, 1:357–366, September 1995. ISBN 0 9521898 2 8.

[115] Farzin Mokhtarian. Silhouette-based isolated object recognition through curvature scale space. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(5):539–544, May 1995.

[116] Farzin Mokhtarian. Silhouette-based object recognition with occlusion through curvature scale space. *Proc. European Conference on Computer Vision*, 1:566–578, 1996.

[117] Farzin Mokhtarian. Silhouette-based occluded object recognition through curvature scale space. *Machine Vision and Application*, 10:87–97, 1997.

[118] Farzin Mokhtarian. A theory of multi-scale, torsion-based shape representation for space curves. *Computer Vision and Image Understanding*, 68(1):1–17, October 1997.

[119] Farzin Mokhtarian, Sadegh Abbasi, and Josef Kittler. *Efficient and Robust Retrieval by Shape Content through Curvature Scale Space*, volume 8 of *Series on Software Engineering and Knowledge Engineering, Series Editor S. K. Chang*. World Scientific Publishing, 1997. Editors Arnold W. M. Smeulders and Ramesh Jain. ISBN 9810233272.

[120] Farzin Mokhtarian, Nasser Khalili, and P. Yuen. Multi-scale 3-D free-form surface smoothing. *Proc. of the* 9^{th} *British Machine Vision Conference*, 2:730–739, September 1998. ISBN 1-901725-04-9.

[121] Farzin Mokhtarian, Nasser Khalili, and P. Yuen. Multi-scale free-form surface description. *Proc. Indian Conference on Computer Vision, Graphics and Image Processing*, pages 70–75, 1998.

[122] Farzin Mokhtarian, Nasser Khalili, and P. Yuen. Free-form 3-D object recognition multiple scales. *Proc. of the* 11^{th} *British Machine Vision Conference*, pages 446–455, September 2000.

[123] Farzin Mokhtarian, Nasser Khalili, and P. Yuen. Curvature computation on free-form 3-D meshes at multiple scales. *Computer Vision and Image Understanding*, 82:1–22, 2001.

[124] Farzin Mokhtarian, Nasser Khalili, and P. Yuen. Multi-scale free-form 3-D object recognition using 3-D models. *Image and Vision Computing*, 19(5):271–281, 2001.

[125] Farzin Mokhtarian, Nasser Khalili, and P. Yuen. Estimation of error in curvature computation on multi-scale free-form surfaces. *International Journal of Computer Vision*, 2002.

[126] Farzin Mokhtarian and Alan Mackworth. Comments on "scale-based description and recognition of planar curves and two-dimensional shapes": Authors' reply. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(5):675, 1986.

[127] Farzin Mokhtarian and Alan Mackworth. Scale-based description and recognition of planar curves and two-dimensional shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(1):34–43, January 1986.

[128] Farzin Mokhtarian and Alan K. Mackworth. A theory of multi-scale, curvature-based shape representation for planar curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(8):789–805, August 1992.

[129] Farzin Mokhtarian and H. Murase. Silhouette-based object recognition through curvature scale space. *Proc. International Conference on Computer Vision*, pages 269–274, 1993.

[130] Farzin Mokhtarian and S. Naito. Scale properties of curvature and torsion zero-crossings. *Proc. Asian Conference on Computer Vision*, pages 303–308, 1993.

[131] O. Monga, R. Lengagne, and R. Deriche. Crest lines extraction in volume 3-D medical images: A multi-scale approach. *INRIA research report no. 2338*, July 1994.

[132] Hiroshi Murase and Shree K. Nayar. Visual learning and recognition of 3-D objects from appearance. *International Journal of Computer Vision*, 14:5–24, 1995.

[133] Kenji Nagao and W. E. L. Grimson. Affine matching of planar sets. *Computer Vision and Image Understanding*, 70(1):1–22, April 1998.

[134] Jason M. Nash, John N. Carter, and Mark S. Nixon. Extraction of moving articulated-objects by evidence gathering. *Proc. of the* 9^{th} *British Machine Vision Conference*, 2:609–618, September 1998. ISBN 1-901725-04-9.

[135] W. Neuenschwander, P. Fua, G. Szekely, and O. Kubler. Velcro surfaces: Fast initialization of deformable models. *Computer Vision and Image Understanding*, 65(2):237–245, February 1997.

