Shape Recovery Using Extended Superquadrics

Talib S Bhabhrawala, Venkat N. Krovi, Frank C. Mendel and Venugopal Govindaraju

Technical Report

Department of Computer Science
State University of New York at Buffalo
Buffalo, New York 14260

Abstract

Reverse engineering of accurate 3D models and 2D contours of real objects from surface measurements is recognized as an important research goal in various communities. Currently the application domains include industrial product design and computer generated imagery for film and multimedia and potential application now also extends to the life sciences realm.

Despite the large body of work on 3D modeling, most models of shape lack the descriptive power to bridge the gap between reconstruction, recognition, and analysis due, mostly, to conflicting requirements. To obtain meaningful information from noisy sensor data reconstruction models, researchers have traditionally used large numbers of parameters. In contrast searching and recognition techniques use shape abstractions, which drastically reduce information. Analysis models require the inclusion of physics, in the form of kinematics, dynamics and FEA, preferably in parametric form to support simulation-based refinement processes.

Thus the goal of our effort is to develop a class of hybrid models whose underlying geometric and computational data structure intimately combines implicit, explicit and parameterized surface representations with volumetric solid representations that are well suited for transitioning to analysis and at the same time enjoy a low order parameterization. This thesis proposes a new shape parameterization for smoothly deformable two and three dimensional objects, such as those found in biomedical images, whose diversity and irregularity make them difficult to represent in terms of fixed features or high fidelity models which are capable for any analysis.

The proposed representation strategy can then represent 3D medical data obtained as point clouds from sources such range imaging devices. To achieve this we have utilized extended-superquadric models which are well-suited for shape representation but require the development of the concomitant methods and benchmarking prior to widespread acceptance. Furthermore, such models are well suited for transitioning to analysis, as for example, in model-based non rigid structure and motion recovery or for mesh generation and simplified volumetric-FEA applications.

Contents

List of Figures

1 Introduction

In the recent years there has been an explosion in availability and form of 3D data. Simultaneously, there has also been rapid growth in ubiquitous availability of computation and communication infrastructure which has made it possible to create, manipulate and distribute such 3D data. The application areas for use of such data have ranged from engineering, life-sciences, visual arts and entertainment.

Unfortunately what we mean by 3D data varies significantly from one application to the next and from one community to the other based on requirements posed by each. For instance, data can be obtained as cloud of points and are often more than adequate for visualization. In many cases a model based reconstruction approach is pursued from this point cloud data to realize additional benefits, such as data reduction, simplified data manipulation, subsequent storage and analysis. Model based shape reconstruction has been the focus of considerable research interest for the past few decades. Efficient, reliable and robust methods are needed for building geometric shape models from raw input data to describe and characterize a variety for shapes and patterns. Numerous general and powerful techniques based on results from range imaging, mathematical morphology and differential geometry are available.

However all modeling is an approximation and there is a critical need for creating low order parametric approximations that are well suited for manipulation, visualization and analysis. However this comes at the additional cost of computation and the type of model required is defined by the final outcome of the application. The final application of the data determines both the selection of model as well as the process of reconstruction. Algorithms which perform very well for niche applications may fail for another. For instance, for visualization purposes it is

important to capture the detailed surface geometric information and various CAGD techniques have been developed. On the other hand if the final application is for dynamic analysis and finite element analysis it is more important to capture the volumetric information within the model. We will focus on examining model-based reconstruction techniques for the latter goal which would be applicable to medical image data.

Regardless of the application and method employed the cardinal requirement remains to create *low order*, *parametric models* to *approximate* the actual data set. Peering a little more into these requirements, firstly while models with multiple parameters have greater expressive power, from a computational view point we want a *low order* models for fitting, manipulating, visualizing and analysis. Secondly, data is infinite dimensional, but for tractability we need to come up with a model which is by its very nature an approximation. Finally, parametric models have been used with great success in engineering and remain the best candidate for such data. The information obtained from such a parametric representation can be utilized of parametric encapsulation of knowledge. In other words, knowledge in the form of parameterized models (shape) can be stored in universally accessible digital libraries. Encapsulation of form within such libraries facilitates distribution of both capabilities and results and provides a mechanism for verification and further improvement. We or other researchers could potentially access our models and rerun our "experiments' to confirm our results. As new tools or concepts are developed, they too could be easily and widely disseminated, tested, and improved through this mechanism. Parameterization is critical to permit indexing within a library/database and retrieval for use in various parametric interpolation/extrapolation analysis schemes (such as shape blending or generation of response surfaces).

If we look at general shape *reconstruction methods* itself, there is a requirement of models with broad geometric coverage. To represent more complex shapes there is a trade off between the increase of the degrees of freedom and expressive power on the one hand, and the compactness and computational complexity on the other. Constructive solid geometry (CSG) representations, created by boolean operations on primitives, can accurately represent simple shapes with few parameters, compared to surface boundary representations. Curve and surface modeling techniques using polygons, B-splines and Fourier descriptors are better suited for reconstruction of a wide range of geometric shapes. But, these have proven inadequate for modeling real-world objects when interior properties are of significance. Further to extract some sort of useful information from noisy data the representations generally have an elaborate representation with large number of parameters[1-4].

Many of these techniques are well developed in the engineering domain for a variety of applications. Product design and testing activities are chiefly reliant on computer simulations and a common requirement for them is to incorporate existing objects into the virtual prototyping environment which can be used for analysis, modifications etc. One important area of application is *reverse engineering* which is the process of producing design details in the form of CAD model from the physical part. The process consists of first scanning the object concerned to create a point cloud that represents the skin of the object. Then using some reconstruction algorithm the surface model of the object is created. The surface model thus created is exported to a Solid modeling package where the surface model is checked for integrity of the model and after modification if necessary, the solid model is generated. The solid model thus created can be then used for analysis, documentation, NC code generation or for prototyping.

Motivated by this we would like to develop and extend similar techniques for use in the life sciences domain. In particular the swift progress in medical imaging is revolutionizing the field of life science tremendously and its role has expanded beyond the simple visualization of anatomic structures. It has become a vital tool for and inspection, planning, simulation, analysis, information indexing and host of other applications. Medical imaging allows scientists and physicians to gather information by looking, analyzing and reverse engineering biological parts.

The requirement of analysis models in the life sciences arena is becoming increasingly important. Natural objects exhibit high levels of irregularities and bridging the gap between recognition and reconstruction increases significantly. However in the field of life sciences, biological examples tend to be of irregular shape and inhomogeneous physiologic and biomechanical properties. Oftentimes this necessitates increased complexity of modeling in order to adequately capture the underlying physics closely, which can create challenges for computational testing. In contrast the engineering and computer graphics community has typically emphasized uniformity of shapes/materials that have facilitated ease of functional testing (physical or virtual).

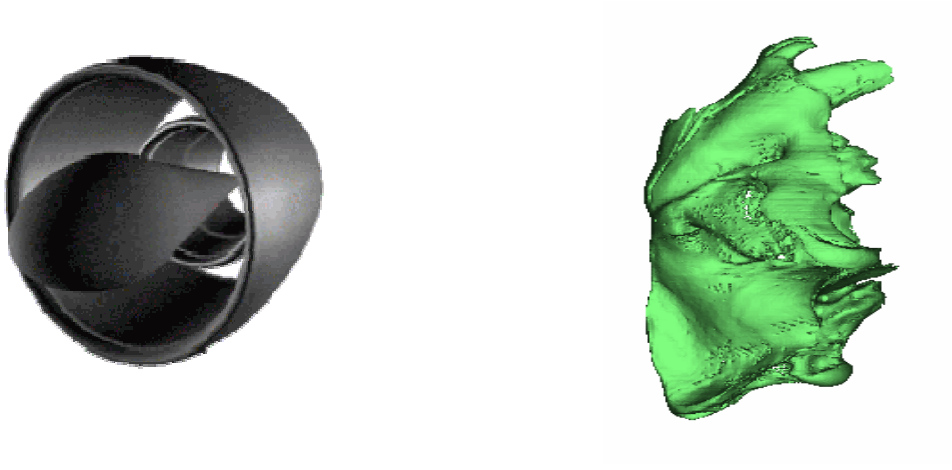


Figure 1-1 Objects found in the Engineering domain vs. the Life Sciences domain

Although modern imaging devices provide exceptional views of internal anatomy, currently the use of computers to quantify and analyze the embedded structures with accuracy and efficiency is limited. There is a of need representations which would not only be able to capture naturally occurring irregularities in an object but should also capable of extracting meaningful information with the noisy data and leaving lesser room for assumptions. These are contrasting requirements and any development towards achieving both of these qualities in a single representation can benefit the ongoing research in reconstruction of biological models immensely. Success in finding suitable means of rendering complex biological models to computationally more efficient algorithms will broaden avenues between the biological and engineering/computational communities to the benefit of both. To be more specific methods developed for treatment of biological problems can help further the creation of new engineered shapes and products.

1.1 Research Issues

From the onset we are interested in both the theoretical challenges that such a problem would generate and in developing a practical methodologies which can be used eventually for naturally occurring objects.

Which kind of a parametric modeling framework would be most suitable for rapid, easy, accurate and computationally inexpensive shape modeling and conversion to volumetric solid model from a dense sampling of the surface?

It is critical that representation schema encompass two features; of adequate shape capture and at the same time remain computationally tractable. These however are highly contrasting requirements and represent a major research issue. But the thing to bear in mind is that they should not only be capable obtaining an "acceptable" shape but maintaining qualities such as

topological accuracy, smoothness, continuous curvature surfaces as well sharp edges. It should be suitable for the following phases of the "design" process like for analysis and manipulation. Combining all these desirable characteristics and exhibit it using a compact representation is the primary and most challenging issue of the effort.

How can we leverage the same framework to additionally parametrically explore multi-resolution hierarchical indexing, storage, searching, reconstruction and retrieval?

Further the challenge lies in ensuring a computational representation of shape (a shape descriptor) which has a potential for which an index can be built and geometric matching can be performed efficiently. Knowledge in the form of parameterized models (form) and corresponding analyses (function) can be stored in universally accessible digital libraries. Encapsulation of both form and function within such libraries facilitates dissemination of both capabilities and results and provides a mechanism for verification and further improvement. Parameterization is critical to permit indexing within a library/database and retrieval for use in various parametric interpolation/extrapolation analysis schemes

1.2 Organization of the thesis

The remainder of the thesis is organized as follows:

Chapter 2 provides background of the current reconstructing algorithms with particular emphasis on the field of medical imaging and some of its salient features. It also provides an exhaustive background of the current research with superquadric representation which is a relatively new area for object representation and has come into focus very recently.

Chapter 3 provides brief preliminaries of the technical background especially pertaining to superquadrics that is required for understanding this thesis.

Chapter 4 lays out the foundation of the various modules associated with the formulation and development of the algorithm. The chapter explains the methodology and its development in a bottom up approach which is also analogous to how it was actually conceptualized. The problem was divided into functional elements with varying importance and then each element was refined and integrated to the whole approach.

Chapter 5 first analyzes various case studies where in the two major aspects of the thesis, Shape Construction & Shape Optimization, are intuitively explained by means of examples and case studies.

Chapter 6 concludes the thesis with the discussion of results obtained in this work and provides suggestions for the future work.

2 Background

Research on 3D data and reconstruction has been done in numerous areas catering to various applications. It is beyond the scope of this thesis to identify and discuss every one. In this chapter we will present an overview of the reconstruction problem and some of the principal and widely used techniques associated with the same. We will also look at some of the scanning technologies present out there which are utilized to generate point cloud data.

In the second part of this chapter we present a comprehensive discussion on the current research and evolution of superquadrics and its utilization in various applications. Superquadrics will be utilized as geometric primitives for shape representation in our methodology. Its is important to understand that the choice of primitive not only imposes a constraint on the representation but also on the recovery procedure as well. All model based approaches are limited and cannot possibly model all of the input data precisely. Inherently, all modeling procedures tend to be an approximation. What is important here is to search and use a representation which balances all requirements to automate a recovery procedure for a particular application.

2.1 Overview of the Reconstruction Problem

Work in the field can be grouped into three broad categories as discussed below. These are generic established methodologies with a number of types and variants beneath it which are used for a plethora of applications.

Piecewise linear reconstruction is governed by the goal of constructing a polygonized surface that interpolates or approximates the given points. Computational Geometry provides a framework of concepts and techniques to attack this problem.

Surface fitting is based on techniques from Computer Aided Geometric Design CAGD and Numerical Analysis. The reconstructed object is represented as a collection of surface patches algebraic NURBS, splines etc.

Physically based modeling have been developed and used especially by the Computer Vision community. In these methods a surface is deformed under the action of applied forces and internal reactions until it approximates the data points.

The subsequent discussion is not meant to be exhaustive but should give a lucid suggestion of the range of techniques that are being used. When reconstructing a model representation from unorganized points a major problem is that of inferring the topological class of the object. Most of the approaches described assume that the topological class of the object and often restrict themselves to genus zero objects.

The following section will present a more detailed look at some methodologies based on this classification but we will restrict our discussion to an overview. A comprehensive discussion of other techniques for other applications may be found in [4, 5].

2.1.1 Piecewise linear reconstruction

Interpolation refers to finding values for points between the given points (i.e. inside their convex hull); if the function should extend over a wider region of the input domain the problem is instead referred to as extrapolation. The method chosen should depend on the properties one desires the resulting surface to have; many interpolation methods are based on Voronoi diagrams and Delaunay triangulations. A piecewise constant approximation, used in rainfall estimation,

can be found by simply choosing the function value in each Voronoi cell to be that of the cell's generating site. The Delaunay triangulation itself provides a piecewise linear continuous function, defined within the convex hull of the input, that minimizes a certain energy functional. "Natural neighbor interpolation" is defined for each point x by adding x as a site to the Voronoi diagram of the original sites, and averaging the sites' values weighted by the fraction of the cell for x previously covered by each other cell. This somewhat complicated procedure results in a continuous function smooth everywhere except at the original sites.

2.1.2 Surface fitting

Probably the best known surface extraction algorithms are so-called iso-surface algorithms[6]. The algorithms belonging to this class take a function *f* from $\mathbb{R}3$ to $\mathbb{R}$ as their input. The zero level set of the function *f* is assumed to be the surface to be approximated. Iso-surface algorithms work by dividing the space $\mathbb{R}3$ into cubes, or into some other polyhedral primitives, and thereafter evaluating the function *f* at every vertex of every primitive. The algorithms belonging to this class take a function *f* from $\mathbb{R}3$ to $\mathbb{R}$ as their input. The zero level set of the function *f* is assumed to be the surface to be approximated. Iso-surface algorithms work by dividing the space into cubes, or into some other polyhedral primitives, and thereafter evaluating the function *f* at every vertex of every primitive. Based on these evaluated values, intersections of the surface and primitive polyhedra are inferred. For example, the Marching Cubes algorithm[7] uses only the information about the sign of the evaluated function values and a look-up table to triangulate the given surface.

The Marching Cubes algorithm offers a straight-forward method for extracting surfaces from volumetric images. Each voxel has simply to be classified either being inside or outside of the surface of interest, a segmentation of the image. If image segmentation is

performed simply by threshold intensity values, the process is sensitive to imaging noise, natural variation of intensity values across the image, and other image artifacts. Especially, the correct surface topology cannot be enforced. Note, that noise, blur, and other artifacts may cause the surface interest appear with wrong topological type in the image.

A significant drawback of this method is that it usually requires large amounts of expert intervention. Furthermore the subsequent analysis and interpretation of the segmented object is hindered by pixel or voxel level structure representations generated by most image processing algorithms. The marching cubes algorithm also falls short because the number of triangles in a triangulation is large and hence renders it computationally very expensive.

2.1.3 Deformable Surface Models

Deformable surface models are advanced techniques for surface extraction that can incorporate soft and hard constraints into the surface extraction problem. Deformable models and their applications are surveyed for example in[8-10]. It is not only a promising model based approach but is being extensively researched for many model based approaches.

The mathematical basics of deformable models represent a synergy of geometry and/or physics and approximation theory. It usually permits broader shape coverage by employing geometric representations that involve many degrees of freedom, such as splines. These models also support highly intuitive interaction mechanisms that allow for local and global deformations. Details of formulation and application for some approaches can be found in [9].

2.2 Scanning Technologies

For a point cloud based approach we need sophisticated methods of extracting surface information from the objects of interest. There are a variety of approaches with varying resolution, accuracy, turn around time and cost.

Range sensors as mentioned earlier capture dense 3D point measurements of an objects surface. Conventional range sensors capture 3D surface measurements with respect to a 2D image plane; this is referred to as a 2.5D range image. Multiple view range images are required to capture data for the entire surface of an object and hence create the required point cloud. This *registration* technique coupled with the right algorithm for reconstruction can be used to model any arbitrary unknown object.

The major devices used towards 3D data collection and object reconstruction are discussed in Appendix A and further information about each can be found at [11-13].

2.3 Superquadrics Literature Survey

Based on the original superellipses, superquadrics have been defined based on the spherical product of two 2-dimensional supershapes and include sub- and superellipsoids, paraboloids, hyperboloids and toroids[14]. Since then superquadrics have been used as a quantitative model for various applications in computer environments, both in computer graphics and increasingly also in computer vision[3, 15-18].

Several researchers have used and modified superquadrics for various applications like object reconstruction, modeling, recognition and motion analysis[3, 17-28]. Barr who first came

up with the initial superquadric representation[14] applied basic deformations such as tapering, bending, and twisting to superquadrics[19].

Salinas and Bajcsy utilized a modified error-of-fit function based on the implicit representation of the superquadric but were restricted to global deformations[23]. Gupta employed a volumetric segmentation scheme which useful for high level vision applications and was association of symmetric superquadrics[22]. Boult and Gross used a gradient descent minimization method for superquadric recovery, but had problems with convergence for cylindrically shaped objects[20]. Gross and Boult also discussed several different error-of-fit functions and compared their impact on the shape recovery[21], but was limited to models utilizing the basic superquadric representation. Zhang improved on previous researchers by using nonlinear deformable superquadric models and was able to encompass a wider range of shapes[25]. Terzapoulos and Metaxas induced local and global deformations by converting data into forces and simulating equations of motion which are capable of modeling nonsymmetrical objects[24]. The blended deformable model which is based on the linear interpolation of two parameterized shapes along their main axes, using a blending function was an extension of his work[29]. However, since deformable superquadrics are physics-based models with internal deformation energies and do not have closed-form equations, it could be very hard to extract a set of invariant features from them to generate a unified object knowledge database for object recognition[28]. Leonardis discussed the recovery of superquadrics directly from unsegmented range data, thus avoiding any pre-segmentation steps[30]. This is an encouraging view as it can reduce a lot of parameters if implemented successfully, however their representation didn't deal with local deformations.

It is clear that superquadrics have received significant attention because of their compact representation and robust methods for recovery of 3D models. Although superquadrics are powerful in describing shapes, they cannot model many naturally occurring objects due to their assumption of intrinsical symmetry. Hence, approaches to extend superquadric are representation powers without losing its advantages are very desirable.

As we have seen increasing the degrees of freedom of the superquadrics has been possible by parametrization [31], by blending multiple models [29] or by free form deformations[32]. Other ways to increase the degrees of freedom have included hyperquadrics [33], ratioquadrics [34], and ray-quadrics[35].

However recent work of Zhou and Kambhamettu presented a breakthrough, because they showed that the exponent need not to be fixed, but could be functions. They studied the use of Bezier curves as exponent functions greatly increasing the potential of superquadrics to describe more complex shapes. In the same paper they also pointed out that closed shapes need not have the same shape everywhere, but that several parts of the shapes may have differing exponents or shapes[18, 36].

