Diss. ETH No. 10979

Description and analysis of 3-D shapes by parametrization of closed surfaces

A dissertation submitted to the
SWISS FEDERAL INSTITUE OF TECHNOLOGY ZURICH

for the degree of
Doctor of Technical Sciences

presented by
Christian Michael Brechbühler-Miškuv
Dipl. Inf. Ing. ETH
born 13th of March, 1964
citizen of Huttwil, Bern

accepted on the recommendation of
Prof. Dr. Guido Gerig, examiner
Dr. Nicholas Ayache, directeur de recherche, co-examiner

1995

Contents

1 Introduction **5**
- 1.1 3D shape 5
- 1.2 Alternative approaches 7
- 1.3 Surface based approaches 7

2 Manifolds **9**
- 2.1 A General Model for Characterizing Manifolds 9
  - 2.1.1 The basic idea 9
  - 2.1.2 Definitions and Notation 9
  - 2.1.3 Various concrete cases 10
- 2.2 Series Expansion 16
- 2.3 Planar curves 17
  - 2.3.1 Splines 18
  - 2.3.2 Bézier Curves 18
  - 2.3.3 Elliptic Fourier Descriptors 19
- 2.4 Space Curves 28

3 Surfaces: Representation and Parametrization **29**
- 3.1 Open and closed Surfaces 29
- 3.2 Closed surfaces in 3D 30
- 3.3 The surface data structure 31
  - 3.3.1 An example of the basic surface data structure . . 33
- 3.4 Parametrization of closed surfaces 36
- 3.5 Parametrization of surface patches 40

4 Optimization **47**
- 4.1 Initial parametrization 47
  - 4.1.1 Co-latitude from diffusion. 48

4.1.2 Determining co-latitude in the example 49
4.1.3 Longitude from diffusion. 51
4.1.4 Determining longitude in the example 52
4.2 Optimization methods 53
4.2.1 Notation 55
4.2.2 A Lagrange-Newton algorithm 55
4.2.3 Constrained optimization in orthogonal spaces . . 57
4.3 Examples and results 68

5 The Spherical Harmonic Descriptor 71
5.1 Spherical harmonics 71
5.1.1 Definition 71
5.1.2 Expressing surface shape 75
5.1.3 Variability of spherical harmonic surfaces 75
5.2 Harmonic shape descriptors 79
5.2.1 Integration over the sphere 79
5.2.2 Invariant descriptors 81
5.2.3 Importance of uniform parametrization 89
5.3 Comparison and Recognition of Shapes 90
5.3.1 Symmetries of the Ellipsoid 92
5.3.2 Limitations 94
5.3.3 Experimental results 94

6 Conclusions 103

7 Appendix: Polar Coordinates for Ω_2 and Ω_3 109

Abstract

The general problem addressed in this thesis is the description of the *form* of an m-dimensional manifold embedded in $\mathbb{R}^n$. A planar curve, e.g., is a one-dimensional manifold in two-dimensional space, hence $m = 1, n = 2$. And $m = 1, n = 3$ stands for a space curve. But the case of particular interest is $m = 2, n = 3$, i.e. a surface in 3D. We achieve the description in two steps. First a continuous one-to-one mapping between the manifold and an appropriate m-dimensional parameter space U is constructed. This step is called *parametrization*; it defines n functions $x_i : U \hookrightarrow \mathbb{R}$, or one vectorial function $\underline{x} : U \hookrightarrow \mathbb{R}^n$. Then each of the functions $x_i(\underline{u})$ is expanded into a series of a fixed set of basis functions defined on U. The coefficients in the series expansions form a descriptor of the shape.

The surface of a simply connected 3D object is a closed surface homeomorphic to the sphere. The unit sphere suggests itself as the parameter space, and spherical harmonics are suitable as basis functions. The parametrization establishes true area ratios in parameter space and minimizes distortions. Standardization yields generic shape descriptors which are invariant to pose and size of the object. Due to the nature of the basis functions, spherical harmonic descriptors capture shape in a global way.

The new methods are illustrated with both simple and complex 3-D test objects. Potential applications are recognition, classification and comparison of convoluted surfaces.

Zusammenfassung

Diese Arbeit beleuchtet das allgemeine Problem der Formbeschreibung einer m-dimensionalen Mannigfaltigkeit, die im Raum $\mathbb{R}^n$ eingebettet ist. Für $m = 1, n = 2$ ist dies eine zweidimensionale Kurve, und $m = 1, n = 3$ steht für eine Kurve im Raum. Der interessanteste Fall aber ist $m = 2, n = 3$, d.h. eine Fläche in 3D. Es sind zwei Schritte zu unterscheiden. Zuerst wird eine stetige eineindeutige Abbildung zwischen der Mannigfaltigkeit und einem geeigneten m-dimensionalen Parameterraum konstruiert. (Es ist nicht immer möglich, U auf einen Teil von $\mathbb{R}^m$ abzubilden.) Dieses definiert n skalare Funktionen $x_i : U \hookrightarrow \mathbb{R}$ oder eine vektorielle $\underline{x} : U \hookrightarrow \mathbb{R}^n$. Danach wird jede der Funktionen $x_i(\underline{u})$ in eine Reihe entwickelt, wobei der selbe Satz von auf U definierten Basisfunktionen zur Anwendung kommt. Die Koeffizienten der Reihenentwicklung bilden einen Deskriptor der Form.

Die Oberfläche eines einfach zusammenhängenden 3D Objektes ist eine geschlossene, zur Kugel homöomorphe Fläche. Die Einheitskugel bietet sich als Parameterraum an, und die sphärischen harmonischen Funktionen bilden eine geeignete Basis. Eine Standardisierung der Koeffizienten ergibt invariante Formdeskriptoren, welche unabhängig von der Grösse und Lage des Objektes sind. Aufgrund der Natur der Basisfunktionen erfassen sphärische harmonische Deskriptoren Form auf eine globale Art.

Sowohl einfache 3D Testobjekte als auch Beispiele komplizierterer Formen aus der Medizin illustrieren die neuen Methoden. Mögliche Anwendungen liegen in der modellbasierten Erkennung, der Klassifikation und im quantitativen Formvergleich auch verschlungener Oberflächen.

Chapter 1

Introduction

1.1 3D shape

With the proliferation of high quality volumetric image data, especially for the medical community, and new segmentation methods for multidimensional image data, segmented 3-D objects become available and are ready for structural analysis. Most often, volumetric objects are represented by a binary voxel representation or by a triangulation of the surface. Although these representations allow a 3-D rendering for visually capturing the object properties, both lack descriptive power as they are based on huge lists of voxels or surface elements. Characterizing and understanding shape properties, however, requires a representation which captures global and local shape features with a small number of parameters. Such a concise description could be useful for a comparison of various objects, for finding dissimilarities, for matching objects to predefined models and for an efficient reconstruction and manipulation of objects.

A shape descriptor must be general enough to handle very different shapes, but should be capable of representing accurately global as well as local features of objects. Shape analysis favors object-centered volume or surface descriptions, e.g. using polynomials, triangulation meshes, generalized cylinders, medial manifolds or spherical harmonics. Object position, orientation, and size should be separated from the object-centered shape description. Hierarchy can organize the description into varying levels of detail, from coarse to fine.

The surface based approach gets most attention in this paper. Some shape description methods based on mesh-like surface models rely on a robust and reproducible surface parametrization in a two-coordinate space. While tracking a contour in 2-D images is easily done, the extension to higher dimensions is non-trivial and requires the development of new concepts. Thus far, representation methods for mapping an object surface onto a sphere have been limited to represent only star-shaped or convex objects, as they start from an initial radial surface function $r(\theta, \phi)$ [1] [22]. Staib and Duncan [37] discuss the use of a parameter space with torus topology, which can be deformed into a tube by squashing the torus cross-section to a thin ribbon. Closed surfaces are obtained by considering tubes whose ends close up to a point. This approach illustrates clearly some principal difficulties which can also be found in other parametrization techniques.

- The idea of warping a torus to a closed surface poses the problem that the *parameters have different meanings*. One parameter defines a kind of spine along which cross-sections are stacked up. The choice of the end-points of this spine decisively determines the solution and even determines whether an object can be parametrized or not.

- Squeezing a circle to a line (as done with the tubular torus cross-section) results in a *nonhomogeneous distribution of parameters* on the object surface. Although continuous, the representation of a line by harmonic functions results in a clustering of parameters near the turning points of the tracing direction. Furthermore, closing a cylinder at both open ends causes a further distortion of the parameter net.

- Warping a torus to a tube and finally to a “closed” surface shows that the parametrization does *not* result in a *one-to-one mapping* of surface points to parameters. The walls of the tube are traced up and down in order to avoid discontinuities at the open edges, visiting each surface point twice.

Our new approach tries to overcome these limitations. We present a new method that allows a uniform mapping of an object surface into a two-coordinate space with spherical topology. Our aim is the parametrization of arbitrarily shaped objects which are simply connected and contain intrusions and protrusions. As a mapping of convoluted surface structures onto surfaces of minimal curvature introduces distortions, optimization of the distribution of nodes in parameter space becomes necessary. This problem is solved by a nonlinear optimization.

Parametrized surfaces can be expanded into spherical harmonics, hierarchically describing global and local shape properties by spatial frequency constituents [7]. A new method for the generation of descriptors which are invariant to translation, rotation and scaling is developed. Invariance is crucial for a comparative analysis of different structures.

1.2 Alternative approaches

Approaches to shape characterization can be divided coarsely into two categories. They are characterized by concentrating either on the space the object takes up, or on its border. The two viewpoints can be termed volume or region based on one hand, and surface or contour based on the other hand. They are dual in the sense that the surface is the boundary of the volume, and the volume is the fill of the surface. Surface based techniques are discussed in more detail below.

Skeletons or medial manifolds are important representants of the region or volume based shape model. They go back to Blum's medial axis [6], and they are closely related to the Voronoi Diagram [28]. The skeleton of a 2D (3D) object is the set of all centers of disks (spheres) that lie completely inside the objet, but touch the border in at least 2 (2 or 3) points. It is a subset of all local symmetry axes (surfaces) of the object. In [27], the medial manifold is developed into a hierarchically structured descriptor for 2D shapes.

Moments form another class of region – or volume – based descriptors [21, 38]. Their extension to 3D has been investigated [32, 25, 11], although invariances of the descriptors are somewhat more intricate than in 2D.

1.3 Surface based approaches

In rendering applications, triangulated surfaces (or, more generally, polygon meshes) are a popular representation. While they fit the purpose of computer graphics well, they do not describe the object. They lack the conciseness of an object description, and they do not help the task of comparing two objects.

Superquadrics can describe a certain class of objects very concisely. The surface defined by

$$\begin{aligned}
\underline{x}(\theta, \phi) &= \left(\begin{array}{ll} a_0 \sin^{\epsilon_1}(\theta) & \cos^{\epsilon_2}(phi) \\ a_1 \sin^{\epsilon_1}(\theta) & \sin^{\epsilon_2}(phi) \\ a_2 \cos^{\epsilon_1}(\theta) & \end{array} \right) \qquad (1.1) \\
0 \leq \;& \theta \leq \pi \\
0 \leq \;& \phi \leq 2\pi
\end{aligned}$$

is the superquadric with size parameters a_0, a_1 and a_2 and squareness parameters ϵ_1 and ϵ_2. One of the benefits of superqadrics is that there is a straight-forward inside-outside function.

$$F(\underline{x}) = \left(\left(\frac{x_0}{a_0} \right)^{\frac{2}{\epsilon_2}} + \left(\frac{x_1}{a_1} \right)^{\frac{2}{\epsilon_2}} \right)^{\frac{\epsilon_2}{\epsilon_1}} + \left(\frac{x_2}{a_2} \right)^{\frac{2}{\epsilon_1}} \overset{?}{\leq} 1 \tag{1.2}$$

Inequality (1.2) holds for all points inside the object, with equality on the surface. For $\epsilon_1 = \epsilon_2 = 1$ the surface is an ellipsoid. In all other cases the surface reflects the strong alignment of the definition with the coordinate axes. The cross section will always show a four ray symmetry, which may be stretched to a two ray symmetry if $a_0 \neq a_1$. The x_2 axis is treated differently than the axes x_0 and x_1. Objects with sharp edges can be generated by appropriate selection of the exponents, but the user cannot control where the edges appear. The scheme lacks generality.

Global deformations of a superquadric [36], like tapering, bending, and twisting greatly enhance the set of representable shapes. But each of these deformations has its own mathematical formulation. For every class of shapes that we wish to represent, a new deformation must be invented. The representation of the shape gets complicated and heterogenic; the beauty of a concise and elegant mathematical definition (of (1.1) or (1.2)) is lost.

Thirion proposes a new concept for the description of smooth surfaces [39]. Two principal curvatures exist in any point of the surface, and the concept relies on their local extrema. Lines where one curvatuvature is extremal link points where both curvatures are extremal. The *extremal mesh* is the graph of these extremal lines and points.

The topological dual of a triangulation of the surface is the *simplex mesh* [9]. It typically consists of mostly hexagons and a few pentagons, quadrilaterals or triangles. Unlike deformable surfaces defined on regular grids, it can have its topology altered locally.

Chapter 2

Manifolds

2.1 A General Model for Characterizing Manifolds

2.1.1 The basic idea

The objects of our interest can be characterized as m-dimensional submanifolds in an n dimensional object space. They are defined as bicontinuous one-to-one **maps** from an m-dimensional parameter space to the object space $\mathbb{R}^n$. Typically, $1 \leq m < n \leq 3$.

2.1.2 Definitions and Notation

$\mathbb{R} \equiv \mathbb{R}^1$	the set of real numbers.
$\mathbb{R}^k$	k dimensional Euclidean space, or the set of all ordered k-tuples of real numbers.
X	object space
$\underline{x}$	point or vector in object space
x_0, x_1, x_2	object space coordinates (Cartesian)
n	dimension number of object space
U	parameter space
$\underline{u}$	point or vector in parameter space
u_0, u_1, u_2	Cartesian parameter space coordinates
θ, ϕ	polar parameter space coordinates
m	dimension number of parameter space
Ω_{m+1}	the set of unit vectors (the “sphere”) in $\mathbb{R}^{m+1}$.

2.1.3 Various concrete cases

A planar curve is described in parametric form as a vector valued function $\underline{x}(u_0)$, i.e. by two scalar functions $x_0(u_0)$ and $x_1(u_0)$, where u_0 is the parameter.

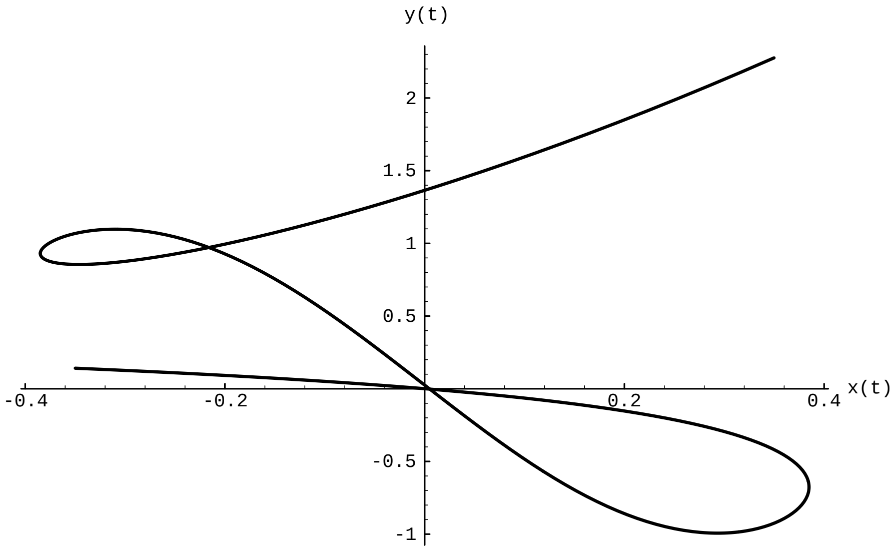


Figure 2.1: An open planar curve, defined as two scalar functions of one real parameter.

Example 1 Figure 2.1 presents an example of an open planar curve. The curve is defined as the two scalar functions $x_0(u_0) = u_0^3 - u_0$ and $x_1(u_0) = \frac{e^{4\,u_0}}{40} + \sin(\frac{5\,\pi\,u_0}{\sqrt{9+16u_0^2}})$. The parameter space is the real interval $U = [-\frac{8}{7}, \frac{8}{7}]$. An alternative interpretation is a single vector-valued function, mapping $U \hookrightarrow \mathbb{R}^2$. The plots of the individual functions $x_0(u_0)$ and $x_1(u_0)$ are combined in Figure 2.2.

Similarly, a space curve $\underline{x}(u_0)$ is given by three functions $x_0(u_0)$, $x_1(u_0)$, $x_2(u_0)$.

Example 2 An open space curve, i.e. a function $\mathbb{R} \hookrightarrow \mathbb{R}^3$, appears in Figure 2.3. The function is $\underline{x}(u_0) = (0.46 + u_0 - u_0^3, -0.1 - u_0^2, 0.75 + u_0 - \frac{u_0^3}{3})^T$, and u_0 ranges over the interval $U = [-1.1, 1.1]$.

A surface, which needs two parameters (u_0, u_1), is defined through three functions $x_0(u_0, u_1)$, $x_1(u_0, u_1)$, $x_2(u_0, u_1)$.

Example 3 In Figure 2.4 we see a surface patch which is given by the vector valued function

$$\underline{x}(\underline{u}) = \left(u_1\,\left(2 + \frac{2\,\sin(u_0^2)}{u_0\,\sqrt{1+4\,u_1^2}}\right), -2\,u_1^2 + \frac{\sin(u_0^2)}{u_0\,\sqrt{1+4\,u_1^2}}, \frac{\cos(u_0^2)}{u_0}\right).$$

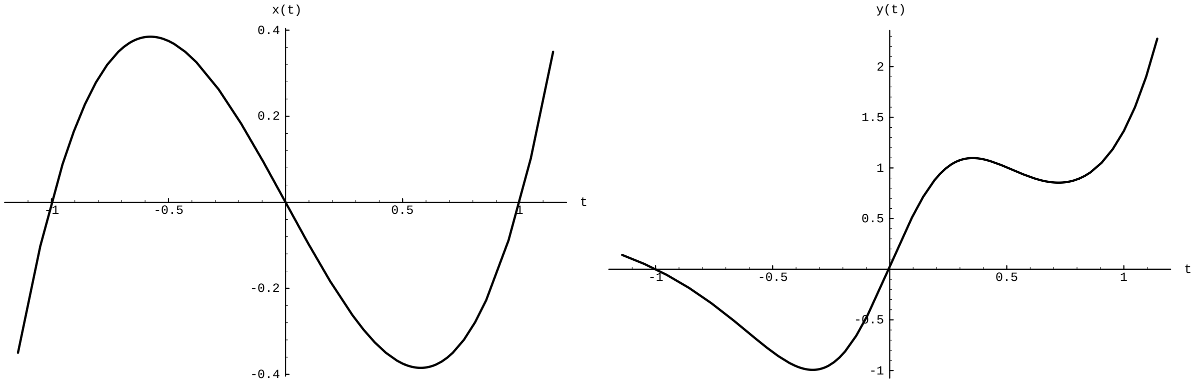


Figure 2.2: Plot of the component functions $x_0(u_0)$ and $x_1(u_0)$ of the Figure 2.1.

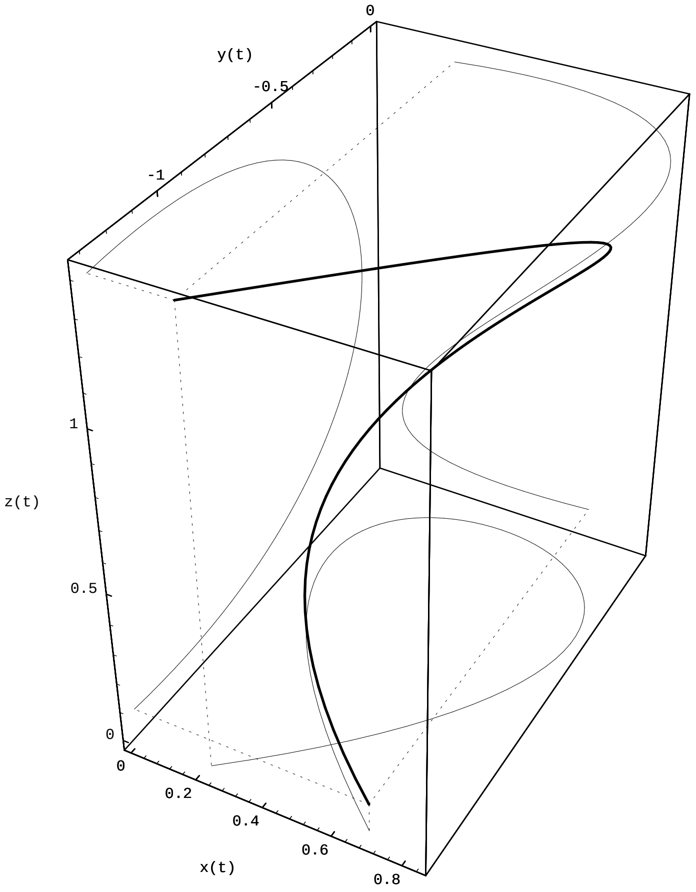


Figure 2.3: Plot of a vector function $\underline{x}(u_0)$ as a space curve. For clarity, the projections of the space curve onto the planes $x_0 = 0$, $x_1 = 0$ and $x_2 = 0$ are added in thin lines. The fine dashed lines indicate how the endpoints of the curve project into these planes.

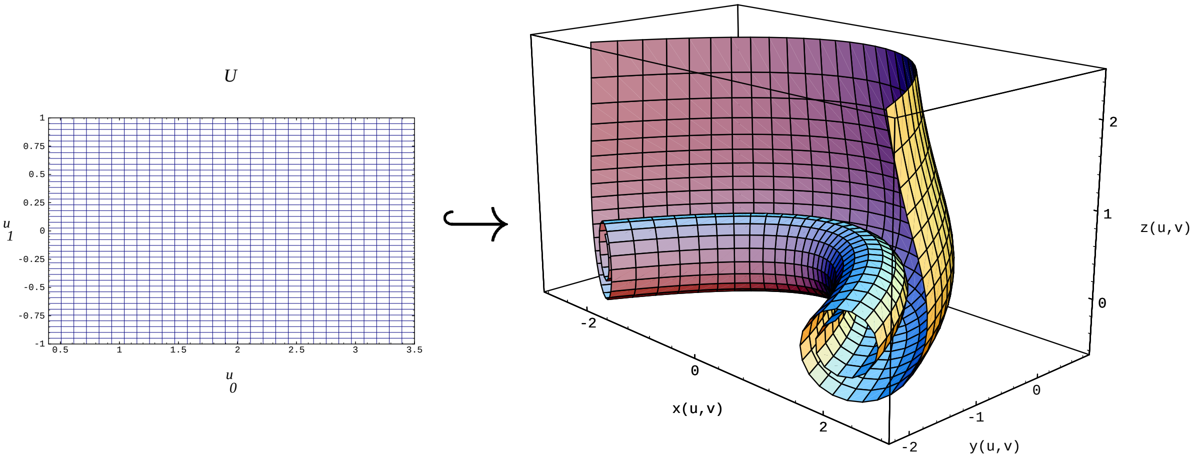


Figure 2.4: A surface patch, defined as a functional mapping from the rectangular parameter region U (left) to $\mathbb{R}^3$ (right).

(or the three scalar component functions) evaluated on a region of $\mathbb{R}^2$. A rectangle is the simplest region; for the figure I chose $U = [0.4, 3.5] \times [-1, 1] \subset \mathbb{R}^2$. Graphing this function raises the question of how the rectangle should be traversed. The most obvious answer is to vary each of the parameters in constant increments independently. This results in the rectangular grid, that is visible in the illustration.

The functions must be continuous. In most applications there is some restriction on the values that the parameters may take. The set of parameter values are called the *parameter space*, denoted by U. It has m dimensions, where $m = 1$ for lines and $m = 2$ for surfaces. The object lies in $\mathbb{R}^n$, the *object space*, where $n = 2$ for the plane and $n = 3$ for three-space. We combine the parameters in a vector $\underline{u} \in U$ and the object space coordinates x_0, x_1 and maybe x_2 in a vector $\underline{x} \in \mathbb{R}^n$. When $\underline{u}$ runs over U, $\underline{x}$ runs over an m-dimensional manifold embedded in $\mathbb{R}^n$. We can write the general idea of a parametric description of manifolds as the function

$$U \hookrightarrow \mathbb{R}^n : \underline{u} \to \underline{x}(\underline{u}) \,. \tag{2.1}$$

The parameter space should reflect the topology of the given manifold. An interval on the real axis (e.g. $[0, 1]$) is appropriate as the parameter space for an open curve (as in examples 1 and 2). A circle (like Ω_2) is the proper parameter space for a closed curve. Although this parameter space is one-dimensional, there is no continuous one-to-one mapping to an interval on $\mathbb{R}$. Relation (7.3(appendix)) can map it to an interval of length 2π, like $[0, 2\pi)$, but this introduces a discontinuity at $\underline{u} = (1, 0)^T$.

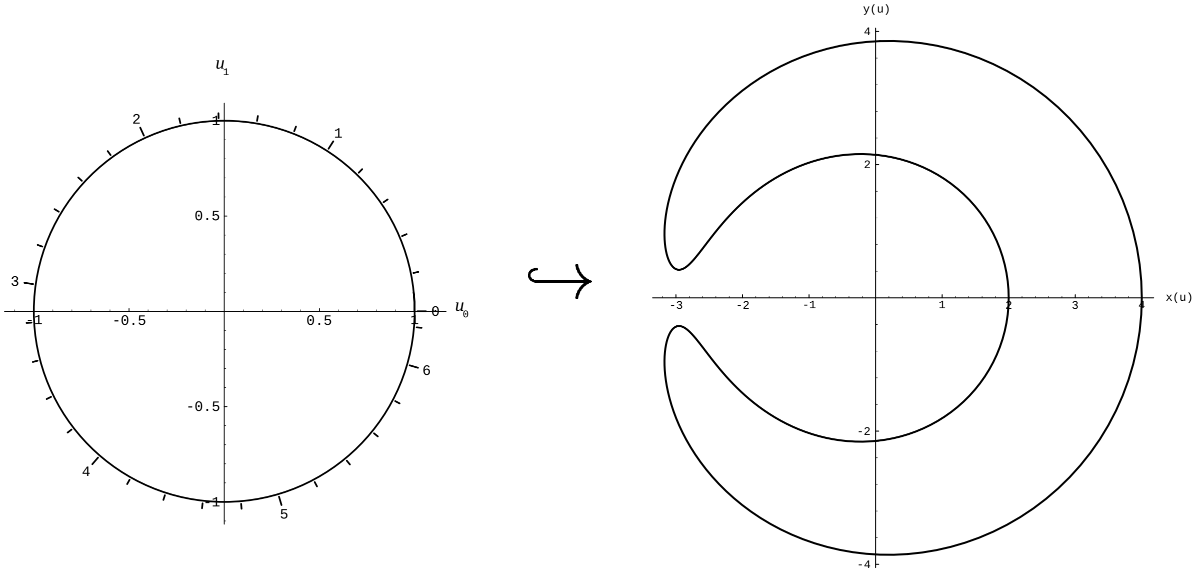


Figure 2.5: Plot of the curve $\underline{x}(\underline{u})$, where $\underline{u} = (u_0, u_1)^T \in U = \Omega_2$ (left).