[136] Alison Noble and Dale Wilson. On computing aspect graphs of smooth shapes from volumetric data. *Computer Vision and Image Understanding*, 66(2):179–192, May 1997.

[137] J. Oliensis. Local reproducible smoothing without shrinkage. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, 15(3):307–311, March 1986.

[138] Peter J. Olver, Guillermo Sapiro, and Allen Tannenbaum. Invariant geometric evolutions of surfaces and volumetric smoothing. *SIAM J. Appl. Math.*, 57(1):176–194, February 1997.

[139] Barrett O'Neill. *Elementary Differential Geometry.* Academic Press, second edition, 1997. ISBN 0125267452.

[140] J. O'Rourke. *Computational Geometry in C.* Cambridge University Press, 1993. ISBN 0521445922.

[141] E.S. Page and L.B. Wilson. *Information Representation and Manipulation in a Computer*. Cambridge Computer Science Texts 2. Cambridge University Press, 1976. ISBN 0521201462.

[142] Xavier Pennec. Toward a generic framework for recognition-based on uncertain geometric features. *Videre: Journal of Computer Vision Research*, 1(2):58–87, Winter 1998.

[143] Alex P. Pentland. Perceptual organization and the representation of natural form. *Artificial Intelligence*, 28:293–331, 1986.

[144] Eric Persoon and King-Sun Fu. Shape discrimination using Fourier descriptors. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, PAMI-8(3):388–397, May 1986.

[145] M. Pilu and R. Fisher. Recognition of geons by parametric deformable contour models. *Proc. European Conference on Computer Vision*, pages 71–82, 1996.

[146] A. V. Pogorelov. *Differential Geometry*. P. Noordhoff N.V., 1960.

[147] J. Ponce, D. J. Kriegman, S. Petitjaan, S. Sullivan, G. Taubin, and B. Vijayakumar. Representations and algorithms for 3-D curved object recognition. *Three-dimensional Object recognition Systems*, pages 17–56, 1993.

[148] Marc Proesmans, Luc Van Gool, and A. Oosterlinck. Active acquisition of 3-D shape for moving objects. *IEEE International Conference on Image Processing*, pages 647–650, 1996. 0-7803-3258-X.

[149] Marc Proesmans, Luc Van Gool, and A. Oosterlinck. One-shot active 3-D shape acquisition. *IEEE Proceedings of ICPR*, pages 336–340, 1996.

[150] N. S . Raja and A. K. Jain. Obtaining generic parts from range images using a multi-view representation. *Computer Vision, Graphics and Image Processing*, 60:44–64, 1994.

[151] Paul L. Rosin. Assessing the behaviour of polygonal approximation algorithms. *Proc. the* 9^{th} *British Machine Vision Conference*, 2:670–679, September 1998. ISBN 1-901725-04-9.

[152] C. A. Rothwell, D. A. Forsyth, A. Zisserman, and J. L. Mundy. Extracting projective structure from single perspective views of 3-D point sets. *Proc. International Conference on computer Vision*, 1993.

[153] Steven J. Ruuth. An algorithm for generating motion by mean curvature. *Proc. 12th International Conference on Analysis and Optimization of Systems Images, Wavelets*, June 1996.

[154] Steven J. Ruuth. Efficient algorithm for diffusion-generated motion by mean curvature. *Journal of Computational Physics*, 144(Article no CP986025):603–625, 1998.

[155] H. Samet. *The Design and Analysis of Spatial Data Structures.* Addison-Wesley, 1990.

[156] A. Saminathan, Andrew J. Stoddart, Adrian Hilton, and John Illingworth. Progress in arbitrary topology deformable surfaces. *Proc. of the* 8^{th} *British Machine Vision Conference*, 2:679–688, September 1997.

[157] Simone Santini and Ramesh Jain. The graphical specification of similarity queries. *Journal of Visual Languages and Computing*, 7:403–421, 1996.

[158] Gregory Satir and Doug Brown. *C++ The Core Language.* O'Reilly, 1995. ISBN 156592116X.

[159] W. J. Schroeder, J. A. Zarge, and W. E. Lorensen. Decimation of triangle meshes. *Computer Graphics*, pages 65–70, 1992.