3 Superquadrics and their Properties

In the previous chapter we presented motivation and current research for the use of superquadrics as a convenient family of parametric shapes which can model a diverse set of objects. Superquadrics have received significant attention because of their compact & simple mathematical representation and robust methods for recovery of a wide variety of 3D shapes using a relatively few number of parameters.

In this chapter we will discuss the features of superquadrics in greater detail and in particular we will examine the various mathematical characteristics that favor the use of such superquadrics for the reconstruction problem.

3.1 Superellipses

A superellipse is a closed curve defined by the following simple equation:

$$\left(\frac{x}{a}\right)^{\varepsilon} + \left(\frac{y}{b}\right)^{\varepsilon} = 1 \qquad (3.1)$$

Where a and b are the size (positive real number) of the major and minor axes and m is a rational number. If $\varepsilon = 2$ and $a = b$, we get the equation of a circle. For larger m, however, we gradually get more rectangular shapes, until for m the curve takes up a rectangular shape (Fig. 2.1). On the other hand, when $\varepsilon = 0$ the curve takes up the shape of a cross[37].

Superellipses are special cases of curves which are known in analytical geometry as Lame curves. Lame curves are named after the French mathematician Gabriel Lame, who was the first who described these curves in the early 19th century[38].

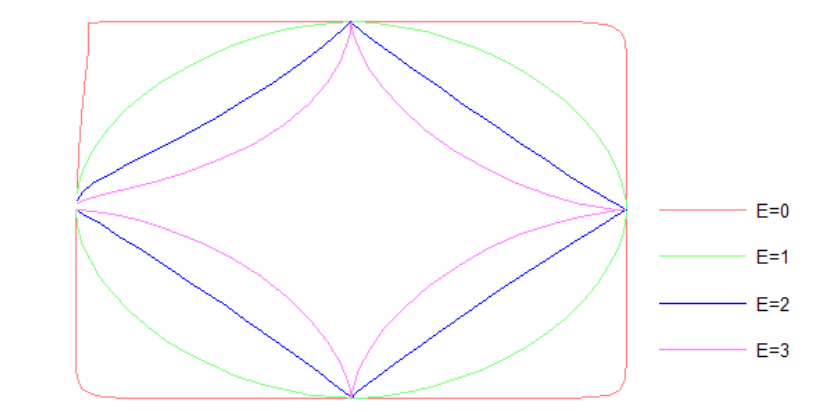


Figure 3-1: A superellipse can change continuously from a star-shape (ε=4) through a circle (ε=1) to a square (ε=3) shape.

3.2 Spherical Products

The spherical product $\otimes$ is defined to operate on two 2D curves

$$h(\omega) = \begin{bmatrix} h_1(\omega) \\ h_2(\omega) \end{bmatrix}, \omega_o \le \omega \le \omega_1$$
$$m(\eta) = \begin{bmatrix} m_1(\eta) \\ m_2(\eta) \end{bmatrix}, \eta_o \le \eta \le \eta_1 \qquad (3.2)$$

which results in a 3D surface.

$$x(\eta, \omega) = m \otimes h = \begin{bmatrix} m_1(\eta) & h_1(\omega) \\ m_1(\eta) & h_2(\omega) \\ m_2(\eta) & \end{bmatrix}, \quad \begin{matrix} \omega_0 \le \omega \le \omega_1 \\ \eta_0 \le \eta \le \eta_1 \end{matrix} \qquad (3.3)$$

Each 2D curve has one degree of freedom, so the resultant surface has 2 degrees of freedom. By adding a scaling term for each spatial direction, we achieve a form with 5 degrees of freedom.

$$x(\eta,\omega)=\begin{bmatrix} a_1 & m_1(\eta) & h_1(\omega) \\ a_2 & m_1(\eta) & h_2(\omega) \\ a_3 & m_2(\eta) & \end{bmatrix}, \quad \begin{matrix} \omega_0 \le \omega \le \omega_1 \\ \eta_0 \le \eta \le \eta_1 \end{matrix} \tag{3.4}$$

We can think of the function h as a horizontal curve which is swept vertically according to the function m. $m_1(\eta)$ scales h, while $m_2(\eta)$ defines the vertical sweeping motion. In this way, we see that the parameter ω affects the surface horizontally, while η affects the surface vertically.

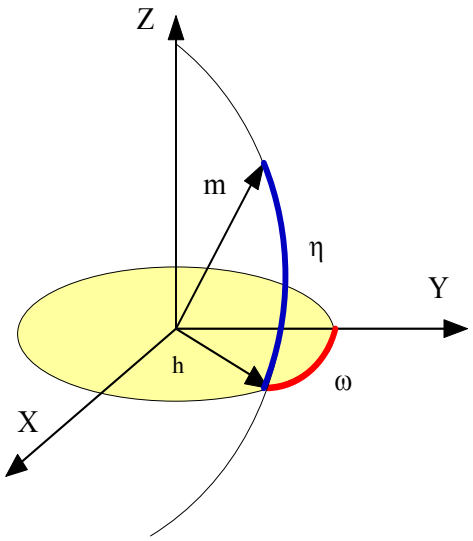


Figure 3-2 Azimuth and elevation angles.

Thinking in terms of the description above a unit sphere, for example, where a half circle in a plane orthogonal to the (x, y) plane is. $\mathrm{h}(\omega)=\begin{bmatrix}\cos\omega \\ \sin\omega\end{bmatrix}, -\pi \le \omega \le \pi$

(3.5)

is crossed with the full circle in (x, y) plane

$$\mathrm{EOF2}=\sum_{i=1}^{N\ data}\left[1-\mathrm{f}\left(\mathrm{x}_i,\mathrm{y}_i\right)^{f(\eta)}\right]^2 \tag{3.6}$$

we obtain a unit sphere

$$r(\eta,\omega)=m(\eta)\otimes h(\omega)=\begin{bmatrix}x\\ y\\ z\end{bmatrix}=\begin{bmatrix}\cos\eta\cos\omega\\ \cos\eta\sin\omega\\ \sin\eta\end{bmatrix}, \begin{matrix}-\pi/2 \le \eta \le \pi/2 \\ -\pi \le \omega \le \pi\end{matrix}$$

(3.7)

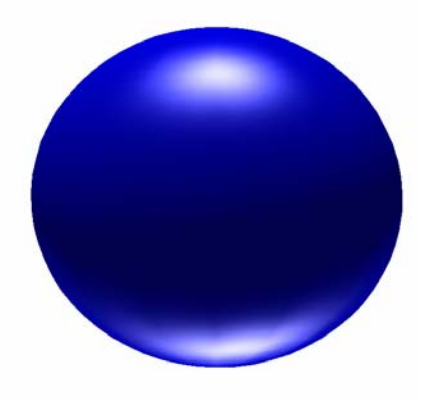


Figure 3-3 A unit sphere generated from the equation 3.7

Analogous to a circle a superellipse may be written as

$$\left(\frac{x}{a}\right)^{\varepsilon}+\left(\frac{y}{b}\right)^{\varepsilon}=1 \tag{3.8}$$

$$s(\theta)=\begin{bmatrix} a\cos^{\varepsilon}\theta \\ b\sin^{\varepsilon}\theta \end{bmatrix}, -\pi \le \theta \le \pi \tag{3.9}$$

3.3 Superquadrics

Furthering our discussion, superellipsoids can be obtained by a spherical product of a pair of such superellipses very similar to how we derived the unit circle and the super ellipse.

$$
\begin{aligned}
& r(\eta,\omega) = s_1(\eta) \otimes s_2(\omega) = \\
& = \begin{bmatrix} \cos^{\varepsilon_1} \eta \\ a_3 \cos^{\varepsilon_1} \omega \end{bmatrix} \otimes \begin{bmatrix} a_1 \cos^{\varepsilon_2} \omega \\ a_2 \cos^{\varepsilon_2} \omega \end{bmatrix} = \\
& = \begin{bmatrix} a_1 \cos^{\varepsilon_1} \eta \; a_1 \cos^{\varepsilon_2} \omega \\ a_2 \cos^{\varepsilon_1} \eta \; a_2 \sin^{\varepsilon_2} \omega \\ a_3 \sin^{\varepsilon_1} \omega \end{bmatrix}, \begin{matrix} -\frac{\pi}{2} \le \eta \le \frac{\pi}{2} \\ -\pi \le \omega \le \pi \end{matrix}
\end{aligned}
\tag{3.10}
$$

Note that exponentiation with is a signed power function such that

$$
\cos^{\varepsilon} \theta = sign(\cos\theta) \left|\cos\theta\right|^{\varepsilon} ! \tag{3.11}
$$

Parameters a_1, a_2 and a_3 are scaling factors along the three coordinate axes. ε_1 and ε_2 are derived from the exponents of the two original superellipses and determine the shape of the superellipsoid cross section parallel and perpendicular to the (x,y) plane, containing z axis.

This flexibility achieved by raising each trigonometric term to an exponent is of particular interest to us. These exponents, in simple terms, control the relative roundness and squareness in both the horizontal and vertical directions.

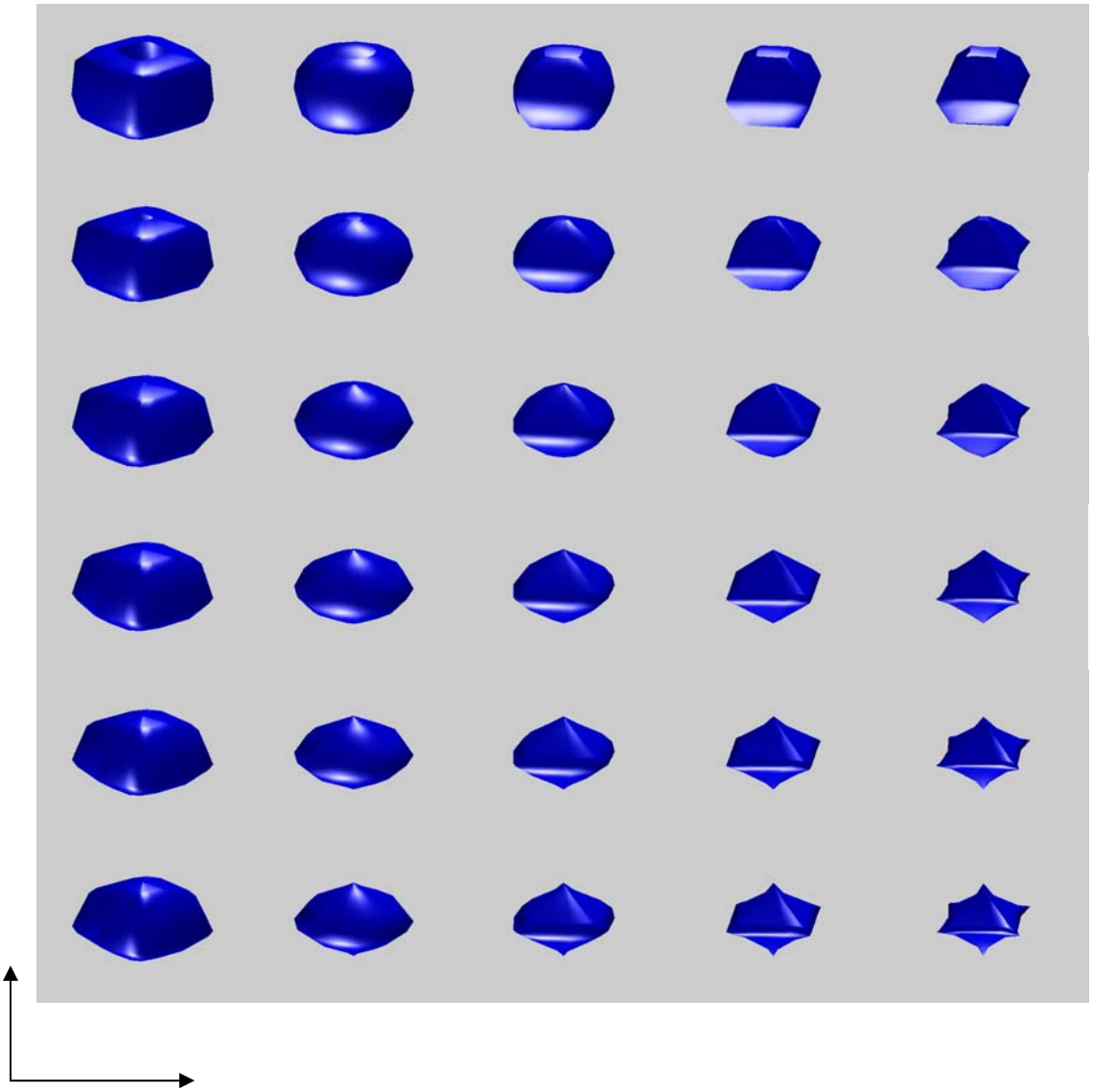


Figure 3-4 The ability of the superquadric to obtain a variety of shapes using low order parameterization

Figure 3-4 **The ability of the superquadric to obtain a variety of shapes using low order parameterization** shows a variety of objects obtained by varying the exponents ϵ_1 and ϵ_2 from 0.3 to 3.0. ϵ_1 controls the vertical component and of each object and increases in value from left to right. ϵ_2 controls the horizontal component and increases from bottom to top.

The term superquadrics was defined by Barr in his seminal paper[14]. Superquadrics are a family of shapes that includes not only superellipsoids, but also superhyperboloids of one piece and superhyperboloids of two pieces, as well as supertoroids. In computer vision literature, it is common to refer to superellipsoids by the more generic term of superquadrics and it will be so in our discussion as well.

3.3.1.1 Implicit Representation

Superquadrics have a valuable single implicit function to measure the relative distance from a given point to a superquadric surface. By eliminating parameter η and ω using equality $\cos^2\theta + \sin^2\theta = 1$, the following implicit equation can be obtained.

$$\left(\left(\frac{x}{a_1}\right)^{2/\varepsilon_2} + \left(\frac{y}{a_2}\right)^{2/\varepsilon_2}\right)^{\frac{\varepsilon_1}{\varepsilon_2}} + \left(\frac{z}{a_3}\right)^{2/\varepsilon_1} = 1$$

$$F(x,y,z) = \left(\left(\frac{x}{a_1}\right)^{2/\varepsilon_2} + \left(\frac{y}{a_2}\right)^{2/\varepsilon_2}\right)^{\frac{\varepsilon_1}{\varepsilon_2}} + \left(\frac{z}{a_3}\right)^{2/\varepsilon_1} \tag{3.12}$$

This function is also called the inside–outside function because it provides an "inside–outside" test that determines the position of a given point $[x,y,z]^T$ relative to the superquadric surface. If F(x, y, z) = 1, the point (x, y ,z) is on the surface. If F(x, y, z)>1, the corresponding point lies outside and if F(x, y, z)< 1, it lies inside the superquadric.

One of the advantages of using the implicit representation is that the reconstructed object can be subjected to any regular Boolean operation. The ease of generating new shapes with the implicit representation allows the user to incorporate the resulting solid into other models.

Further, with the implicit representation, the object is continuous everywhere, and the test for point membership classification can be performed with an evaluation of the function

3.3.1.2 Boundary Values

We should take note of the boundary conditions of η and ω vary for each superquadric type is very important. Sometimes the boundary values are included, and sometimes they are excluded. This depends on two factors. Firstly, a trigonometric function used in a quadric formula may have certain values at which it is not defined. For example, the superhyperboloids use sec functions which are undefined at $\frac{\pi}{2}$ and $\frac{-\pi}{2}$. Secondly, the boundary values may be defined so that the object "wraps around" and meets itself. For example, in the superellipsoid, the boundary for ω is defined so that a complete circle is formed without overlapping on itself.

3.3.1.3 Normals and Tangents

In practice, superquadrics are an easy class of objects to use because they have well defined normal and tangent vectors. Normal vectors are used in intensity calculations during rendering. Both the normal and tangent vectors are used to calculate the curvature of the surface. Sampling of points used to represent an object is often defined according to curvature: in regions of higher curvature, more points are used to achieve a better representation of the surface region.

3.4 Extended Superquadrics

An extended superquadric[18] is defined by a set of points (x,y,z) such that:

$$\begin{bmatrix} a_1 \cos^{f(\eta)} \eta \, a_1 \cos^{f(\omega)} \omega \\ a_2 \cos^{f(\eta)} \eta \, a_2 \sin^{f(\omega)} \omega \\ a_3 \sin^{f(\eta)} \omega \end{bmatrix}, \begin{matrix} -\frac{\pi}{2} \le \eta \le \frac{\pi}{2} \\ -\pi \le \omega \le \pi \end{matrix} \tag{3.13}$$

η and ω are used to represent the longitude and latitude angles and the exponents are now functions of these angles.

Whereas the exponents of superquadrics are constants, the extended superquadrics have exponents changing according to $f(\omega)$ *and* $f(\eta)$. Parameters $f(\omega)$ *and* $f(\eta)$ correspond to latitude and longitude angles expressed in the object-centered spherical coordinate system. $f(\omega)$ lies in the *x*–*y* plane, while $f(\eta)$ corresponds to the angle between the *z* axis and the *x*–*y* plane. Like regular superquadrics, scale parameters a_1, a_2, a_3 define the size of the extended superquadrics in the *x, y, z* directions, respectively.

As we have seen earlier owing to the constant values of their exponents, the superquadrics are symmetric about the three axes and hence, without deformations, can model only symmetric objects. Hence this modified mathematical representation in contrast, whose exponents vary with can model a much broader range of shapes that are not symmetric.

3.4.1 Implicit Representation for an Extended Superquadric

Based on our discussion of regular superquadrics, the inside-outside function of extended superquadrics can be defined as:

$$\left(\left(\frac{x}{a_1} \right)^{\frac{2}{f(\omega)}} + \left(\frac{y}{a_2} \right)^{\frac{2}{f(\omega)}} \right)^{\frac{f(\omega)}{f(\eta)}} + \left(\frac{x}{a_1} \right)^{\frac{2}{f(\omega)}} = 1$$

$$\eta = \operatorname{atan} 2\left(\frac{z}{\sqrt{x^2 + y^2}} \right) \qquad (3.14)$$

$$\omega = \operatorname{atan} 2\left(\frac{y}{x} \right)$$

3.4.2 Control of Exponent Functions

Since the shape of the extended superquadric surface changes according to the exponent functions, it is very important to make sure that the shapes of the exponent functions are controllable. Also, the exponent functions must be continuous in order to ensure that the extended superquadric model deforms continuously and thus has a smooth surface.

With such properties for exponent functions, we introduce the spline as the exponent function. The concept here is that this interpolated curve as in Figure 3-5 acts like a look up table for the algorithm, where the value of a control point corresponding to any latitude and longitude angle can be easily determined. Here we can obtain the exponent value for the superquadric equation anywhere between $\frac{\pi}{2}$ and $\frac{-\pi}{2}$.