Example 4 A closed curve, as the one in Figure 2.5, is represented as a function $\Omega_2 \hookrightarrow \mathbb{R}^2$, in this case

$$\underline{x} = ((3 + u_1)\ \cos(3\,u_0), (3 + u_1)\ \sin(3\,u_0))^T\ .$$

But the representation can also take the form of a function $[0, 2\pi) \hookrightarrow \mathbb{R}^2$, where $u' \in U' = [0, 2\pi) \to \underline{u} = (\cos u', \sin u')^T \in \Omega_2$. The resulting equivalent representation is

$$\underline{x}(u') = (3 + \sin u') \begin{pmatrix} \cos\ 3 \cos u' \\ \sin\ 3 \cos u' \end{pmatrix}\ .$$

A function defined on any real interval might work equally well, if continuity up to the needed order (C_0, C_1 or higher) was established artificially between the endpoints of the interval, which coincide logically.

Example 5 Figure 2.6 shows a closed space curve, defined through the function $\underline{u} \to (1.1 + \sin(3\,u_0), -1.1 + \sin(3\,u_1), u_0^2 + 1)^T$.

The case is similar for the closed surface of a simply connected object, which has the topology of Ω_3. Relation (7.4(appendix)) maps the parameter space $\Omega_3 \subset \mathbb{R}^3$ to $[0, \pi] \times [0, 2\pi) \subset \mathbb{R}^2$, introducing a line of discontinuity at $\phi = 0 \equiv 2\pi$ (or $u_1 = 0, u_0 \geq 0$) and two poles at $\theta = 0, \pi$ (or $\underline{u} = (0, 0, \pm 1)^T$). Depending on the context, either representation may be preferable.

Example 6 Figure 2.7 introduces the case that is most important for the work presented in this thesis, a mapping from the sphere Ω_3 to $\mathbb{R}^3$. The function

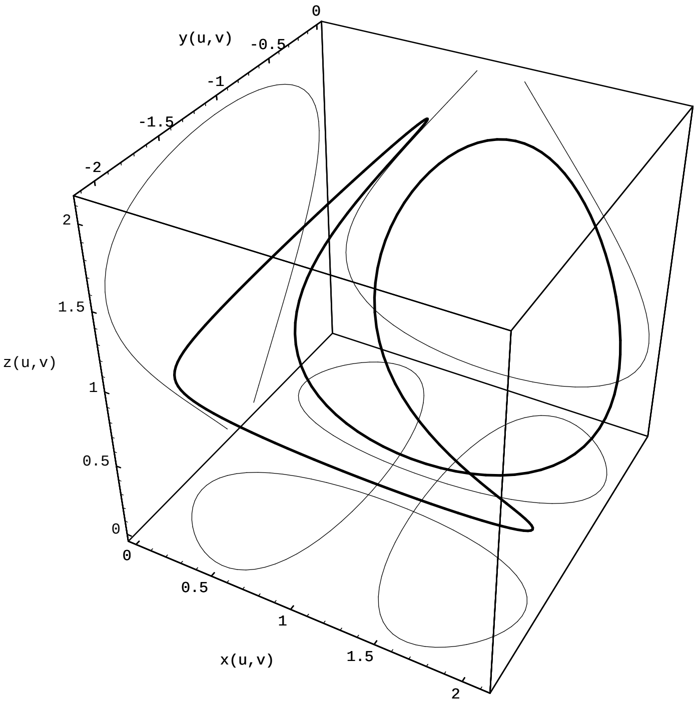


Figure 2.6: This closed space curve is defined by a function $\Omega_2 \hookrightarrow I\!R^3$. The projections of the curve onto the coordinate planes give a better 3D impression.

plotted — $(u_0^3 + \frac{u_0}{2}, u_1^3 + \frac{u_1}{2}, u_2^3 + \frac{u_2}{2})^T$ — might be defined anywhere on $I\!R^3$, but we deliberately restrict its domain to $U = \Omega_3$. Tiling the parameter space is more debatable with the sphere than with a rectangle. There is no obvious optimal tessellation. Polar coordinates (eq. 7.4(appendix)), as used in Fig. 2.7 are just one possibility, and they have the disadvantage that they introduce poles at two arbitrary points on the sphere (and hence on the surface) that are not intrinsically special in any way. A rotated grid in Ω_3 leads to a different subdivision of the surface in $I\!R^3$, as in Fig. 2.8. And a completely different covering of the parameter space with triangles, as shown in Fig. 2.9, brings along the corresponding triangular parcellation of the object surface. The triangulation is based on the icosahedron, where each of the 20 triangles is replaced by 64 small triangles. All three figures plot the same function, which represents the same surface, and is parametrized identically. Just the polygonal subdivision of the parameter space U varies.

For open surface patches, a region of $I\!R^2$ is an appropriate parameter space (cf. example 3). In the case of the closed surface of an object with a handle, which is not simply connected, a toroidal parameter space is needed. As with the sphere, a rectangular part of $I\!R^2$ suffices, with

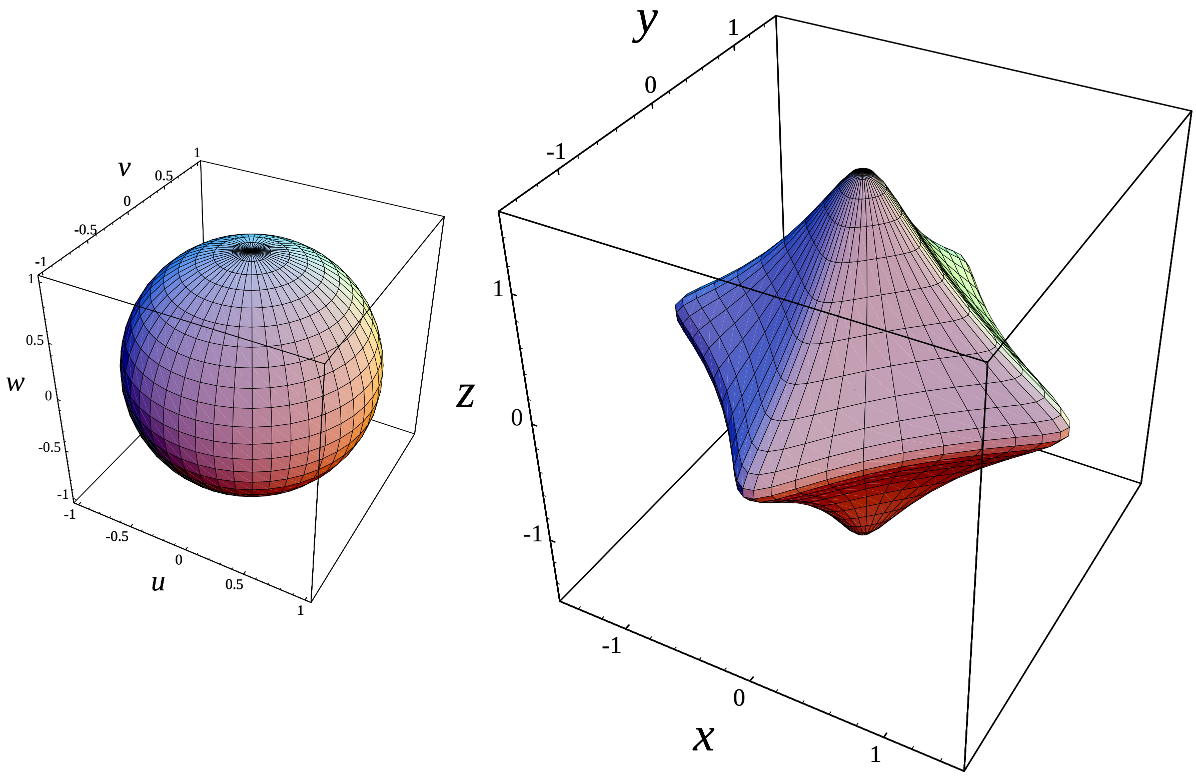


Figure 2.7: *Left:* An arbitrary parcellation of the domain $U = \Omega_3$ of function $U \hookrightarrow \mathbb{R}^3$. *Right:* The function maps the unit sphere to the given surface, which is similar to an octahedron.

implied periodicity at the border: top wraps around to bottom, and left to right. I will not discuss these surfaces in further detail.

There are many possibilities to define the same manifold, which differ in the parametrization of the manifold. A re-parametrization is a different mapping between the manifold and the parameter space. It may or may not bring along a change of the parameter space. For an illustration, assume the left diagram of Fig. 2.7 is the parcellation of $U = \Omega_3$ and the right diagram of Fig. 2.8 is the corresponding surface in $\mathbb{R}^3$; the same parameter space is used. In any case, re-parametrization will lead to different functions and hence to a different representation of the identical manifold. In section 3.4 I will present a method for a standardized, reproducible parametrization of surfaces homeomorphic to Ω_3.

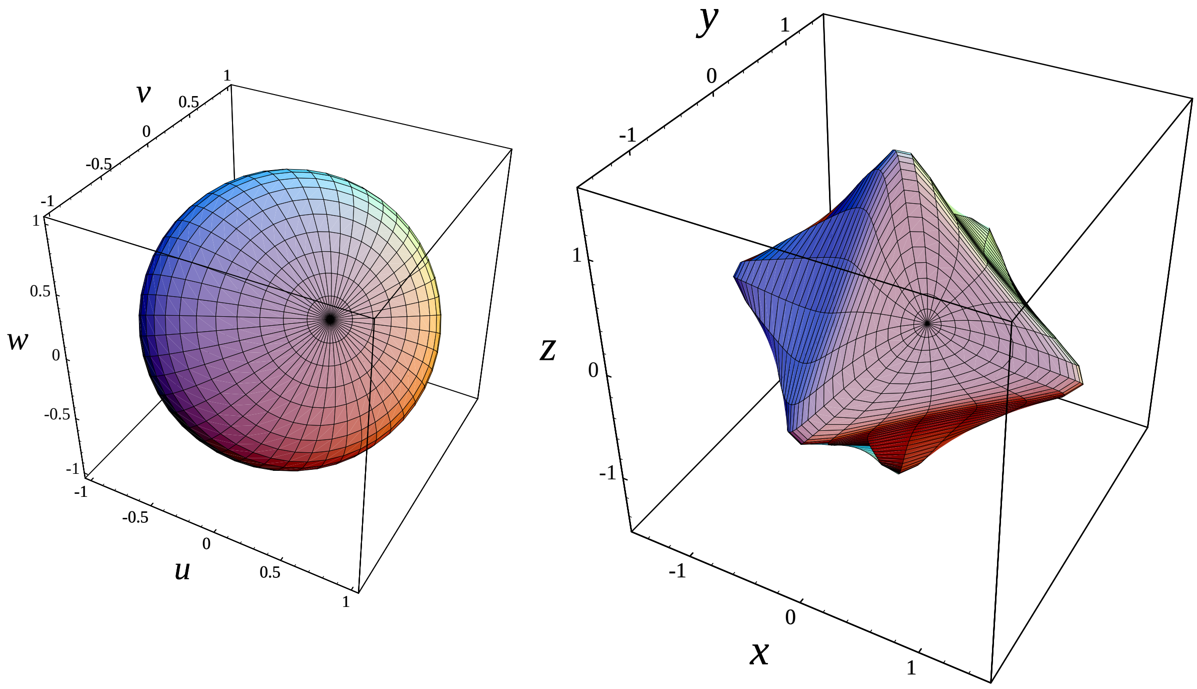


Figure 2.8: The same function as in Figure 2.7. *Left:* the rotated parcellation of Ω_3. *Right:* The corresponding surface.

2.2 Series Expansion

Many applications define the function $\underline{x}(\underline{u})$ as a linear combination of a collection of basis functions f_i. The set of basis functions may be finite or infinite (in which case an initial part of the series is used for practical purposes). Their domain is the parameter space, i.e.

$$f_i : U \hookrightarrow \mathbb{R}$$

The individual (scalar) coordinate functions take the following form.

$$\begin{aligned} x_0(\underline{u}) &= \textstyle\sum_i c_{i\,0} f_i(\underline{u}) \\ x_1(\underline{u}) &= \textstyle\sum_i c_{i\,1} f_i(\underline{u}) \\ &\vdots \end{aligned} \tag{2.2}$$

All n functions $x_k(\underline{u})$ use the same set of basis functions. The scalar coefficients $c_{i\,0}, c_{i\,1}$ (etc.) of one particular basis function f_i constitute an n-vector, $\underline{c}_i$. One vector equation can thus summarize the above equations

$$\underline{x}(\underline{u}) = \sum_i \underline{c}_i f_i(\underline{u}) \,. \tag{2.3}$$

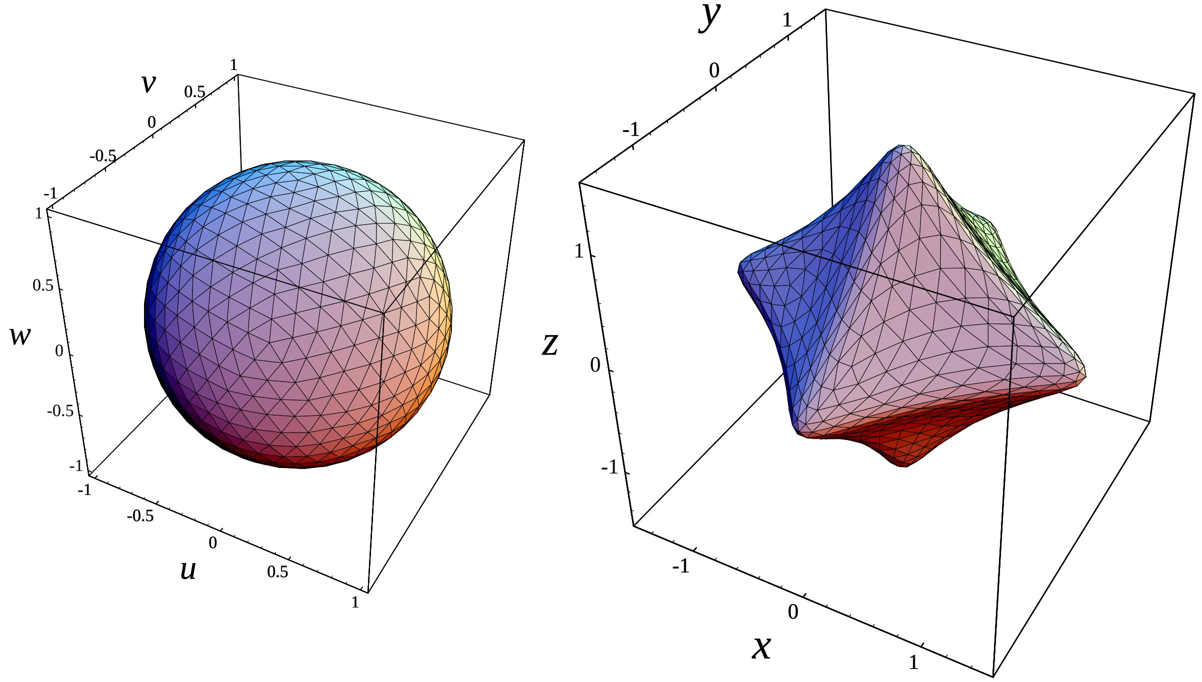


Figure 2.9: The same function as in Figure 2.7 with the surfaces facetted in yet another way.

In most applications, the set of basis functions is fixed. In this case, the collection of coefficient vectors determines the manifold completely: a list of numbers captures the definition of the manifold, and hence its shape, and even its parametrization. When the parametrization for an object is well-defined (and thus reproducible) and invariant to certain geometric transformations of the object, corresponding points on two transformed copies of the same manifold will map to the same point in parameter space. The resulting coefficient vectors can be transformed geometrically to a standard coefficient set or *descriptor*. The same shape will then invariably lead to the same descriptor, irrespective of any geometric transforms.

2.3 Planar curves

With $m = 1, n = 2$, the scheme (2.3) takes the following form.

$$\underline{x}(u) \quad = \quad \sum_i \begin{pmatrix} c_{i\,0} \\ c_{i\,1} \end{pmatrix} f_i(u) \tag{2.4}$$

We have two-dimensional coefficient vectors $\underline{c}_i$, and the basis functions f_i take one scalar argument u. This situation defines a planar curve. Popular basis functions include polynomials in u, i.e. linear combinations of powers u^k. The powers themselves are not suited as a basis in general, as they are ill conditioned, i.e. almost linearly dependent. The polynomials can be designed with many interesting and useful properties; alas their discussion is outside the scope of this text.

2.3.1 Splines

The basis functions of uniform cubic B-splines [2] take the form of piecewise cubic polynomials in (scalar) u.

$$B_0(u) = \frac{1}{6} \begin{cases} 0 & u < 0 \\ u^3 & 0 \leq u < 1 \\ -3u^3 - 12u^2 + 12u - 4 & 1 \leq u < 2 \\ 3u^3 - 24u^2 + 60u - 44 & 2 \leq u < 3 \\ -u^3 - 12u^2 + 48u - 64 & 3 \leq u < 4 \\ 0 & 4 \leq u \end{cases} \tag{2.5}$$

The individual basis functions $f_i(u) = B_i(u) = B_0(u-i)$ are all translated copies of this prototypic function. The generic function might also be given in a centered form (see Figure 2.10):

$$B_2(u) = \begin{cases} 0 & & u & < -2 \\ (2+u)^3 & -2 \leq & u & < -1 \\ 4 - 6u^2 - 3u^3 & -1 \leq & u & < 0 \\ 4 - 6\,u^2 + 3\,u^3 & 0 \leq & u & < 1 \\ (2-u)^3 & 1 \leq & u & < 2 \\ 0 & 2 \leq & u & \end{cases} \quad . \tag{2.6}$$

Many other kinds of B-splines, and of orders different from 3, can be defined.

2.3.2 Bézier Curves

Bernstein polynomials [2] of degree d are the summands in the binomial expansion of $((1-u)+u)^d = 1$.

$$P_{i,d} = \binom{d}{i} u^i (1-u)^{d-i} \tag{2.7}$$

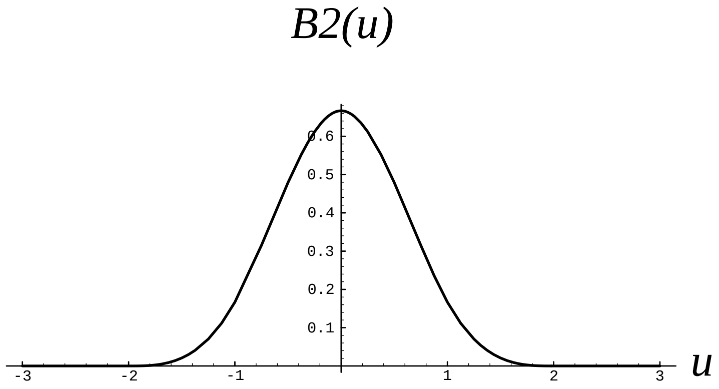


Figure 2.10: The prototype basis function of the uniform cubic B-splines. The smooth curve consists of four different cubics, extended by a straight line.

The parameter u must take a value in the interval $[0,1]$. These polynomials (for some fixed d) can serve as basis functions $f_i = P_{i,d}$ in the composition scheme (2.3) where $u \in [0,1]$ and $\underline{x} \in \mathbb{R}^2$ or $\underline{x} \in \mathbb{R}^3$. The resulting curves are called Bézier Curves [5] [4] [3]. Like the B-splines, they have many other nice properties.

Polynomials in u and related basis functions find applications mostly in graphics rather than in recognition. The definition of a reproducible parametrization of curves given as point lists is not of great importance in this area.

2.3.3 Elliptic Fourier Descriptors

The contour of a simply connected 2D region is a closed curve. (If the region is not simply connected, it has several closed contour curves.) The unit circle Ω_2 is the adequate parameter space for a closed curve, and the polar coordinate ϕ can parametrize it (7.3(appendix)). The position $\underline{x}(\underline{u})$ on the contour is then a periodic function of ϕ. This makes the harmonic circle functions $\frac{1}{\pi} \cdot \{\sqrt{2}, \cos\phi, \sin\phi, \cos 2\phi, \sin 2\phi, \ldots\}$ the preferred set of basis functions (the Fourier basis). The method of elliptic Fourier descriptors has been applied to object characterization and recognition [29, 23]. Friedrich [14] used this method among others for the shape characterization of particle collectives. He considered only the outline of a 2D picture of the particles he investigated.

The method of elliptic Fourier descriptors receives a somewhat more detailed discussion here. It plays a seminal role for the 3D surface description, and it introduces several concepts which are also relevant in 3D.

The basis consists of trigonometric functions – or complex exponentials – when we specify the position in parameter space with the polar coordinate ϕ. But when the position is given in Cartesian coordinates, the basis takes the simple form of polynomials. The link is again equation (7.3). In the following list, the common normalizing factor $1/\sqrt{\pi}$ is left out for readability.

$$\begin{array}{lcc lcc}
\cos\,\phi & = & u_0 & \sin\,\phi & = & u_1 \\
\cos 2\phi & = & u_0^2 - u_1^2 & \sin 2\phi & = & 2u_0 u_1 \\
\cos 3\phi & = & u_0^3 - 3u_0 u_1^2 & \sin 3\phi & = & 3u_0^2 u_1 - u_1^3 \\
\cos 4\phi & = & u_0^4 - 6u_0^2 u_1^2 + u_1^4 & \sin 4\phi & = & 4u_0^3 u_1 - 4u_0 u_1^3
\end{array}$$

$$\vdots$$

These relations are obtained through Euler's relation, using $\cos\, n\phi + j \sin\, n\phi = e^{jn\phi} = (e^{j\phi})^n = (u_0 + ju_1)^n$. The harmonic circle functions of degree n in ϕ are polynomials of degree n in u_0 and u_1. They can be written as *homogeneous* polynomials, as in the above list. All of these functions are orthogonal on the unit circle, and if all are divided by $\sqrt{\pi}$, the basis is orthonormal.

The construction of *invariant* descriptors for closed planar curves introduces the problem of parametrization. Arclength is the simplest and most straightforward choice. An arbitrary point on the contour is chosen as the starting point. When the whole curve has the length L, the point at arclength s from the starting point – say counterclockwise – is assigned parameter $\phi = \frac{2\pi s}{L}$; s is a scalar.

The mathematical treatment is simpler in the complex plane, i.e. when the parameter space is the circle $U = \{u \in \mathbb{C} \mid u^*\, u = 1\}$ and the mapping $z : U \hookrightarrow \mathbb{C}$ represents the curve. The relation between s, ϕ and u is

$$u \;=\; u_0 + j\, u_1 \quad = \quad e^{j\phi} \;=\; e^{\frac{2\pi j s}{L}}\;; \tag{2.8}$$

j is the imaginary unit.

Fourier expansion

The function $z(u)$ is represented as a series of complex exponentials.

$$z(u) \quad = \quad \sum_{n=-\infty}^{\infty} z_n u^n \tag{2.9}$$

We will refer to this central formula several times. We can express the complex coefficient z_n in polar notation, i.e.

$$z_n = r_n e^{j\psi_n} \ , \tag{2.10}$$

with $r_n \in \mathbb{R}$, $r_n \geq 0$, and $\psi_n \in \mathbb{R}$.

Determining the coefficients The calculation of z_n for a given contour $z(u)$ of practical interest. The following deduction summarizes a few general properties of Fourier series. As the parameter space is a subset $U \subset \mathbb{C}$ of the complex numbers, an infinitesimal step du of length $|du| = d\phi$ must be restricted to the direction counterclockwise tangential to U (or orthogonal to u), i.e.

$$du \quad = \quad ju|du| = ju\frac{2\pi}{L}ds \tag{2.11}$$

In (2.9) we substitute m for the summation variable. We multiply both sides with u^{-n} and then integrate $\oint |du|$. The integral in the sum evaluates to $2\pi\delta_{m,n}$, where $\delta_{..}$ denotes Kronecker's Delta[1]. The value of z_n follows immediately.

$$\begin{aligned}
z(u) &= \sum_{m=-\infty}^{\infty} z_m u^m && (2.12)\\
z(u)u^{-n} &= \sum_{m=-\infty}^{\infty} z_m u^m u^{-n} = \sum_{m=-\infty}^{\infty} z_m u^{m-n} \\
\oint z(u)u^{-n}|du| &= \sum_{m=-\infty}^{\infty} z_m \oint u^{m-n}|du| \\
&= \sum_{m=-\infty}^{\infty} z_m 2\pi\delta_{m,n} = 2\pi z_n \\
z_n &= \frac{1}{2\pi}\oint z(u)u^{-n}|du| && (2.13)
\end{aligned}$$

In most applications, $z(u)$ describes a polygon and often we are not interested in the center of gravity of the contour. In this case, it is simplest to start from the derivative $\frac{d}{|du|}$ of (2.9) and to derive a formula for z_n

---

[1] Note the difference between $\oint u^l|du| = 2\pi\delta_{l,0}$ and $\oint u^l du = 2\pi j\delta_{l,-1}$.

exactly as above.

$$
\begin{aligned}
z'(u) = \frac{d}{|du|}z(u) &= ju\frac{d}{du}z(u) \;=\; ju \sum_{m=-\infty}^{\infty} z_m \, m u^{m-1} && (2.14)\\
&= \sum_{m=-\infty}^{\infty} jmz_m \, u^m &&\\
z_n &= \frac{1}{jn}\oint z'(s)u^{-n}|du| && (2.15)
\end{aligned}
$$

Computing the Fourier descriptor for a closed polygon When the curve is given as a polygon (e.g. by its Freeman-code [13]), it can be written as a sum of straight line pieces. In the same way the integral (2.15) breaks up into a sum of partial integrals. The M sample points $z(u_k),\ k = 0 \ldots M$ of the Freeman curve (the corners of the M-gon) define the transitions between the partial integrals. The arclength parametrization of the curve implies that the point $z(u)$ traverses the curve with constant speed; on each individual straight line piece this leads to

$$z'(u) = \frac{\Delta z}{|\Delta u|} = \frac{L\,\Delta z}{2\pi\,|\Delta u|} = const\ .$$

We anticipate that this relation will simplify the expression for z_n significantly. We note

$$
\begin{array}{rcll}
u_0 &=& 1 & \\
u_M &=& e^{2\pi j} = 1 & \\
z(u_0) &=& z(u_M), & \text{closed polygon}\\
\Delta z[k] &=& z(u_{k+1}) - z(u_k) & \\
z'[k] &=& \frac{L\,\Delta z[k]}{2\pi\,|\Delta z[k]|} & \\
z'(u) &=& z'[k], & \text{for } u \text{ between } s_k \text{ and } s_{k+1}
\end{array}
$$

and get

$$
\begin{aligned}
z_n &= \frac{1}{jn}\oint z'(u)u^{-n}|du|\\
&= \frac{1}{jn}\sum_{k=0}^{M-1}\int_{u_k}^{u_{k+1}} z'(u)u^{-n}|du|\\
&= \frac{1}{jn}\sum_{k=0}^{M-1} z'[k]\int_{u_k}^{u_{k+1}} u^{-n}|du|
\end{aligned}
$$

$$\begin{aligned} &= \frac{1}{jn}\sum_{k=0}^{M-1} z'[k]\left[\frac{1}{jn}u^{-n}\right]_{u_k}^{u_{k+1}} \\ z_n &= -\frac{1}{n^2}\sum_{k=0}^{M-1} z'[k]\,u^{-n}\Big|_{u_k}^{u_{k+1}} \quad . \end{aligned} \tag{2.16}$$

To evaluate this sum, it is enough to calculate one complex exponential for each term. As the lower bound of each term is equal to the upper bound of the previous term, the value of u^{-n} can be re-used.