[160] Will Schroeder, Ken Martin, and Bill Lorensen. *The Visualization Toolkit.* Prentice Hall, 1996. ISBN 0131998374.

[161] M. Seibert and A. M. Waxman. Adaptive 3-D object recognition from multiple views. *IEEE Trans Pattern Analysis and Machine Intelligence*, 14:107–124, 1992.

[162] J. A. Sethian. *Level Set Methods.* Cambridge University Press, 1996. ISBN 0521572029.

[163] Ken Shoemake. Euler angle conversion. *Graphics Gems IV*, pages 222–229, 1994. Paul S. Heckbert (Ed.). ISBN 0123361559.

[164] T. M. Silberberg, L. Davies, and H. Harwood. An iterative Hough procedure for three dimensional object recognition. *Pattern Recognition*, 17:621–629, 1984.

[165] S. S. Sinha and R. Jain. Range image analysis. *Handbook of Pattern Recognition and Image Processing: Computer Vision*, 2:185–237, 1994. T. Y. Young, ed.

[166] Franc Solina and Ruzena Bajcsy. Recovery of parametric models from range images: The case for superquadrics with global deformations. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, 12(2):131–147, February 1990.

[167] Milan Sonka, Vaclav Hlavac, and Roger Boyle. *Image Processing, Analysis and Machine Vision.* Chapman and Hall, 1993. ISBN 0412455706.

[168] B. I. Soroka and R. K. Bajcsy. Generalized cylinders from serial sections. *Proc. IJCPR*, 1976.

[169] G. Soucy and F.P. Ferrie. Surface recovery from range images using curvature and motion consistency. *Computer Vision and Image Understanding*, 65(1):1–18, January 1997.

[170] Marc Soucy and Denis Laurendeau. Multi-resolution surface modelling-based on hierarchical triangulation. *Computer Vision and Image Understanding*, 63(1):1–14, January 1996.

[171] J. Stansfield. Conclusions from the commodity export project. *memo 601, MIT AI Lab, Cambridge, MA*, 1980.

[172] Fridtjof Stein and Gerard Medioni. Structural indexing: Efficient three dimensional object recognition. *Three Dimensional Object Recognition Systems*, pages 353–373, 1993. A. K. Jain and P. J. Flynn (Editors).

[173] Ian Stewart and David Tall. *The Foundations of Mathematics*. Oxford University Press, 1977. ISBN 0198531656.

[174] G. Stockman. Object recognition and localization via pose clustering. *Journal of Computer Vision, Graphics and Image Processing*, 40:361–387, 1987.

[175] Andrew J. Stoddart and M. Baker. Reconstruction of smooth surfaces with arbitrary topology adaptive splines. *Proc. European Conference on Computer Vision*, 2:241–254, 1998.

[176] Andrew J. Stoddart and M. S. Baker. Progressive splines. *IEEE Workshop on 3-D Image Analysis and Synthesis*, pages 295–298, 1998.

[177] Andrew J. Stoddart and M. S. Baker. Surface reconstruction and compression using multi-resolution arbitrary topology G^1 continuous splines. *ICPR*, pages 788–791, 1998.

[178] Andrew J. Stoddart, S. Lemke, Adrian Hilton, and T. Renn. Estimating pose uncertainty for surface registration. *Image and Vision Computing*, 16:111–120, 1998.

[179] Bjarne Stroustrup. *The C++ Programming Language*. Addison-Wesley, second edition, 1991. ISBN 0201539926.

[180] P. Suetens, P. Fua, and A. J. Hanson. Computational strategies for object recognition. *ACM Computing Surveys*, 24(2):5–61, 1992.

[181] D. L. Swets. The self-organizing hierarchical optimal subspace learning and inference framework for object recognition. *Ph.D. thesis, Michigan State University, Dept of Computer Science, East Lansing, Michigan*, 1996.

[182] Gabriel Taubin. Parametrizing and fitting bounded algebraic curves and surfaces. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 103–108, 1992.

[183] Gabriel Taubin. Curve and surface smoothing without shrinkage. *IEEE International Conference on Computer Vision*, pages 852–857, June 1995.

[184] Gabriel Taubin. Estimating the tensor of curvature of a surface from a polyhedral approximation. *IEEE International Conference on Computer Vision*, pages 902–907, 1995.