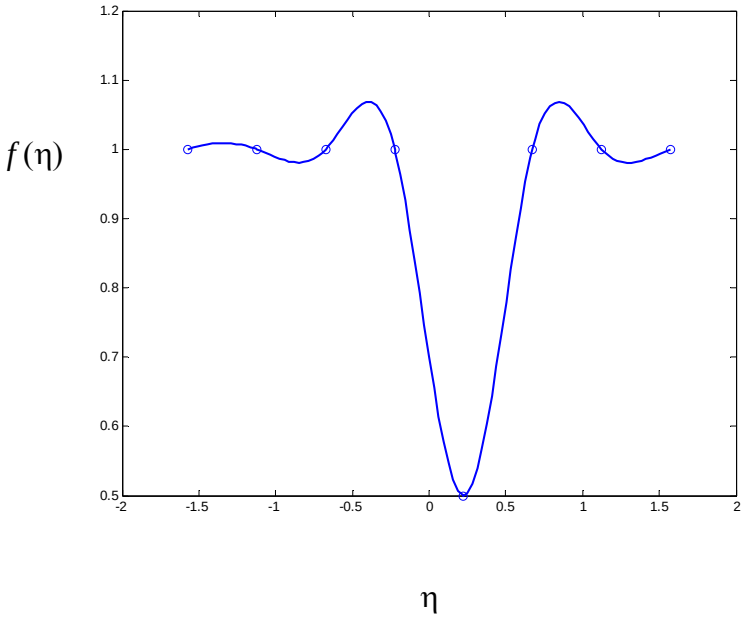


Figure 3-5 Curve between f(η) and η acting as a lookup table

The most attractive feature of the splining in the exponential space is that any defined curve is continuous and smoothly follows the control points without erratic oscillations. Hence, by adjusting the control points, it is very easy to control the shape of the spline curve and then the shape of the whole extended superquadric model. Some examples will be presented in the next section.

3.4.3 Examples

In the previous discussion we noted that individual control points have the ability to induce local deformations on a superquadric surface. Here we will further explore this aspect using simple examples which are illustrated in Figure 3-6 and Figure 3-7.

In each of the four subplots in the top row, that value of $f(\omega)$ is varied from its nominal value of one (which represents a circle/sphere) of the control points. The effect of corresponding change is then shown on the actual curve/surface in the subplots in the bottom rows.

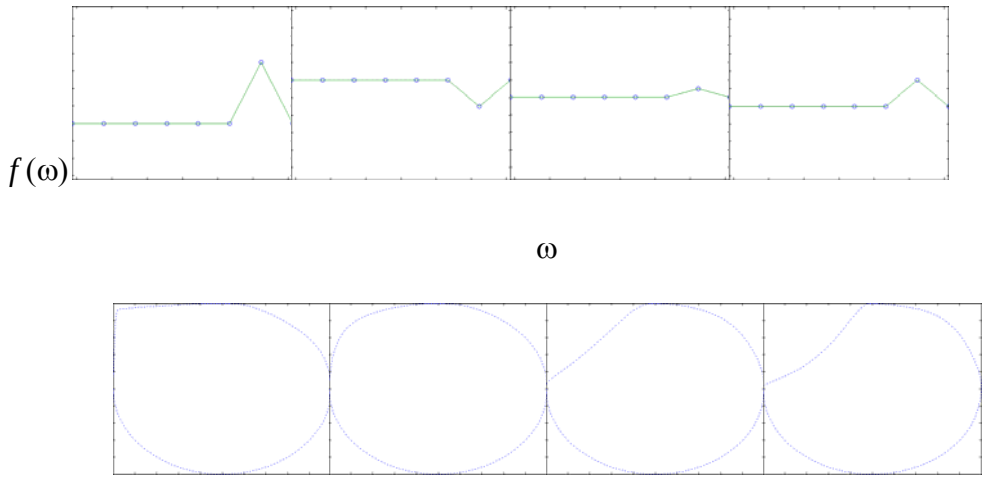


Figure 3-6 Local Deformations in 2D using Extended Superquadrics

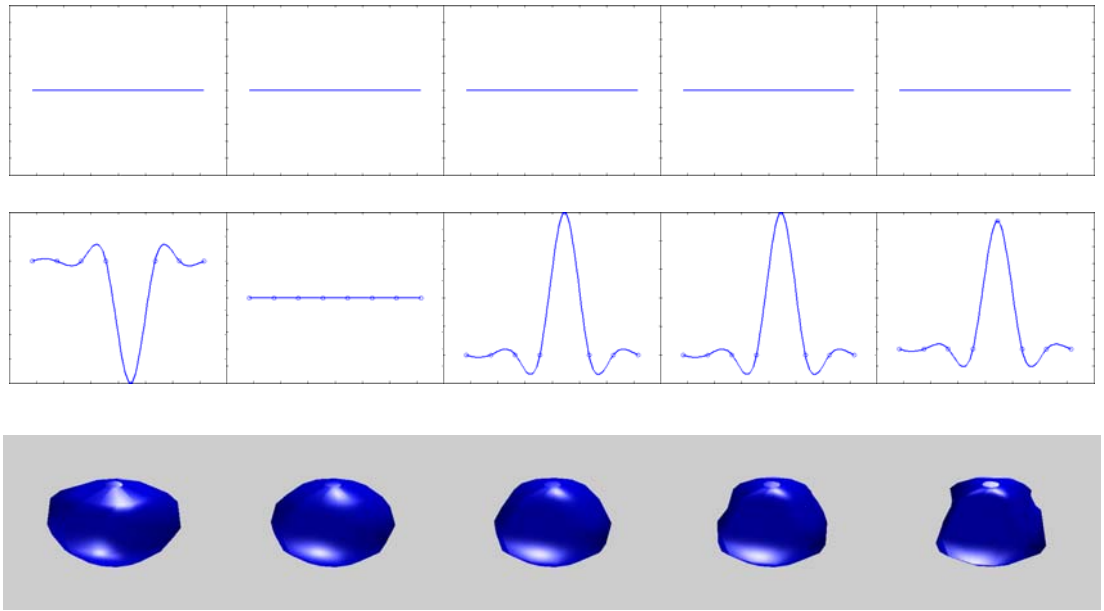


Figure 3-7 Local Deformations in 3D using Extended Superquadrics

4 Development of Scheme Shape Recovery using Superquadrics

In the previous chapter we established the preliminaries which are going to lay the basis for the developed scheme for recovering a superquadric from unstructured 2D and 3D data.

In this chapter, we will discuss the model definition, its formulation and the hierarchical algorithm which enables us to utilize the powerful superquadric representation for fitting shapes to a cloud of points.

4.1 Problem Statement

The problem we wish to solve is that of recovering a superquadric model from a set of 2D points. Given a set of N points $\sum_{i=1}^{N}(x_i, y_i, z_i)$, the task is to minimize

$$\vec{P} = (\underset{\sim}{X} - f(\underset{\sim}{P}))^T (\underset{\sim}{X} - f(\underset{\sim}{P}))$$

Where $f(\underset{\sim}{P})$ is the superquadric model that most closely fits those points i.e. such that average distance of the points from its surface minimized.

4.2 Initial Model Definition

We will initially develop our model based on regular superquadrics which, as discussed earlier, will enable us to obtain symmetrical shapes only. A superquadric positioned in a local coordinate system is defined here by 5 parameters, specifying its shape, size.

$$\begin{bmatrix} x_\omega \\ y_\omega \\ z_\omega \end{bmatrix} = F(a_1, a_2, a_3, \varepsilon_1, \varepsilon_2) \qquad (4.1)$$

where a_1, a_2, a_3 denote the superquadric size and $\varepsilon_1, \varepsilon_2$ are the exponent values determining the over all shape of the superquadric.

The problem is to find suitable values for each of these parameters, such that they describe the superquadric that best fits the points. In order to do this, we need a function F that can compute how close a given point is to the superquadric's surface. By evaluating this function over all the points, and taking the average, we get an error-of-fit (EOF) measure for the model.

4.3 Superquadrics in General Position

The inside-outside function discussed earlier in 3.3.1.1 defines the superquadric in an object oriented system (x_s, y_s, z_s). However in our scope of application we are looking at dealing with a cloud of points from various sources which are expressed in the world coordinate system. Hence it is required that we transform the points to the object coordinated system with a homogeneous coordinate transformation T^{-1} as in Figure 4-1.

$$\begin{bmatrix} x_s \\ y_s \\ z_z \end{bmatrix} = T^{-1} \begin{bmatrix} x_\omega \\ y_\omega \\ z_\omega \end{bmatrix} \qquad (4.2)$$

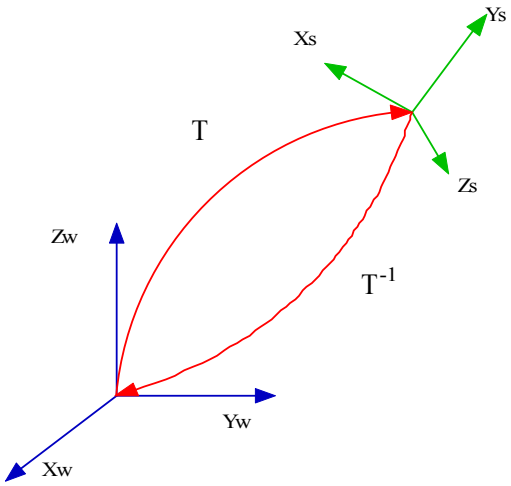


Figure 4-1 Coordinate transforms T and T^{-1} link the object and world coordinate systems.

T is the homogeneous transformation matrix which rotates and then translates a point from the origin of the world coordinate system for $[p_x, p_y, p_z, 1]^T$. By inverting the homogeneous transformation matrix and using Euler angles (Φ, θ, ψ) to express the elements of the rotational part of transformation matrix T we get the inside-outside function for superquadrics in the general position.

$$\begin{bmatrix} x_s \\ y_s \\ z_s \end{bmatrix} = F(a_1, a_2, a_3, \varepsilon_1, \varepsilon_2, \theta, \psi, \phi, p_x, p_y, p_z) \tag{4.3}$$

The expanded function has an additional six parameters which define the position and orientation of the superquadric in space. Furthering our discussion in 3.3.1.1 we have a total of 11 parameters and the appropriate number of control points determining the shape and position of a locally deformable superquadric.

4.4 Case of Extended Superquadrics

Furthering our above discussion, for the extended superquadric we will treat ε_1 as a function of η and ε_2 as a function of ω. While there are numerous ways for providing a basis for expressing ε_1 and ε_2 we will choose a spline representation which interpolates a set of control points. Adjusting these control points provides a natural and user friendly method for altering the shape of the function as discussed in section3.4.2.

However converting each exponent into a corresponding function interpolating 'p' control points now increases the total number of parameters by 2(p-1).

Thus the expanded inside-outside function will be,

$$\begin{bmatrix} x_s \\ y_s \\ z_s \end{bmatrix} = F(a_1, a_2, a_3, f(\eta), f(\omega), \theta, \psi, \phi, p_x, p_y, p_z) \tag{4.4}$$

where the number of parameters are now 9+2(p-1).

4.5 Error of Fit Function

Most superquadric-based shape recovery and representation methods work by finding optimum parameters for a superquadric by minimizing an appropriately defined objective function (EOF). The EOF serves as an error function to evaluate how accurately the recovered model fits the data. Both the precision of the recovered superquadric model, computational efficiency, sensitivity to noise etc. of the data fitting process largely depend on the objective function used.

There has been some research in the area of comparing objective functions [21, 39]. With the exception of the two objective functions explored in this paper, the other functions

investigated have lost their popularity due to either unstable performance or high computational cost. The first objective function is based on the implicit definition of superquadrics [23], and the other on radial Euclidean distance [21].

Most of these investigated only regular superquadrics and deformable superquadrics were not addressed. Recently there has been some research in comparing of objective functions for deformable superquadrics and the objective function based on the implicit definition of the superquadric performed better than the others[40].

However for our choice of superquadrics i.e. extended superquadrics, there is no closed-form error-of-fit function based on the true Euclidean distance[18]. Hence the objective function based on the implicit representation was to form the basis of our EOF function.

$$EOF = \sum_{i=1}^{N\,data} \left[1 - f\left(x_i, y_i, z_i\right)\right]^2 \tag{4.5}$$

where f is the inside out side function which was discussed in section 3.4.1.

It is clear that the function varies quickly when the point (*x*,*y*,*z*) moves to the parts where the exponents are large and slowly where the exponents are small making it bias. An additional exponent is added to *EOF* to obtain an approximately uniform function.

$$EOF2 = \sum_{i=1}^{N\,data} \left[1 - f\left(x_i, y_i, z_i\right)^{f(\eta)}\right]^2 \tag{4.6}$$

4.6 Ambiguity in Superquadrics Shape Description

It was understood that superquadric parameters can produce identical shapes. Besides symmetries with respect to rotation a set of exponent functions in conjunction with scaling

parameters can generate the same shape as another set. A simple example is that the same parallelepiped with slightly rounded edges ($a_1=a_2=1$, $\varepsilon_1=0.1$ and $\varepsilon_2=0.1$) can be represented after a rotation of $\frac{\pi}{4}$ around axis z of the object centered coordinate system, with $a_1=a_2=\sqrt{2}$, $\varepsilon_1=0.1$ and $\varepsilon_2=1.9$ [23].

Most efforts of recovering models are used to represent space occupancy or path planning disregard this issue as their interest being in the shape recovered only. They can accept this lack of uniqueness.

This issue is of particular interest to us because if the superquadric parameters are to act as some form digital signatures to shapes which will promote comparison and indexing, there has to be uniqueness in the parameters and the corresponding shape recovered. This could be handled in the objective function by constraining the parameter space during recovery and forces the convergence to unique parameters.

Hence to compensate incomplete information in a single view and solve the ambiguity, the minimum volume constraint is added to the objective function. The scalar value $a_1.a_2.a_3$ is positively proportional to the volume which leads to the resulting superquadrics fitting the data and having minimum volume as well. The our EOF function becomes,

(4.7)

$$EOF3 = \sqrt{a_1 a_2 a_3} \sum_{i=1}^{N\,data} \left[1 - f\left(x_i, y_i, z_i\right)^{f(\eta)} \right]^2$$

4.7 Optimization Problem in Standard Form

We can now go ahead and set up the fitting model as an optimization problem.

$$F = \sqrt{a_1 a_2 a_3} \sum_{i=1}^{N\,data} \left[1 - f\left(x_i, y_i, z_i\right)^{f(\eta)} \right]^2$$

Minimize

where

$$f\left(a\tan 2\left(\eta\right)\right) > 0$$
$$f\left(a\tan 2\left(\omega\right)\right) > 0$$
$$a1, a2, a3 > 0$$

The variables being

Scaling Parameters: a_1, a_2, a_3

Control Points: $C1_i, C2_j$ *where* i and j = Number of control points.

The side constraints are that the exponents and the scaling parameters must always be positive.

4.8 Choice of Optimization method

Now that we have completely defined our optimization problem, the choice of method used for obtaining accurate results is very crucial as well. We have three principal issues to consider prior to selection of an optimization method.

I. Highly non- linear objective function which exhibits multiple local minima.

II. A method which is more or less independent of initial estimates of the model.

III. We do not have a fixed number of variables like regular superquadrics the number of control points may increase till exact shape representation is obtained.

Keeping these in mind the following sections will discuss the issues un greater detail.

4.8.1 Conventional Approach

The conventional method of fitting superquadric models to 3D data is through nonlinear least-squares minimization of an error-of-fit function using an algorithm such as that proposed

by Levenberg-Marquardt[41]. The algorithm then proceeds iteratively attempting to converge on a minimum in parameter space after being supplied with a set of model parameters.

Least-squares minimization techniques such as the LM algorithm follow a rigidly defined strategy for finding a minimum in parameter space and the minimization process can, in general, only guarantee convergence to a local minimum. This can be a significant problem when the parameter space is of complicated topology, i.e. exhibits multiple local minima, as is the case when fitting superquadric models to 3D points. The initial set of model parameters supplied to the algorithm determines to which minimum it will converge. Unless minimization is started in the vicinity of the global minimum it is unlikely to reach it, and will converge to a minimum local to the starting point instead. The success of the algorithm in reaching the global minimum depends greatly on its starting point in the parameter space, and this reliance on the initial model parameters represents the main limitation of the least-squares minimization approach. Most researchers attempt to maximize the effectiveness of the LM algorithm by providing it with a set of initial model parameters that predict in some way how the model will fit the data points.

However this becomes an issue with our model because besides the parameters required for a regular superquadrics we need a fair estimate of the number of control points and give them "good" initial points. Since we have no way of extracting such kind of information prior to the optimization routine this becomes a severe limitation for the LM and our initial results with 3D were very discouraging.

4.8.2 Genetic Algorithms

Genetic algorithms, first introduced by Holland [42], are class of robust search and optimization procedures based on the ideas of genetics and the Darwinian theory of evolution. Genetic algorithms are now widely applied in science and engineering as adaptive algorithms for

solving practical problems. Certain classes of problem are particularly suited to being tackled using a GA based approach.

The general acceptance is that GAs are particularly suited to multidimensional global search problems where the search space potentially contains multiple local minima. Unlike other search methods, correlation between the search variables is not generally a problem. The basic GA does not require extensive knowledge of the search space, such as likely solution bounds or functional derivatives. A task for which simple GAs are not suited is rapid local optimization; however, coupling the GA with other search techniques to overcome this problem is trivial.

In contrast to least- squares minimization techniques, a GA explores many areas of a parameter space in parallel. GA works by discovering, emphasizing and recombining good "building blocks". Their efficiency derives from the highly parallel fashion in which this occurs. This idea requires that good solutions consist of good building blocks that are particular combinations of parameters which relate to high fitness. This property allows a GA to avoid becoming trapped in a local minimum, even when searching a topologically complex parameter space. Hence the best choice of optimization algorithm after several test runs was decided to be a GA.

4.8.2.1 Implementing a Genetic Algorithms

The important elements which govern the fate of a GA are the concepts of parent selection, crossover and mutation. We have utilized a binary encoding scheme for the variables.

GA selection operators perform the equivalent role to natural selection. The overall effect is to bias the gene set in following generations to those genes which belong to the most fit individuals in the current generation. There are numerous selection schemes described in the literature; "Roulette wheel" selection, tournament selection, random selection, stochastic

sampling. These, in essence, mimic the processes involved in natural selection. In our scheme we have chosen the Roulette wheel selection.

There is also a multitude of methods to create a coherent set of genes from two parent sets. If following the schema formalism, crossover is the process by which good schema get combined. The most commonly used forms are shown in Figure 4-2 and have been used in our algorithm.

Mutations enable the GA to maintain diversity whilst also introducing some random search behavior. As for crossover, many types of mutation operator may be conceived depending upon the details of the problem and the chromosomal representation of solutions to that problem. We have generally used a single point mutation.