In the case of the Freeman-code, Δz can only take eight different values, namely $1, 1+j, j, -1+j, -1, -1-j, -j$ and $1-j$. The same holds for z': these as well are completely determined by the code $0\ldots7$ (and L). The representation as an array of constants is probably preferable, although we can find the following closed expressions for a given code c.

$$\begin{aligned} z' &= \frac{L}{2\pi}e^{\frac{cj\pi}{4}} \\ \Delta s = |\Delta z| &= \sqrt{1 + c \bmod 2} \\ \Delta z &= \sqrt{1 + c \bmod 2}\;\; e^{\frac{cj\pi}{4}} \end{aligned}$$

Harmonic contributions to the contour In equation (2.9) we match terms with opposite indices in pairs

$$z(u) = z_0 + \sum_{n=1}^{\infty} \underbrace{z_n u^n + z_{-n}u^{-n}}_{elli_n(u)} \tag{2.17}$$

All harmonic contributions – except z_0 – describe an ellipse when taken by themselves. The ellipse $elli_n(u)$ is covered n times when u runs over the circle U. The vertex of the ellipse, i.e. the maximum of the absolute value of $elli_n$, is reached where both terms of the sum have the same phase; at this point the triangle inequality

$$|elli_n(u)| \leq |z_n u^n| + \left|z_{-n}u^{-n}\right| \tag{2.18}$$

holds with equality.

Dependence from the starting point The notation $|^V$ marks the quantities that result from shifting the starting point by $\lambda = \frac{L}{2\pi}\theta$ on the contour. We note the following relations.

$$
\begin{aligned}
s &= s|^V + \lambda && (2.19)\\
u &= u|^V e^{j\theta} && (2.20)\\
z(u) &= z(u|^V e^{j\theta}) = \sum_{n=-\infty}^{\infty} z_n \left(u|^V e^{j\theta}\right)^n && (2.21)\\
&= z|^V(u|^V) = \sum_{n=-\infty}^{\infty} \underbrace{z_n e^{jn\theta}}_{z_n|^V} \left(u|^V\right)^n && (2.22)\\
z_n|^V &= z_n\, e^{jn\theta} && (2.23)
\end{aligned}
$$

Eqn. 2.10 leads to $z_n|^V = r_n e^{j(\psi_n + n\theta)}$. With the choice of θ (or λ), we can move the starting point to an arbitrary position on the contour, e.g. to the vertex of $elli_1$:

$$
elli_1|^V(u=1) = r_1 e^{j(\psi_1+\theta)} + r_{-1} e^{j(\psi_{-1}-\theta)} \overset{\text{vertex}}{=} (r_1 + r_{-1}) e^{j\psi} \qquad (2.24)
$$

The second equation holds only at the vertex, i.e. when the phases match; the common phase ψ is left to determine. From the equality of the phases we conclude

$$
\begin{aligned}
\psi_1 + \theta &= \psi_{-1} - \theta = \psi && (2.25)\\
2\theta &= \psi_{-1} - \psi_1 && (2.26)\\
\theta &= \frac{\psi_{-1} - \psi_1}{2} \qquad \text{and hence} && (2.27)\\
\psi &= \frac{\psi_1 + \psi_{-1}}{2} && (2.28)
\end{aligned}
$$

Thus θ defines the necessary shift of the starting point, and ψ gives the (unchanged) position of the main axis of $elli_1$. One can also view the shifting of the starting point as a rotation of the object in parameter space U, the unit circle. The notion of rotating in parameter space will extend naturally to 3D surfaces in section 5.2.2.

The half main axes of $elli_n$ have the lengths $r_n + r_{-n}$ and $|r_n - r_{-n}|$. Only the long main axis of $elli_1$ will be used in the following.

Dependence from the rotational position of the object We rotate the coordinate system about the angle ψ or we rotate the object about $-\psi$, which is equivalent. In the complex notation, a rotation is simply a multiplication and applying (2.9) immediately reveals the coefficients of the rotated object.

$$z|^R(u) = z(u)\, e^{-j\psi} = \sum_{n=-\infty}^{\infty} z_n e^{-j\psi} u^n \tag{2.29}$$

$$z_n|^R = z_n e^{-j\psi} \tag{2.30}$$

All coefficients are multiplied by the same complex number. This achieves a rotation in object space. We have anticipated the special choice of the rotation angle by naming it "ψ" – it rotates the main axis of $elli_1$ to the horizontal real axis.

Scale dependence The scale factor α, in analogy to (2.29), leads to

$$z|^S(u) = \alpha\, z(u) = \sum_{n=-\infty}^{\infty} \alpha\, z_n u^n \tag{2.31}$$

$$z_n|^S = \alpha\, z_n \tag{2.32}$$

To normalize half the major axis of $elli_1$ to 1, we set $\alpha = \frac{1}{r_1+r_{-1}}$.

Invariant Fourier descriptors

Ignoring z_0, that is setting $z_0|^T = 0$, achieves translation invariance. We sum up all standardizations; the invariant coefficients are denoted $\tilde{z}_n$ for brevity:

$$z_n|^{V,R,S,T} = \tilde{z}_n = z_n \frac{e^{j(n\theta-\psi)}}{r_1 + r_{-1}} \tag{2.33}$$

$$\tilde{z}_0 = 0 \tag{2.34}$$

In particular, we observe the first degree coefficients.

$$\tilde{z}_1 = \frac{e^{j(\theta-\psi)}}{r_1 + r_{-1}} r_1 e^{j\psi_1} = \frac{r_1}{r_1 + r_{-1}} \tag{2.35}$$

$$\tilde{z}_{-1} = \frac{e^{j(-\theta-\psi)}}{r_1 + r_{-1}} r_{-1} e^{j\psi_{-1}} = \frac{r_{-1}}{r_1 + r_{-1}} \tag{2.36}$$

In addition we consider the number of degrees of freedom we can eliminate by standardizing, and how they reflect in properties of the coefficients.

Translation		2	$\tilde{z}_0 = 0$
Rotation	ψ	1	$\tilde{z}_1, \tilde{z}_{-1} \in \mathbb{R}$
Starting point	$\lambda = \frac{L\theta}{2\pi}$	1	
Scale		1	$\tilde{z}_1 + \tilde{z}_{-1} = 1$

Both $\tilde{z}_1$ and $\tilde{z}_{-1}$ are positive, but this does not correspond to a whole degree of freedom, but so to speak only to one bit for each assertion. One last bit of freedom remains though: θ and ψ together are only determined modulo π. This corresponds to the fact that an ellipse has two vertices, and there is no reason to prefer either one when only considering $elli_1$. Taking higher harmonics into account could resolve this ambiguity. The other possibility is to consider both instances in the comparison. It follows from equation (2.33) that the coefficients with even index change their sign when θ and ψ both flip about π.

The aggregate of all $\tilde{z}_n$ constitutes a shape based descriptor or feature vector, which can serve for object recognition.

Truncating the expansion If only for practical reasons, the infinite sum (2.9) must be truncated at some maximum degree, say N. We define the partial sum $\check{z}$ and the complex-valued deviation $f(u)$ as follows.

$$\check{z}(u) = \sum_{n=-N}^{N} z_n u^n \tag{2.37}$$

$$f(u) = z(u) - \check{z}(u) = \sum_{|n|>N} z_n u^n \tag{2.38}$$

The maximal $\underline{x}$ or y deviation ϵ, which is sometimes analyzed in the literature, can also be defined in our complex notation.

$$\epsilon = \max\left(\sup_{\forall u} |\mathrm{Re} f(u)|, \sup_{\forall u} |\mathrm{Im} f(u)|\right) \tag{2.39}$$

But the mean square error ϵ_2 is better suited to the theory of orthogonal functions. We define its square.

$$\epsilon_2^2 = \oint |f(u)|^2 \, |du| \tag{2.40}$$

$$= 2\pi \sum_{|n|>N} |z_n|^2 = 2\pi \sum_{|n|>N} r_n^2 \tag{2.41}$$

The nice relation (2.41) results from applying (2.38) after a short calculation. It is also known as Parseval's theorem.

Of all standardizations, only scaling affects the above considerations. It multiplies the error ϵ or ϵ_2, respectively, by α. (Rotation can slightly change ϵ, by a factor of $\sqrt{2}$ at most, but ϵ_2 never changes.)

Classification of objects

Model based object recognition is an important application of shape descriptors. For each model object g, a descriptor, i.e. a collection $\{\tilde{z}_n\}[g]$, is computed in the so-called training phase. In the same way a descriptor $\{\hat{z}_n\}$ is determined for the unknown object. We define a *classification distance* (or its square, respectively).

$$D[g] = \sum_{n=-N}^{N} |\tilde{z}_n[g] - \hat{z}_n|^2 \tag{2.42}$$

We decide for the model g with minimal $D[g]$.

Relations to real-valued notation

In the expression for $elli_n$ we collect real and imaginary parts, which correspond to the $\underline{x}$ and y coordinate. The argument $\frac{2\pi ns}{L}$, which is the same for all trigonometric functions, is omitted for brevity.

$$\begin{aligned} elli_n(s) &= z_n u^n + z_{-n} u^{-n} \\ &= (\operatorname{Re} z_n + j \operatorname{Im} z_n)(\cos + j\ \sin) \qquad (2.43) \\ &\quad +(\operatorname{Re} z_{-n} + j \operatorname{Im} z_{-n})(\cos - j\ \sin) \\ &= \quad (\operatorname{Re} z_n + \operatorname{Re} z_{-n}) \cos + (\operatorname{Im} z_{-n} - \operatorname{Im} z_n) \sin \\ &\quad + j\ ((\operatorname{Im} z_n + \operatorname{Im} z_{-n}) \cos + (\operatorname{Re} z_n - \operatorname{Re} z_{-n}) \sin) \qquad (2.44) \end{aligned}$$

We define

$$\begin{aligned} a_n &= \operatorname{Re} z_n + \operatorname{Re} z_{-n} && (2.45) \\ b_n &= -\operatorname{Im} z_n + \operatorname{Im} z_{-n} && (2.46) \\ c_n &= \operatorname{Im} z_n + \operatorname{Im} z_{-n} && (2.47) \\ d_n &= \operatorname{Re} z_n - \operatorname{Re} z_{-n} && (2.48) \end{aligned}$$

and write

$$\begin{aligned} x_n(s) = \operatorname{Re}(elli_n(s)) &= a_n \cos + b_n \sin && (2.49) \\ y_n(s) = \operatorname{Im}(elli_n(s)) &= c_n \cos + d_n \sin && (2.50) \end{aligned}$$

or, in matrix notation,

$$\begin{pmatrix} x \\ y \end{pmatrix}_n = \begin{pmatrix} a_n & b_n \\ c_n & d_n \end{pmatrix} \begin{pmatrix} \sin \frac{2\pi n s}{L} \\ \cos \frac{2\pi n s}{L} \end{pmatrix} \tag{2.51}$$

It becomes obvious that $elli_n$, as an affine image of the unit circle, is an ellipse.
We can solve the definitions (2.45...2.48) for z_n and z_{-n}.

$$z_n = \frac{1}{2}(a_n + d_n) + \frac{j}{2}(c_n - b_n) \tag{2.52}$$

$$z_{-n} = \frac{1}{2}(a_n - d_n) + \frac{j}{2}(c_n + b_n) \tag{2.53}$$

Finally, the following relations hold for the special case $n = 0$.

$$z_0 = a_0 + j\, c_0 \tag{2.54}$$

$$a_0 = \mathrm{Re}\, z_0 \tag{2.55}$$

$$c_0 = \mathrm{Im}\, z_0 \quad . \tag{2.56}$$

2.4 Space Curves

When the coefficient vectors in (2.4) are three-dimensional instead of two-dimensional, we get space curves. Their treatment is almost identical to that of plane curves. The same basis functions may be used. And as in 2D, there can be open and closed curves, with $\mathbb{R}$ and Ω_2 as their respective, one-dimensional parameter spaces.

In the case of the Fourier basis, the first degree reconstruction is an ellipse (as in 2D) and hence *planar*, but it can have an arbitrary orientation in space. The rotation in parameter space (shift of the starting point) is identical to that in the 2D case. The rotation in object space has now three degrees of freedom instead of one. But it could take place much the same way as above after rotating the plane of the first order ellipse into the x_0-x_1 plane.

Chapter 3

Surfaces: Representation and Parametrization

3.1 Open and closed Surfaces

The superficies of a 3D voluminous object is a closed surface; these are the objects of research of this thesis. There exist closed surfaces which cannot be the surface of any object because they self-intersect. These include the non-orientable surfaces. They will not receive further attention here. In the following discussion we will concentrate further on the surfaces of simply connected objects or, equivalently, the ones with trivial topology, e.g. a ball. Only piecewise smooth surfaces are considered in this thesis.

Surfaces are two-dimensional manifolds. When standing at a point on a surface, there are two independent directions in which we can walk without leaving the surface. Correspondingly, it takes two real numbers to specify a point uniquely on a surface, if this is to be done in a bi-continuous way. The two numbers are called the *parameters* of the point. The requirement of bi-continuity means intuitively that points that lie close to each other in the surface must have similar coordinates, and vice versa. In some applications piecewise bi-continuity is sufficient. The task of assigning a parameter pair to all points of a given surface in this fashion is referred to as *parametrization* of the surface.

In contrast to the closed surfaces there are bordered surfaces. They shall be called surface *patches*. They can result from cutting a region from the closed surface of some object. When patches are simply connected, they have only one connected border. Of course there are “patches” with much fancier topology, e.g. unresolvable knots in 3D, but they are not dealt with here.

A rectangular portion of $\mathbb{R}^2$ as parameter space U is often sufficient for a discussion of (2.3) with $n = 3$ and $m = 2$ for open surfaces. If necessary in an application, U is trimmed to a subregion of the rectangle, which yields a part of the surface with the same topology (connectivity) as the subregion, except for self-intersections. The products of one-dimensional Bézier and B-spline basis functions form a basis for Bézier and B-spline patches, respectively. For Bézier patches the following basis functions result,

$$P_{i_0,d_0;i_1,d_1}(u_0, u_1) = P_{i_0,d_0}(u_0) \cdot P_{i_1,d_1}(u_1)\,. \tag{3.1}$$

In one patch, d_0 and d_1 are fixed. The $(d_0 + 1)(d_1 + 1)$ different basis functions result from varying i_0 from 0 to d_0 and i_1 from 0 to d_1. More complicated surfaces are patched together from several patches; conditions for smooth joints have been set up.

Section 3.5 elaborates on the parametrization of surface patches of discrete volumetric objects.

3.2 Closed surfaces in 3D

Let x_0, x_1 and x_2 denote Cartesian object space coordinates and θ and ϕ polar parameter space coordinates. The parametrization gives us three explicit functions defining the object surface as follows:

$$\underline{x}(\theta, \phi) = \begin{pmatrix} x_0(\theta, \phi) \\ x_1(\theta, \phi) \\ x_2(\theta, \phi) \end{pmatrix}$$

When the free variables θ and ϕ run over the whole sphere (e.g. $\theta = 0 \ldots \pi, \phi = 0 \ldots 2\pi$), the point $\underline{x}(\theta, \phi)$ runs over the whole surface of our object. The sphere Ω_3 is considered a perfectly symmetric surface without any singular points or preferred directions. The specification (e.g. by two polar coordinates) of a point $\underline{u}$ in parameter space is not important: the characterization of the surface lies completely in the mapping from the parameter space Ω_3 to the object space $\mathbb{R}^3$, that is, in the function $\underline{x}(\underline{u})$, which leads back to the general definition of manifolds (2.1).

3.3 The surface data structure

Medical CT or MRI images are examples of volumetric data. For each cuboidal cell (volume element or voxel) in a certain volume we have one or more measurements. When segmentation succeeds, one anatomical unit can be characterized by a binary data volume, in which every voxel contains either 1, which means it belongs to the unit, or 0, meaning it is in the background. The object is then the set of "1" voxels and can be pictured as a collection of small cubes. As stated above, we assume simple connectedness. In addition we exclude the *singular* or *bad* points where voxels touch on an edge or vertex only, without a 6-path between them of length three or four, respectively (see Figure 3.1)[1]. Section 2 of [24] contains a good discussion of singular points in binary images. The surface of a voxel object is a set of unit squares, all parallel to one of the three coordinate planes x_1x_2, x_2x_0 or x_0x_1. The edges and vertices that bound the faces are also parts of the surface.

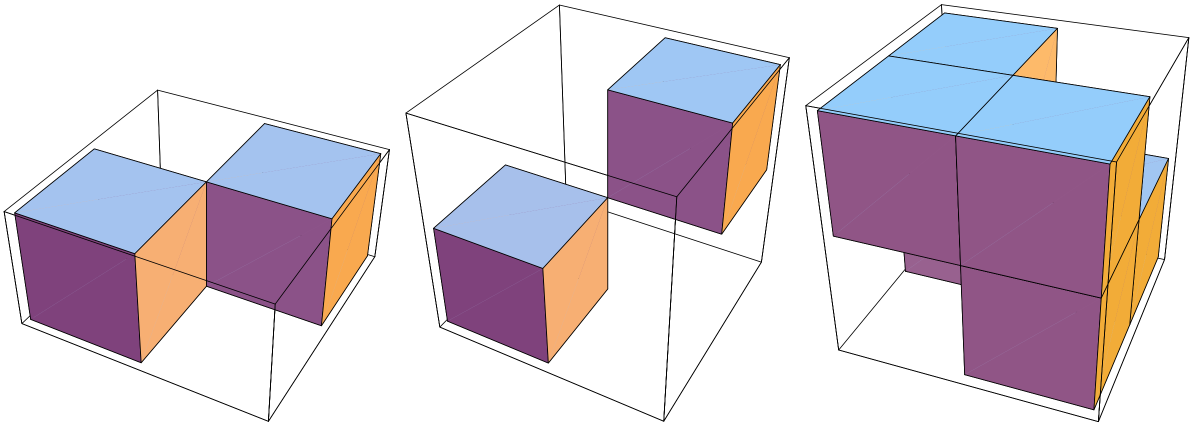


Figure 3.1: In a binary data volume, singular configurations of edge type (left) and of vertex type (middle and right) can occur.

Our input data is a single simply 6-connected object. It is represented as a 3-D binary image. We adopt the *cuberille* notion[19]. The surface of the object must be represented as a data structure that reflects the geometry as well as topological relations. This allows working effectively with the surface. The data structure stores information about all the square faces separating object and background voxels, and all the edges and vertices bounding these faces. This initial data structure constitutes

---

[1] The first implementation simply rejected a data set that contained any singularities. An improved algorithm that handles all possible configurations correctly has been developed [34].

a complete representation of the object surface; similar structures are often used for 3-D display purposes [17].

For a n_{x_0} by n_{x_1} by n_{x_2} data volume, the integer voxel coordinates have the ranges $0 \leq x_0 < n_{x_0}$, $0 \leq x_1 < n_{x_1}$ and $0 \leq x_2 < n_{x_2}$. We interpret each voxel as a cube extending from x_0 to $x_0 + 1$, from x_1 to $x_1 + 1$ and from x_2 to $x_2 + 1$, where (x_0, x_1, x_2) are the nominal coordinates of the voxel. The voxel center is placed at $(x_0 + \frac{1}{2}, x_1 + \frac{1}{2}, x_2 + \frac{1}{2})$ to retain integer coordinates for the corners.

The surface data structure is organized around the vertices. Each surface vertex is listed once in the structure, which gives it a unique identifying number (an *id*). The entry of a vertex specifies its Cartesian coordinates (x_0, x_1, x_2) and the list of its direct and diagonal neighbors. The neighbors are given in counterclockwise sequence around the vertex when viewed from outside the object; therefore, direct and diagonal neighbors alternate (see Figure 3.2).

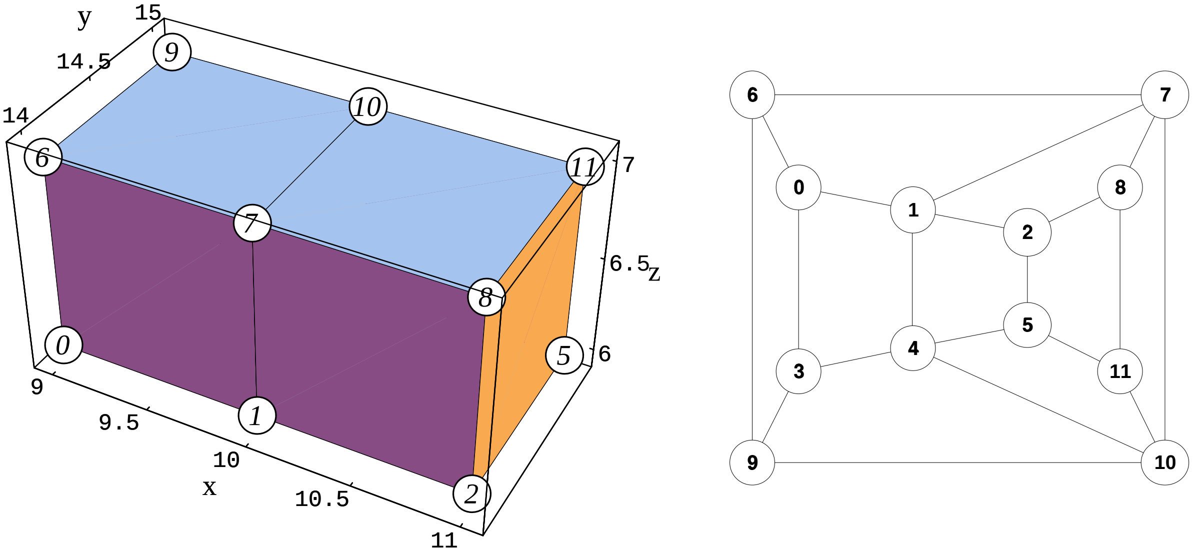


Figure 3.2: *Left:* A two-voxel example object illustrating the surface data structure, which focuses on the vertices. The numbers of the vertices are shown in circles (vertices 3 and 4 are hidden). The data structure entry for a vertex represents its Cartesian coordinates and a list of neighboring nodes. The entry for vertex 7 is {{x_0 =10, x_1 =14, x_2 =7}, *neighbors*={6, 0, 1, 2, 8, 11, 10, 9}}. The table below lists the complete surface data structure of this object. — *Right:* A flat diagram of the surface net for the same object.

3.3.1 An example of the basic surface data structure

The "two-voxel" object (cf. Figure 3.2) is used here and in the examples for the initial parametrization of section 4.1. The surface data structure of this object is given in Table 3.1.

node nr.	x_0	x_1	x_2	neighbors
0	{{{ 9,	14,	6},	{ 1, 7, 6, 9, 3, 4}},
1	{{ 10,	14,	6},	{ 0, 3, 4, 5, 2, 8, 7, 6}},
2	{{ 11,	14,	6},	{ 1, 4, 5, 11, 8, 7}},
3	{{ 9,	15,	6},	{ 4, 1, 0, 6, 9, 10}},
4	{{ 10,	15,	6},	{ 3, 9, 10, 11, 5, 2, 1, 0}},
5	{{ 11,	15,	6},	{ 4, 10, 11, 8, 2, 1}},
6	{{ 9,	14,	7},	{ 7, 10, 9, 3, 0, 1}},
7	{{ 10,	14,	7},	{ 6, 0, 1, 2, 8, 11, 10, 9}},
8	{{ 11,	14,	7},	{ 7, 1, 2, 5, 11, 10}},
9	{{ 9,	15,	7},	{ 10, 4, 3, 0, 6, 7}},
10	{{ 10,	15,	7},	{ 9, 6, 7, 8, 11, 5, 4, 3}},
11	{{ 11,	15,	7},	{ 10, 7, 8, 2, 5, 4}}}

Table 3.1: The complete surface data structure of the two cube object.

The following algorithm generates the surface data structure. Object voxels touching the border of the data volume would require a special treatment; therefore, we make sure that all border voxels belong to the background. We define an *interior vertex* as the common corner of eight voxels of the data volume. The $2 \times 2 \times 2$ voxels form the *neighborhood* of the interior vertex. The neighborhood contains six *half-neighborhoods*, as Figure 3.3 illustrates. Two half-neighborhoods of the same vertex are *orthogonal* if they share exactly two vertices, like e.g. NORTH and WEST. In three nested loops we visit all the interior vertices. Let the x_2-loop be the outermost, so that x_2 numbers the current plane. If the neighborhood of a vertex is homogeneous, i.e. all eight voxels are in the object or all are in the background, the vertex is ignored. Otherwise (if the incident voxels are mixed), a new surface vertex has to be entered into the data structure. A surface vertex in plane x_2 may have neighboring surface vertices in the planes $x_2 - 1$, x_2 and $x_2 + 1$. Therefore surface vertices must be detected and assigned identifying numbers one plane ahead of time, and the ids of the current (x_2) and previous ($x_2 - 1$) plane must

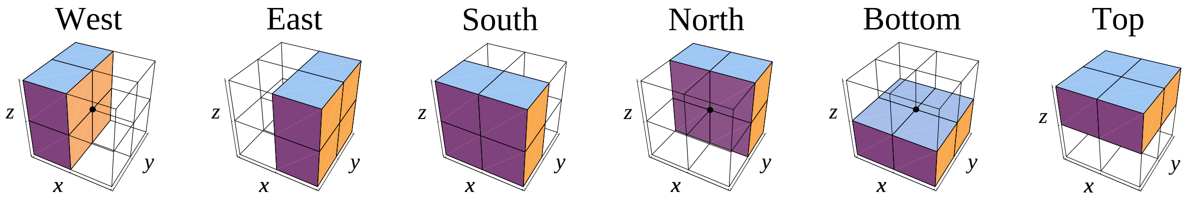


Figure 3.3: A vertex has six half-neighborhoods. Each one of them consists of four voxels. The vertex is marked with a black dot.

be kept at hand as well. When assigning ids, singular points should be checked for. If any half-neighborhood is checkered, this signals an edge type singularity. If two diagonally opposite voxels are the only object or background voxels in the neighborhood, there is a vertex type singularity.

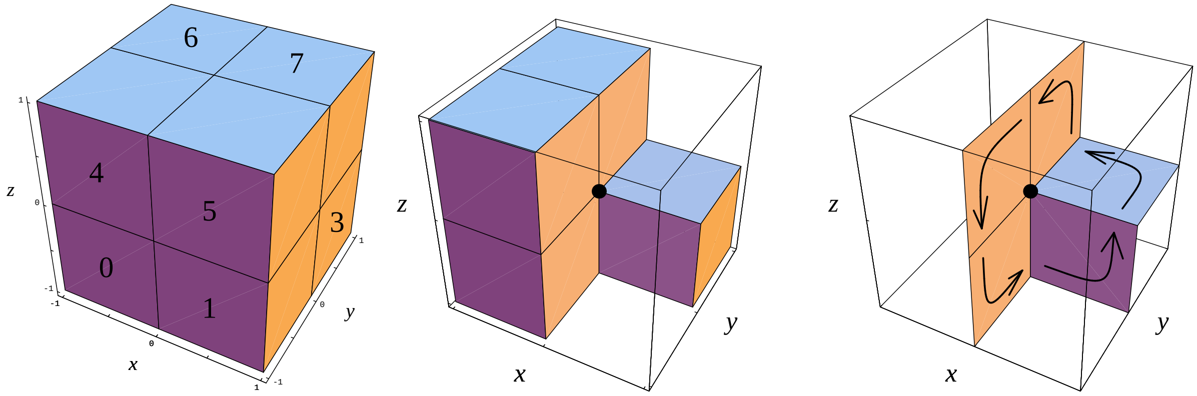


Figure 3.4: *Left:* Numbering of the voxels incident at a vertex. Voxel 2 is hidden at the lower back. Coordinates are relative to the vertex being examined. *Middle:* The neighborhood of a vertex where five object and three background voxels meet; an example. *Right:* The five surface facets that meet at the vertex are traced counterclockwise in the order E, N, T, S, B (,E).