[185] Gabriel Taubin. A signal processing approach to fair surface design. *SIGGRAPH*, pages 351–358, 1995.

[186] Gabriel Taubin. Optimal surface smoothing as filter design. *Proc. European Conference on Computer Vision*, 1996.

[187] Bart M. ter Haar Romeny (Ed.). *Geometry-Driven Diffusion in Computer Vision*. Kluwer Academic, 1994. ISBN 0792330870.

[188] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography - a factorization method. *International Journal of Computer Vision*, 9(2):137–154, 1992.

[189] E. Trucco, R. B. Fisher, and A. W. Fitzgibbon. Direct calibration and data consistency in 3-D laser scanning. *Proc. of the* 5^{th} *British Machine Vision Conference*, 2:489–498, September 1994. ISBN 95218981X.

[190] Frank C. D. Tsai. A probabilistic approach to geometric hashing using line features. *Computer Vision and Image Understanding*, 63(1):182–195, January 1996.

[191] P. W. M. Tsang. A genetic algorithm for affine invariant object shape recognition. *Proc. IMechE*, 211:385–392, 1997.

[192] Shimon Ullman and Ronen Basri. Recognition by linear combinations of models. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, 13(10):992–1006, October 1991.

[193] S. Umeyama. Parametrized point pattern matching and its application to recognition of object families. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, 15(2):136–144, 1993.

[194] Florian-Horia Vasilescu. Quaternionic Cayley transform. *Journal of Functional Analysis*, 164(Article ID jfan.1999.3389):134–162, December 1998. 0022-1236/99.

[195] Michael W. Walker, Lejun Shao, and Richard A. Volz. Estimating 3-D location parameters using dual number quaternions. *CVGIP: Image Understanding*, 54(3):358–367, November 1991.

[196] John J. Weng, N. Ahuja, and T. S. Huang. Learning recognition and segmentation of 3-D objects from 2-D images. *International Conference on Computer Vision*, 1:121–128, April 1993. 0818638702.

[197] Lance R. Williams and Allen R. Hanson. Perceptual completion of occluded surfaces. *Computer Vision and Image Understanding*, 64(1):1–20, July 1996.

[198] A. P. Witkin. Scale space filtering. *Proc. IJCAI*, pages 1019–1023, 1983.

[199] Haim J. Wolfson. Model-based object recognition by geometric hashing. *Proc. European Conference on Computer Vision*, pages 526–536, 1990.

[200] A. K. C. Wong, S. W. Lu, and M. Rioux. Recognition and shape synthesis of 3-d objects-based on attributed hypergraphs. *IEEE Trans Pattern Analysis and Machine Intelligence*, 11:279–290, 1989.

[201] www.cselt.it/mpeg. MPEG-7 standard. September 2001.

[202] Gang Xu. A unified approach to image matching and segmentation in stereo, motion, and object recognition via recovery of epipolar geometry. *Videre: Journal of Computer Vision Research*, 1(1):22–55, Fall 1997.

[203] P. Yuen, Nasser Khalili, and Farzin Mokhtarian. Curvature estimation on smoothed 3-D meshes. *Proc. of the* 10^{th} *British Machine Vision Conference*, 1:133–142, September 1999. ISBN 190172509X.

[204] P. Yuen, Farzin Mokhtarian, and Nasser Khalili. Multi-scale 3-D surface description: Open and closed surfaces. *Proc. Scandinavian Conference on Image Analysis*, pages 303–310, 1999.

[205] P. Yuen, Farzin Mokhtarian, Nasser Khalili, and John Illingworth. Curvature and torsion feature extraction from free-from 3-D meshes at multiple scales. *IEE Proc. Vision, Image and Signal Processing*, 147(5):454–462, 2000. UK ISSN 1350-245X/IVIPEK 147(5)385-492.

[206] Chaohuang Zeng and Milan Sonka. Local three-dimensional shape-preserving smoothing without shrinkage. *IEEE 1st International Conference on Image Processing, Icip '97*, pages 393–396, October 1997.

[207] Hongbin Zha, Hideki Nanameg, and Tadashi Nagata. Recognition 3-D objects by using a Hopfield-style optimization algorithm for matching patch-based descriptions. *Pattern Recognition*, 31(6):727–741, 1998.

Index