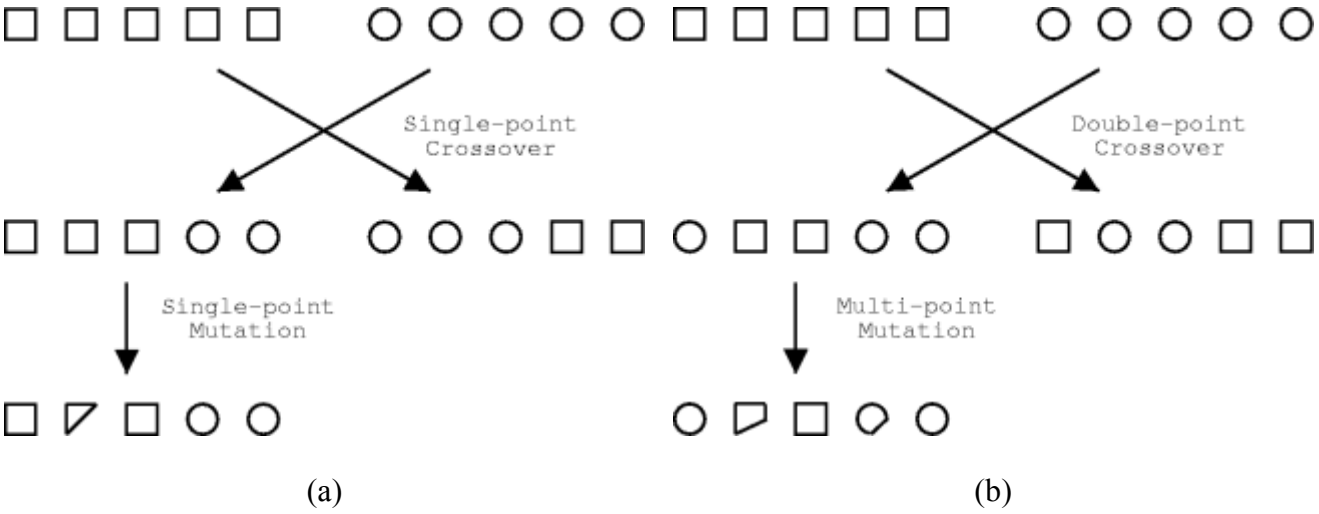


Figure 4-2 Types of Crossover and Mutation Schemes

Given the discussion above, use of a GA would appear to be a viable alternative to LM, avoiding the drawbacks associated with that approach. However, there is a significant obstacle to applying a GA to a complex nonlinear optimization problem such as that described here,

specifically the high computational cost due to the slow convergence rate of the GA. In general, a GA will take significantly longer to converge than least-squares minimization routine applied to the same problem. This is primarily because a GA does not make direct use of local parameter-space information to determine the most promising search direction. Rather, the GA explores the parameter space in a less directional fashion, considering many different regions in parallel.

In our own comparisons of the performance of a GA with that of LM when applied to the problem of superquadric fitting (especially in 2D), we have found that, for the same data set, the GA takes significantly longer to converge. However incase of more complex 3D scenarios LM method yielded poor results and the only way was to accept the high computational cost of a GA to achieve reasonable results.

4.9 Computation for initial estimates for parameters

When we see our formulation so far it is apparent that we do not have a fair estimate of the control points for the optimization algorithm, however by analyzing the points it may be possible to estimate the size, position and orientation of the model that most closely fits them, with reasonable accuracy. This "initial fit" of the model to the data should provide the algorithm with a reasonably good route to the minimum.

4.10 Moment Based Estimation

Moment based techniques have a well established tradition in object recognition and pose estimation[43]. This is of particular interest to us because we need to model the superquadric based in arbitrary space as discussed in 4.3 for any practical application. Further such a

technique can also give some fair initial estimate on the size of the superquadric in terms of the scaling along all three axes.

The original position of the initial ellipsoid based on the cloud of points is going to be the mean of each coordinate i.e. $(\bar{x}, \bar{y}, \bar{z})$.

The first step is to translate the object from the world coordinate system with origin at $(\bar{x}, \bar{y}, \bar{z})$ to the object coordinate system with origin at (0, 0, 0). The next step would be to calculate the orientation of the object centered coordinate system. To accomplish this we first need to assemble the matrix of central moments.

$$M = \begin{pmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{yx} & I_{yy} & -I_{yz} \\ -I_{zx} & -I_{zy} & I_{zz} \end{pmatrix} \tag{4.8}$$

Where

$$I_{xx} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \bar{y})^2 + (z_i - \bar{z})^2$$

$$I_{yy} = \frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2 + (z_i - \bar{z})^2$$

$$I_{zz} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \bar{y}) + (x_i - \bar{x})^2$$

$$I_{xy} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \bar{y})(x_i - \bar{x})$$

$$I_{xz} = \frac{1}{N}\sum_{i=1}^{N}(z_i - \bar{z})(x_i - \bar{x})$$

$$I_{xz} = \frac{1}{N}\sum_{i=1}^{N}(z_i - \bar{z})(x_i - \bar{x})$$

The central moments are with respect to $(\bar{x}, \bar{y}, \bar{z})$ which is the center of gravity. The axes of coordinate system are rotated so that the axes are aligned along the axes of minimal and

maximal moment of inertia. This rotation produces coordinate system, and inertia matrix computed in the world frame is diagonal[15]. We use singular value decomposition to obtain the rotation matrix and eigen vectors.

We decided to orient the object such that the z coordinate axis coincides with the axis which exhibits some extent of rotational symmetry. However since we are dealing with highly asymmetrical objects, this luxury of favorable orientation as done by some researchers dealing with symmetrical objects[23] [15] , is not always available to us.

Once the orientation is determined the size of the initial superquadric is simply the of the farthest range point along each coordinate axis. The initial estimates of these parameters are sometimes very close to the actual value and aid the fitting procedure tremendously.

4.11 Algorithm for Shape Recovery

Based on formulation and discussion above with regards to various aspects of shape recovery using superquadrics, the following algorithm was developed and used for superquadric fitting.

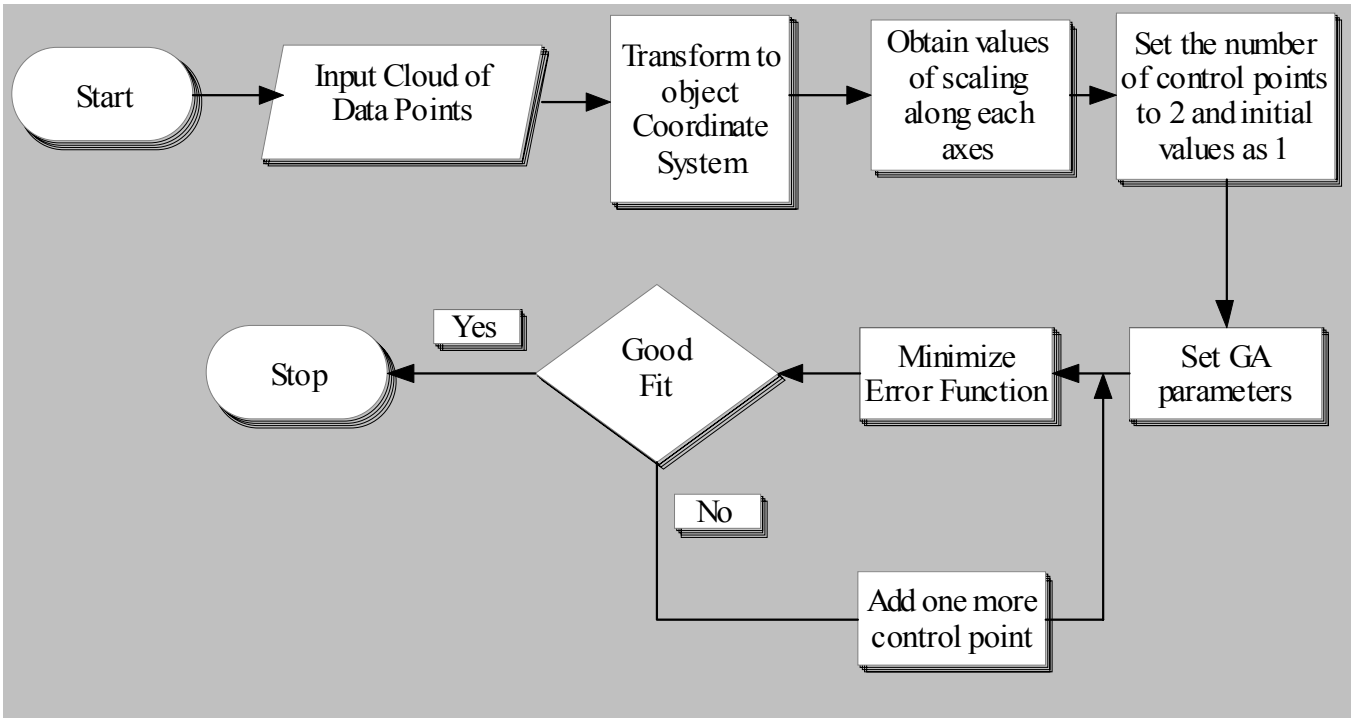


Figure 4-3 Flow Chart depicting shape recovery Algorithm

4.12 Iterative Segmentation and Recovery

We have already seen how regular superquadrics have some drawbacks in approximating to irregular shapes and hence we utilized extended superquadrics to over come these limitations. The other approach is to derive a structural description by first segmenting an object into primitives and then model each primitive with a single superquadric[17, 26, 27, 30, 44-47].

We have tried to incorporate in our algorithm the benefits of both approaches, by establishing a segmentation procedure prior to applying the extended superquadric recovery. Once we have initial estimates to the scaling parameters from the algorithm, we fit a single superquadric to the whole data-set. The object is then divided into two parts by using a error distribution based on this initial fit. The orientation of the dividing line is determined by a

hierarchical optimization algorithm which orients it self to a cluster of points which have minimum standard deviation of the error distribution. This proves to be very useful in approximation of shapes that exhibit deep concavity in their shapes which is very difficult to represent/capture using a single superquadric.

4.12.1 Working of the Segmentation routine

As mentioned earlier, it might be difficult to fit a highly complex model using a single extended superquadric. The most obvious alternative is to have a two step process of first segmenting the object into two components and then fit it to each part.

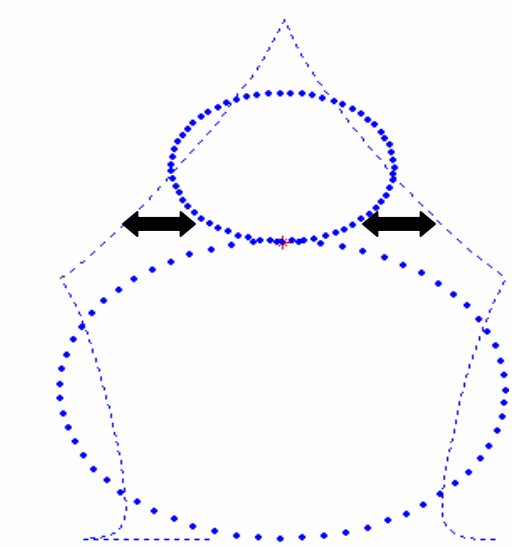


Figure 4-4 Eight control point extended Superquadric.

Extended superquadrics are not very good at representing deep concavities in objects as seen in the figure above where an eight control point superquadric completely fails to represent the shape at hand. Hence the concavity makes the most attractive property to use for identifying a location for partitioning the data. Hence after the initial superquadric fit the algorithm searches for points with maximum error. This can be easily identified as our objective function is an error

function within itself. We compute the error distribution for the data-set by using the EOF function.

$$EOF2 = \sum_{i=1}^{N\ data} \left[1 - f\left(x_i, y_i\right)^{f(\eta)} \right]^2$$

If the point lies within the superquadric its value is positive, otherwise it is zero or negative. Considering that we are interested in the concave points, which will be positive, a direct measure of the point with largest error will give us the point through which the dividing line must pass.

In the second phase we need to select an orientation for the line passing through the identified point. To obtain the orientation we first define two lines at a distance *d* from the existing line $ax + by + c = 0$, having the same orientation. Considering that M number of points lie between these lines, the evaluation function for standard deviation of their errors will be:

$$\sum_{i=1}^{M\ data} \left[\frac{(EOF)^2}{M} \right]$$

We setup a small optimization problem where the above function is minimized to obtain the orientation such that the cluster of all points with high error values is partitioned by the line. The variables are the coefficients of the equation of the line *a, b, c* and the distance *d* of the enclosing lines from the original line.

The line thus obtained is utilized to partition the data into two sets and individual superquadrics are approximated to the same. The partitioning is not always appropriate, but is extremely influential in certain cases. We will revisit this in one of our case studies in the following chapter. This was an attempt to improve the existing methodologies, but requires more detailing and inclusion of further intelligence to be integrated completely in the algorithm for all cases.

4.13 Summary

We began by considering shape as a mapping between sets of 2D and 3D points into the parametric domain an appropriately defined vector space of model parameters of deformable superquadrics[23-25]. The assumption of intrinsically symmetry often fails in modeling numerous real-world biological objects. Hence we represented the changing exponents as spatially-varying functions, creating the class of extended superquadrics[18, 36]. The resulting extended superquadrics retained many desired properties of superquadrics such as compactness, controllability, and intuitive meaning, which are all advantageous for shape modeling, recognition, and reconstruction. We also adopted the recently developing trend of modeling the varying exponents in terms of Cubic-spline curves, parameterized by the control points. The methodology was developed towards achieving the goal to integrate parametric curve surface theory that allows for local control with volumetric and contour shape representation that permits users to easily and quickly capture a fair estimate of the shape of a solid object in a physically plausible fashion.

5 Case Studies

In the previous chapter we discussed the utilization of extended-superquadric models and integrated with an automated method to achieve robust shape recovery.

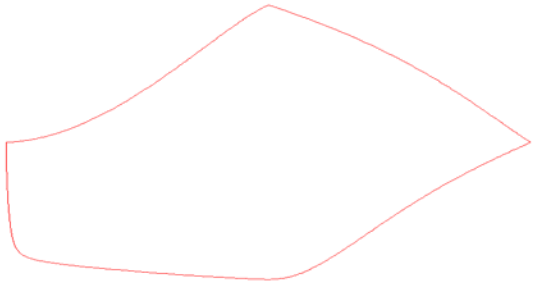
In this section we hope to present some of the results obtained and provide some sort of benchmarking to promote its eventual widespread acceptance. Before moving onto 3d data, we present the results on sets of 2D data to show the whole fitting process clearly.

5.1 2D Case

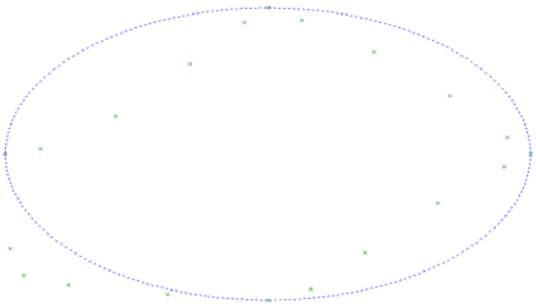
The following case exhibits the progression of the algorithm from the initial estimate through the various stages and improving the fit till the best solution has been achieved. The test curve chosen exhibits both convex and concave local deformations to test our EOF in both extremes. 20 points were extracted from the test curve which constitutes our cloud of points. The cloud of points have been generated about the origin hence no transformation is required for this example.

The following EOF used is for the recovery as was motivated in 4.3 which suffices for 2d recovery and doesn't require the volumetric constraint.

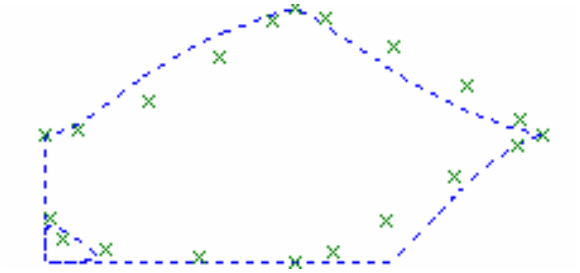
$$\text{EOF2} = \sum_{i=1}^{N\,data} \left[1 - f\left(x_i, y_i\right)^{f(\eta)} \right]^2$$



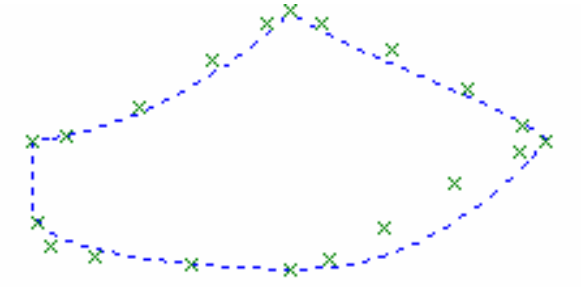
(a) Test curve



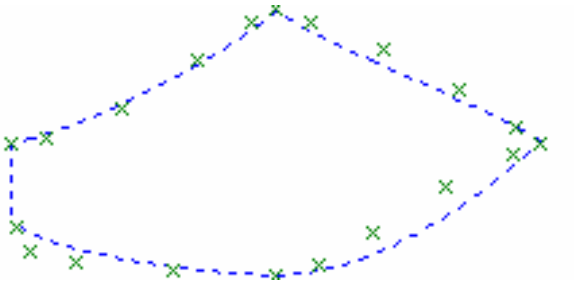
(b) Initial fit of symmetric SQ for initial estimates



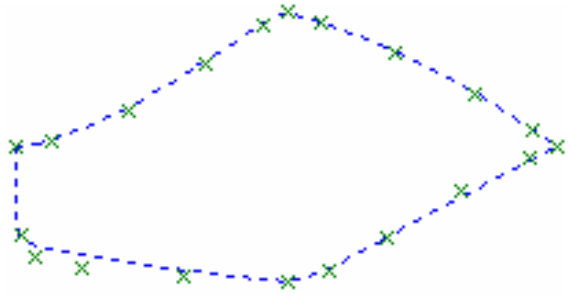
(c) Number of Control Points = 2
EOF = 0.2207



(d) Number of Control Points = 3
EOF = 0.1684



(e) Number of Control Points = 4
EOF = 0.1121



(f) Number of Control Points = 5
EOF = 0.0102

Figure 5-1 An example exhibiting 2d recovery process.

Table 1 Depicting the values of variables at each stage of the recovery of the curve above

Iteration	Xaxes Scaling	Yaxes Scaling	Control Point 1	Control Point 2	Control Point 3	Control Point 4	Control Point 5
1	8.2711	4.0929	1.0000				
2	8.2711	4.0929	0.2552	3.0000			
3	8.2711	4.0929	0.2500	1.7590	2.5301		
4	8.2711	4.0929	0.2500	1.3308	2.2084	2.3470	
5	8.2711	4.0929	0.2645	0.7500	1.5772	2.0651	2.5322

An interesting thing to observe from the above results is that the initial estimates of the scaling values are very accurate and remain unchanged through the course of the algorithm. This affirms that initial parameter estimation is highly accurate and can be used to assist the optimization algorithm to converge to the global minimum. This holds of very high relevance in highly irregular shapes which require large number of control points.

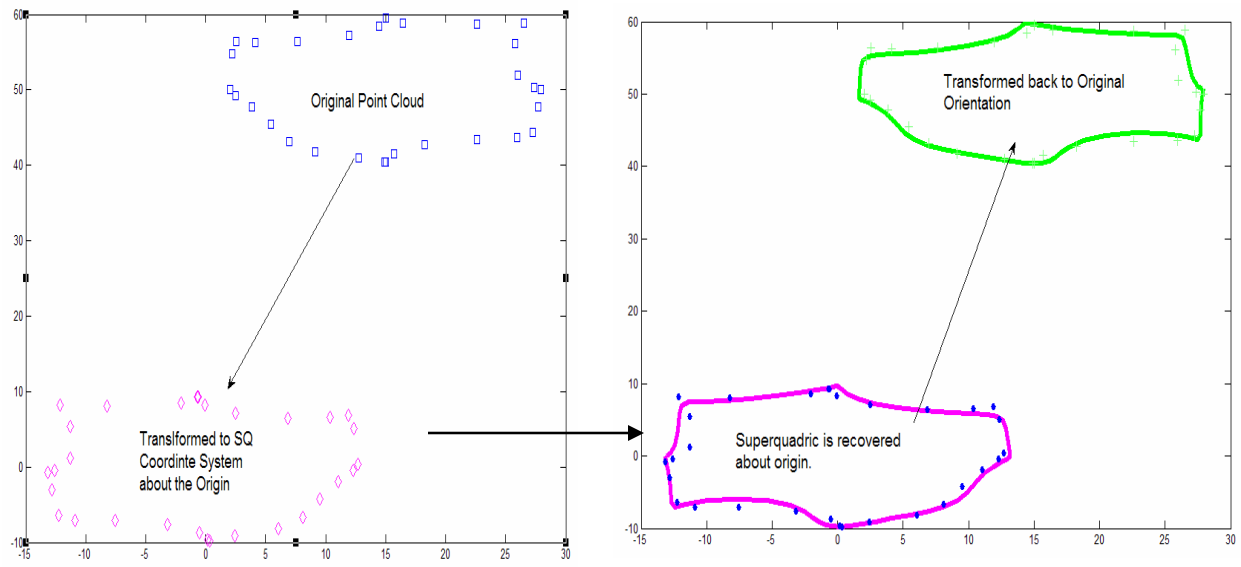
A very highly accurate representation was obtained using 7 parameters. This gives a feel of the compactness of the representation, in addition to the broad geometric coverage which has been motivated in previous discussion. The representation performs very well for 2D scenarios with a small number of points describing the contour and encompasses enough topographic accuracy to suit many practical scenarios and objects with closed shapes. However when the data set is large the approximation increases and then the trade off between low end parameterization and accuracy becomes more evident.