For each surface vertex in the current plane, the Cartesian coordinates are put into the entry of the vertex. Starting at a 6-neighbor surface vertex, we cycle around the central vertex, going over all incident faces until we are back at the starting neighbor vertex. Figure 3.4 (middle and right) illustrates this procedure. The half-neighborhoods are inspected to find a starting direction. A 6-neighbor is a surface vertex, if and only if the corresponding half-neighborhood is not homogeneous. In the example in Figure 3.4, the "West" half-neighborhood is homogeneous, but "East" is not, so this is the starting direction. Table 3.2 controls advancing to adjacent directions. The left diagram in Figure 3.4 defines the numbering of the neighborhood voxels. When facing any direction, the four adjacent directions are always arranged counterclockwise. For example, when we face north, we see E,T,W,B in counterclockwise succession.

Direction	half space	voxels	adjacent
West	$x_0 < 0$	0,2,4,6	B,N,T,S
East	$x_0 > 0$	1,3,5,7	S,T,N,B
South	$x_1 < 0$	0,1,4,5	B,W,T,E
North	$x_1 > 0$	2,3,6,7	E,T,W,B
Bottom	$x_2 < 0$	0,1,2,3	E,N,W,S
Top	$x_2 > 0$	4,5,6,7	S,W,N,E

Table 3.2: The six main directions are each associated with a direct neighbor vertex, a half space, a half-neighborhood of voxels (cf. Fig. 3.4 left), and four orthogonal adjacent directions

The key step of the algorithm is advancing from one direction to the next. The center vertex remains the same, we just face another neighbor (direction). The step from the current to the new direction must advance counterclockwise (seen from outside) on a facet of the object. Equivalently, the intersection of the object voxels in the full neighborhood, the old half-neighborhood and the new half-neighborhood must be equal to the intersection of the old half-neighborhood, the new one and the one cyclically next in the old direction's "adjacent" list. The latter is the intersection of three mutually orthogonal half-neighborhoods, so it always consists of a single voxel. Thanks to the exclusion of edge singularities (see footnote on page 31), there is exactly one successor direction for each direction reached by this scheme, and the direction always returns to the starting direction after 3 to 6 steps. The exclusion of vertex singularities guarantees that there is only one such loop.

The first move in the example is from direction "East" ($+x_0$) to "North" ($+x_1$). The neighborhood {0,2,3,4,6} intersected with old E {1,3,5,7} and new N {2,3,6,7} is {3}, the same as the intersection of E, N and B {0,1,2,3}, the successor of N.

The diagonal neighbor between two direct neighbors is reached by adding the vectors pointing to the direct neighbors. The list of the ids of all visited direct and diagonal neighbor vertices completes the entry of the central vertex.

Only vertices are stored in the data structure, but information about edges and faces of the surface is also present in the lists of neighbors. For instance, vertex 7 in Figure 3.2 has the neighbor list {`6, 0, 1, 2, 8, 11, 10, 9`}. Taking every second number in this list – 6, 1, 8, 10 – reveals that edges to nodes 6, 1, 8 and 10 emanate from node 7. Overlapping triples of neighbors – {6, 0, 1}, {1, 2, 8}, {8, 11, 10} and {10, 9, 6} (by wrapping around to the first neighbor) – give the four faces {7, 6, 0, 1}, {7, 1, 2, 8}, {7, 8, 11, 10} and {7, 10, 9, 6}, all written counterclockwise. In this way, every face of the surface is mentioned four times, once by each corner. Our algorithm, however, requires an additional table that lists every face exactly once. We generate it by visiting all vertices, putting a face in the table only when it is defined from the corner with the smallest id, i.e. when it is mentioned for the first time.

As the object lies completely within our data volume, we always find a closed surface. All vertices in the list are inner vertices of the surface, and therefore they all have an even number of neighbors. The data structure generalizes in a very natural way to the case of open surfaces, or surface patches, where vertices on the border of the surface have an odd number of neighbors. This possibility is not exploited here, but only later in the parametrization of surface patches (cf. section 3.5).

The correspondence between the surface net and a graph becomes clear when a vertex is interpreted as a node in the graph, an edge as an arc, and a face as a mesh (four-cycle)[40]. For a simply connected object we get a planar graph with the following topological properties: Four edges bound each face, and each edge bounds two faces and is bounded by two vertices. Depending on the local connectivity, each vertex bounds three to six edges. There are exactly two more vertices than faces; this follows from Euler's relation.

A surface with its two degrees of freedom is characterized by a polygonal description based on vertex coordinates with three spatial coordinates. Seeking for an appropriate parametrization, however, requires a description based on two parameters.

3.4 Parametrization of closed surfaces

A key step in the description of the form of a surface is the mapping of the surface to the parameter space, in our case the sphere. A one-to-one mapping must be constructed, i.e. any point on the surface has to map to exactly one point on the sphere, and vice versa. The location on the sphere corresponding to a surface point defines the *parameters* of the point. It can be represented in a computer as two polar or three Cartesian coordinates. Mapping a surface to the sphere assigns parameters to every surface point; therefore I also call it parametrization. The mapping must be continuous, i.e. neighboring points in one space must map to neighbors in the other space. It is possible and desirable to construct a mapping that preserves areas. The use of the cuberille notion gives special importance to case of square facets, which map to spherical quadrilaterals. Figure 3.5 symbolically illustrates this mapping of a selected facet from the object surface to a portion of space U. We recall from subsection 2.1.3, and in particular from example 6, that the parameter space $U = \Omega_3$ is a subset of of $\mathbb{R}^3$, but that it could be related to the rectangle $[0, \pi] \times [0, 2\pi) \subset \mathbb{R}^2$ through the bijection (7.4(appendix)). It is not possible in the general case to map every surface facet to a spherical square. Distortions cannot be avoided, but they should be minimal.

It emerges that the parametrization, i.e., the embedding of the object surface graph into the surface of the unit sphere, is a constrained optimization problem. The following paragraphs define the meaning of *variables*, *objective* (goal function), *constraints* and *starting values* in this context.

Variables The coordinates of all vertices can vary in the optimization. Using two (e.g. spherical) coordinates per vertex would be the most economic representation with respect to storage space, but this would make the equal treatment of all spatial directions difficult and pose the problem of discontinuity and singularities in the parameter space. Therefore we prefer Cartesian coordinates (u_0, u_1, u_2) for representing a location on the sphere, introducing one virtual degree of freedom per vertex. The number of variables is three times the number of vertices.

Constraints Two kinds of equalities and one kind of inequalities constrain the values that the variables can take.

1. The Euclidean norm of the coordinates of any vertex must be 1. This constraint compensates for the virtual degree of freedom and forces every vertex to lie on the unit sphere in parameter space.

2. We ask for *area preservation*, which in our context means that any object surface region must map to a region of proportional area on the sphere. To satisfy this requirement, we include one constraint

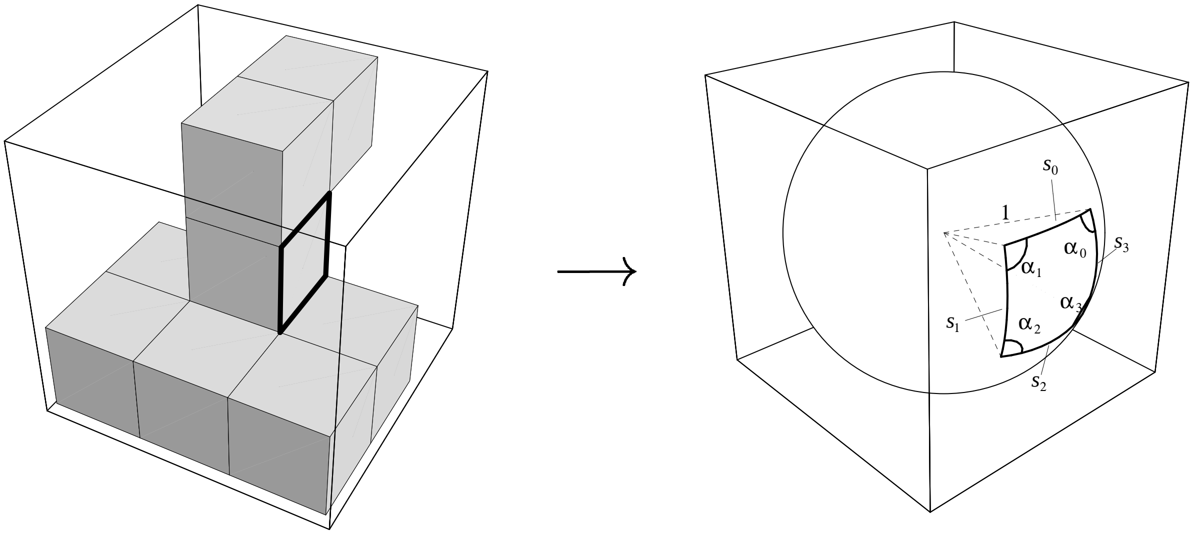


Figure 3.5: Every single face on the object's surface is mapped to a spherical quadrilateral. The sides of a spherical polygon are geodesic arcs on the sphere surface. As the sphere has unity radius, the length of a side s_i is equal to the corresponding center angle (in radian). The quadrilateral in this illustration is special in that its four sides $s_0 \cdots s_3$ are equal and its four angles $\alpha_0 \cdots \alpha_3$ are equal: it is the spherical analogue of a square.

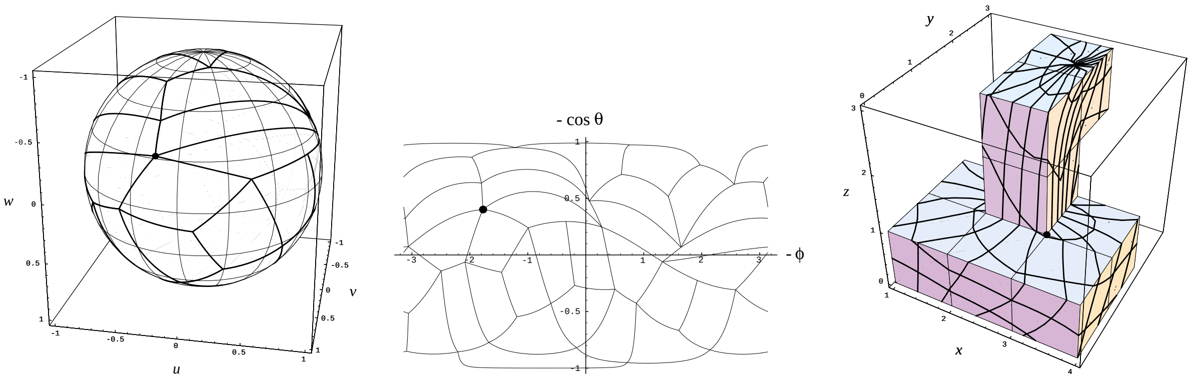
for each elementary facet (see Figure 3.5): the area of the corresponding spherical quadrilateral must be 4π divided by the total number faces.

3. No angle α_k of any quadrilateral on the sphere must become negative or exceed π.

Objective Function The *goal* is to minimize the distortion of the surface net in the mapping. It is conceptually similar to angle preservation, and it must tend to make the shape of all the mapped faces as similar to their original square form as possible. To fulfill this goal perfectly, a facet should map to a "spherical square" (see Figure 3.5). This can never be reached exactly for all faces except when the object has a very special form, e.g., consists of one single voxel. There are several ways to trade off between the distortions made at different vertices. We observe that the ideal shape of any face, a spherical square, minimizes the circumference $\sum_{i=0}^{3} s_i$ of any spherical quadrilateral with a given area. At the same time it maximizes $\sum_{i=0}^{3} \cos s_i$. These two measures are similar, but not equivalent if summed over the whole net, as they trade off among distortions differently. The second measure punishes too long sides more and honors too short sides less than the first one, which is a desirable effect. It is also simpler to calculate; the cosine of a side - and of the respective central angle - is the dotproduct of the vectors from the sphere center to the endpoints of the side.

Starting Values The variables in our optimization are the positions on the unit sphere to which the vertices are mapped. Therefore, *initial values* in this context means an first rough mapping of the object's surface to the sphere. It is important for the optimization algorithm that the sphere be completely covered with faces and none of them overlap, even in the beginning.

Chapter 4 describes the construction of an initial parametrization satisfying the last requirement, and it elaborates on the technical details of the optimization procedure. Figure 3.6 anticipates the parametrization result for the small object from Figure 3.5, called "duck" in the sequel.



(*a*) (*b*) (*c*)

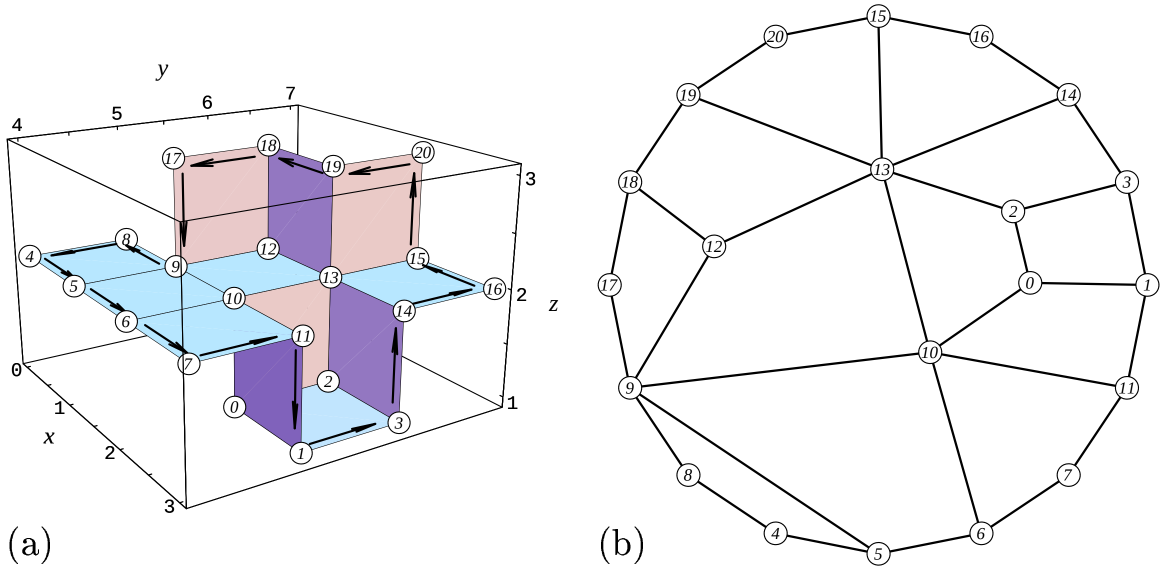
Figure 3.6: The parametrization achieved by the optimization is visualized three different ways. *(a)* The surface net is plotted on the spherical parameter space. Thick lines depict the edges of the original square faces. The equidistance for both θ and ϕ is $\frac{\pi}{8}$. *(b)* Cartesian interpretation of $(\phi, \cos\theta)$ gives an equal-area cylinder projection. The horizontal lines at ±1 are the poles. *(c)* Conversely, the polar coordinate grid is drawn over the object.
For comparison, one vertex is marked with a black dot in all diagrams.

3.5 Parametrization of surface patches

In many applications we are not interested in the closed surface of a whole object but in an open surface. Unfolding the human cerebellar cortex is an active topic of research, simply to name one. An open surface has at least one border. It does not enclose a defined volume. A circular (or square) surface patch has one border. When a hole is cut in such a patch, we get one with two borders; it is topologically the same as the outer curved surface of a cylinder. This last example illustrates that in three dimensions we cannot in general identify a border as an "outer" or "inner" contour: nothing discriminates the top and bottom rim of the cylinder surface. Orientable connected surface patches are treated in the following; this excludes, e.g., the Moebius strip. The set of permissible surfaces is even a bit more restricted; patches that have "handles" are not fit for embedding in $\mathbb{R}^2$. A surface patch can have any number of borders which are all considered equivalent. When the patch is mapped to a planar parameter space, one arbitrarily chosen border will be the outer boundary and the other ones will appear as holes.

The surface data structure is suitable also for the representation of surface patches, as explained above in section 3.3. It retains the notion of an "inner" and "outer" side of the surface. A vertex on the border of the patch is distinguished by its odd number of neighbors. Both the first and the last one in the list of neighbors are direct neighbors, and they are border vertices as well. Several consistency checks are done based on this observation. Figure 3.7a shows a small surface patch with only one border as an example. It has five interior vertices and sixteen border vertices. The following procedure maps the surface patch to the inner of a regular polygon, the corners of which lie on Ω_2.

All vertices are inspected sequentially. As soon as one of them has an odd number of neighbors the search stops and the border containing that vertex is considered the "outer" border. This choice is arbitrary if there is more than one border; having one selected in advance and specified by one of its vertices would be another possibility. The first neighbor in the neighbor list of a boundary vertex is the "previous" vertex, and the last neighbor is the "next" vertex. This convention is indicated by the arrows in Figure 3.7a. It induces a unique sense for cyclically traversing the border(s) of the patch. Starting at the start vertex and always proceeding to the next border vertex, the procedure eventually returns to the start vertex and collects a list of all vertices on the outer boundary. In the example we would start at vertex 1 and collect the boundary

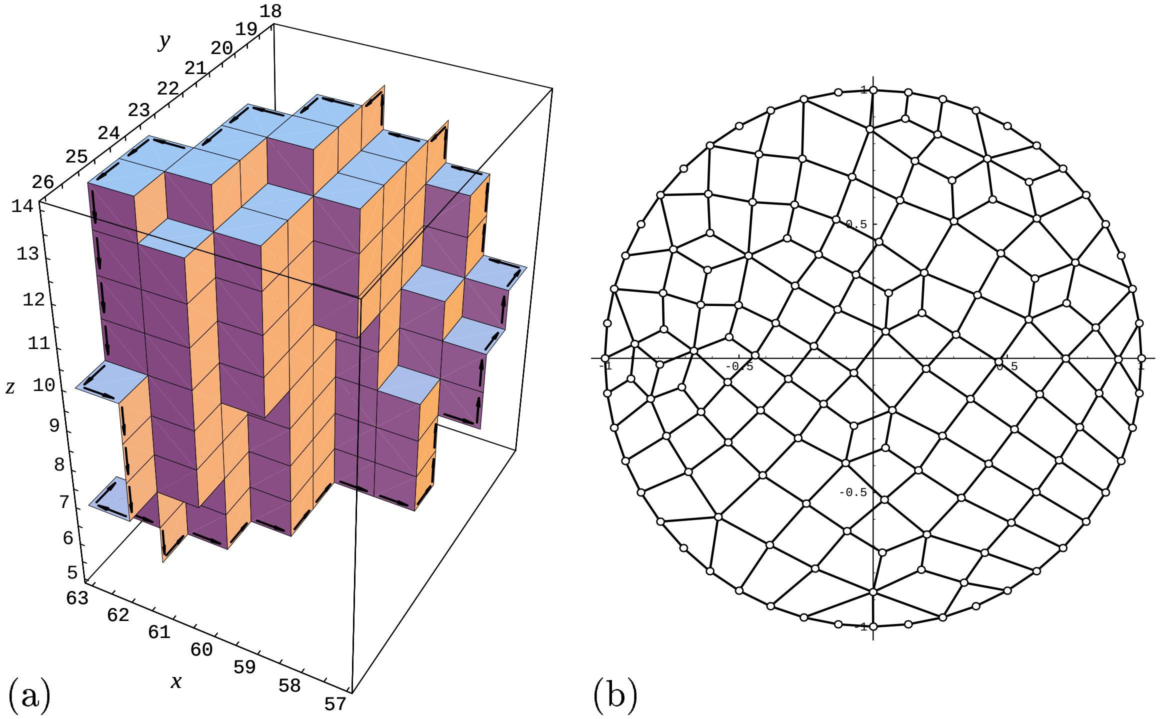


(a) (b)

Figure 3.7: (a) This tiny surface patch consists of 21 vertices, 12 facets + 1 border, and 32 links. The interior vertex 2 has the neighbors {0, 1, 3, 14, 13, 10}. The border vertex 9 has nine neighbors ({8, 4, 5, 6, 10, 13, 12, 18, 17}), an odd number. Its successor is vertex 8, its predecessor 17. Arrows illustrate the previous–next relation. (b) The same patch flattened, i.e. mapped to a circular portion of $I\!R^2$.

cycle $\{1, 3, 14, 16, 15, 20, 19, 18, 17, 9, 8, 4, 5, 6, 7, 11\}$. These vertices take fixed positions at equal distances on Ω_2 in the u_0, u_1 parameter space. If there are *length* vertices on the border, the vertex i gets coordinates $u_0[i] = \cos\frac{2\pi i}{length}$ and $u_1[i] = \sin\frac{2\pi i}{length}$, where $0 \le i < length$. These points form a regular *length*-gon. The u_0 and u_1 parameters for all other vertices follow from these boundary values in the same way as θ does in the initial parametrization of closed surfaces (cf. subsection 4.1.1). Border vertices which are not part of the outer boundary receive no special treatment; they will line inner holes. The parameter of each non-boundary vertex must equal the average of the parameters of the direct neighbors. All conditions form a sparse symmetric system of linear equations, which is solved by PILS. Exactly the same steps – with the respective boundary conditions – independently yield u_0 and u_1. Figure 3.7b presents the result.

Patches from medical objects will serve as less trivial examples for the patch parametrization. Admittedly the origin of the data does not really matter for the patch sizes I cut out for the following illustrations. But



(a) (b)

Figure 3.8: (a) A patch from the surface of the patella that appears in the top left diagram of Figure 5.22. The patch is at the top right end in that projection. (b) The parametrization assigns a unique $\underline{u}$ coordinate pair to every vertex. Small circles indicate the vertices. The flat net does not fold over or otherwise overlap itself.

they show that the method works on natural data. I chose small patches to produce readable diagrams with enough resolution to distinguish individual vertices. Figure 3.8a presents a small patch from the surface of a patella. Figure 3.8b illustrates the parametrization of this patch, i. e. its mapping to the unit circle in the u_0-u_1 plane.

Figure 3.9a introduces the case of a surface patch with more than one border. One would probably consider the longest of the three borders as the outer border and the other two as holes, as indicated in the graphics. This choice is actually arbitrary; when a surface in 3D has more than one border, they all are equivalent. The parametrization works for this surface without modification; Figure 3.9b displays the result. As before, every vertex in the planar embedding of the net in $\mathbb{R}^2$ is the center of gravity of its neighbors, except for the vertices of the chosen exterior border, which are fixed on Ω_2. It follows that the other borders – or the

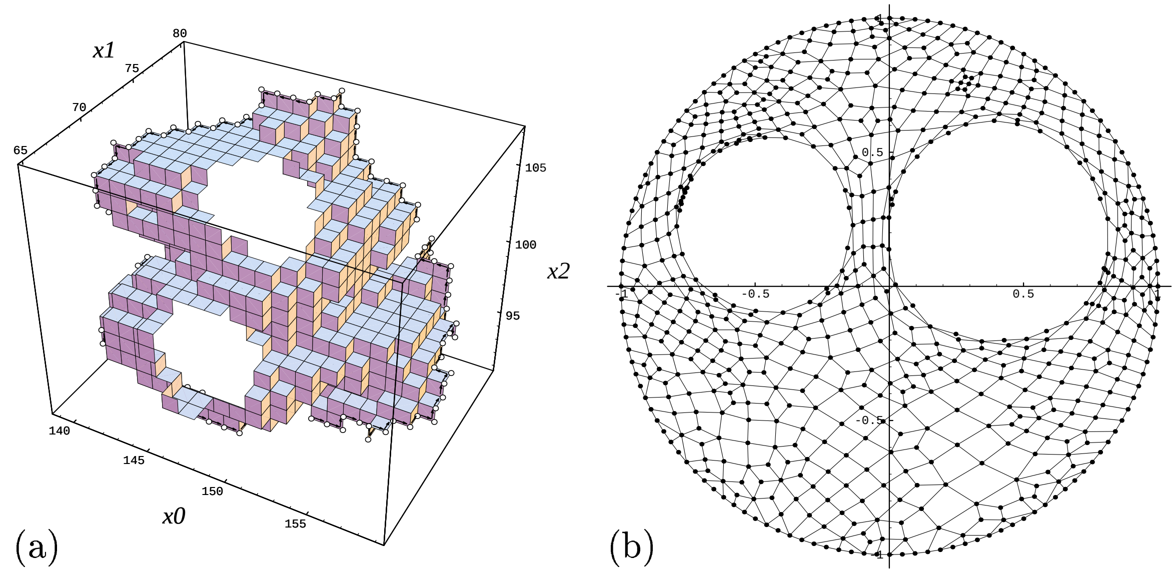


Figure 3.9: (a) A patch with three borders (holes). It is cut out from the surface of a brain dataset. The longest one of the three borders is arbitrarily selected as the “outer” border. White balls mark it, and a chain of arrows traces it. The other two borders are, by consequence, treated as “inner” holes. (b) The same patch parametrized, i.e. mapped to the unit circle. The holes map to convex polygons.

holes – map to convex polygons, just as the ordinary faces, which map to convex quadrilaterals.

Figure 3.10 elaborates on the arbitrary choice of one “outer” boundary. A different boundary was chosen this time, resulting in a different possible parametrization. It apparently suffers from more distortion than the one of Figure 3.9, but it is nevertheless a valid planar embedding of the surface net.

The spreading out of the patch can be useful in several ways. It can be used to present a property that varies over the surface in an overview. The whole patch can thus be presented at one glance. This is of particular interest in investigations of the cerebral cortex. Mental activities are going on in all of the grey cortex, including the sulci and their ramifications, not just the outer parts of the surface we see when watching a whole brain as a solid 3D object from outside. It is generally assumed that the folding of the brain surface was driven by the need for more processing ability and hence cortex area. Most of the new area lies hidden in the sulci and structures like the insula. Parametrization of surface patches

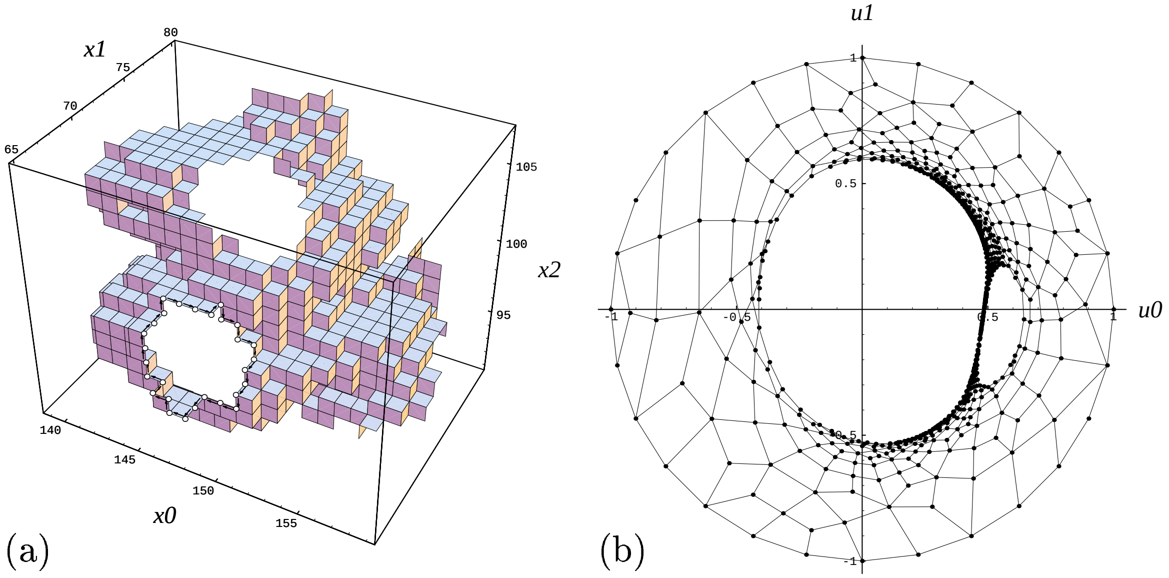


Figure 3.10: (a) The same patch as in Figure 3.9. This time the shortest of the three borders plays the role of the “outer” border. (b) Again, the surface patch maps to a part of $\mathbb{R}^2$, inside the regular 28-gon representing the “outer” border.

can help to create or refine a map of the cortex. Parametrization has the potential to reproduce proximity in the cortex surface, which is related to functional proximity, in contrast to spatial closeness in 3D. One might depict properties like curvature of the surface, deepness of the sulci (to make orientation easier), or activity of the corresponding region of the cortex. As a very simple example, the distance of the corresponding surface element from the observer paints the facets in Figure 3.11.