5.1.1 General Position

As we had earlier disused in section 4.3 the data points obtained from various sources will not be in the same coordinate frame as the recovered superquadric and hence we discussed

how the algorithm is designed to recover the superquadric when data points are placed in the world coordinate system. Further this also exhibits the functioning of moment based estimation as discussed in 4.10 to obtain the orientation and translation of the data points in the object centered coordinate system. This helps the recovery by aligning the axes to assist the recovery process.

The following example deals with one such case and depicts pictorially the recovery procedure with emphasis on the above operation. It can be observed that after the forward transform the outermost data points are oriented along the x-axes and their measure can be used as an initial estimate for the scaling parameter for the superquadric.

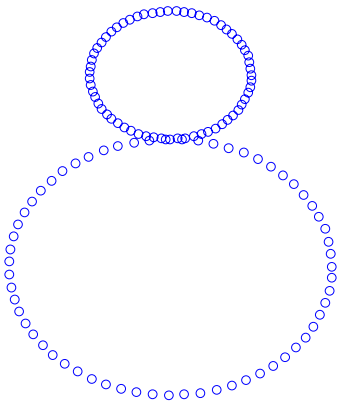


(a) **(b)**

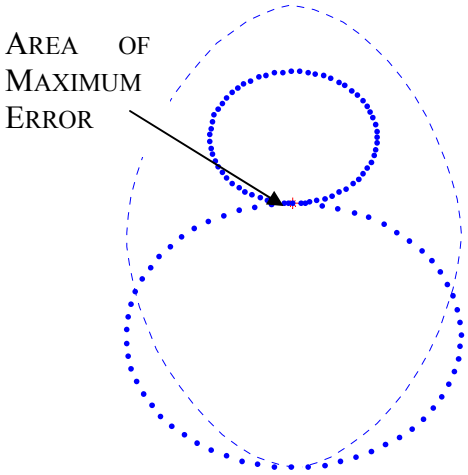
Figure 5-2 (a) Forward Transformation from the World Coordinate System to Object Centered System and (b) Inverse transformation after recovery.

5.1.2 Segmentation and Recovery

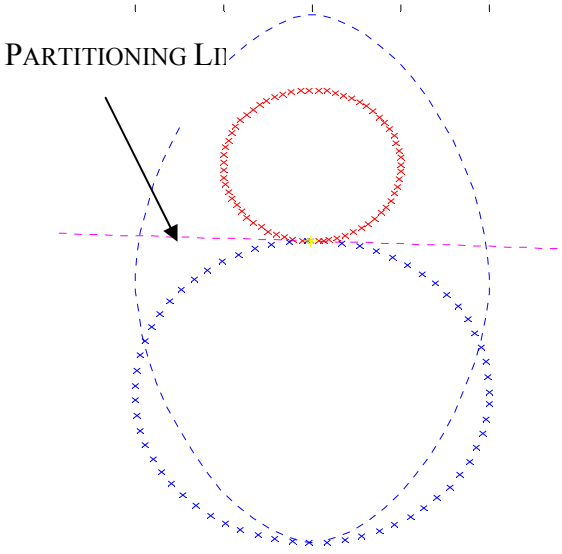
In Section 4.12 we discussed the shortcomings of a single superquadric in approximating shapes with deep concavities and hence formulated an iterative segmentation scheme which could be integrated into the developed algorithm. Following figures indicate the application of the iterative segmentation to a test case of a snow man.



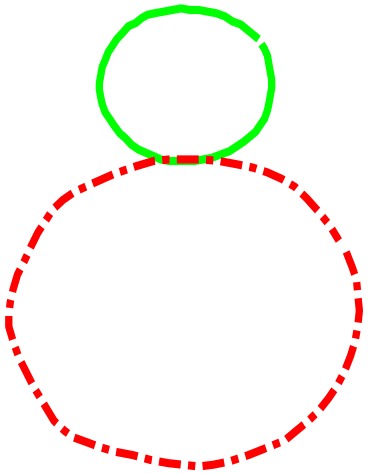
(a) Point Cloud Data



(b) Initial Superquadric Fit - EOF= 40



(a) Data Segmentation based on initial fit



(b) Two Superquadrics to approximate the data
EOF1= 0.575
EOF2= 0.326

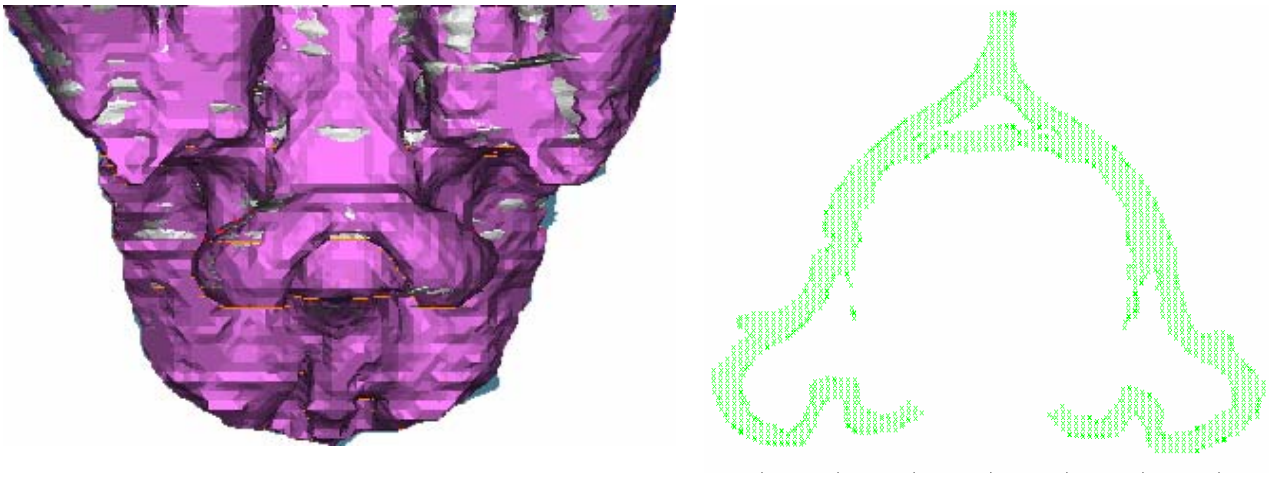
Figure 5-3 Iterative segmentation and recovery

As can be observed the initial superquadric fails to fit the data with 6 control points and per our previous test runs is not able to approximate it to an acceptable level at higher control points either. However once the data is segmented at the cluster of points, with the measuring index being the error values, two individual superquadrics can easily be utilized to approximate the entire data set with much higher degree of accuracy.

This is presents it self to be a very powerful and useful addition to the algorithm, however its performance is highly case dependent and the partitioning is not always appropriate.

5.1.3 Test Case

To test the algorithm on large dataset and to emulate occurring objects, we have utilized a part of the lower jaw of a skull of a saber tooth tiger which exhibits sufficient irregularities.

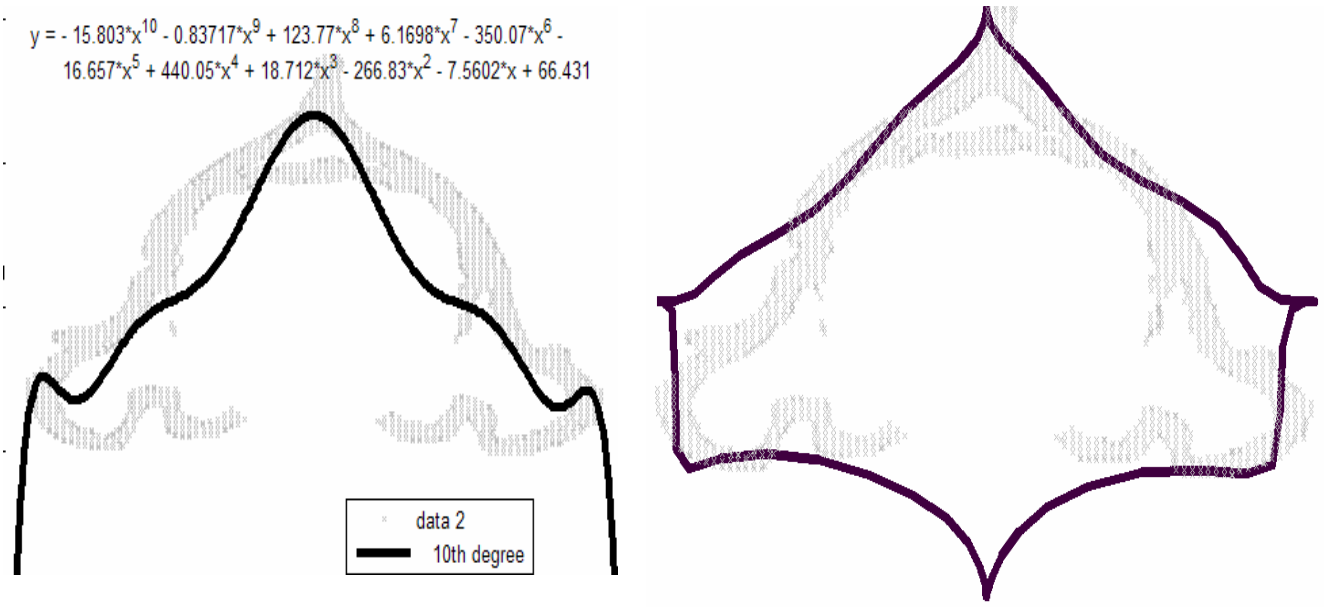


(a) (b)

Figure 5-4 (a) Lower jaw of a saber tooth cat (b) A horizontal section of the jaw with point cloud data of 1890 points.

We utilized the top most section of the part which can be considered as a top view and 1890 data points were extracted to act as our 2D point cloud data. The data set is more of a cluster and doesn't outline the exact contour. Hence the problem now becomes one of approximating a contour amongst a cluster of points while obtaining minimum EOF.

First a 10 degree polynomial was used to fit a curve to the data and the result is presented in the figure below. It hardly provides any visual representation of the desired shape. A higher degree polynomial; in order of hundreds or more; may approximate it better, but we are trying to emphasize on compactness of representation. The crux of the matter is to obtain maximum shape abstraction with minimum parameters.



(a) **(b)**

Figure 5-5 (a) 10th Degree Polynomial Approximation. (b) Shape recovery from a section of the lower jaw using superquadrics.

On utilizing our methodology the following shape abstraction is obtained. The scheme utilized 12 control points for the curve and presents a much better visual representation with minimal parameters. Hence this algorithm has application areas where the accuracy of closed shape can be compromised for compactness and geometric flexibility.

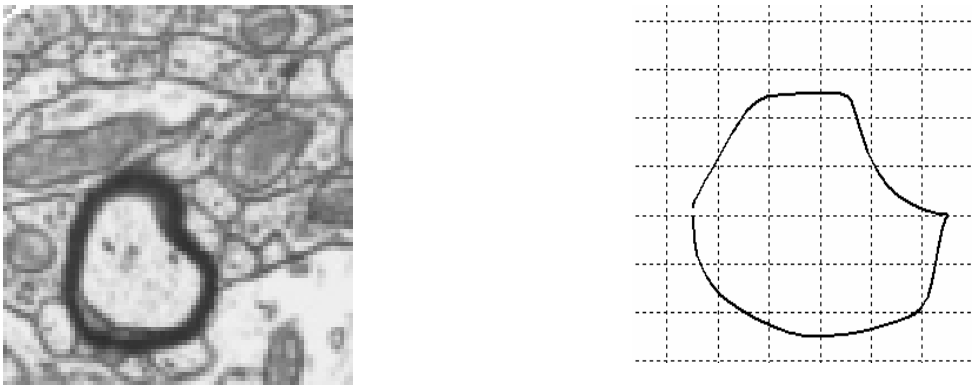
5.1.4 Image Segmentation with Deformable Curves

The segmentation of anatomic structures—the partitioning of the original set of image points into subsets corresponding to the structures—is an essential first stage of most medical image analysis tasks, such as registration, labeling, and motion tracking. These tasks require anatomic structures in the original image to be reduced to a compact, analytic representation of their shapes. However this is a vast research area within itself and has a whole range of applications. We just wanted to motivate an offshoot application while discussing our findings and results.

As shown in the figure below a possible extension and application of our scheme can be to approximate the locations and shapes of object boundaries in images if some identifying points are available. The mathematical representations of such shapes in a succinct manner which can further be utilized for a number of operations like segmentation of healthy tissue surrounding a pathology for enhanced visualization or possibly create MEMS equipments based on shapes obtained from MRI images.

The superquadric derived is a simple yet accurate representation utilizing just four control points from a data set of just 10 points. Most clinical segmentation is currently performed using manual editing. In this scenario, a skilled operator, using a computer mouse or trackball, manually traces the region of interest on each slice of an image volume. It is quite apparent that manual slice editing suffers from several drawbacks. These include the difficulty in achieving

reproducible results, operator bias and fatigue. If such a scheme is integrated with the approach the operator needs to pick just a few points and get a mathematical representation of the boundary/shape which can be used for several post processing steps.



(a) (b)

Figure 5-6 (a) Cell membrane in an EM photomicrograph [44] (b) Shape abstraction using a planar SQ.

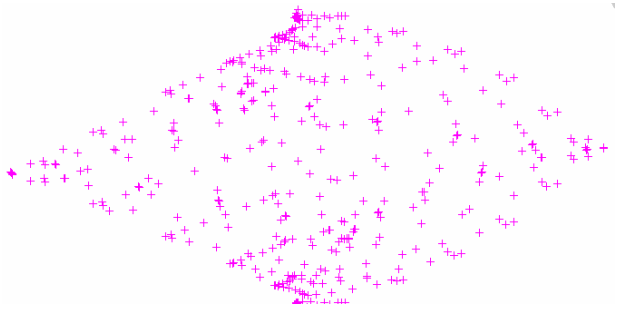
5.2 3D Case

Now we move on to the discussion and performance of the algorithm for 3D scenarios where the input data are in the form of an unstructured 3D point cloud. The algorithm follows the same route as in some of the cases discussed above, but now everything extends to the [x y z] domain. Some obvious modifications are present beginning with the EOF function.

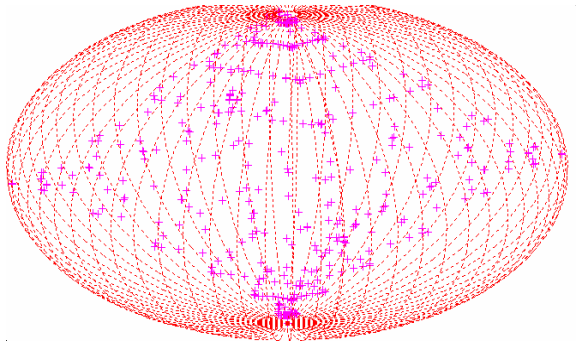
As discussed in section 4.3 to compensate incomplete information in a single view and solve the ambiguity, the minimum volume constraint is added to the objective function. The scalar value $a_1.a_2.a_3$ is positively proportional to the volume which leads to the resulting superquadrics fitting the data and having minimum volume as well. The our EOF function becomes,

$$F = \sqrt{a_1 a_2 a_3} \sum_{i=1}^{N\,data} \left[1 - f\left(x_i, y_i, z_i\right)^{f(\eta)} \right]^2$$

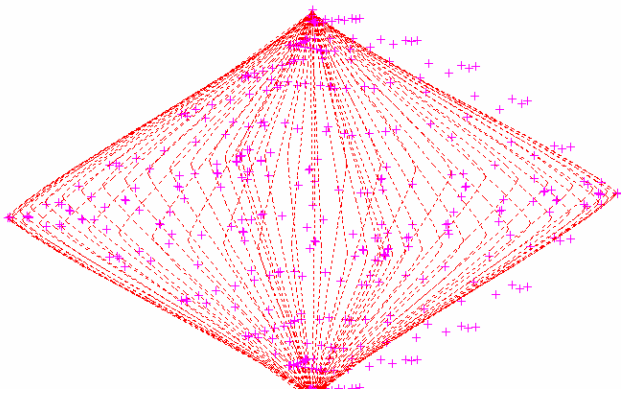
Analogous to the 2D we will see the algorithm improves iteratively till the best fit is obtained. The test object is such that it exhibits both concave and convex behaviors at different locations to test the flexibility in terms of local deformation of the recovery process.



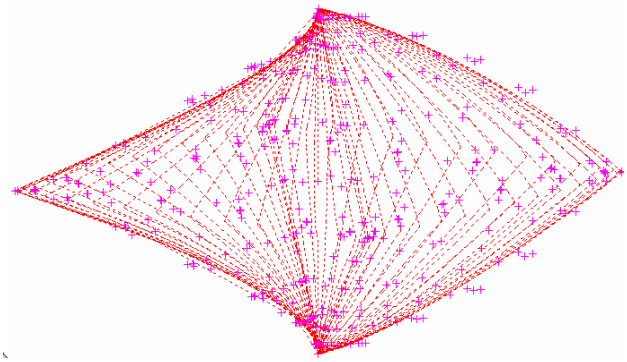
(a) Point Cloud Data of 50 points



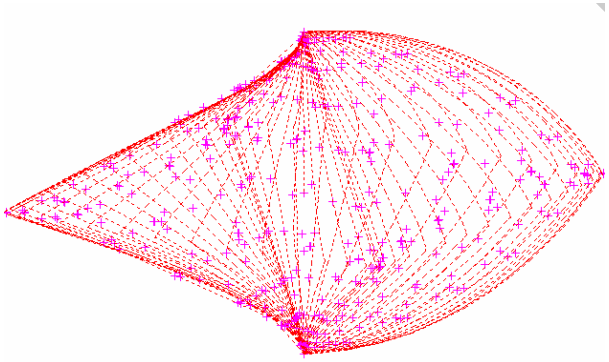
(b) Initial fit of symmetric SQ for initial estimates



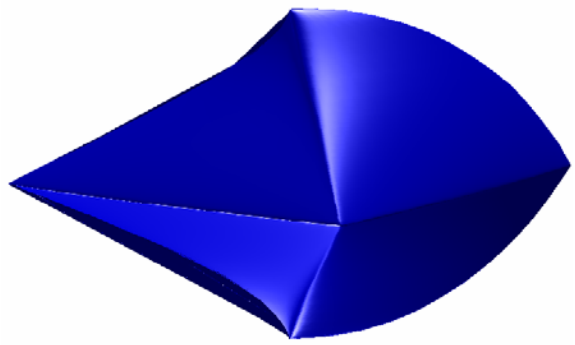
(c) Number of Control Points = 4
EOF = 21.972



(d) Number of Control Points = 8
EOF = 6.397



(e)Number of Control Points = 10
EOF = 2.235



(f) Surface Model of Recovered SQ

Figure 5-7 An example exhibiting 3D recovery process.