The mapping also defines the vector-valued function $U \hookrightarrow \mathbb{R}^3 : \ \underline{u} \rightarrow \underline{x}(\underline{u})$ from the unit circle disk (possibly with holes) to object space. The three component functions $x_0(\underline{u})$, $x_1(\underline{u})$, and $x_2(\underline{u})$ are plotted in Figure 3.12. The parametrization of Figure 3.9 was used.

To approximate the data with by a smooth function, we can determine a least-squares fit to the empirical function $\underline{x}(\underline{u})$ as a linear combination of some set of basis functions. The coefficients of the linear combination are vectors of $\mathbb{R}^3$. The aggregate of these vectorial coefficients forms a descriptor for the shape of the surface patch. If the patch has more than one border, the fit extrapolates a value to the regions of $\mathbb{R}^2$ corresponding to the resulting holes: the holes are filled. This happens in Figure 3.13 where a Bézier patch (Eq. 3.1) fits the functions from Figure 3.12 (and

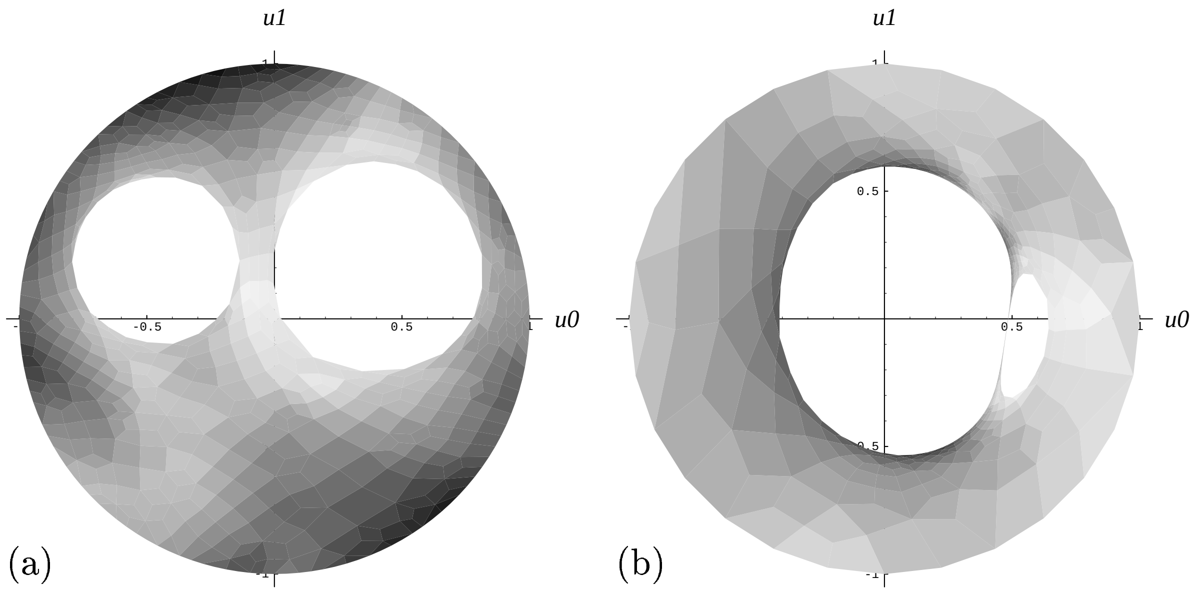


Figure 3.11: Displaying a surface property in parameter space: The grey level represents the distance of the corresponding surface element from the observer. Bright means close and dark means further away. The results with both parametrizations are presented. In (a) the longest border is selected as outer border, in (b) the shortest one.

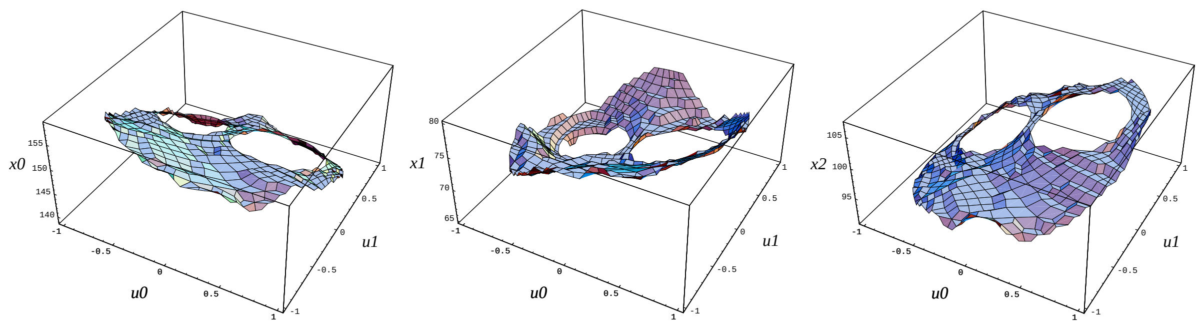
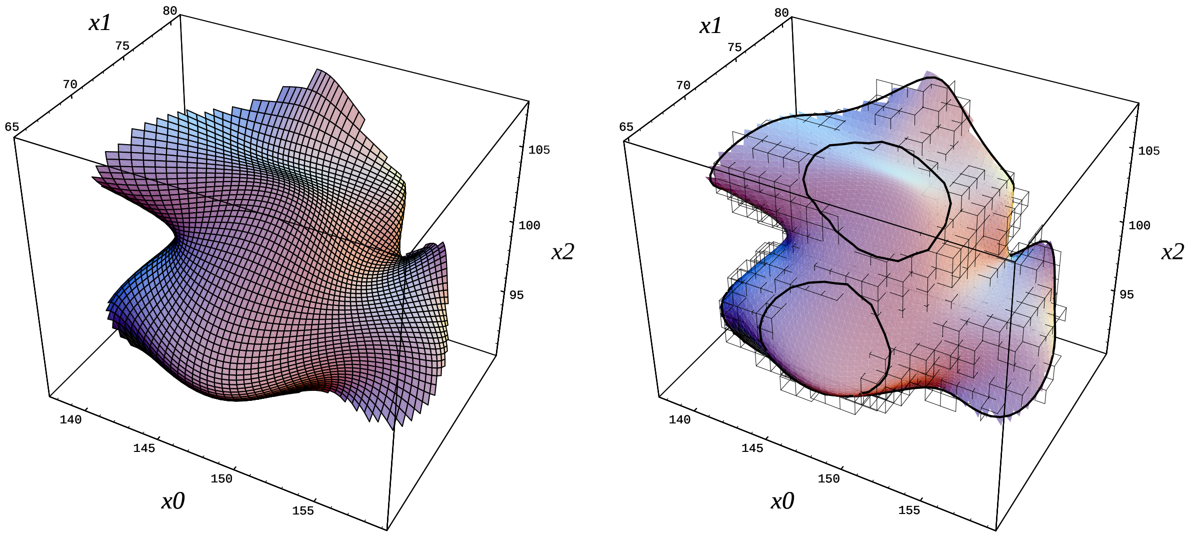


Figure 3.12: The parametrization from Figure 3.9 implies a function $\underline{x} : U \hookrightarrow \mathbb{R}^3$. These three diagrams plot the three components of the function $\underline{x}(\underline{u})$. The function is undefined on the holes.



(a) (b)

Figure 3.13: Reconstruction of the surface from the fit function. (a) $\underline{u}$ varies over the disk $\|\underline{u}\|_2 = 1$ in $I\!R^2$. The parameter grid shows on the surface: it consists of the iso-lines of u_0 and u_1. (b) The parameter grid is suppressed. Instead, the original 3D patch is superimposed as a wireframe. The thick lines result from evaluating the fit function at the parameter values assigned to the borders of the patch.

3.9). The fit uses products of Bernstein polynomials up to degree 5 in u_0 and u_1. We see a parametric plot of the fit function, to the left with the u_0, u_1 coordinate grid, to the right with a superimposed wireframe of the original data and the lines to which the fit function maps the parameter values of the borders.

When the surface patch is topologically sound, i.e. planar, no two points of the surface will map to the same place in the unit circle disk; there can be no overlap. A planar embedding of the surface graph is thus found, but it may have significant distortions. Any protrusion of the surface maps to a small image graph squeezed together in the image of the foot of the protrusion. It is not obvious how the net should be allowed to relax towards a more even state. Stating the task as a minimization problem faces more difficulties than in the case of closed surfaces If we made all facets work towards equal (e.g. unit) area and shape as square-like as possible, some surface nets would tend to self-overlap. Such overlaps can happen between distant parts of the net and are not locally detectable. If they were detected, elaborate schemes for their prevention would need to come into action. Their discussion exceeds the scope of this publication.

Chapter 4

Optimization

The discretization of Laplace's equation on a given surface net and the solution of the corresponding Dirichlet problem constructs an initial parametrization. The differences between closed and open surfaces are limited to the border conditions. This approach is explained in Section 4.1. The optimization itself is a demanding task, due to the size of the problem. Section 4.2 discusses solutions, details of the most successful approach, and strategies that had to be developed to cope with various problems that can arise.

This chapter may be a bit technical in some parts. The later chapters will not rely on the material presented here.

4.1 Initial parametrization

An earlier version of the procedure is described in [8].

The first mapping or parametrization is done in polar coordinates. The two polar coordinates θ and ϕ are determined for all vertices in two separate steps. Two vertices have to be selected as the *poles* for this process. The choice of these poles is not critical, as the subsequent optimization process will remove all its influences except a rotation in parameter space. Selecting two poles which lie close together, however, results in a poor initial parametrization. The optimization will converge to the same solution, but it takes more iteration steps. We select the two vertices with lexically maximal and minimal (x_2, x_1, x_0) coordinates in object space; i.e. x_2 is weighed most and x_0 least. The poles are the first and the last vertex in our numbering of the surface vertices. When n_{vert} denotes the total number of vertices, the north pole has number 0 and the south pole has number $n_{\text{vert}} - 1$.

4.1.1 Co-latitude from diffusion.

Co-latitude θ should grow smoothly from 0 at the north pole to π at the south pole. In this context, θ is not a free variable but rather an unknown function (of the location on the object) that we are looking for. To assign a co-latitude value with the desired property to every node, we formulate the corresponding continuous problem as Laplace's equation $\nabla^2\theta = 0$ (except at the poles), with Dirichlet conditions $\theta(\text{northpole}) = \theta_{\text{north}} = 0$, $\theta(\text{southpole}) = \theta_{\text{south}} = \pi$ for co-latitude θ[31]. A physical analogy is heat conduction: we heat the south pole up to temperature π , cool the north pole to temperature 0 and ask for the stationary temperature distribution on the heat-conducting surface. As usual in the discrete case, the Laplacian is approximated by finite second differences of the available direct neighbors, which in our case implies that every node's co-latitude (except the poles') must equal the average of its neighbors' co-latitudes. These conditions form a sparse set of linear equations, which can be written in the form $A'\underline{\theta}' = \underline{b}'$, where A' is a $n_{\text{vert}} \times n_{\text{vert}}$ matrix, $\underline{\theta}' := (\theta_0, \theta_1 \ldots \theta_{n_{vert}-1})^T$ and $\underline{b}'$ is an n_{vert} vector of constants. The border conditions supply two equations and the average property defines $n' := n_{\text{vert}} - 2$ equations. Applying the border conditions $\theta_0 = \theta_{\text{north}}$ and $\theta_{n_{\text{vert}}-1} = \theta_{\text{south}}$ results in the reduced $n' \times n'$ system $A\underline{\theta} = \underline{b}$, where $A = (a_{1,1}, a_{1,2} \ldots a_{n',n'})$ is symmetric and $\underline{\theta} := (\theta_1 \ldots \theta_{n'})^T$

The algorithms that are used to set up the matrix A and the right hand side vector $\underline{b}$ make use of the well-organized surface data structure, as illustrated below in pseudo-code notation. The sparsity of the matrix is exploited; only the few non-zero entries (4 to 7 per row) are stored. This saving is essential to be able to handle larger objects.

Set up matrix A:

```
for vertex = 1...n
    $a_{vertex,vertex}$:= number of direct neighbors;
    for the direct neighbors of vertex
        if the neighbor is not a pole
            $a_{vertex,neighbor}$:= -1;
```

```
Set up constant vector $b$:
    set all entries of $b$ to 0;
    for the direct neighbors of south_pole
        $b_{neighbor}$ := $\pi$;
```

This sparse symmetric linear system of equations is solved with the Pils package [30]. The solution has the important property that co-latitude θ varies monotonically between the poles, since there can be no local extremum by virtue of the maximum principle [26]. Figure 4.1a shows the resulting θ for the simple test object named "duck".

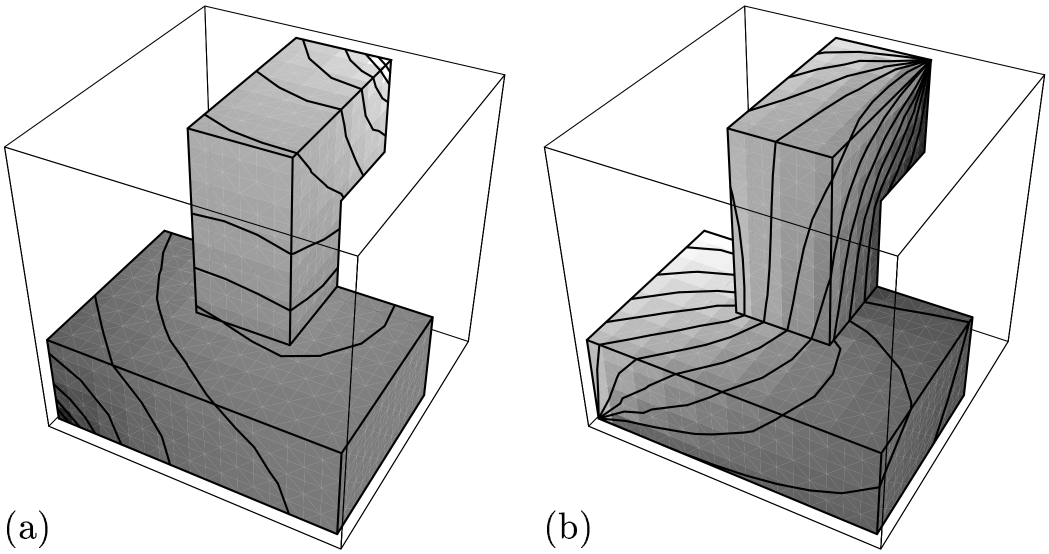


Figure 4.1: The simple object "duck" consisting of nine voxels is used for illustrating the initial parametrization. The north pole is at the lower left, the south pole at the upper right. Co-latitude is mapped on the object's surface as a grey value in *a*; iso-latitude lines are drawn every $\frac{\pi}{16}$. Longitude is shown in *b*; iso-longitude lines ("meridians") are $\frac{\pi}{8}$ apart.

4.1.2 Determining co-latitude in the example

A fully worked example with the "two-voxel" object of Figure 3.2 is presented here. It serves as a concrete and detailed illustration of the general concepts exposed above. The border conditions

$$\theta_0 \quad \equiv \quad \theta_{\text{north}} = 0$$

$$\theta_{11} \equiv \theta_{\text{south}} = \pi$$

and the average properties

$$\begin{aligned}
\theta_1 &= (\theta_0 + \theta_4 + \theta_2 + \theta_7)/4 \\
\theta_2 &= (\theta_1 + \theta_5 + \theta_8)/3 \\
\theta_3 &= (\theta_4 + \theta_0 + \theta_9)/3 \\
\theta_4 &= (\theta_3 + \theta_{10} + \theta_5 + \theta_1)/4 \\
&\vdots \\
\theta_{10} &= (\theta_9 + \theta_7 + \theta_{11} + \theta_4)/4
\end{aligned}$$

can be arranged in matrix notation.

$$\begin{pmatrix}
1 & 0 & & 0 & & & 0 & & & & & \\
-1 & 4 & -1 & & -1 & & & -1 & & & & \\
& -1 & 3 & & & -1 & & & -1 & & & \\
-1 & & & 3 & -1 & & & & & -1 & & \\
& -1 & & -1 & 4 & -1 & & & & & -1 & \\
& & -1 & & -1 & 3 & & & & & & -1 \\
-1 & & & & & & 3 & -1 & & -1 & & \\
& -1 & & & & & -1 & 4 & -1 & & -1 & \\
& & -1 & & & & & -1 & 3 & & & -1 \\
& & & -1 & & & -1 & & & 3 & -1 & \\
& & & & -1 & & & -1 & & -1 & 4 & -1 \\
& & & & & 0 & & & 0 & & 0 & 1
\end{pmatrix}
\begin{pmatrix}
\theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \\ \theta_5 \\ \theta_6 \\ \theta_7 \\ \theta_8 \\ \theta_9 \\ \theta_{10} \\ \theta_{11}
\end{pmatrix} =$$

$$(\theta_{\text{north}}, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, \theta_{\text{south}})^T \qquad (4.1)$$

Eliminating the first and the last row (boundary conditions) leads to the following reduced, symmetric system.

$$\begin{pmatrix}
4 & -1 & & -1 & & & -1 & & & \\
-1 & 3 & & & -1 & & & -1 & & \\
& & 3 & -1 & & & & & -1 & \\
-1 & & -1 & 4 & -1 & & & & & -1 \\
& -1 & & -1 & 3 & & & & & \\
& & & & & 3 & -1 & & -1 & \\
-1 & & & & & -1 & 4 & -1 & & -1 \\
& -1 & & & & & -1 & 3 & & \\
& & -1 & & & -1 & & & 3 & -1 \\
& & & -1 & & & -1 & & -1 & 4
\end{pmatrix}
\begin{pmatrix}
\theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \\ \theta_5 \\ \theta_6 \\ \theta_7 \\ \theta_8 \\ \theta_9 \\ \theta_{10}
\end{pmatrix}
=
\begin{pmatrix}
\theta_{\text{north}} \\ 0 \\ \theta_{\text{north}} \\ 0 \\ \theta_{\text{south}} \\ \theta_{\text{north}} \\ 0 \\ \theta_{\text{south}} \\ 0 \\ \theta_{\text{south}}
\end{pmatrix} \qquad (4.2)$$

$$= (0, 0, 0, 0, \pi, 0, 0, \pi, 0, \pi)^T$$

The solution is $\underline{\theta} = (\frac{2\pi}{5}, \frac{3\pi}{5}, \frac{3\pi}{10}, \frac{\pi}{2}, \frac{7\pi}{10}, \frac{3\pi}{10}, \frac{\pi}{2}, \frac{7\pi}{10}, \frac{2\pi}{5}, \frac{3\pi}{5})^T$, or $\underline{\theta}' = \frac{\pi}{10}(0, 4, 6, 3, 5, 7, 3, 5, 7, 4, 6, 10)^T$, where the first and last value correspond to the poles.

4.1.3 Longitude from diffusion.

Unlike latitude, longitude is a cyclic parameter: When we walk around a sphere counterclockwise (seen from the north), longitude keeps increasing monotonically all the time, but there must be a place where longitude leaps back by 2π. A global longitude parameter always has a discontinuous line running from pole to pole, and the step height is 2π. Consider, as an analogy, local time on every spot of the globe; the date line is a 24 hour discontinuity, but not a meridian. In our problem, the choice of the date line is immaterial: it just has to connect the two poles. The date line is chosen as a path with steepest co-latitude ascent in each of its nodes. The values crossing the date line from west to east are decremented by 2π, values propagated to the west are incremented by 2π. The poles and all links to them are removed from the net, making the topology of the net that of a tube. Longitude remains undefined for the poles. The cyclic Laplace equation $\nabla^2\phi = 0$ (with date line) again corresponds to a system of linear equations in the discrete case.

The new system of linear equations is structurally identical to the one for co-latitude. Typically, only a small part of the equations is different. The matrix for longitude ϕ differs from the one for co-latitude θ only by the values of six diagonal entries, corresponding to the three neighbors of each pole. This similarity simplifies setting up and solving the new system.

Due to the cyclic boundary conditions, the solution ϕ is defined only up to an additive constant. The linear equations are dependent, and the system is singular. To make it regular, we have to specify the longitude of one vertex. We set $2phi\_1 = 0$ arbitrarily. This equation can be added to any row of the system. We add it to the first row.

The following portions of pseudo-code update the matrix and generate a new right hand side vector.

Modify matrix A:

```
for both poles
    for the direct neighbors of pole        cut link to pole
        $a_{neighbor,neighbor}$ -= 1;
$a_{0,0}$ += 2;                              add $\phi_1 = 0$
```

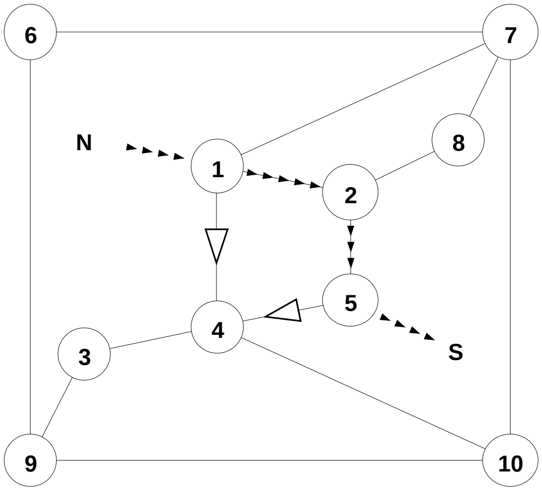
Set up constant vector b:

```
for row := 1...n do
    $b_{row}$ := 0;
previous := north pole;
here := 1;                                    any nbr of north pole
maximum := 0.0;
while (here != south pole)                    walk on date-line
    for the direct neighbors of here
        if $\theta_{neighbor}$ > maximum then
            maximum := $\theta_{neighbor}$;
            nextpos := position of neighbor;
        if neighbor == previous then
            prevpos := position of neighbor;
    for all direct neighbors $\in \{prevpos \overset{\cdots}{\text{clockwise}} nextpos\}$ do
        add $2\pi$ to $b_{neighbor}$;
        subtract $2\pi$ from $b_{here}$;
    previous := here;
    here := neighbor of here indicated by nextpos;
```

In spherical coordinates, longitude is undefined at the poles. We arbitrarily set $\phi_{\text{north}} = \phi_{\text{south}} = 0$. Figure 4.1b illustrates the resulting ϕ for the "duck" test object.

4.1.4 Determining longitude in the example

The path 0, 1, 2, 5, 11 is used as the date line; it is indicated by a row of small black triangles. Links extending from the date line to the west are 1 → 4 and 5 → 4; they are marked with white triangles. The poles, "N" (vertex 0) and "S" (vertex 11), are no longer part of the net. The following equations result.



$$\begin{aligned} 3\phi_1 &= (\phi_4 - 2\pi) + \phi_2 + \phi_7 \\ 3\phi_2 &= \phi_1 + \phi_5 + \phi_8 \end{aligned}$$

$$\begin{aligned} 2\phi_3 &= \phi_4 + \phi_9 \\ 4\phi_4 &= \phi_3 + \phi_{10} + (\phi_5 + 2\pi) + (\phi_1 + 2\pi) \\ &\vdots \\ 3\phi_{10} &= \phi_9 + \phi_7 + \phi_4 \end{aligned}$$

These equations, together with $2\phi_1 = 0$, can be put into matrix notation as follows.

$$\begin{pmatrix} 5 & -1 & & -1 & & & -1 & & & \\ -1 & 3 & & & -1 & & & -1 & & \\ & & 2 & -1 & & & & & -1 & \\ -1 & & -1 & 4 & -1 & & & & & -1 \\ & -1 & & -1 & 2 & & & & & \\ & & & & & 2 & -1 & & -1 & \\ -1 & & & & & -1 & 4 & -1 & & -1 \\ & -1 & & & & & -1 & 2 & & \\ & & -1 & & & -1 & & & 3 & -1 \\ & & & -1 & & & -1 & & -1 & 3 \end{pmatrix} \begin{pmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \phi_4 \\ \phi_5 \\ \phi_6 \\ \phi_7 \\ \phi_8 \\ \phi_9 \\ \phi_{10} \end{pmatrix} = \begin{pmatrix} -2\pi \\ 0 \\ 0 \\ 4\pi \\ -2\pi \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad (4.3)$$

The solution is $\underline{\phi} = (0, 0, \frac{5\pi}{4}, \frac{3\pi}{2}, \frac{-\pi}{4}, \frac{3\pi}{4}, \frac{\pi}{2}, \frac{\pi}{4}, \pi, \pi)^T$

For every node of the net, we now have computed a co-latitude θ and a longitude ϕ. This defines a continuous, unique mapping from the surface of the original object to the surface of a sphere, which is illustrated by Figure 4.2. Instead of specifying a location on Ω_3, the spherical parameters ϕ and θ can also be used in a flat Cartesian coordinate system as shown in Figure 4.2b, which gives an overview of the whole unfolded net. The right border wraps around to the left border, and vice versa. The top and bottom borders stand for the poles. The θ axis points up to produce the same orientation as in the other figures; the south pole is at the top. The polar coordinates are transformed to 3D Cartesian by (7.4(appendix)) and yield starting values for the optimization.

4.2 Optimization methods

Powerful methods for nonlinear constrained minimization are known[16]. But the commonly available optimization routines can not be used for larger objects because they are not suited for such a large problem, as they do not take advantage of its sparsity and information available about

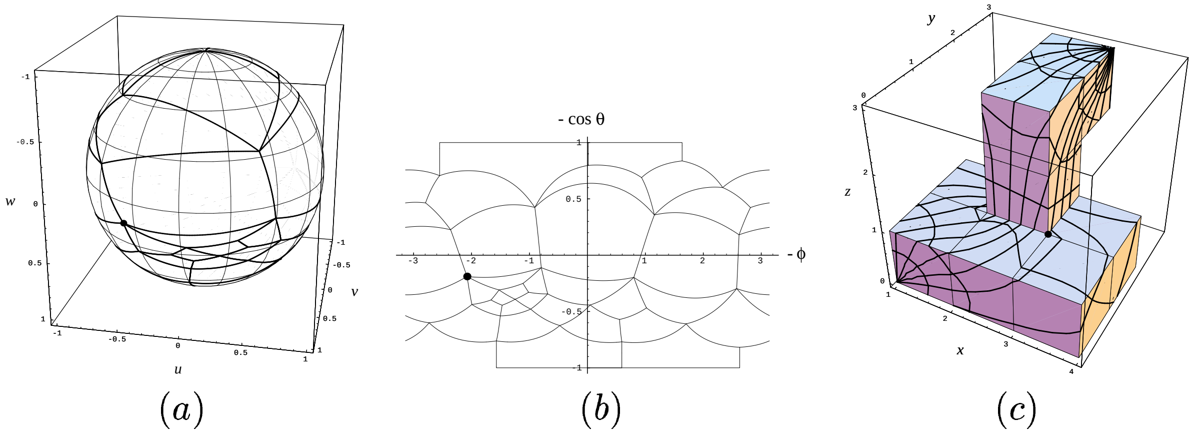


Figure 4.2: Diffusion yields the initial parametrization, which is plotted in the same three ways as the final result in Figurefig:opt. *(a)* The surface net is plotted on the spherical parameter space. The thick lines depict the edges of the original square faces. The equidistance for both θ and ϕ is $\frac{\pi}{8}$. *(b)* ϕ and $\cos\theta$ are interpreted as Cartesian coordinates. The monotonic cosine function is applied to give a true-area cylindrical projection. The horizontal lines at ±1 are the poles. *(c)* Conversely, the globe coordinate grid is drawn over the object. For comparison, one vertex is marked with a black dot in all diagrams.

the constraints. I had to develop and implement my own optimization algorithms. They are a key part of surface parametrization, and receive some more detailed discussion here.

The sparse linear solver package Pils[30] is used for solving the linear systems of equations that arise inside a minimization algorithm. I have implemented two algorithms. One is a Lagrange-Newton method[12], also known as sequential quadratic programming (SQP). The other one is based on Conjugate Gradients [31] for finding a minimum and the Newton (or Newton-Raphson) scheme for satisfying the constraints. This second method has been more successful in solving the kind of optimization problems that arise from surface parametrization.

Both algorithms use a form of the Newton scheme for finding a zero x^* of a nonlinear function $f(x)$: $f(x^*) = 0$. The Newton scheme follows from the Taylor series of order 1 of f about x:

```
Repeat
    $x := x - f'(x)^{-1} \cdot f(x)$
Until convergence
```

An initial x must be supplied by guessing. When x and $f(x)$ are scalars, f' is the derivative of f. But $\underline{x}$ and $\underline{f}$ may also be vectors of the same dimension n. In this case $\underline{f}'$ is the $n \times n$ Jacobian matrix $\nabla \cdot \underline{f}^T$ of $\underline{f}$.

As the goal function, the constraints, and their derivatives have to be calculated repeatedly, it is important that we can use an efficient data structure that holds information about adjacency (for the goal function and matrix sparsity) and surface facets (for the area constraints). The solution of the nonlinear program defines the optimal parametrization of our object's surface.

4.2.1 Notation

Let $\underline{x}$ be the variables of the optimization, in our case a vector of length $3n_{\text{vert}}$. The goal function $f(\underline{x})$ is scalar. And $\underline{c}(\underline{x})$ denotes the vector of constraints; its length is not always the same; see below. The general problem is

$$\text{NEP:} \quad \underset{\underline{x} \in \mathbb{R}^{3n_{\text{vert}}}}{\text{minimize}} \quad f(\underline{x}) \qquad \text{subject to} \quad \hat{\underline{c}}(\underline{x}) = \underline{0} \ ,$$

and the solution is denoted as $\underline{x}^*$.

The corresponding Lagrangian function is defined as

$$\mathcal{L}(\underline{x}, \underline{\lambda}) = f(\underline{x}) - \underline{\lambda}^T \, \underline{c}(\underline{x}) \ .$$

The augmented Lagrangian function includes a penalty term, the sum of the squares of the active and equality constraints multiplied by a penalty factor ρ.

$$\mathcal{L}_A(\underline{x}, \underline{\lambda}, \rho) = f(\underline{x}) - \underline{\lambda}^T \, \underline{c}(\underline{x}) + \rho \underline{c}(\underline{x})^T \cdot \underline{c}(\underline{x})$$

At the solution $(\underline{x}^*, \underline{\lambda}^*)$, $\mathcal{L}(\underline{x}, \underline{\lambda})$ is stationary with respect to $\underline{x}$ and $\underline{\lambda}$.

Let ∇ be the gradient with respect to $\underline{x}$ only, and $\bar{\nabla}$ the gradient with respect to both $\underline{x}$ and $\underline{\lambda}$. We define the objective gradient $\underline{g} = \nabla f(\underline{x})$ as well as $A = \nabla \cdot \underline{c}^T(\underline{x})$, the Jacobian of $\underline{c}$, which is the matrix of constraint normals. At the solution no component of the gradient can be tangent to the constraints, i.e. the gradient is a linear combination of the constraint normals: $\underline{g} + A\underline{\lambda} = \underline{0}$. Let $\underline{g}_Z = \underline{g} - A\underline{\lambda}$ be the gradient projected to the tangent subspace.

4.2.2 A Lagrange-Newton algorithm

The idea of this optimization method is presented in [12], so the following is a brief summary. The condition that $(\underline{x}^*, \underline{\lambda}^*)$ is a stationary point of

$\mathcal{L}(\underline{x}, \underline{\lambda})$ can be written in the form $\bar{\nabla}\mathcal{L}(\underline{x}, \underline{\lambda}) = \underline{0}$. Finding a zero of these equations with the Newton method leads to the updates $\underline{\delta x}$ and $\underline{\delta \lambda}$, defined by the following equation, where the matrix on the left is the Hessian of $\mathcal{L}$,

$$\bar{\nabla}\bar{\nabla}^T\mathcal{L}\begin{pmatrix}\underline{\delta x}\\ \underline{\delta\lambda}\end{pmatrix} = -\bar{\nabla}\mathcal{L}\,.$$

This can be transformed to the equivalent system

$$\begin{pmatrix} W & -A \\ -A^T & 0 \end{pmatrix}\begin{pmatrix}\underline{\delta}\\ \underline{\lambda}\end{pmatrix} = \begin{pmatrix}-\underline{g}\\ \underline{\hat{c}}\end{pmatrix}, \tag{4.4}$$

where

$$W = \nabla\nabla^T f(\underline{x}) - \sum_i \lambda_i \nabla\nabla^T \hat{c}_i(\underline{x})$$

is the Hessian $\nabla\nabla^T\mathcal{L}$. Now $\underline{\delta}$ is the $\underline{x}$ increment, and $\underline{\lambda}$ is the new Lagrange multiplier estimate.

There are n_{vert} vector length constraints, keeping all vertices in Ω_3, and $n_{\text{face}} - 1$ area constraints, so $\underline{c}(\underline{x})$ has length $n_{\text{vert}} + n_{\text{face}} - 1$. The matrix in (4.4) is square of dimension $3n_{\text{vert}} + n_{\text{vert}} + n_{\text{face}} - 1 = 5n_{\text{face}} - 3$, symmetric, and sparse.

The algorithm in its simplest form accepts an initial guess for $\underline{x}$ and $\underline{\lambda}$. Then it repeatedly solves the system (4.4) and updates $\underline{x} := \underline{x} + \underline{\delta}$, until some convergence criterion is satisfied.

I use the sparse linear solver Pils[30] to exploit the sparsity of the system. But this solver cannot work if any element of the diagonal is zero. Many such elements exist in the lower right of the matrix by definition. Therefore columns of the matrix have to be permuted to put only non-zeroes on the diagonal. The inverse of this permutation must be applied to the result vector after solving to get the true result. An ad hoc strategy decides which columns should change places; it uses specific knowledge about the sparsity structure of the matrix, which is influenced by the connectivity structure of the object surface net.

Without preventive measures, the iteration becomes unstable, with rapidly diverging $\underline{\lambda}$ and the constraints being violated ever worse. Two measures can often overcome the problem. When $\|\underline{g}_Z\|^2$ exceeds a certain threshold, the current estimate $\underline{\lambda}$ is considered unreliable, and the least squares solution of the overdetermined system $A.\underline{\lambda} = -\underline{g}$ is a better estimate for $\underline{\lambda}$. The threshold 10^{16} turns out to work reasonably well. To make sure the constraints will eventually be satisfied, occasionally an iteration step only tries to satisfy the constraints better. Doing this every 8 iterations turned out to work best.

4.2.3 Constrained optimization in orthogonal spaces

Two things must happen concurrently; the constraints have to be satisfied, and the goal function needs to be minimized. The first means solving a system of nonlinear equations. This system is typically underdefined, i.e. there are more variables than equations. The remaining degrees of freedom are all used for picking among the many solutions one where the goal function is minimal. Linearizing the constraints yields a first order Taylor series: $\underline{c}(\underline{x} + \underline{dx}) \approx \underline{c}(\underline{x}) + A^T \cdot \underline{dx}$. It reveals that the two processes operate in two locally orthogonal subspaces, in the column space of A and in its orthogonal complement, the corresponding null space of vectors orthogonal to the columns of A. The matrix A is not fixed over all iterations, it rather depends on $\underline{x}$, and so do the two vector spaces. The linearization is only valid in a neighborhood of $\underline{x}$.

The Newton method can solve a nonlinear system of equations; it moves in the column space of A. The Conjugate Gradient scheme is effective for minimization; it works in the null space of A. Both methods merge to a constrained optimization algorithm. The combined method performs a single step of either method alternatingly and converges towards the solution.

Three types of constraints come into play, equality constraints on vector lengths and areas, and inequality constraints on angles (see 3.4).

Vector lengths are handled separately by continuous projection. The need for a vertex to remain in Ω_3 is not explicitly represented as a constraint but is silently enforced in any changes to $\underline{x}$. Whenever a displacement from $\underline{x}$ to $\underline{x} + \underline{dx}$ is requested, the new position of every vertex is projected onto Ω_3; each triple $\underline{t}$ of coordinates from the new $\underline{x}$ is divided by $||\underline{t}||_2$, i.e. normalized.

All faces must have equal area. The surface of Ω_3, which is 4π, must be equally partitioned, and each face gets $\frac{4\pi}{n_{\text{face}}}$. Imposing this area for all faces would be redundant; the areas of the faces always sum to 4π. It suffices to specify the area of all faces but one, and this last one will get the remaining area, also $\frac{4\pi}{n_{\text{face}}}$.

To determine the area of a spherical quadrilateral $Q : ABCD$, we may pretend to cut it in two triangles $T_1 : ABD$ and $T_2 : BCD$. The angle at B is split, $\beta = \angle ABC = \beta_1 + \beta_2 = \angle ABD + \angle DBC$, and so is $\delta = \delta_1 + \delta_2$. The area of a spherical triangle on Ω_3 equals its spherical excess, the quantity by which the sum of its three angles exceeds π (the angle sum in a planar triangle). The area of T_1 is $A_1 = \alpha + \beta_1 + \delta_1 - \pi$; for T_2 we have $A_2 = \beta_2 + \gamma + \delta_2 - \pi$. The diagonal BD joins T_1 and

T_2 to Q without overlap or gap, and Q has the area $A_Q = A_1 + A_2 = \alpha + \beta_1 + \delta_1 - \pi + \beta_2 + \gamma + \delta_2 - \pi = \alpha + \beta + \gamma + \delta - 2\pi$.

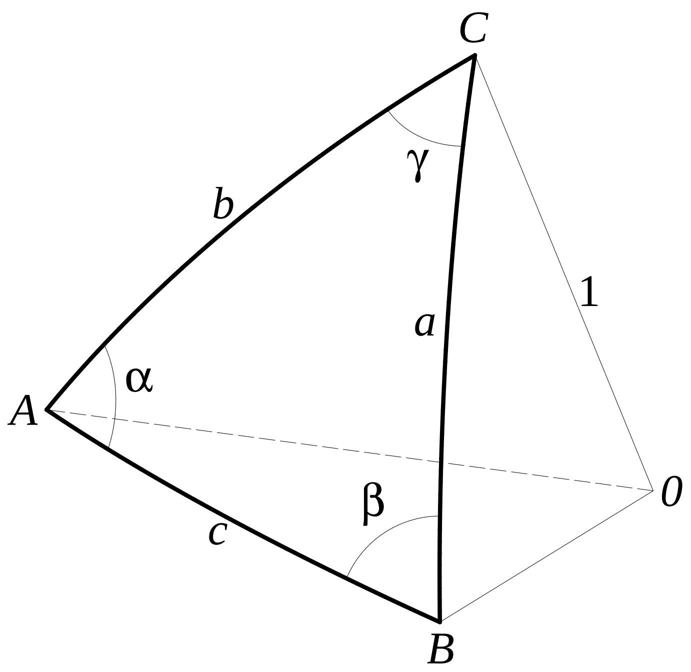


Figure 4.3: A spherical triangle, and how I label its parts. The three vertices A, B, and C are at unit distance from O.

For calculating one specific angle it is enough to consider a triangle, ABC (see Fig. 4.3). The name of a vertex shall denote at the same time the vector pointing from O – the origin and the center of Ω_3 – to that vertex. The side of a spherical triangle is the same as the corresponding center angle, i.e. $a \equiv \angle BOC, b \equiv \angle COA, c \equiv \angle AOB$. We use the law of cosines of spherical trigonometry.

$$\begin{aligned} \cos a &= \cos b \cos c + \sin b \sin c \cos \alpha && (4.5) \\ x_\alpha \overset{\text{def}}{=} \cos \alpha \sin b \sin c &= \cos a - \cos b \cos c && (4.6) \\ &= B^T C - A^T C\ A^T B \end{aligned}$$

Another relation stems from the proof of the law of sines[15].

$$y_\alpha \overset{\text{def}}{=} \sin \alpha \sin b \sin c = A^T (B \times C) = (A, B, C) \qquad (4.7)$$

Neither $\sin b$ nor $\sin c$ can be negative, because a distance on Ω_3 is never negative and it can never exceed π. In a non-degenerate case, where both $\sin b$ and $\sin c$ are positive, x_α and y_α determine α uniquely, e.g. with the

C math library function atan2(y_α, x_α). In our application no angle of any quadrilateral must become negative or exceed π. This claim leads to one inequality constraint $y_\alpha \geq 0$, for each angle α.

An inequality constraint may be active or inactive at a specific point, and this depends on whether violating the constraint is an issue. Inequalities may switch from inactive to active or back a number of times in the course of the optimization. The constraints that influence the direction in which the optimization proceeds at any time are the equalities and the active inequalities.

The Newton method, as explained above, takes a step towards a solution of $\underline{\hat{c}} = \underline{0}$, where $\underline{\hat{c}}$ comprises all equalities and the active inequalities. The function $\underline{\hat{c}}(\underline{x})$ takes the place of f in the generic method, and its Jacobian A corresponds to f'. We assume the matrix A has full rank. The system $A^T(\underline{x})\underline{dx} = -\underline{c}(\underline{x})$ has many solutions, and we need only one of them. A pertinent choice is to take the shortest $\underline{dx}$; this choice restricts $\underline{dx}$ to the column space of A. Any component of $\underline{dx}$ orthogonal to this space would not change the validity of the solution, but make it longer. $\underline{dx}$ can then be written as $\underline{dx} = A\underline{dy}$, and $\underline{dy}$ is determined from the resulting square symmetric positive definite system $A^T\underline{dx} = A^TA\underline{dy} = -\underline{c}(\underline{x})$. After tentatively moving (which includes projecting) by the step $\underline{dx}$ suggested by the Newton method, all constraints are re-evaluated. If any of the inactive inequalities was violated, or the violation of any equality or active inequality increased more than a small threshold value, then the length of $\underline{dx}$ is halved and the move is repeated from the previous position with this shorter step size until an acceptable step is found. Of course, the improvement on $\underline{\hat{c}} = \underline{0}$ is poorer in this case.

Finite differences estimate all the derivatives for the constraint Jacobian A. The continuous projection takes place when each component of $\underline{x}$, one at a time, moves a little bit by δ, where the relevant constraints are re-evaluated, and then the component assumes its previous value again. The difference in the constraint values, divided by δ, yields an estimate for the derivative. As motions out of Ω_3 are silently suppressed, each triple of any estimated constraint normal is orthogonal to the corresponding triple of $\underline{x}$. (And thus all constraint normals are orthogonal to $\underline{x}$.) The gradients of the constraint functions can be derived analytically. This has been done in the previously sketched method, but it leads to very complicated expressions, that fill several pages of source code and are expensive to evaluate.

The objective function is simply a quadratic form, and as such has a straightforward analytic gradient.

$$f(\underline{x}) \quad = \quad \sum_{\underline{v}} \sum_{\underline{n} \in \mathrm{nbr}(v)} \underline{v}^T \cdot \underline{n} \tag{4.8}$$

$$\nabla_{\underline{v}} f(\underline{x}) \quad = \quad \sum_{\underline{n} \in \mathrm{nbr}(v)} \underline{n}\,, \tag{4.9}$$

where $\underline{v}$ runs over all vertices, i.e., all triples of coordinates in $\underline{x}$, and neighbors(v) is the set of neighbors of vertex v. Each triple $\underline{t_g}$ of this gradient is again projected into the subspace orthogonal to the corresponding triple $\underline{t_x}$ of $\underline{x}$, as in $\underline{t_g} - = \underline{t_x} \cdot \underline{t_x}^T \cdot \underline{t_g}$.

Projecting $\underline{g}$ to $\underline{g}_Z$ in the null space of A involves the estimation of the Lagrange multipliers. The objective gradient $\underline{g}$ is explained as well as possible as a linear combination of constraint normals. To this end, we solve the overdetermined system $A\underline{\lambda} = \underline{g}$ in the least squares sense. The solution is defined by $A^T A \underline{\lambda} = A^T \underline{g}$; it minimizes the Euclidean norm $\|\underline{g}_Z\|_2$ of the residual $\underline{g}_Z = \underline{g} - A\underline{\lambda}$. The sign of a Lagrangian multiplier λ_i is an important indicator of the role the corresponding constraint is playing, when it is an inequality. When the multiplier is positive, the objective gradient would tend to move $\underline{x}$ into the “forbidden” region and positive “force” is being applied to stop it from doing so. On the other hand, a negative multiplier λ_i indicates that $\underline{x}$ would move into the allowed region $c_i \geq 0$ if we released the constraint, that keeps c_i close to 0. If the corresponding constraint i is not violated too badly, i.e. if the current value $c_i(\underline{x})$ is above some slightly negative threshold, it is inactivated at this point. When any constraint is inactivated, that iteration terminates prematurely and the conjugate gradient step is not carried out.

The conjugate gradient method constructs a new search direction in a multidimensional minimization problem from the current gradient and from the information from all previous gradients, which is condensed in the previous search direction. It does not require the storage of a matrix and will find the minimum of a quadratic form in a finite number of steps (assuming exact mathematics). It is described in [35], in [31] (pages 316 ff.), in [16] (pages 144 ff.), in [30] (pages 29 ff.) and in [20]. Usually it is applied to the case of unconstrained optimization. I adapt it to the constrained case by supplying the projected gradient $\underline{g}_Z$ as the new gradient (in the subspace) in every iteration. Projecting the old search direction into the zero space of the current constraint gradients might improve the performance of the optimization. This has not yet been tested out.

The vector $\underline{p}$ that the conjugate gradient scheme supplies defines the direction in which the new line search is to proceed. The line search tries to find on the straight line $\underline{x}+\alpha\underline{p}$ the point that minimizes the augmented Lagrangian function $\mathcal{L}_A(\alpha) = \mathcal{L}_A(\underline{x} + \alpha\underline{p}, \underline{\lambda}, \rho)$. The scalar α is the step length. The minimum point must not violate any inactive inequalities. A binary search finds the minimum. It starts at $\alpha = 0$ and initializes the step size s to the final step size of the previous search, multiplied by the constant factor $step_{\text{large}}$, which is larger than 1, e.g. 1.6. The initial figure of merit is $\mathcal{L}(\alpha = 0)$. Repeatedly, the left figure of merit $\mathcal{L}(\alpha - step)$ and the right one $\mathcal{L}(\alpha + step)$ are evaluated. Among the left, middle and central position the one with the minimal figure of merit, but not violating any inactive constraint, becomes the new α. At the central position α none of the inactive constraints is violated, by construction. If a negative inactive constraint $c_i < 0$ prevents the selection of an outer position, it is flagged as stopping progress, until a deliberate (smaller) step in the opposite direction is taken. After the step (or staying in place) the step size is halved. This is repeated until the step size falls below the tolerance *line_tol* and the minimum of $\mathcal{L}_A$ or the zero $c_i = 0$ is defined with sufficient accuracy. An invariant during line search is that the final *alpha* can reach a point in the interval $[\alpha - 2s..\alpha + 2s]$, narrowed at most by *line_tol*. As a safeguard against shrinking the step size too much, the retained copy for the next iteration may not be reduced by more than the factor $step_{\text{small}}$, which is smaller than 1, e.g. 0.08. If progress of the line minimization was stopped by an inactive inequality becoming negative, then this inequality is activated, i.e. added to the active set.

The penalty factor ρ in $\mathcal{L}_A$ ideally should adapt itself according to the success of satisfying $\underline{c}(\underline{x}) = \underline{0}$. The following scheme turned out to work satisfactorily, but is certainly not optimal. Initially ρ has the value ρ_0, and after each line search it is updated if the current point $\underline{x}$ changed (if $\alpha \neq 0$). Define *badness* as the maximum increase in the square of any active or equality constraint c_i during line search (a different definition of *badness* may be superior, e.g. based on $\|\underline{c}\|_2^2$). Constraints are not considered seriously violated when the violation is less than *c_tol*. The constant ρ_{limit} stops ρ from going towards zero; ρ will always stay above this limit. Then ρ is replaced by

$$\rho \cdot \left(\frac{\mathit{c\_tol} \cdot c_{1\rho}}{\mathit{c\_tol} \cdot c_{0\rho} - \mathit{badness}} + c_{2\rho}\right) + \rho_{\text{limit}} \quad .$$

During the typical optimization, ρ is decreasing most of the time, first faster, then asymptotically approaching $\rho_{limit}/(1 - c_{1\rho}/c_{0\rho} - c_{2\rho})$.

At a solution of the constrained minimization problem, all constraints are satisfied, and there is no descent direction in the subspace tangent to the constraints, i.e. the goal function is stationary. These are two necessary conditions for a minimum: $\underline{c}(\underline{x}^*) = \underline{0}$ and $\underline{g}_Z(\underline{x}) = \underline{0}$. In practical experiments, $\|\underline{c}\|$ goes towards zero approximately proportional to the square of $\|\underline{g}_Z\|$. Thus $cost \overset{\text{def}}{=} \|\underline{c}\|^2 + \|\underline{g}_Z\|$ is a valid measure of the overall progress of the optimization. When *cost* drops below a fixed threshold, like 10^{-6} or 10^{-10}, the optimization is assumed to have terminated successfully. Obviously a reasonable scaling of the problem functions is important for the validity of this termination criterion.

Influences of parameters on performance

A number of constants, or parameters, controls the optimization. A good set of parameters will work robustly on a wide range of problems. The constants were tuned so that several different objects are parametrized effectively. To judge the performance by means of comparable figures, one further object (see Figure 4.4) as parametrized with the controlling constants found before as “typical” values. The nonnegative *cost* captures the achievements of the optimization. Figure 4.5 shows the decrease of *cost* in the course of the optimization with the typical parameters. The optimization takes a few iterations to get started. This includes the identification and activation of the constraining inequalities. One inequality becomes active in the first iteration and inactive again in the third iteration. The fourth iteration activates a different constraint, which stays active until the fifteenth iteration. An iteration that inactivates one or more constraints does not define a *cost*, therefore a gap results in the graph. After this initial phase, *cost* decreases steadily to a level of about 10^{-10}. A look at the individual data reveals that $\|\underline{g}_Z\|$ and $\|\underline{\hat{c}}\|$ decrease in parallel and equally contribute to the progress. The progress of the optimization stagnates after about 60 iterations, apparently because of the limited precision of the calculations. Switching to symmetric differences for the estimation of gradients might defer the problem and allow a little further progress. For our object, no inequality is active at the solution.

A quantitative measure of the performance is the value of *cost* after a fixed number of iterations. Allowing a fixed amount of CPU time might be more adequate, but is harder to control in a multi tasking environment.

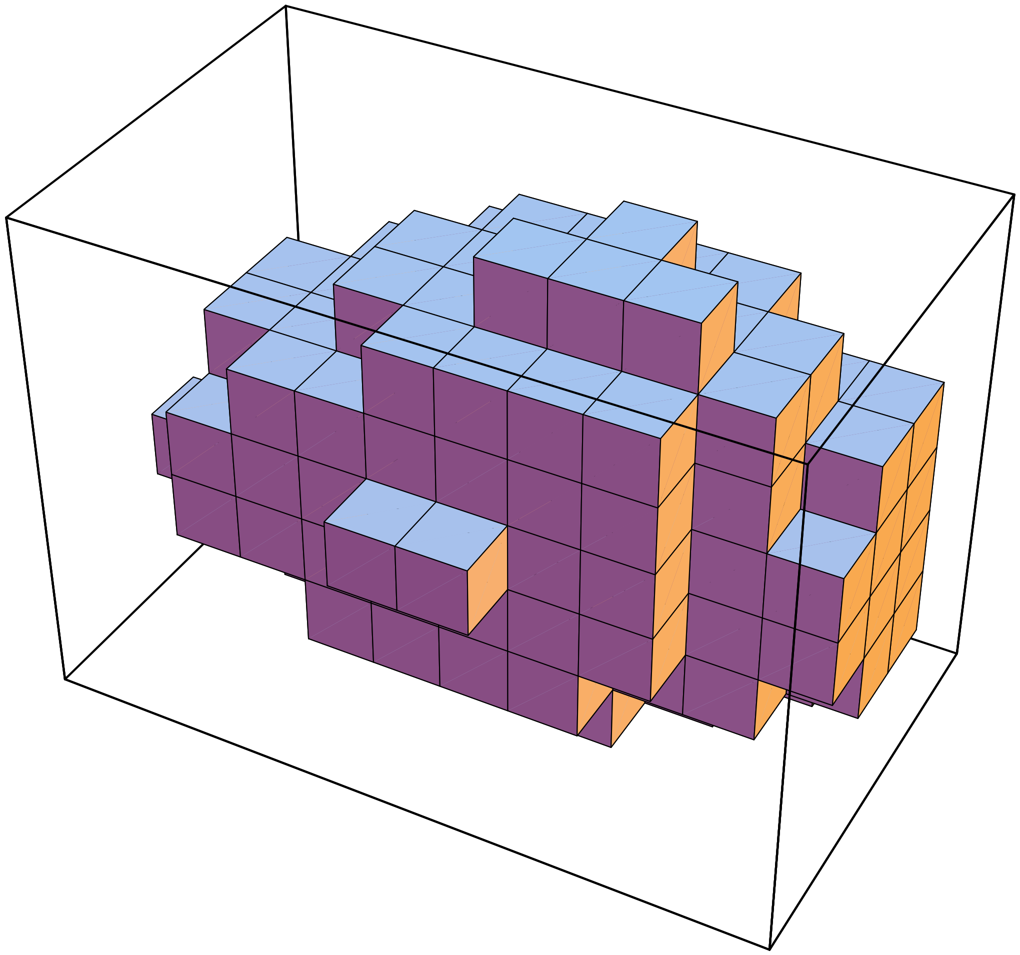


Figure 4.4: The object used in the evaluation of the optimization parameters. In the segmented CT data set of a human knee, the patella was selected. To reduce the computation load, the data was subsampled by a factor of 3 in all three dimensions. This blob-like object is the result.

As the diagram (Fig. 4.5) indicates, 40 iterations were allowed for each optimization. The optimization is stopped in the middle of the effective phase, to get the most expressive figures. In the following diagrams, the final *cost* after forty iterations is plotted against the one parameter being varied; the other parameters are kept fixed. It is desirable to attain a small value for *cost* as fast as possible. The parameter value we pick should lead to a low *cost* (after 40 iterations) for efficiency and it should lie in a flat region for robustness, so that a change in the problem won't spoil the efficiency.