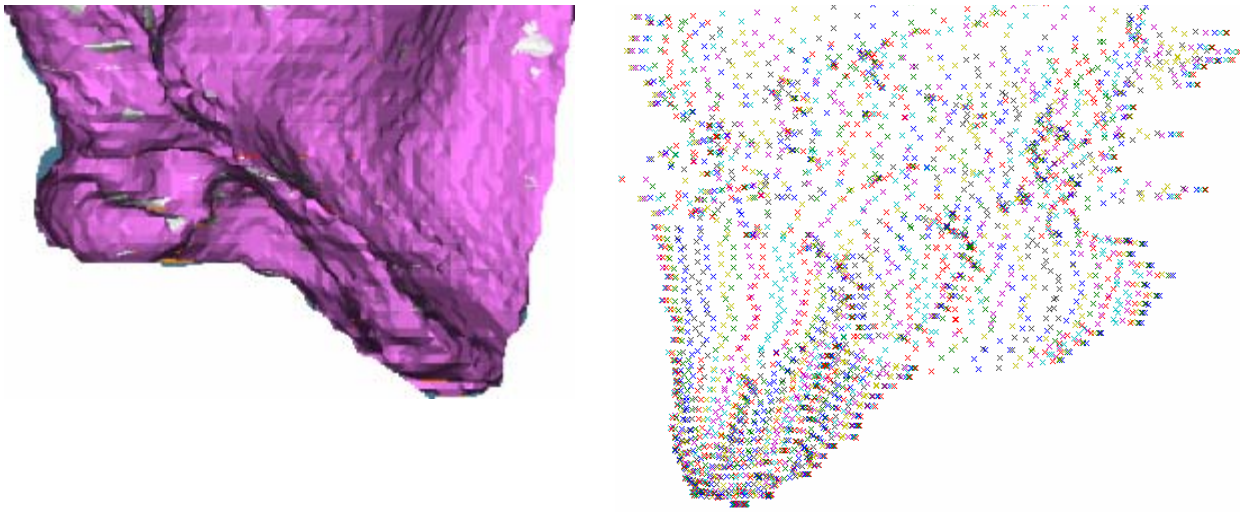
As depicted above the final superquadric was recovered with ten control points capturing both concave and convex features. Looking at the problem from an optimization point of view the genetic algorithm performed 9820 function evaluations to converge to the minimum, for the iteration with 10 control points alone. The time taken was 345.417000 seconds for the last iteration for a 512 MB RAM and 2.0 GHz processor speed. In comparison the derivative based LM method not only improved the solution EOF=1.754 but was able to accomplish the optimization in 12 iterations with 310 function evaluations. This was done in 26.335 seconds.

5.3 Volume Segmentation with Deformable Superquadric Curves

Segmenting 3D image volumes slice by slice, both manually or by applying 2D contour models, is a laborious process and requires a post-processing step to connect the sequence of 2D contours into a continuous surface. Furthermore, the resulting surface reconstruction can contain inconsistencies or show rings or bands. As seen in above examples, the use of a true 3D deformable surface model like ours on the other hand, can result in a faster, more robust recovery technique which ensures a globally smooth and coherent surface. However the capability of representation is limited and cannot work for highly complicated shapes.

As an alternative the following example deals with segmenting noisy data into 2d planes and then uses the algorithm to recover the contour shape of each level. This defines the capability of the method to obtain topological accuracy if sufficient numbers of parameters are increased.

Once all the contours are obtained they are stacked over one another. This evidently does not go hand in hand with our goals of reducing the parameters; however it does exhibit one more way of shape representation using superquadrics.



(a) (b)

Figure 5-8 (a) Shape of the actual Model (b) The sliced contour fit utilizing 4350 points from the original model.

As exhibited in the figure above, around 4350 points were taken from original model in no particular pattern. The algorithm then parses through and divides the point cloud data into equivalent sections; here each contour has been taken at an increase of unit measure in the z direction. The image was then constructed from 47 layers of 2D contours generated using superquadrics. It is visually evident that the intricacy and accuracy of the recovery is way higher. However we lose the advantage of low end parameterization and introduce a costly post processing step.

Bearing this trade offs in mind, the thing to appreciate here is the flexibility of the representation to accommodate for different scenarios by slight variation in the algorithm whilst maintaining automated operation.

6 Interfaces

Interactive technological tools have tremendous potential to address a broad range of objectives from representation, exploration i.e. providing opportunities to explore alternative choices and vary system parameters within a constrained system, to analysis. Our goal was to create and evaluate such an interactive tool in a prototype framework, to explore possible ubiquitous applications of our algorithm.

A PC based GUI interface was developed and utilized for this application which serves a twofold purpose. First it acts as a test bed for the current/future developers to test and tabulate the results. This is accomplished by giving the user various choices for input and alternatives of techniques which could be utilized to perform the calculations. Secondly it caters to the general user by giving him an option to utilize the default settings which are adequate in most situations and at the same time give him more practical methods of inputting data to serve his purpose. It has been developed for a 2D implementation as that is the most intuitive for the user and can be easily extended to the 3D scenario when the application calls for it. A detailed explanation of the working of the GUI will be presented in the following sections.

We also examined extension to web enabled framework to permit users from across the globe to access this system.

6.1 Software

The overall interactive and computational environment is developed within a MATLAB environment, leveraging various toolboxes, for example, GUIDE, Optimization, Web server and

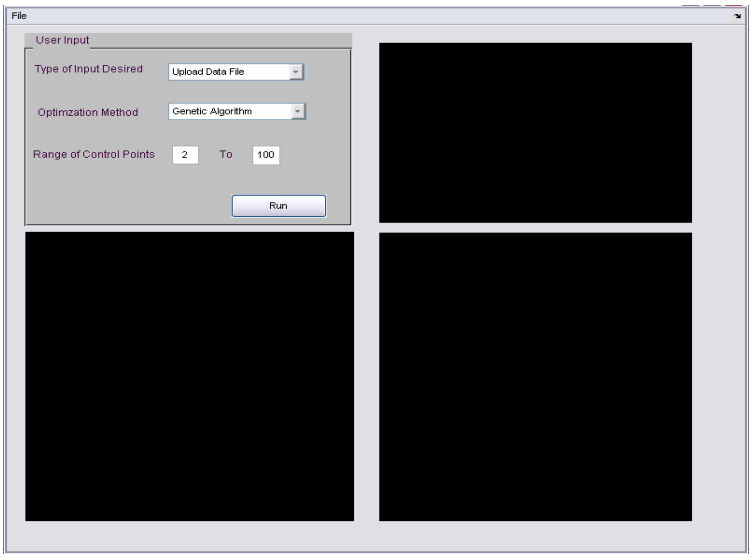
Spline toolbox. This greatly accelerates the initial implementation of the prototype framework. However the utilization of an interpreted language like MATLAB reduces the computational speed drastically. Despite this being an important issue, it takes a back seat during initial stages when the requirement is to run various if-then scenarios and explore various techniques to arrive at the most optimum and suitable for the application.

6.2 PC Based GUI

First we will discuss the various aspects and features of our PC based interface which caters utilization, development and testing of 2D contours.

6.2.1.1 Layout

Figure 6-1 depicts the layout of the GUI and explains the various elements of the interface.



USER INPUT PANEL	PROGRESS/STATUS WINDOW
GRAPHICAL INPUT WINDOW	OUTPUT WINDOW

(a) (b)

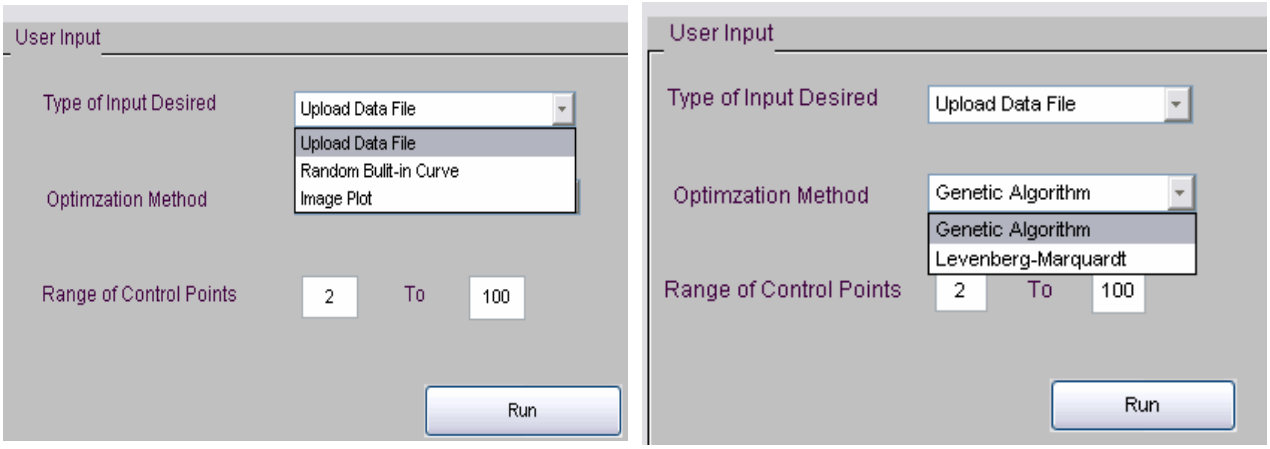
Figure 6-1 (a) Initial GUI when application is started (b) Matrix depicting the Layout

6.2.1.2 Input Panel

The input panel is the first interaction with the user which enables him/her to opt for method of input and set the preferences which will drive the algorithm.

The user may wish to input data in a few different ways and we have tried to narrow it down to three different ways that may be most beneficial to a researcher and or a practitioner in the field. The three forms of input as depicted in Figure 6-2 (a) are:

- *Upload file* - A tab limited data file can be used as input with Cartesian coordinate information of the data points. This is ideal for situations where the data was probably generated manually once and needs to be re entered whilst varying the other parameters of the algorithm. From a more practical stand point, there are various optical gauging devices (Appendix A) which generate a data file with point cloud information after scanning an object.
- *Random Curve Generator* – This option is mainly utilized by a developer or user who wants to learn about/test the topological flexibility of the algorithm.
- *Image Based Plot* - This option supports our discussion in 5.1.4. The biologist and/or operator may upload an existing image of an MRI scan which is mapped as a background reference on the graphical input window and pick points manually to generate the required data.



(a) **(b)**

Figure 6-2 (a) Choice of Input (b) Choice of Optimization Algorithm

We have discussed in considerable detail the effects and effort that have gone in selection of the appropriate optimization technique. The GUI extends the capability to the user to opt between a conventional numeric non-linear method (Levenberg-Marquardt) and a heuristic optimization algorithm (Genetic Algorithms). We have discussed the salient features and limitations of both in Section 4.8. The user can switch between either for performance testing or simply because one failed to give encouraging results.

The flexibility to tweak various parameters of a Genetic Algorithm is established in MATLAB before hand in form of a comprehensive GUI and user can invoke it at any time to set change parameters using the 'gatool' command. The details of which may be found at[48]

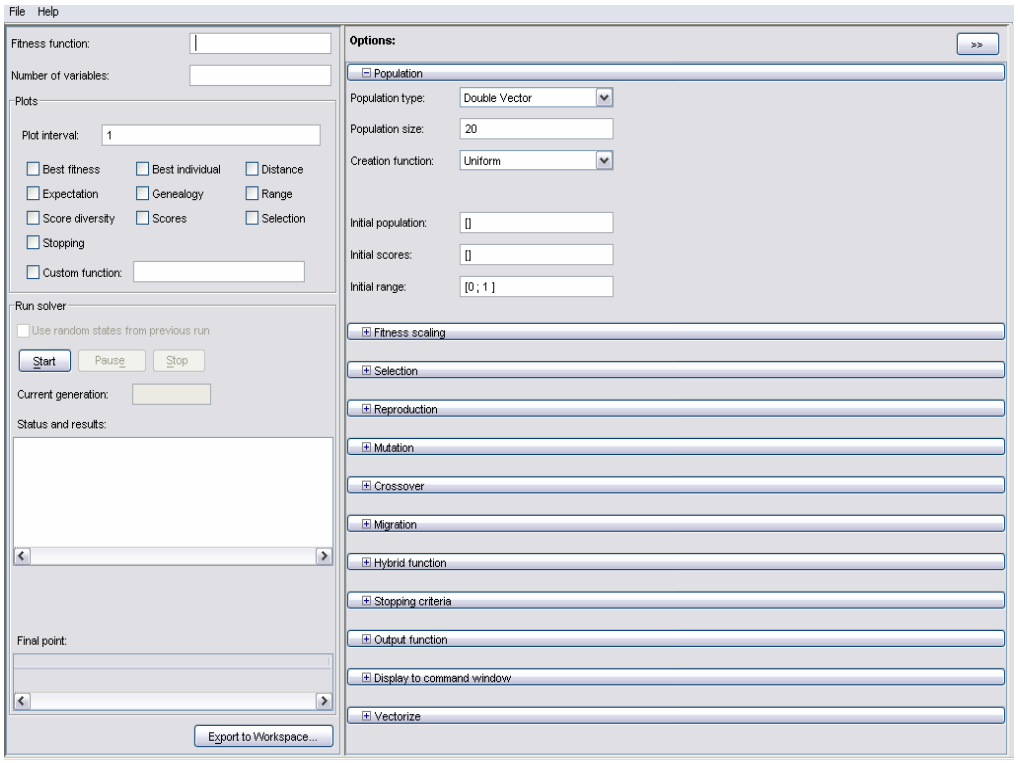


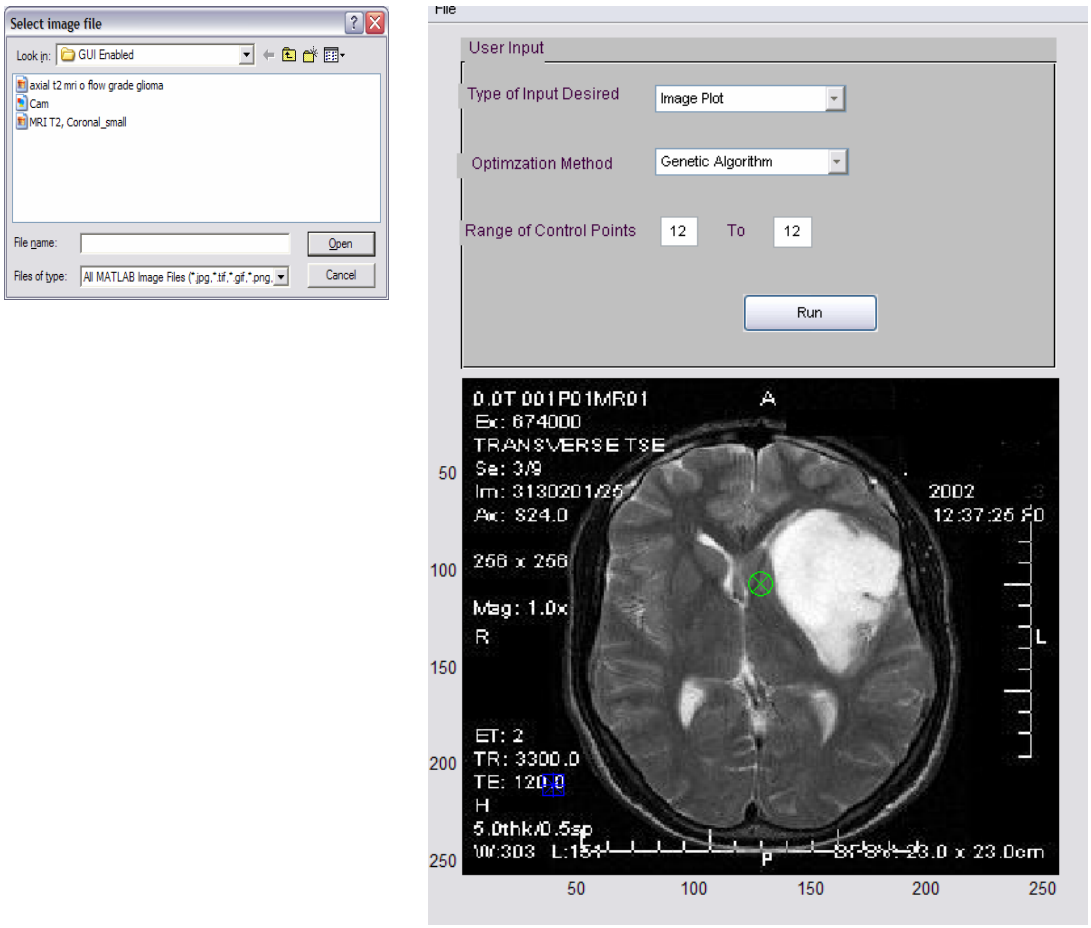
Figure 6-3 (a) GUI invoked by 'gatool' which is built in MATLAB

The third field in the input panel determines the range of control points that the user wishes the algorithm iterates between. A user may find need of this criterion once he has begun having an intuitive understanding of the number control points required to address a certain complexity in the shape. This will allow him to save on some computation time or the flexibility if he is interested in studying the effect of a certain number of control points. A regular user may want to leave the default values which are more than adequate for most situations.

6.2.1.3 Graphical Input

The first two modes of inputs require no further data inputs, however if the user opts for the image plot option, he will be required to input all the points manually. The GUI supports the graphical interaction and the data acquisition to act as input to the fitting algorithm.

The user will first be prompted to upload the appropriate image file, which can be in anyone of the widely used formats like .jpg, .tif, .gif, .png, .bmp. The image file is subsequently displayed in the user interaction window for further operations.



(a) **(b)**

Figure 0-1 (a) Input dialog box for image file (b) Image file displayed for user interaction.

The user at this stage is now able to pick points anywhere on the image. The interface guides the user through various dialogue/input boxes which help the algorithm establish a coordinate system for the underlying image. The required fields are shown in Figure 0-2 and are self explanatory.

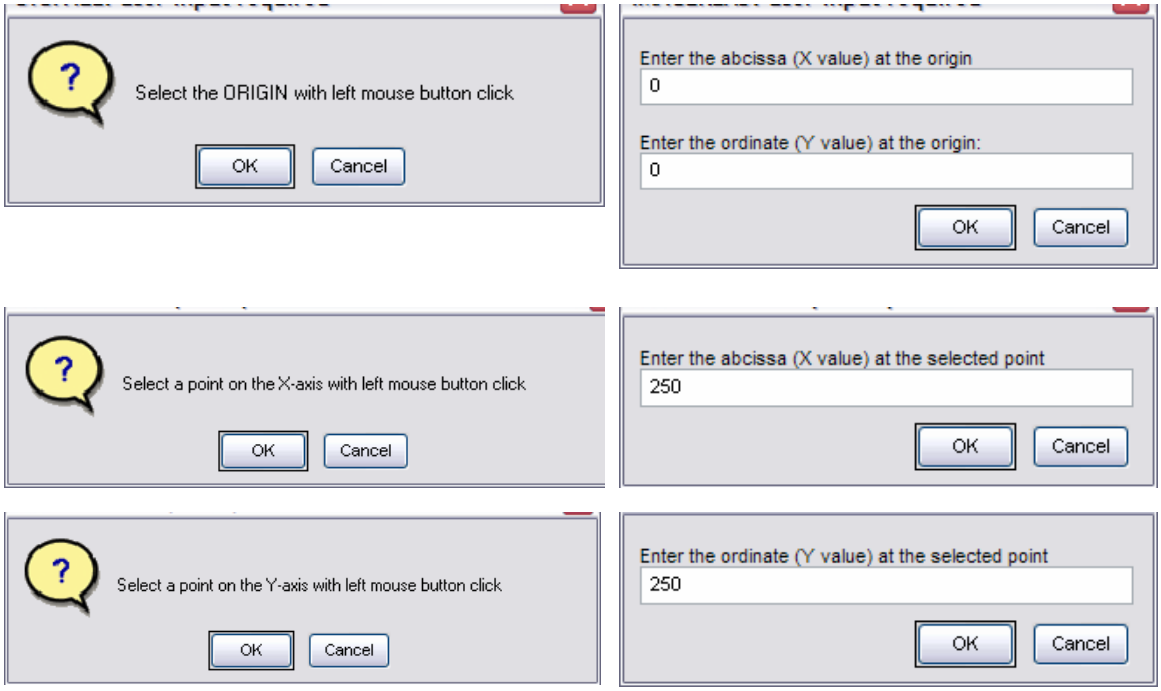


Figure 0-2 Series of dialog boxes required to establish the coordinate system for the image.

Based on these inputs the coordinate frame of the image is scaled and rotated to obtain points with the frame of reference established by the user. Now the user is prompted to pick the points around the region of interest (ROI) and utilize these data points to generate the 2D contour. As shown in Figure 0-3 the sample points are isolated and presented to the user in the adjoining window for verification and subsequently the recovery process may be started. At this point the user is given an option to save the manually entered data points to a file which can utilized repeatedly by using the first input option, hence saving the user from going through the annual process all over again.

Similarly for the earlier two inputs either the data file is parsed or the random curve generator engine is invoked and the point cloud data is extracted or generated. The data is exhibited as in the above case and upon user verification the recover process is initiated.

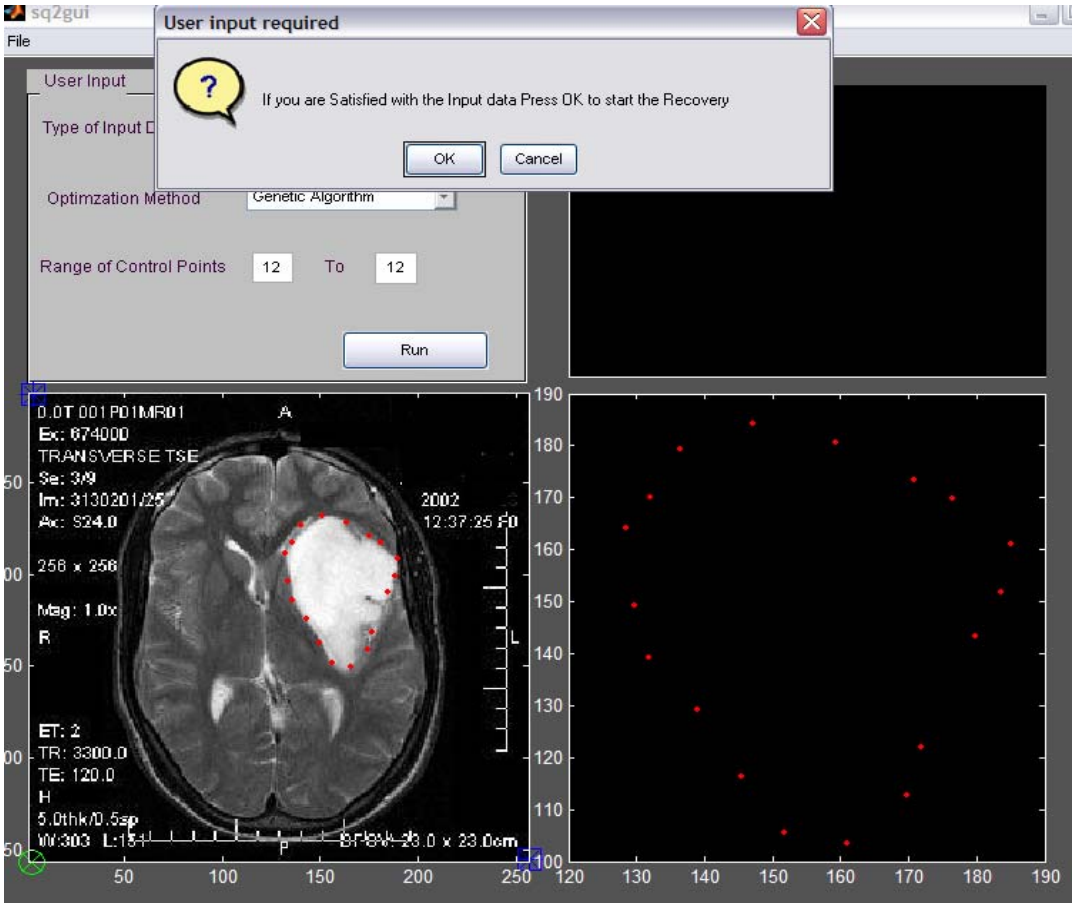


Figure 0-3 State of the GUI after the user manually identifies the ROI

6.2.1.4 Display of Progress

The user can monitor the progress of the optimization algorithm in the top right hand corner as shown in Figure 0-4. The window displays the progress of the algorithm by displaying the function value vs. iteration number. The current function value and number of control points being utilized are displayed on top of the graph.

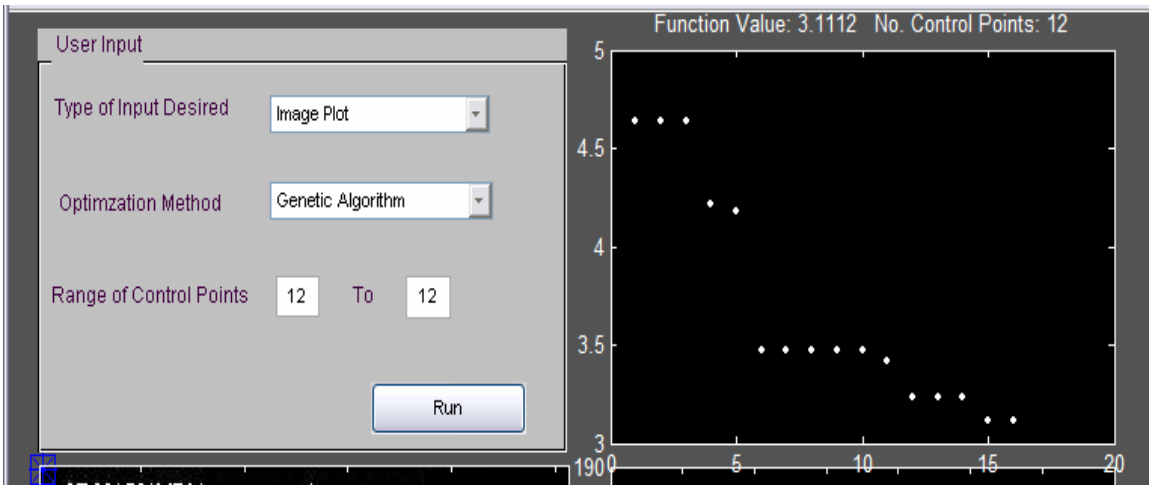


Figure 0-4 GUI showing the progress of the optimization algorithm in the top corner

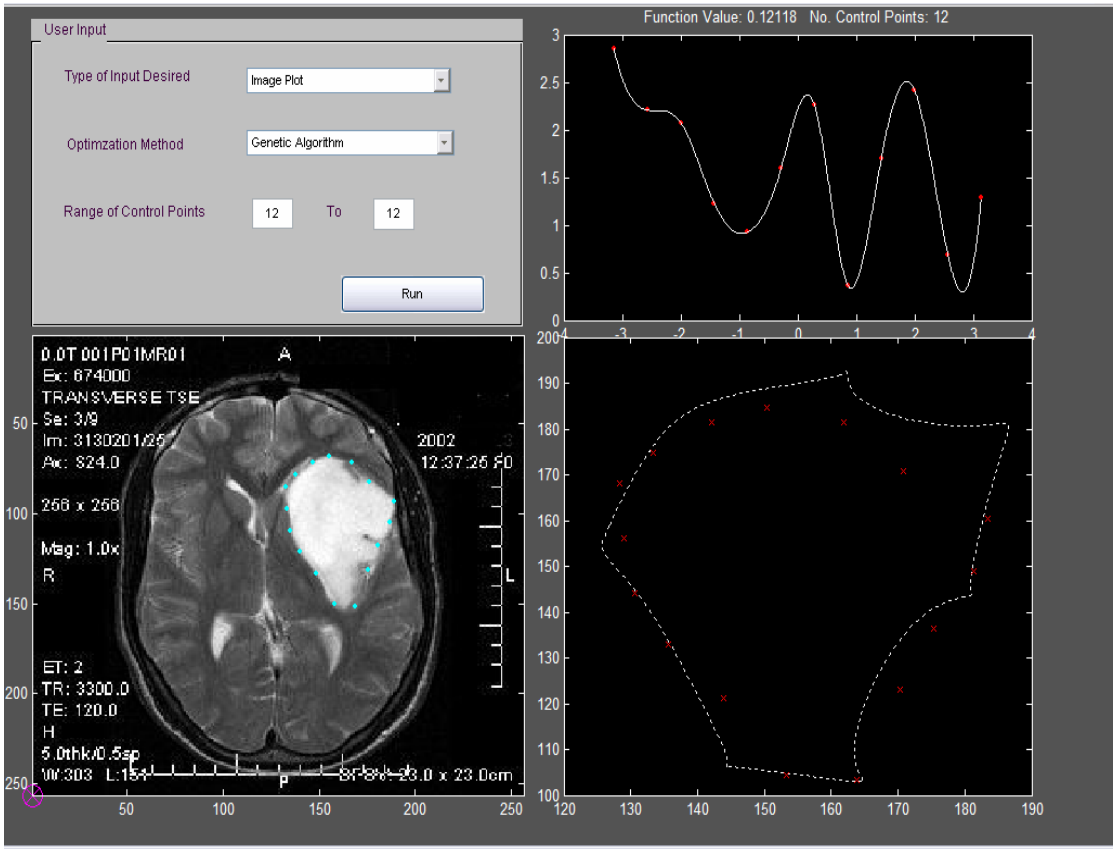


Figure 0-5 GUI displaying the results

6.2.1.5 Display of Results

Figure 0-5 shows the results, with the generated curve in comparison to the data points. The function plot is replaced by the control points along with the interpolated spline, this was essentially the look up table utilized to generate the deform curve. In addition to this the user is prompted to save all the data and parameter values in a text file, this basically is the entire workspace dump. This is invaluable for testing and tabulation of results.

6.2.2 Web Based Interface

Ubiquitous computing refers to an emerging computing paradigm that aims at providing offering user-friendly information and services, anywhere and anytime. Web-based applications offer a range of advantages over traditional, stand alone desktop applications. Using internet technologies based on world-wide standards, it's possible to achieve a far greater level of interoperability between applications than with isolated desktop systems. This means that it's possible to rapidly integrate them within existing infrastructures and platforms. And unlike traditional applications, web systems are accessible anytime, anywhere, via a PC with an Internet connection. Users can access time-critical information from any web browser, including wireless devices, wherever they might be.

6.2.2.1 Motivation

Models of biological objects that we might develop and the output of our shape representation could be placed in web based libraries which make them highly accessible. Researchers could potentially access our models and rerun "experiments' to confirm our results. Moreover, they could expose models to different scenarios, expand the regions of anatomy included in the model, or add to or alter our models and rerun the 'experiment". As new tools or

concepts are developed, they too could be easily and widely disseminated, tested, and improved through this mechanism. We imagine that as such digital libraries grow so will the capacity to quickly and easily perform routine functions such as determining the anatomical element even from very incomplete specimens; identifying similar adaptations across wide taxonomic boundaries, or even just determining whether a specimen is new to science.

6.2.2.2 Architecture of the System

This effort was implemented using a MATLAB web server which enables you to create applications that use the capabilities of the World Wide Web to send data for computation and to display the results in a Web browser. The web server depends upon TCP/IP networking for transmission of data between the client system and MATLAB.

The client computer runs a web browser only, which opens a web page loaded from the server. On the front end we have incorporated a user friendly, high-level block diagrammatic, web interface which can be accessed at any internet enabled computer with a supported browser which gives user access to simulation, testing and refinement. The remote user also views the shape representation of the supplied point cloud data/ parameters in the browser window.

The server computer runs the interface scripts, which communicate with MATLAB and which in turn generates the 3D models which is powered by its visualization capabilities. Connection between client application and MATLAB is shown in Figure 0-6. Client application, usually a web browser, loads an Initial Hypertext Markup Language (HTML) document from Hypertext Transfer Protocol Daemon (HTTPD). After editing the parameters is completed, the client sends data to the HTTPD, which loads matweb through Common Gateway Interface (CGI). Matweb connects to the server by means of an internal protocol. The server loads the requested M-file into a separated copy of MATLAB.

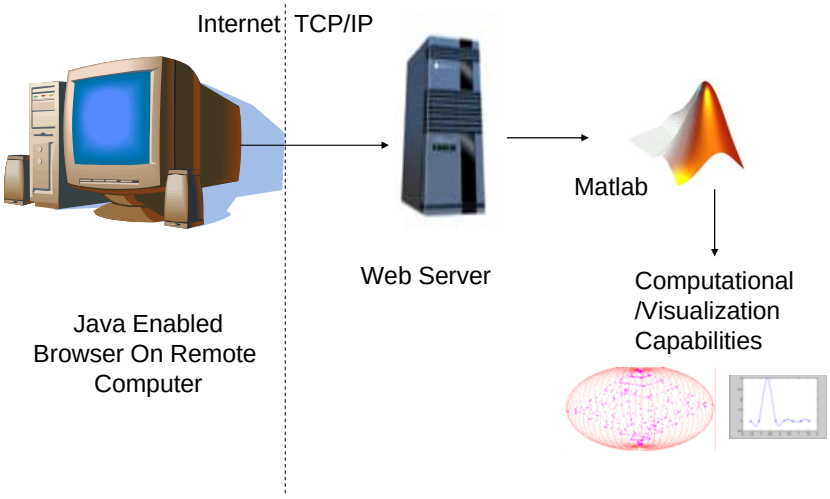


Figure 0-6 Architecture of the web-based framework

6.2.2.3 Web Based Application for Shape Generation

This interface allows the user to generate various 3D shapes by varying the number of control points and randomly assigning values to the same. The web server after initiating the calculations returns a static image to the browser. The user may manipulate the views by changing the rotation angles along each axis.

This application was meant to test the framework to support calculations for 3D data and at the same time educate the user regarding the flexibility in representation of extended superquadrics for asymmetrical objects. The following parametric equation of an extended superquadric is utilized for shape generation which was discussed in Section 3.4.

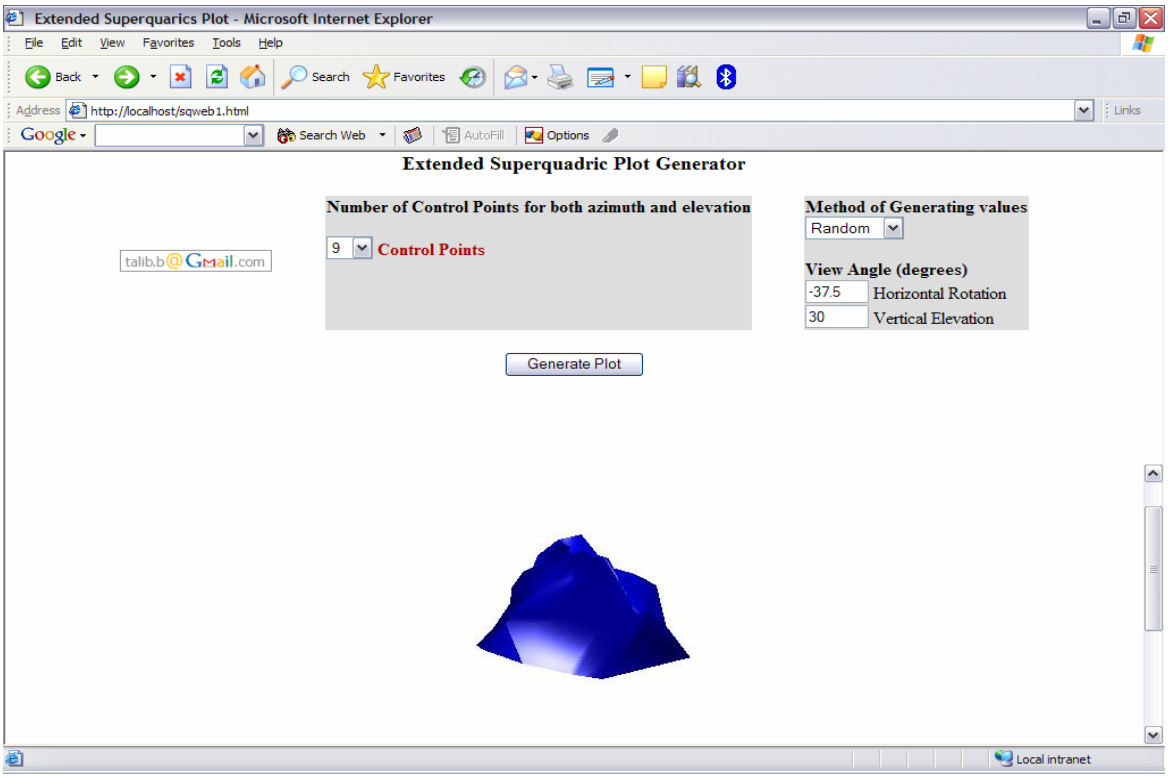


Figure 0-7 Web Based interface for shape generation using extended superquadrics.

6.2.2.4 Web Based 3D Shape Recovery

This application as of now supports just the input of data via a tab limited file which can be uploaded to the server. The algorithm on the server side then parses through the file to generate the input point cloud data for the algorithm and in turn a superquadric is recovered to best fit the data points.