If the finite difference step δ (cf. Fig. 4.6 left) is too large, the secant is not a good approximation to the tangent, because higher order terms disturb the estimation of the derivative. On the other hand a step that

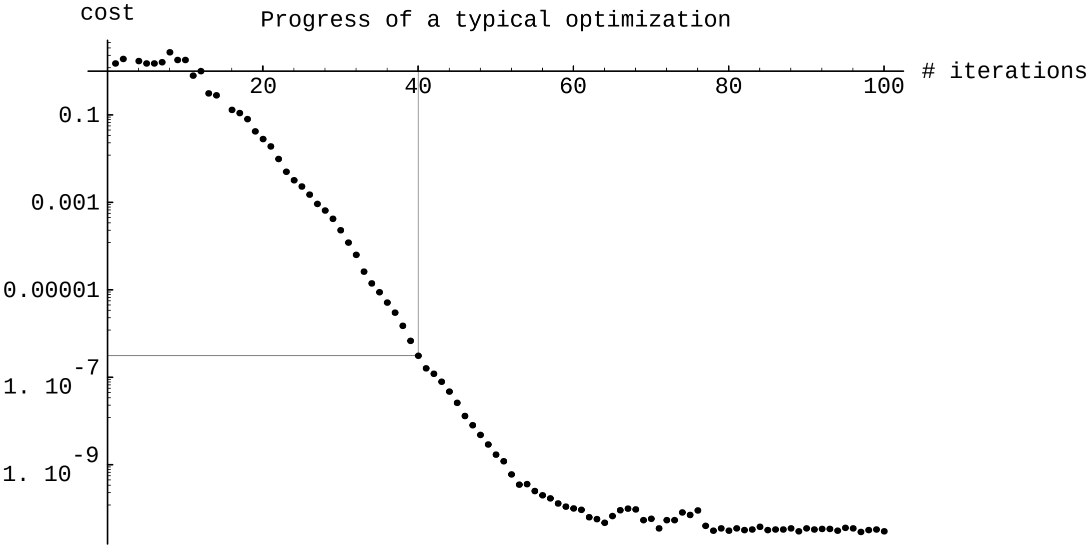


Figure 4.5: The progress of a small, but otherwise typical optimization. The *cost* quantifies the discordance of the current approximation with the necessary conditions for a solution. It decreases over the iterations, and would ideally become zero. The ordinate has a logarithmic scale.

is too short leads to almost equal function values at both sample points and hence to a loss of precision due to cancellation. But there is a wide middle range of δ settings that can be used with confidence.

Constraint tolerance (Fig. 4.6 right) limits *badness* of any step in the line search. A tolerance that is too tight can lead to very small steps in the line search due to nonlinearities in the constraints, and hence to slow progress. The preset value appears reasonable based on experiments.

The graph of *line_tol* shows plateaux because it determines the number of steps taken in the binary search, which is $\lceil {}_2\log \frac{line\_tol}{stepLarg} \rceil$. A tolerance that is too large will lead to an inexact line search and will deteriorate the overall performance. A narrow tolerance will do an exact line search, but no improvement is gained from an extremely precise line search, so this leads to a waste of time.

The quantity *badness* is defined as the maximal increase in the square of the violation of any equality or active inequality constraint. The tolerance for “badness” in the Newton step (Fig. 4.7,right) should not be chosen too small lest it stop the steps towards $\underline{\hat{c}}(\underline{x}) = \underline{0}$ unnecessarily. A large tolerance does no harm; we wouldn’t need this safeguard if we only wanted to solve the model problem. But it can provide some robustness in particularly difficult problems with highly nonlinear constraints.

Activation and inactivation of constraints is done with hysteresis. Ac-

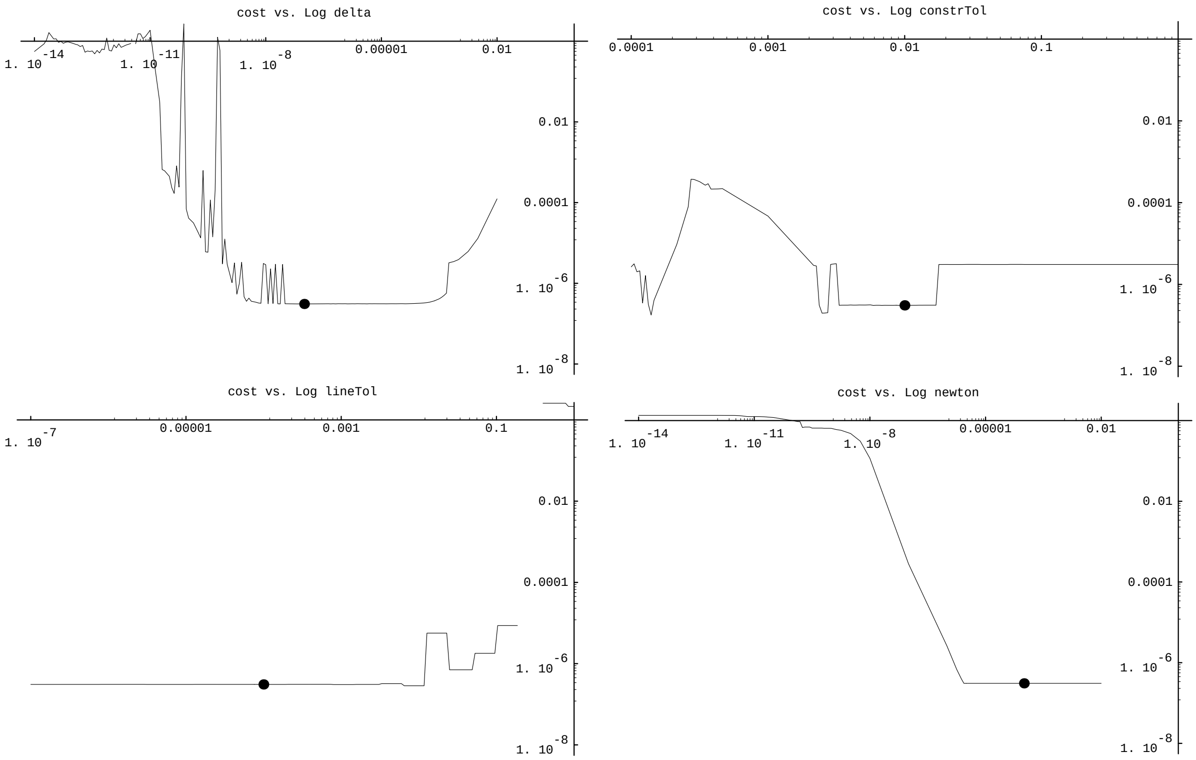


Figure 4.6: The value of *cost* after 40 iterations, plotted against each one of the four selected parameters, which are, however, held constant during one optimization run.

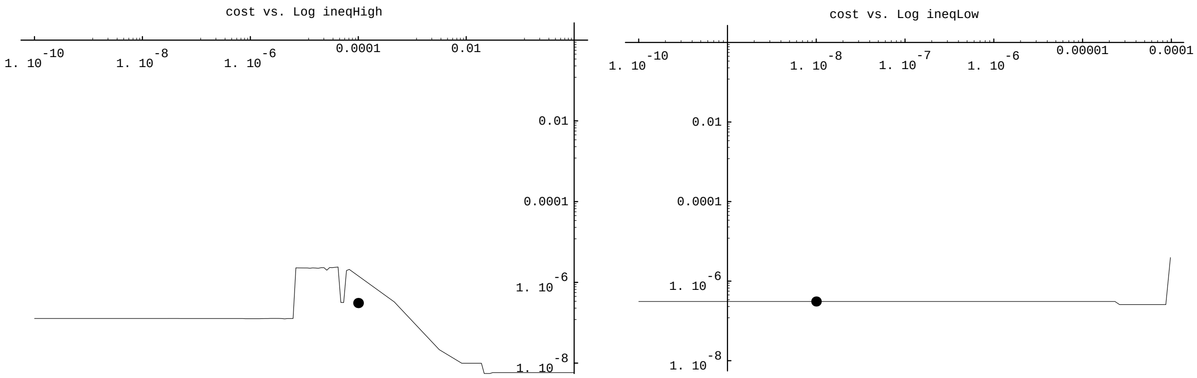


Figure 4.7: The value of *cost* after 40 iterations, plotted against the inequality activation hysteresis thresholds.

tivation occurs at the upper threshold of violation, *ineq_high* (Fig. 4.7 left). Inactivation is only allowed when the violation $-c_i(\underline{x})$ has dropped below *ineq_low* (Fig. 4.7 middle). With the current settings of the other parameters, *ineq_low* has no big influence on overall performance, unless it is increased a lot. When *ineq_high* is set very tolerant then no inequality will ever become active. This gives a low figure for *cost*, which does not take inactive inequalities into account. But it is not desirable to allow arbitrary violation of inequality constraints - that would be ignoring them. *ineq_high* might be set a bit lower (to be more strict) than the currently preset value. A narrow span between *ineq_high* and *ineq_low* can cause a constraint to rapidly switch from inactive to active and back again, which is inefficient because the dimension and the connectivity graph of the system matrix change, and the system has to be set up again each time. I hope a reasonable gap between the two thresholds will provide some hysteresis and a more continuous behavior.

In very large or difficult problems it can be impossible to make any significant progress towards minimizing the goal function or satisfying the equality constraints without setting *ineq_high* to a rather lenient value. The following adaptive scheme allows the optimization to begin with a tolerant value of *ineq_high*, reducing it gradually, and rigorously enforcing all inequality constraints in the end. The variable *ineq_high* is initialized to *ineq_init*. At each accepted position (all inequalities $\geq -\mathit{ineq\_high}$) the value *min_ineq* of the most violated inactive inequality is determined. Then *ineq_high* is reduced to the extent that *min_ineq* permits; it takes the following new value.

$$\max(\min(\mathit{ineq\_high}, -\mathit{ineq\_slack} \cdot \mathit{min\_ineq}), \mathit{ineq\_final})$$

The scheme ensures that *ineq_high* can never drop below *ineq_final* or increase again above the value it has reached. At some point, the optimization may need to move slightly against the worst inactive inequality. The factor $\mathit{ineq\_slack} > 1$ avoids blocking progress in this case.

The three parameters subtly control the adaptation of ρ, which trades off between reductions of $\mathcal{L}(\underline{x})$ and $||\underline{\hat{c}}(\underline{x})||^2$. The ratio $c_{1\rho}/c_{0\rho}$ is more important than the values. This can be seen from the definition of the adaption scheme, and it shows in the two (logarithmic) plots: they are mirror images of each other. The graph for $c_{2\rho}$ is very similar to them as well; with respect to $c_{1\rho}$, it is compressed for large values of $c_{2\rho}$ and stretched a lot for small ones (due to the logarithmic scale). The influence of these three constants is not to be neglected, it can change *cost* by one or two orders of magnitude. It is not easy to find a low constant region. Increasing $c_{0\rho}$ to 1.7 or reducing $c_{1\rho}$ to 0.2 might be advantageous.

A smaller than the current value for ρ_{init} (Fig. 4.9, left) might seem appropriate. Indeed reducing it to about 100 might be beneficial. I would not set it in the low flat region to the left, because a conservative (high) value is a more robust starting point in difficult problems; it won't allow

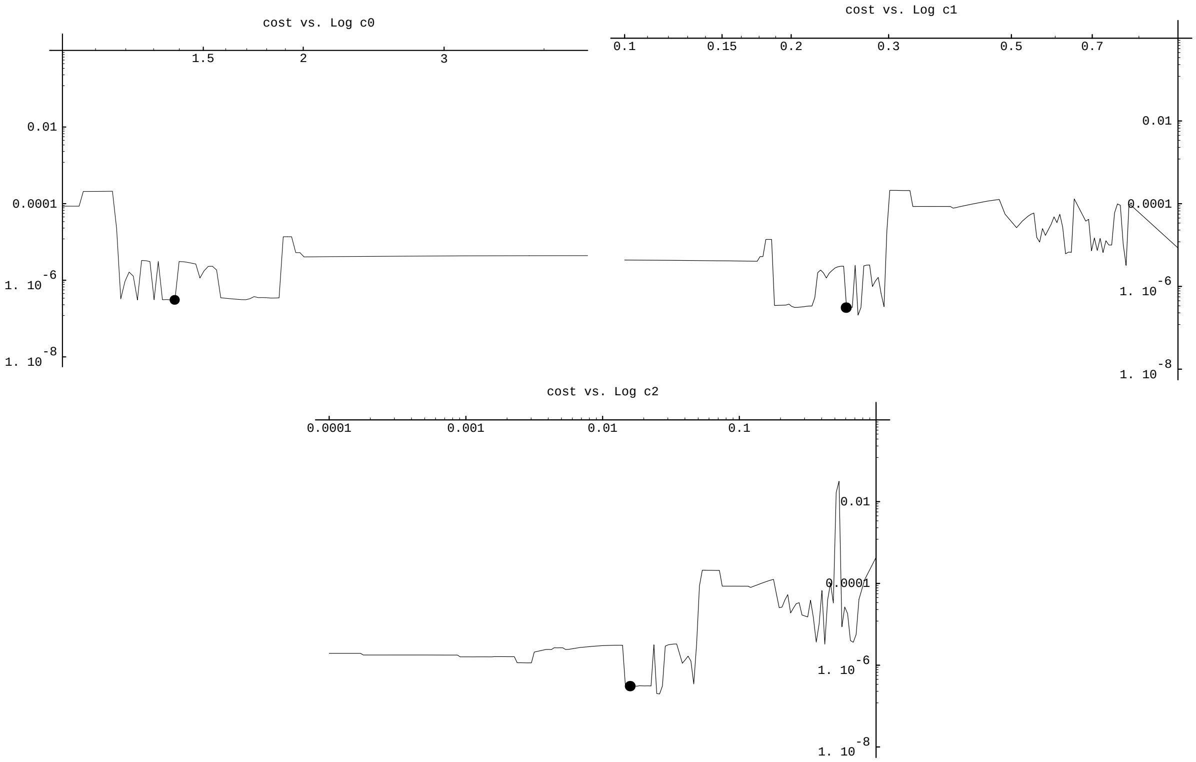


Figure 4.8: Influence of three constants in the ρ adaption scheme on *cost* after 40 iterations.

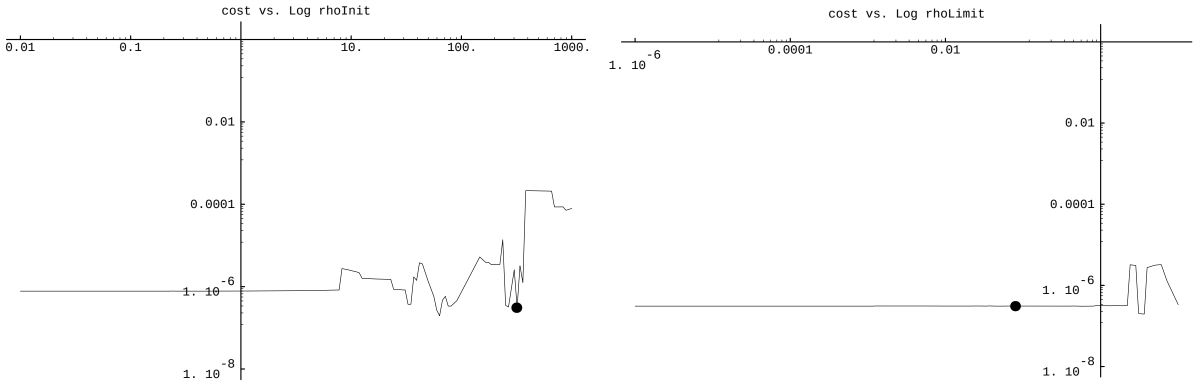


Figure 4.9: Influence of two other constants in the ρ adaption scheme on *cost* after 40 iterations.

much increase in any violation for a certain decrease in the objective function. If the problem turns out to be easy, ρ will soon decrease and adapt to the situation. If the initial ρ is small, the first step may be too large in spite of the constraint violations this might bring along; it can then be difficult for the program to find its way back.

The purpose of ρ_{limit} (Fig. 4.9, right) is to stop ρ from becoming zero even in very well-behaved problems. The choice of this constant is not sensitive, any small value will do. An extremely small value is not the best, because ρ should not be allowed to drop too low, so that it can respond promptly to a changing situation (which doesn't come up in the example problem).

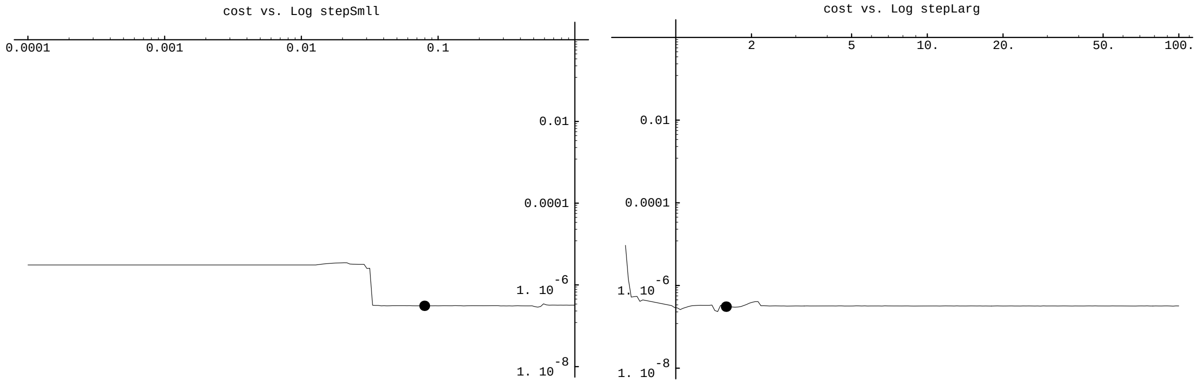


Figure 4.10: Influence of two other constants in the ρ adaption scheme on *cost* after 40 iterations.

Control of the step size in the line search (the ratio between the actual step $\Delta \underline{x}$ and $\underline{g}_Z$) is based on the heuristic that the new step is likely of a similar size as the previous one. The starting step size in the line search adapts to the final step size of the previous, but is cannot change too much at a time. The old step is multiplied by *stepLarge* (Fig. 4.10, right), and the line search can then expand it to almost twice this size. Any value above 1 will do, but a very large step will require many bisections and waste time. Similarly the value of *stepSmall* is not critical; it is the smallest factor by which the step size may be multiplied in one step. This is a safeguard against the occasional very small, zero, or negative step.

4.3 Examples and results

Figure 4.2 shows the starting point for the optimization; the right diagram is a superimposition of the line patterns in Figure 4.1 (the iso-latitude distance was increased to $\frac{\pi}{8}$). Figure 3.6 visualizes the result of the optimization in different ways. The same vertex as in Figure 4.2 is marked. In contrast to that figure, the areas of all elementary facets in parameter space are now equal, and local distortions are minimized. The rotational position of the net on the sphere is arbitrary. After the optimization, the former poles have lost their prominent role. They have now the same importance as any other point in parameter space and could lie anywhere on the surface. However, the use of polar coordinates for visualization still gives them a conspicuous appearance in Figure 3.6c. Figure 4.11 shows several stages of the optimization process. The starting configuration, "0", corresponds to Fig. 4.2a. The final result is obtained after 64 iterations; it has the label "64" and is the same as Fig. 3.6a.

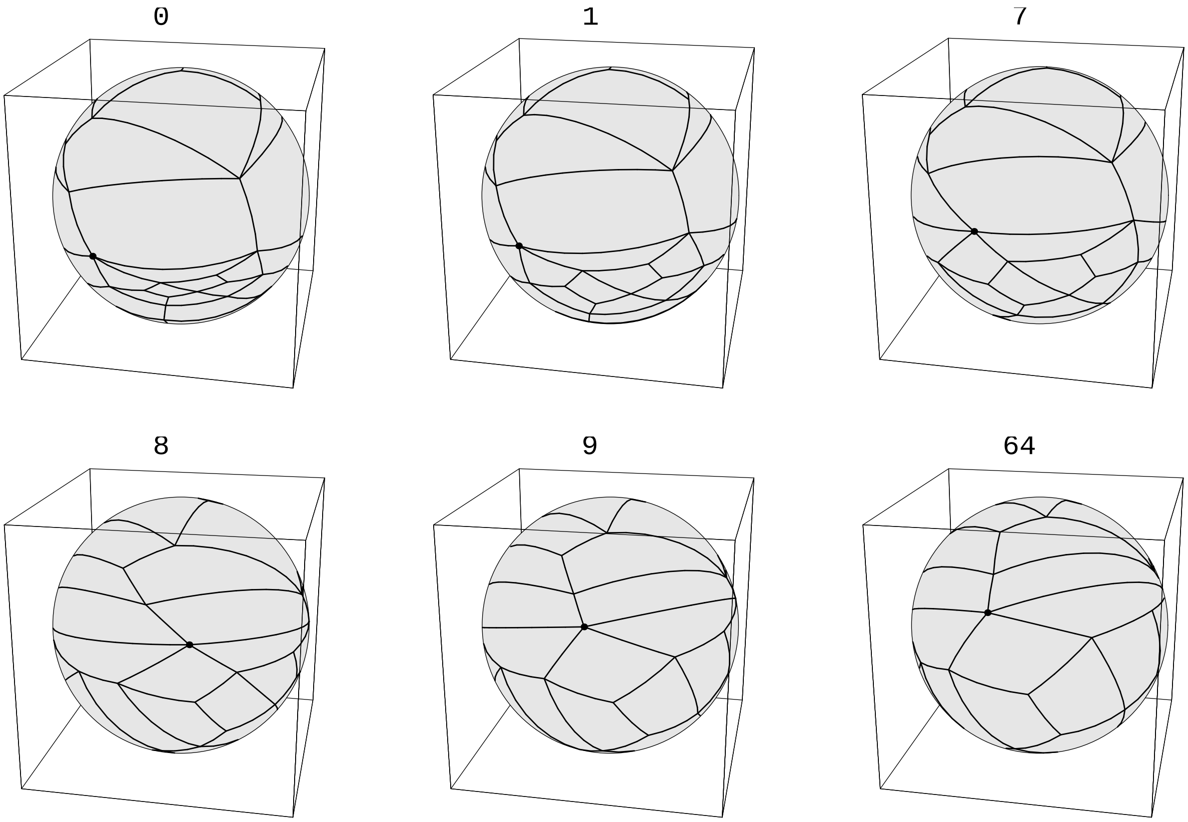


Figure 4.11: Stages from an optimization run. The labels indicate the number of iterations. "0" marks the starting configuration, where the areas of the facets vary considerably. "64" is the final result; all facets have now equal area, and distortions are minimized.

Chapter 5

The Spherical Harmonic Descriptor

5.1 Spherical harmonics

There are many possibilities for choosing a set of basis functions which are defined on the sphere. The spherical harmonic functions are a popular choice because they are relatively simple and have a number of nice mathematical properties. They are introduced e.g. by Greiner [18].

5.1.1 Definition

Y_l^m denotes the spherical harmonic function of degree l and order m. The following definitions agree with [31]. The variable w is a scalar and will correspond to u_2 below.

Legendre polynomials

$$P_l(w) \quad = \quad \frac{1}{2^l l!} \, \frac{d^l}{dw^l} \, (w^2 - 1)^l \tag{5.1}$$

Associated Legendre polynomials

$$\begin{aligned} P_l^m(w) \quad &= \quad (-1)^m (1 - w^2)^{\frac{m}{2}} \frac{d^m}{dw^m} P_l(w) \\ &= \quad \frac{(-1)^m}{2^l l!} (1 - w^2)^{\frac{m}{2}} \frac{d^{m+l}}{dw^{m+l}} (w^2 - 1)^l \end{aligned} \tag{5.2}$$

$$Y_l^m(\theta,\phi) \quad = \quad \sqrt{\frac{2l+1}{4\pi}\,\frac{(l-m)!}{(l+m)!}} \quad P_l^m(\cos\theta) \quad e^{im\phi} \tag{5.3}$$

$$Y_l^{-m}(\theta,\phi) \quad = \quad (-1)^m Y_l^{m*}(\theta,\phi) \tag{5.4}$$

A list of the spherical harmonics up to degree 3 (in table 5.1) exemplifies these definitions. Both polar coordinates (θ,ϕ) and Cartesian coordinates (u_0,u_1,u_2) are used. The Cartesian notation reveals that spherical harmonics are just polynomials, in spite of the $\frac{m}{2}$ exponent, which means a square root if m is odd.

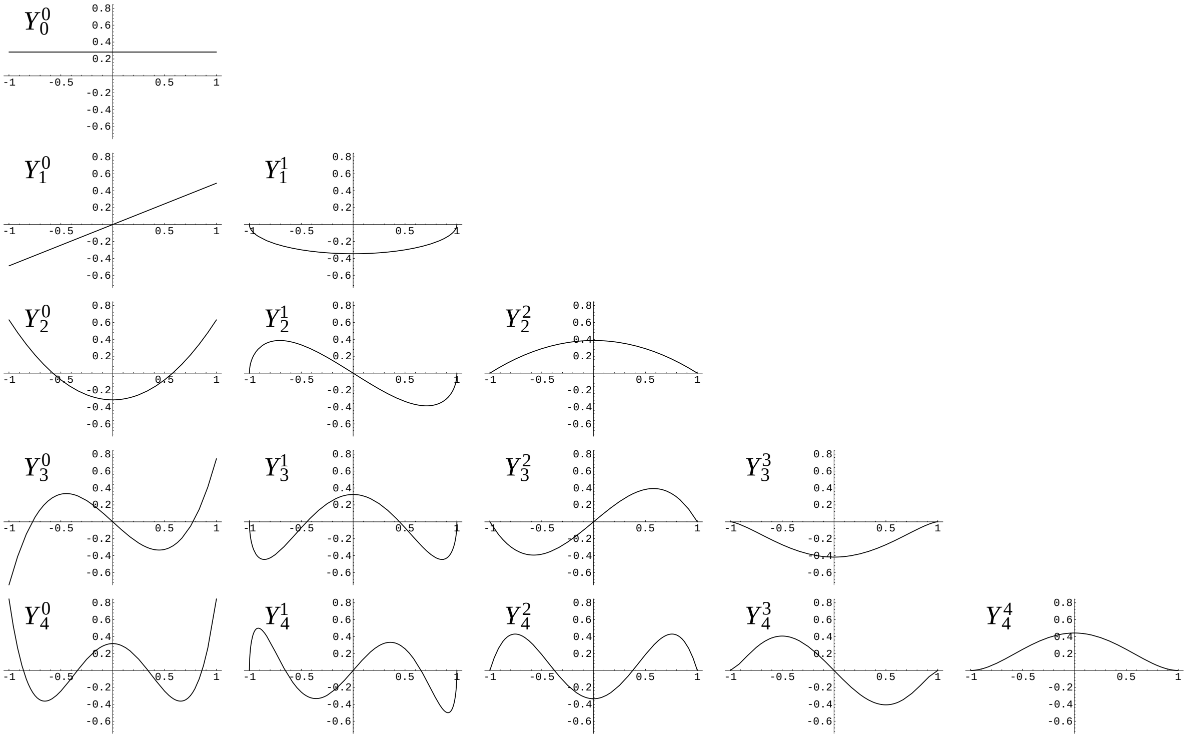


Figure 5.1: Plots of $Y_l^m(\theta = \arccos w, \phi = 0)$ up to degree 4. These diagrams are scaled plots of the associated Legendre polynomials.