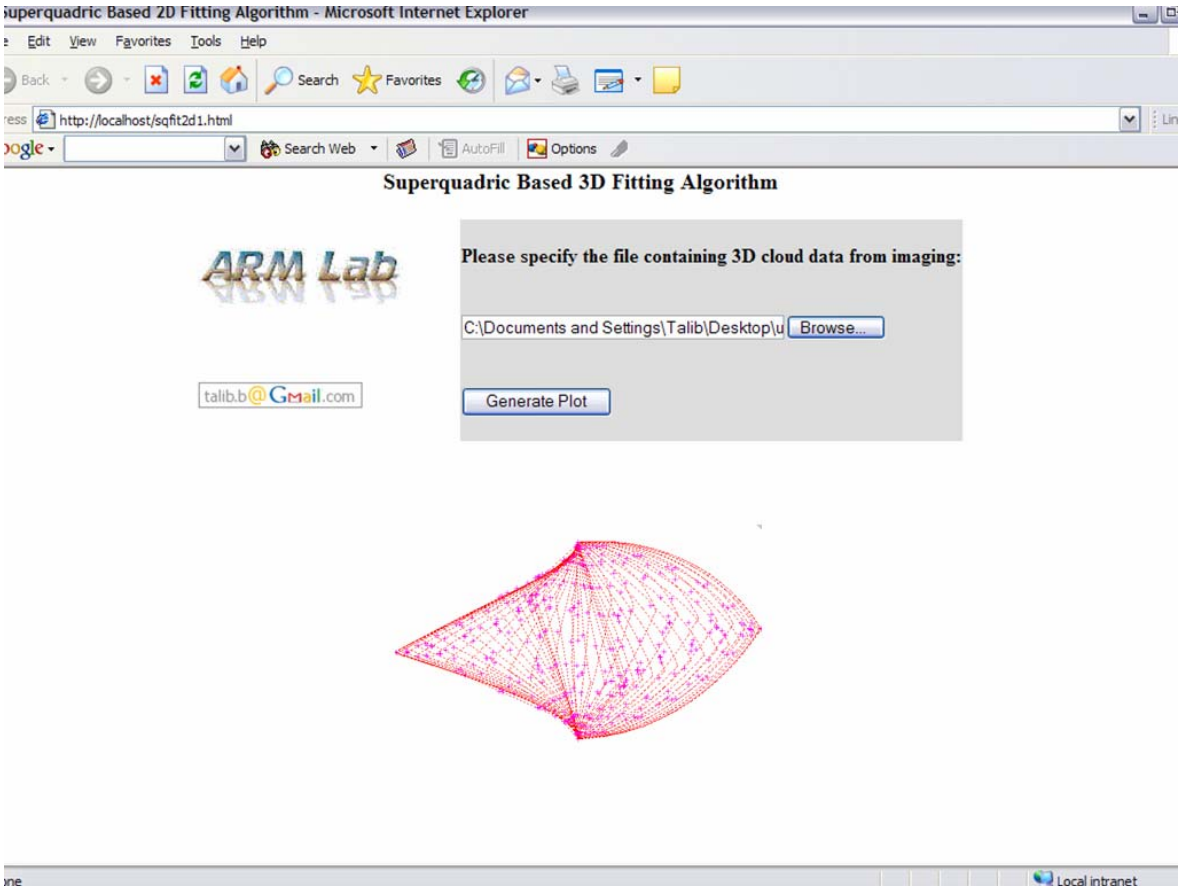


Figure 0-8 Web Based interface for 3D shape recovery using extended superquadrics.

6.2.2.5 Limitations

Since the TCP/IP protocol cannot guarantee transfer rates, real-time demands may not be fulfilled. The response times depend considerably on actual utilization of the network. During the development and testing phase some stability problem emerged in connection with the server. Especially in servers this behavior is not tolerable. A serious disadvantage of the MATLAB Web Server is that, although it is possible to program 3D animations in MPEG- or AVI-format, the response time of applications is too long (1 min. or more). This is impractical considering current Internet conditions. This means there isn't any way to show processes in their extensions in time

or space, only the end-result of a process. This narrows the value of the simulations. Another limitation of MATLAB Web Server is that it requires considerable hardware, because MATLAB is an interpreter, it needs a high performance CPU and also a lot of RAM.

7 Discussion

As presented in the earlier chapters, several experiments and test runs were performed in both 2D and 3D domain. Our results are very encouraging and show that the methodology established can provide an inexpensive, flexible, and potent method for describing surfaces and shapes of two and three dimensions that deform smoothly on spheres. The underlying assumption also being that the data represents enclosed surfaces and shapes. In this chapter we will discuss some salient features of the representation and the recovery method and hopefully motivate some promising research directions.

7.1 Research Questions Revisited

Which kind of a parametric modeling framework would be most suitable for rapid, easy, accurate and computationally inexpensive shape modeling and conversion to volumetric solid model from a dense sampling of the surface?

Our proposed reconstruction approach is flexible enough to incorporate both smooth and sharp features of an asymmetric object while enjoying the benefits of a low end parameterization. The process is fully automatic and is guaranteed to be continuous and within acceptable error when the sampling requirements are satisfied. The resulting model is suitable for CAD modeling and analysis operations owing to the underlying superquadric representation used which is one of the most powerful and compact ways of representing shapes. These parameterized models offer great compactness and broad coverage. However owing to low parameterization the topological flexibility is limited.

How can we leverage the same framework to additionally parametrically explore multi-resolution hierarchical indexing, storage, searching, reconstruction and retrieval?

Parametric formulation of a deformable model should not only yield an accurate description of the object, but it should also provide quantitative information about the object in an intuitive, convenient form. That is, the model parameters should be useful for operations such as measuring, matching, modification, and higher-level analysis or geometric reasoning. This "parameter descriptiveness" criterion may be achieved in a post processing step by adapting or optimizing the parameterization to more efficiently or more descriptively match the data.

However, in our shape representation the descriptive parameterization is directly incorporated into the model formulation. Generally speaking, shape descriptors should be: easy to compute, compact, easily indexable, discriminating of shape differences at many scales, insensitive to noise and possess invariance such as to Euclidean transformations, object representation, tessellation, or genus. The variable coefficients of the continuous exponents used in our representation (spline- based extended superquadrics) offer a compact parameter space. The computationally-tractable the fitting process offers a systematic way to extract and store characteristic signatures of shape in hierarchically-consistent manner, while maintaining the desired level of accuracy. Comparisons between shapes reduces to simply computing the Euclidean distance between the descriptors, retrieving the K best matches for a 3D query model becomes the equivalent of solving the K nearest-neighbors problem in a the low-dimensional parameter space (as opposed to the high-dimensional data space).

7.2 Future Work

Broadly there are two classes of algorithms for shape recovery, Interactive (semi-automatic) algorithms and fully automatic algorithms. Our algorithm depicts an automatic recovery of shapes in an iteratively improving manner which high is a desirable; no matter how difficult it is. This is because it can potentially increase the accuracy, consistency, and reproducibility of the analysis. To achieve this functionality we had to simplify the procedure at some points and this has of course lead to increase in computational time as the algorithm searches for possible number of control points from the beginning each time. Further development would be to utilize more intelligent precursor steps to improve convergence speed of the algorithm, at the same time maintaining it as an automated process.

The representation offers compactness and large geometric coverage for the same, however lacks flexibility while representing highly irregular surfaces. For example objects with a simple, fixed topology and without significant protrusions, these parameterized models coupled with local (spline) and/or global deformations schemes provide a good compactness-descriptiveness tradeoff. On the other hand, modeling of complex, multipart objects such as arterial or bronchial "tree" structures, or topologically complex structures such as vertebrae would require a powerful segmentation and recovery routine. In course of this thesis we have presented a brief introduction to a possible segmentation scheme and should provide a framework for development of a more intuitive and robust scheme.

Our deformable models are capable of representing a broad range of shapes and be useful in a wide array of general medical applications. Alternatively, highly specific, "hand-crafted" or constrained deformable models appear to be useful in applications. Certainly attempts to completely automate the processing of medical images would require a high degree of

application and model specificity. A promising direction for future study appears to be techniques for learning "tailored" models from simple general purpose models.

7.3 Conclusion

The increasingly important role of medical imaging in the diagnosis, treatment and analysis has opened an array of challenging problems centered on the computation of accurate geometric models of anatomic structures or naturally occurring objects from medical images. Such deformable models offer an attractive approach to tackling such problems, because these models are able to represent the broad shape variability of structures. Deformable models overcome many of the limitations of traditional low-level image processing techniques, by providing compact and analytical representations of object shape. The continued development and refinement of these models should remain an important area of research into the foreseeable future.

.

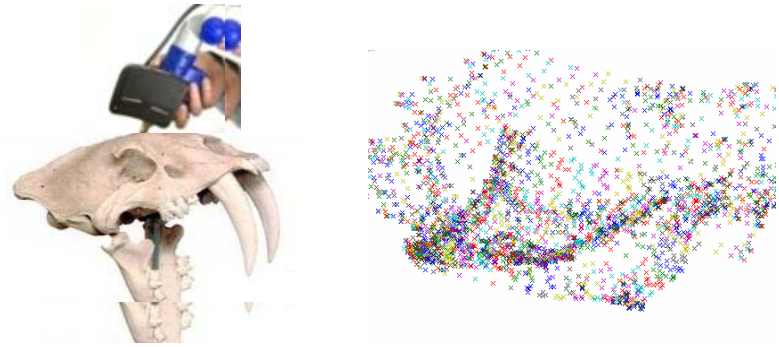
Appendix A

A.1 Touch Probes

Touch Probes have a measurement tip attached via several lymbs and joints. The location of the tip relative to the base can be calculated by knowing the lymb lengths and the joint angles.

The configuration of the number of joints and lymbs will determine the reach and the ability of the touch probe. Touch probes can range from small desktop models capable of measuring 50" sphere up to large industrial models that can measure up to a 12' sphere. They typically have up to seven degrees of freedom. Optical encoders are used at each joint to determine the angle.

An advantage of touch probes is that they are intuitive to use. You move the tip to the location you want to measure, and tell the computer to acquire a point. Touch probes can reach behind and into objects. The disadvantage is that it can time-consuming to measure a lot of points. If the tip used is not a point, then offset calculations are required. Joint and angles are measured by electronic or optical sensors and these measures are combined and transformed into x y z coordinates. Automatic measurement machines are also widely used but mostly in the mechanical manual manufacturing industry for example for quality control.



(a) (b)

Figure 7-1 Use of a probe (a) to obtain the corresponding cloud data (b)

A.2 Laser Sensors

Laser Tracking Interferometers are portable Coordinate Measurement Machines (CMM). These devices can accurately measure large objects from one meter to hundreds of meters in size. A "Laser Tracker" uses a laser interferometer to measure distance and angular encoders to determine the orientation of the center of a spherical mirror retro-reflector (SMR) with respect to the tracker. Combining this information gives you a position. The laser beam can not be broken as the SMR is moved to the measurement location. The device can be accurate to +/- 0.001" when operating within several yards of the tracker. The tracker software must help manage offsets because of diameter of the SMR.

A.3 Profile Scanners

Profile scanners use a technique similar to what television station weather people use to "stand" in front of the weather map. In reality the weather person is standing in front of a colored background (typically blue or green). A special piece of video equipment subtracts the background color out of the video.

The profile scanners typically place a computer controlled turn-table in front of a colored background. The object to be scanned is placed on the turn-table. At every so many degrees, a video camera captures an image of the object. The scanner calculates the profile of the object by looking for the transition of the background color to something else. Using the (i, j) coordinates of the image profile and the turn-table rotation, 3D coordinates can be calculated. The non-background colors are used to create a texture map of the object.

While profile scanners tend to be inexpensive, they have a major drawback. They can not capture concave parts of the object.

A.4 Total Stations and Theodolites

Total Stations and Theodolites can be used to measure discreet points. They are made of several components. First is a mechanism to determine the azimuth of the point of interest compared to the device. Second is a mechanism to determine the elevation of the point of interest compared to the device. The third mechanism determines the distance to the point interest.

A.5 Structured Light

Structured light scanners typically project a known light pattern on to the 3D surface of the object to be modeled. The light pattern is distorted by the surface relief. Different light patterns can be used (i.e. grids, circles, sinusoidal). Triangulation is used to calculate the 3D points.

Structured light scanners capture complete surfaces from a particular point of view. The data from multiple points of view can be combined to create a complete 3D cloud model. The light source is usually an ordinary halogen white light, so there are no safety concerns as with laser scanners.

A.6 Inference

Some things to notice here are that acquired data immensely depends on the scanning technology used and the method of their use. Handheld scanners can be used for extraction of fewer but more accurate points whilst laser scanner is good for digitizing objects with a simple geometry.

It is also important to understand that this is list of scanning technologies is far from comprehensive. It is just meant to give an understanding of some of the common technologies seen in literature. Some other useful references besides the manufacturer websites are [5, 49, 50] and the references with in.

Bibliography

1. Loncaric, S., *A Survey of Shape Analysis Techniques.* Pattern Recognition, 1998. **31**(8): p. 983-1001.

2. Alt, H., and Guibas, L.J., *Discrete Geometric Shapes: Matching, Interpolation, and Approximation: A Survey*, in *Technical Report*. 1996, Institute of Computer Science, Freie Universität: Berlin.

3. Metaxas, D.N. *Physics-Based Deformable Models: Application to Computer Vision, Graphics and Medical Imaging*. in *Kluwer Academic*. 1997. Norwell, MA.

4. Bolle, R.M. and B.C. Vemuri, *On Three-Dimensional Surface Reconstruction Methods.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 1991. **13**(1): p. 1-13.

5. Bernardini, F., et al., *Automatic Reconstruction of 3D CAD Models from Digital Scans*, in *Technical Report CSD-97-012*. 1997, Department of Computer Sciences, Purdue University.

6. Ning, P. and J. Bloomenthal, *An evaluation of implicit surface tilers.* IEEE Computer Graphics and Applications, 1993. **13**(6): p. 33-41.

7. Lorensen, W.E. and H.E. Cline, *Marching cubes: a high resolution 3D surface construction algorithM.* Computer Graphics (SIGGRAPH 87), 1987. **21**(4): p. 163-169.

8. Tohka, J., *Global optimization-based deformable meshes for surface extraction from medical images*, in *Digital Media Institute/ Signal Processing*. 2003, Tampere Univeristy of Technology: Tampere.

9. Terzopoulos, D., A. Survey, and T. McInerney, *Deformable Models in Medical Image Analysis.* Medical Image Analysis, 1996. **1**(2): p. 91-108.

10. Lai, K.F. and R.T. Chin, *Deformable contours - modelling and extraction.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 1995. **17**(11): p. 1084-1090.

11. *Solarius Development Inc*, in *http://www.solarius-inc.com/html/laserscan.html*.

12. *Axila Co.* (http://www.axila.com/).

13. *Apisensor.* **http://www.apisensor.com/**.

14. Barr, A.H., *Superquadrics and Angle-Preserving Transformations.* IEEE Computer Graphics and Applications, 1981. **1**: p. 11-22.

15. Jaklic, A. and F. Solina, *Moments of superellipsoids and their application to range image registration.* Systems, Man and Cybernetics, Part B, IEEE Transactions on, 2003. **33**(4): p. 648-657.

16. Liu, W. and B. Yuan. *Superquadric based hierarchical reconstruction for virtualizing free form objects from 3D data*. in *Signal Processing Proceedings, 2000. WCCC-ICSP 2000. 5th International Conference on*. 2000.

17. Zha, H., T. Hoshide, and T. Hasegawa. *A recursive fitting-and-splitting algorithm for 3-D object modeling using superquadrics*. in *Pattern Recognition, 1998. Proceedings. Fourteenth International Conference on*. 1998.

18. Zhou, L. and C. Kambhamettu. *Extending superquadrics with exponent functions: modeling and reconstruction*. in *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.* 1999.

19. Barr, A.H., *Global and local deformations of solid primitives.* Computer Graphics, 1984. **18**(3): p. 21-30.

20. Boult and Gross. *Recovery of superquadrics from depth information.* in *Workshop on Spatial Reasoning and Multi-Sensor Fusion*. 1987.

21. Gross, A. and T. Boult. *Error of fit measures for recovering parametric solids*. in *In: Internat. Conf. Computer Vision,*. 1988.

22. Gupta, A., L. Bogoni, and R. Bajcsy. *Quantitative and qualitative measures for the evaluation of the superquadric models*. in *Interpretation of 3D Scenes, 1989. Proceedings., Workshop on*. 1989.

23. Solina, F. and R. Bajcsy, *Recovery of parametric models from range images: the case for superquadrics with global deformations.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, 1990. **12**(2): p. 131-147.

24. Terzopoulos, D. and D. Metaxas, *Dynamic 3D models with local and global deformations: deformable superquadrics.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, 1991. **13**(7): p. 703-714.

25. Zhang and B. Yuan. *Representation and Reconstruction of 3-D Objects Using Nonlinear Deformable Superquadric Models*. in *IAPR Workshop on Machine Vision Applications*. 1992.

26. Jacklic, A., A. Leonardis, and F. Solina, *Segmentation and recovery of superquadrics.* Kluwer Academic Publisher, Dordrecht). 2000.

27. Pentland, A., *Segmentation and recovery of superquadrics.* Kluwer Academic Publisher,Dordrecht, 2000: p. 14-16.

28. Gielis, J., *Superquadrics and Generalized Minkowski Metrics*, in *GENIAAL Techincal Report*. 2002.

29. Metaxas., D. and D. DeCarlo., *Shape evolution with structural and topological changes using blending. IEEE Transactions.* Pattern Recognition and Machine Intelligence, 1998. **20**(11): p. 1186-1205.

30. Leonardis, A., A. Jaklic, and F. Solina, *Superquadrics for segmenting and modeling range data.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, 1997. **19**(11): p. 1289-1295.

31. H. Löfelman, E.G., *Parameterizing superquadrics.* Winter School of Computer Graphics and Visualization (Univ. West Bohemia) WSCG'95, 1994(1): p. 162-172.

32. Bardinet, E., L.D. Cohen, and N. Ayache. *Fitting 3-D data using superquadrics and free-form deformations*. in *Pattern Recognition, 1994. Vol. 1 - Conference A: Computer Vision & Image Processing., Proceedings of the 12th IAPR International Conference on*. 1994.

33. Hanson., A.J., *Hyperquadrics - deformable shapes with complex polyhedral bounds.* Computer Vision, Graphics and Image Processing, 1988. **44**: p. 191-210.

34. Blanc, C. and C. Schlick, *Ratioquadrics: an alternative method for superquadrics.* The Visual Computer, 1996. **12**(8): p. 420-428.

35. Akleman., E. *Constructing convex and star shapes with yontsal functions*. in *International Symposium on Computer and Information Sciences*. 1993.

36. Zhou, L. and C. Kambhamettu. *Representing and recognizing complete set of geons using extended superquadrics*. in *Pattern Recognition, 2002. Proceedings. 16th International Conference on*. 2002.

37. Solina, F., *Superuqadrics and their properties.*

38. Gridgeman, N.T., *Lamé ovals.* The Mathematical Gazette, 1970. **54**: p. 31.

39. Van Dop, E. and P.L. Regitien. *Fitting undeformed superquadrics to range data: Improving model recovery and classification*. in *Internat. Conf. Computer Vision and Pattern Recognition*. 1998.

40. Zhang, Y., *Experimental comparison of superquadric fitting objective functions.* Pattern Recognition Letters, 2003. **24**: p. 2185–2193.

41. Marquardt, D., *An Algorithm for Least-Squares Estimation of Nonlinear Parameters.* SIAM J. Appl. Math., 1963(11): p. 431-441.

42. Holland, J.H., *Adaptation in Natural and Artificial Systems*. 1975: University of Michigan Press.

43. R.J.Prokop and A.P.Reeves, *A survey of moment based techniques for unoccluded object representation and recogniti.* Computer Vision Graph, Imae Processing, 1992. **54**(5): p. 438-460.

44. Carlbom, I., D. Terzopoulos, and K. Harris, *Computer-assisted registration, segmentation, and 3D reconstruction from images f neuronal tissue sections.* IEEE Trans. on Medical Imaging, 1994. **13**(2): p. 351–362.

45. Hongbin Zha, T.H., Tsutomu Hasegawa, *Superquadric based object modelin by an iterative segmentation and recovery algorithm.* Inteligent Robot and computer vision, 1997: p. 518-529.

46. Horikoshi, T. and S. Suzuki. *3D parts decomposition from sparse range data using information criterion*. in *Computer Vision and Pattern Recognition, 1993. Proceedings CVPR '93., 1993 IEEE Computer Society Conference on*. 1993.

47. Zhang, Y., A. Koschan, and M.A. Abidi, *Superquadric representation of automotive parts*

applying part decomposition. Journal of Electronic Imaging, 2004. **13**(3): p. 411– 417.

48. Mathworks, in *www.mathworks.com*.

49. Klette, R., K. Schluns, and A. Koschan, *Computer Vision : Three-Dimensional Data from Images*.

50. Simple3D, *www.simple3d.com*.