Figure 5.1 plots $Y_l^m(\arccos w, 0)$ for $0 \leq m \leq l < 5$. The functions Y_l^m take real values for $\phi = 0$. The graphs give a quantitative impression of the zonal amplitude variation of Y_l^m. At the same time, they are plots of the associated Legendre polynomials, except for a scaling constant (cf. (5.3)). The leftmost column corresponds to the Legendre polynomials themselves. Figure 5.2 gives a more qualitative impression of our basis functions and their signs. The real parts of the spherical harmonics up to degree 5 are displayed as gray levels on the surface of the unit sphere,

l	$m=0$	$m=1$	$m=2$	$m=3$
0	1			
1	$\cos\theta$	$e^{i\phi}\ \sin\theta$		polar
2	$-1+3\cos^2\theta$	$e^{i\phi}\ \cos\theta\ \sin\theta$	$e^{2i\phi}\ \sin^2\theta$	
3	$-3\cos\theta+5\cos^3\theta$	$e^{i\phi}(1-5\cos^2\theta)\sin\theta$	$e^{2i\phi}\cos\theta\ \sin^2\theta$	$e^{3i\phi}\sin^3\theta$
0	1			
1	u_2	$u_0+i\,u_1$		Cartesian
2	$-1+3\,{u_2}^2$	$(u_0+i\,u_1)\,u_2$	$(u_0+i\,u_1)^2$	
3	$-3\,u_2+5\,{u_2}^3$	$(u_0+i\,u_1)(1-5{u_2}^2)$	$(u_0+i\,u_1)^2u_2$	$(u_0+i\,u_1)^3$
0	1			
1	u_2	u_0		Re Y
2	$-1+3\,{u_2}^2$	$u_0\,u_2$	${u_0}^2-{u_1}^2$	
3	$-3\,u_2+5\,{u_2}^3$	$u_0\,(1-5\,{u_2}^2)$	$({u_0}^2-{u_1}^2)u_2$	$u_0^3-3u_0u_1^2$
0	0			
1	0	u_1		Im Y
2	0	$u_1\,u_2$	$u_0\,u_1$	
3	0	$u_1(1-5{u_2}^2)$	$u_0\,u_1\,u_2$	$u_1^3-3u_0^2u_1$
0	$1/\sqrt{4\,\pi}$			
1	$\sqrt{3/4\,\pi}$	$\sqrt{3/8\,\pi}$		
2	$\sqrt{5/16\,\pi}$	$\sqrt{15/8\,\pi}$	$\sqrt{15/32\,\pi}$	
3	$\sqrt{7/16\,\pi}$	$\sqrt{21/64\,\pi}$	$\sqrt{105/32\,\pi}$	$\sqrt{35/64\,\pi}$

Table 5.1: Explicit expressions of the spherical harmonics up to degree 3, in both polar and Cartesian form. The last part of the table gives the common normalizing constants, e.g., $Y_1^0 = \sqrt{3/4\pi}\,u_2$.

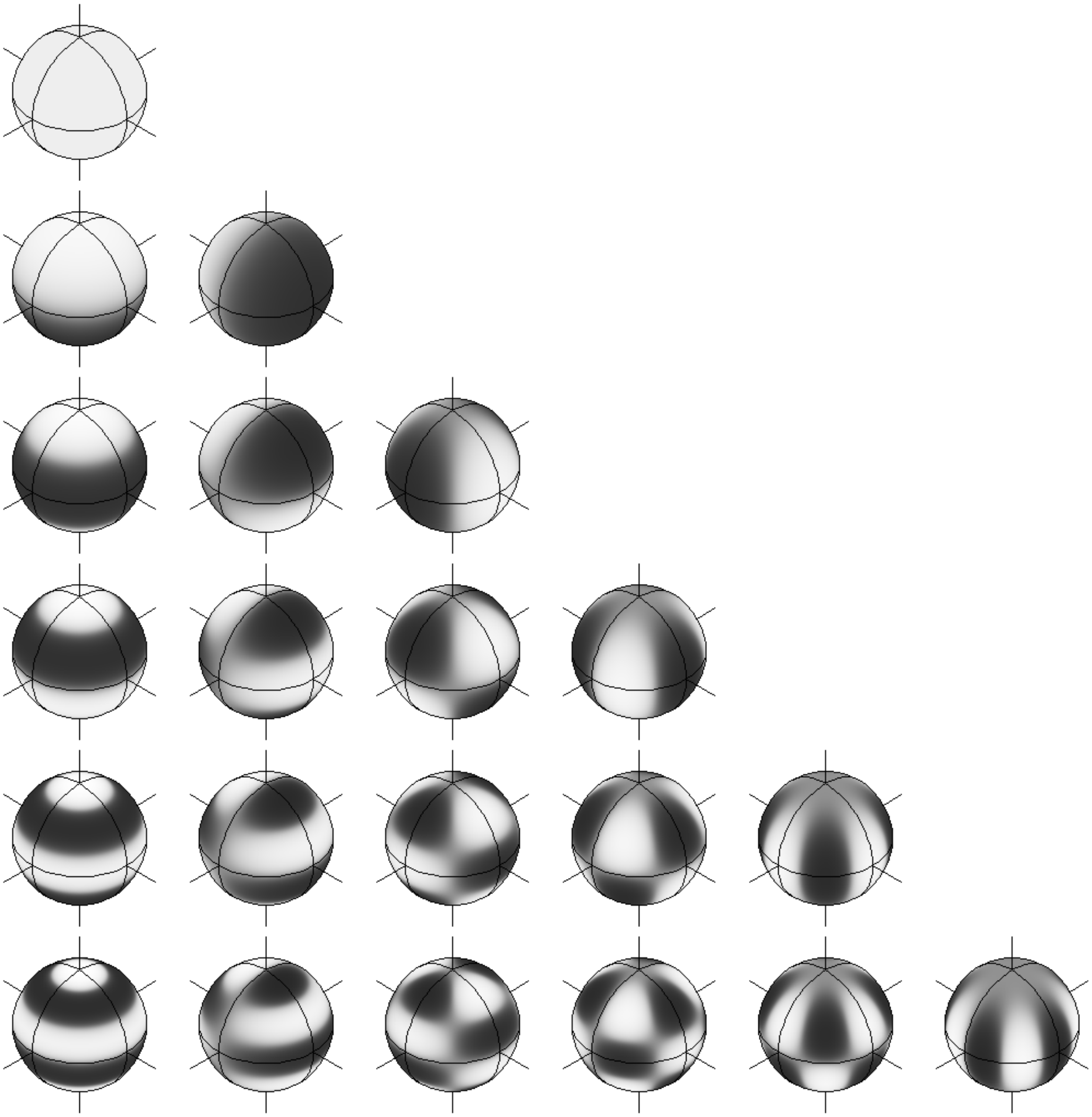


Figure 5.2: The real parts of the spherical harmonic functions Y_l^m, with l growing from 0 (top) to 5 (bottom), and m ranging from 0 (left) to l in each row. The function value is mapped on the spheres, light grey representing positive values and dark grey negative values.

which is the domain of Y_l^m. The real-valued functions Y_l^0 appear by themselves in the left column (m =0). For $m > 0$, the imaginary part is the same as the real part, rotated by $-\frac{\pi}{2m}$.

A spherical harmonic function of degree l is a polynomial of degree l in u_0, u_1 and u_2. It can be written as a homogeneous polynomial of degree l (using the identity ${u_0}^2 + {u_1}^2 + {u_2}^2 = 1$, on Ω_3).

In some cases, we have to adopt an indexing scheme $j(l, m)$ that assigns a unique index j to every pair l, m, like e.g. $j(l, m) := l^2 + l + m$. When the degree of the spherical harmonics is limited, i.e. $0 \le l < n_l$, j is also limited by $j < n_j = n_l^2$.

5.1.2 Expressing surface shape

With spherical harmonics the series (2.3) takes the form

$$\underline{x}(\theta, \phi) = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} \underline{c}_l^m \; Y_l^m(\theta, \phi) \; . \tag{5.5}$$

The coefficients

$$\underline{c}_l^m = \begin{pmatrix} c_{l\,0}^m \\ c_{l\,1}^m \\ c_{l\,2}^m \end{pmatrix}$$

in this series are three-dimensional vectors. Their components, $c_{l\,0}^m$, $c_{l\,1}^m$ and $c_{l\,2}^m$, are complex numbers for $m \ne 0$ in general; they are real numbers for $m = 0$.

For convenience, the real and imaginary parts of the complex basis functions Y_l^m can be used as independent real valued basis functions. The set of functions $\{Y_l^0, \mathrm{Re}(Y_l^m), \mathrm{Im}(Y_l^m)\}$ (where $l \ge 0$ and $0 < m \le l$) is orthogonal but not normalized. For $m > 0$, $\oint \mathrm{Re}(Y_l^m)^2 = \oint \mathrm{Im}(Y_l^m)^2 = \frac{1}{2}$. The real functions might be scaled by $\sqrt{2}$ for $m \ne 0$. There is the same number of functions as with the complex basis, namely $2l + 1$ for any non-negative l, or n_l^2 in total.

5.1.3 Variability of spherical harmonic surfaces

All of the following shapes are defined as spherical harmonic surfaces of degree up to three, i.e. $n_l = 4$. Each of x_0, x_1 and x_2 are defined by the 16 coefficients in the corresponding series. They illustrate the wide variability that can be achieved even with a low degree. Typically only

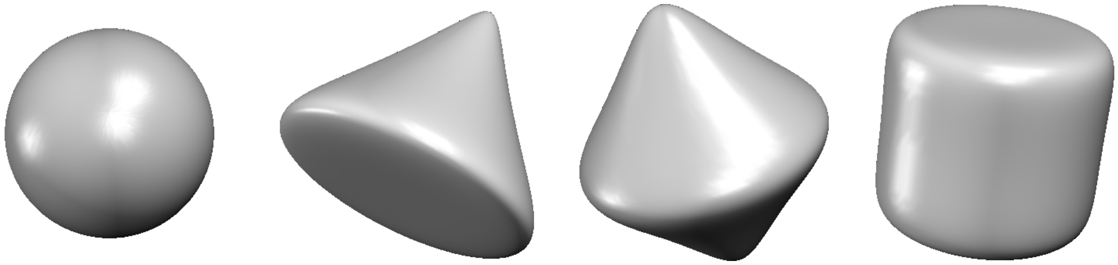


Figure 5.3: Smooth rotational geometrical objects: vector valued series of spherical harmonics represent a sphere, and they approximate a cone, a double cone, and a cylinder.

a few (like 5 or 9) of the 48 coefficients are different from zero in the examples.

The basis functions are smooth. The objects described by a truncated series of the form (5.5) tend to be smooth as well (Fig. 5.3). But they can have sharp edges or cusps as well, as Figure 5.4 illustrates. The first three platonic polyhedra (Figure 5.5) can be modelled with the truncated set of basis functions. The polyhedra in Figure 5.5 have rotational symmetries of order two and three, and the cube and the octahedron have even fourth order symmetry. More generally, an object using spherical harmonics up to degree $n_l - 1$ can have at most n_l-ary rotational symmetry. Examples for $n_l = 2 \ldots 4$ appear in Figure 5.6.

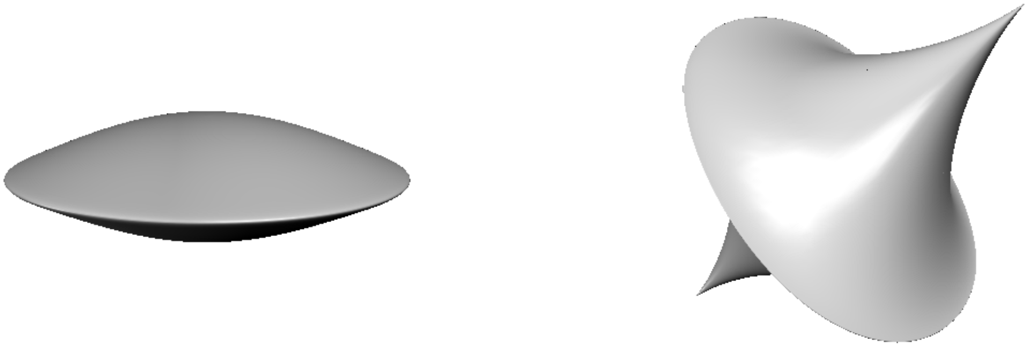


Figure 5.4: The lens has high curvature at the rim, and the top has a sharp edge and points. Yet they are composed of low degree spherical harmonics.

There is no reason to limit ourselves to star-shaped objects. Figure 5.7 lists a few that are not. The Dodecahedron and the Icosahedron would need higher harmonics: but the tetrahedron uses only degree one and

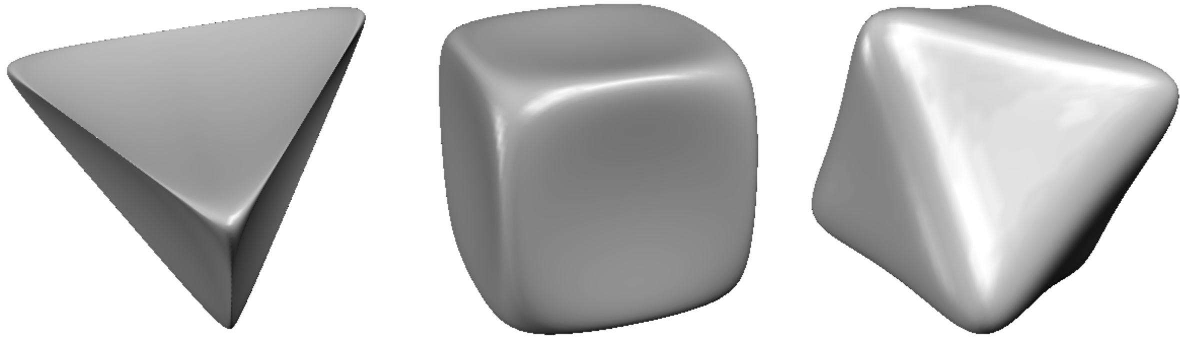


Figure 5.5: The tetrahedron, the cube or hexahedron, and the octahedron are the three simpler ones of the five platonic polyhedra.

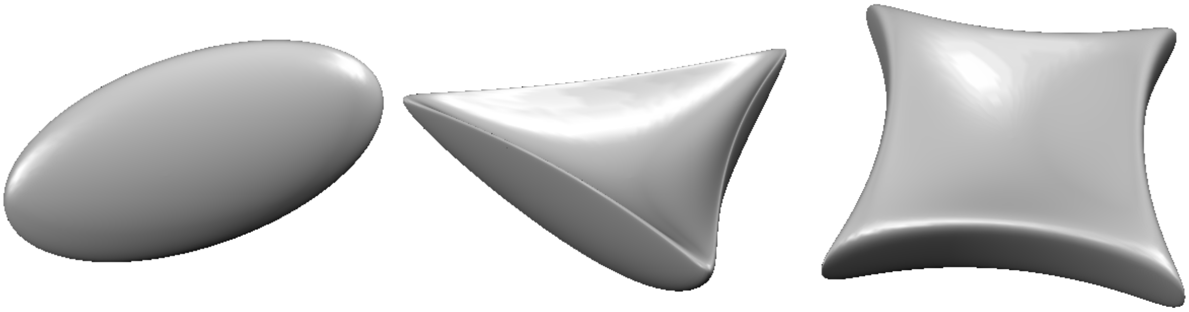


Figure 5.6: Objects involving spherical harmonics up to degree 1, 2 and 3 can have a 2-, 3- and 4-fold rotational symmetry.

two. The cube and the octahedron have no second degree component. They only differ in the sign of the third degree contributions: starting from a sphere, the points that move out on the cube move in on the octahedron, and vice versa. The upper signs in (5.6) correspond to the cube, the lower ones to the octahedron.

$$\begin{pmatrix} x_0 \\ x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} -0.7071\ \mathrm{Re}(Y_1^1) \pm 0.0745\ \mathrm{Re}(Y_3^1) \pm 0.098\ \mathrm{Im}(Y_3^3) \\ -0.7071\ \mathrm{Im}(Y_1^1) \pm 0.0745\ \mathrm{Im}(Y_3^1) \pm 0.098\ \mathrm{Re}(Y_3^3) \\ 0.5\ Y_1^0 \mp 0.0863\ Y_3^0 \end{pmatrix} \tag{5.6}$$

Objects with symmetries have been used in the examples up to now because their whole shape can be perceived or at least guessed from a single view. The chosen objects are special cases with respect to symmetry and not representative in that sense for shapes expressable with spherical harmonic descriptors. Figure 5.8 gives an example of a general object.

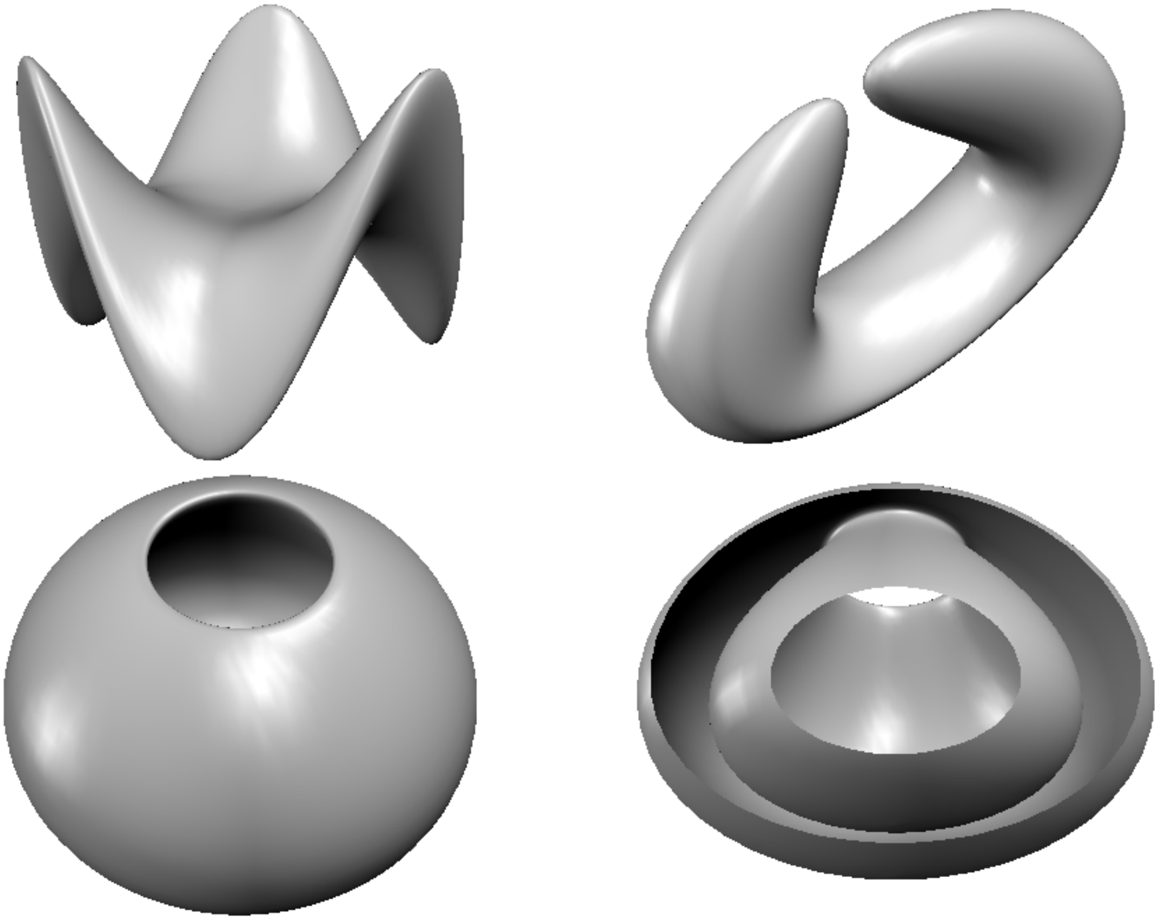


Figure 5.7: Neither of these is star-shaped: A monkey saddle, a croissant and a vase. In the last image, the vase is cut open to reveal the folding in of the surface.

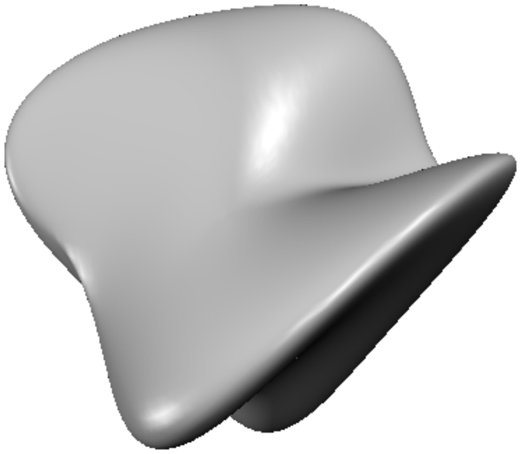
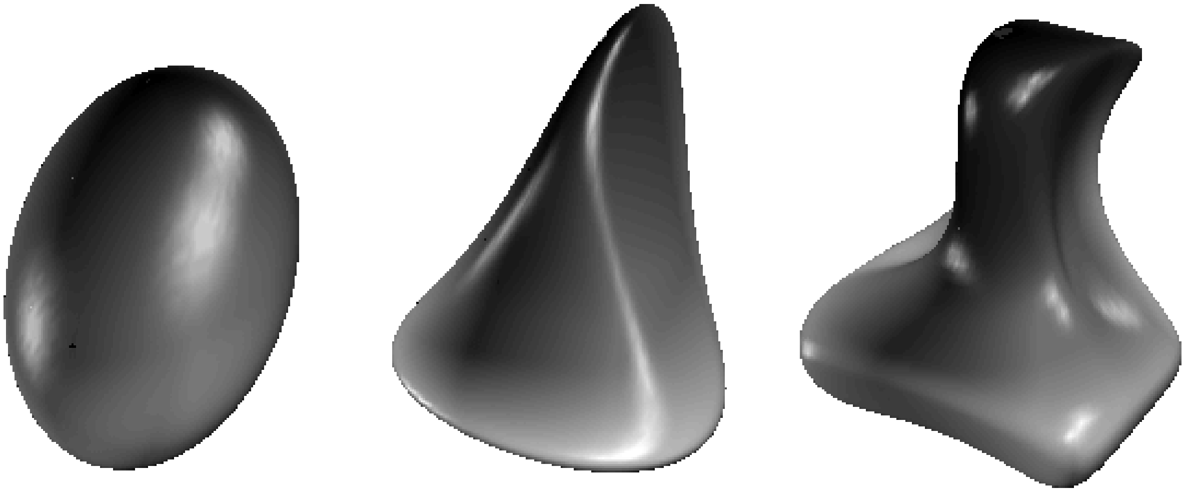


Figure 5.8: A “random” asymmetric blob shaped object

5.2 Harmonic shape descriptors

Section 3.4 introduced the parametrization of the surface of a simply connected object. The parametrization provides a correspondence between any surface vertex x_i and its parameter u_i, which can be interpreted as the sampling of a function $\underline{x}(\underline{u})$, i.e. $\underline{u}[i] \rightarrow \underline{x}(\underline{u}[i]) = \underline{x}[i]$. Now $\underline{x}(\underline{u})$ is expanded into a series of spherical harmonics as in (5.5), or into any other set of basis functions

The coefficients of the spherical harmonic functions of different degrees provide a measure of the spatial frequency constituents of the structure. Partial sums of the series (5.5) for the "duck" test object are plotted in Figure 5.9. The sums are truncated by limiting l to $0 \leq l < n_l$, where $n_l = 2, 4, 8$. As higher frequency components are included, more detailed features of the object appear.



(a) (b) (c)

Figure 5.9: Global shape description by expansion into spherical harmonics: The figures illustrate the reconstruction of the partial spherical harmonic series, using coefficients up to degree 1 (a), to degree 3 (b) and 7 (c).

5.2.1 Integration over the sphere

The use of orthonormal basis functions is convenient for the calculation of the expansion coefficients. Formally, the coefficients are calculated by forming the inner product of $\underline{x}$ with the basis function in question:

$$\underline{c}_l^m \quad = \quad \langle \underline{x}(\theta,\phi), Y_l^m(\theta,\phi)\rangle = \int_0^\pi \int_0^{2\pi} \underline{x}(\theta,\phi)\, Y_l^m(\theta,\phi)\, d\phi \sin\theta\, d\theta \qquad (5.7)$$

The parametrization defines the function $\underline{x}(\theta,\phi)$ only for the parameter coordinates of the vertices. Only $\underline{x}(\theta_i,\phi_i) = \underline{x}_i$ is defined, where i is the index of a vertex, $0 \le i < n_{\text{vert}}$. For the evaluation of the integral (5.7) we would have to define an interpolating function between these sample points; an adaptation of bilinear interpolation could be used for this purpose. But this would introduce an artificial sub-voxel resolution that is not supported by the input data. On the other hand, the straightforward discretization of the integral is

$$\underline{c}_l^m \quad \approx \quad \sum_{i=0}^{n_{\text{vert}}-1} \underline{x}_i\, Y_l^m(\theta_i,\phi_i)\, \Delta\Omega \qquad , \qquad (5.8)$$

with the finite $\Delta\Omega$ replacing $d\Omega = d\phi \sin\theta d\theta$. There are two possibilities for choosing $\Delta\Omega$. Setting $\Delta\Omega = \frac{4\pi}{n_{\text{vert}}}$ distributes the area of Ω_3 evenly over all vertices. But it might be argued that the area of a facets is relevant, and hence should be distributed to its four corners. Then $\Delta\Omega = \frac{4\,\pi\, count_i}{4\, n_{\text{face}}}$, where $count_i$ is the number of neighbors of vertex i. But neither of these schemes in general gives the precise coefficients of a series representing our object. The reason is that although the functions Y_l^m are orthonormal, their values evaluated at some set of parameter pairs (θ_i,ϕ_i) will generally not form an orthonormal set of vectors.

We can arrange all needed values of our basis functions in a $n_{\text{vert}} \times n_j$ matrix B where $b_{i,j(l,m)} = Y_l^m(\theta_i,\phi_i)$. In the usual case where n_j is significantly smaller than n_{vert}, the columns of B are approximately orthogonal. We further arrange the object space coordinates of all vertices in an $n_{\text{vert}} \times 3$ matrix $X = (\underline{x}_0, \underline{x}_1, \ldots \underline{x}_{n_{\text{vert}}-1})^T$ and all coefficients in the $n_j \times 3$ matrix $C = (\underline{c}_0^0, \underline{c}_1^{-1}, \underline{c}_1^0 \ldots)^T$. The equations (5.8) for all l and m take the compact form $C \approx \frac{4\pi}{n_{\text{vert}}} B^T X$. But what we really want is a spherical harmonic series that passes near the real positions of our vertices, i.e. $X = BC + E$ where the error matrix E should be small. These so-called normal equations are solved with least square sums over the columns of E by

$$C \quad = \quad (B^T B)^{-1} B^T X \ . \qquad (5.9)$$

The formula 5.9 mathematically states the use of the pseudo-inverse of B: it does not imply a numerical evaluation of the matrix expressions on the right. The global approximation error is the square of the Frobenius norm of $E = BC - X$, which is also minimized. This is not too different from (5.8) because the symmetric $n_j \times n_j$ matrix $\frac{4\pi}{n_{\mathrm{vert}}} B^T B$ is close to the identity matrix.

5.2.2 Invariant descriptors

The coefficients obtained thus far still depend on the relative position of the parameter net of the object surface and on the orientation of the object in space (Figure 3.6). We can get rid of these dependencies by rotating the object to canonical positions in parameter space and object space. This needs three rotations in parameter space and three rotations in object space, when rotations are described using Euler angles. All rotations result in new linear combinations of the components of the harmonic descriptors.

The relations between the Cartesian and the spherical coordinates of the parameter space are $u_0 = \sin\theta \, \cos\phi$, $u_1 = \sin\theta \, \sin\phi$ and $u_2 = \cos\theta$. To define a standard position we consider only the contribution of the spherical harmonics of degree $l = 1$ in equation (5.5).

$$\underline{x}_1(\theta, \phi) = \sum_{m=-1}^{1} \underline{c}_1^m \; Y_1^m(\theta, \phi) \tag{5.10}$$

This sum involves the basis functions $Y_1^{-1} = \frac{\sqrt{3}}{2\sqrt{2\pi}}\,(u_0 - iu_1)$, $Y_1^0 = \frac{\sqrt{3}}{2\sqrt{\pi}} u_2$ and $Y_1^1 = -\frac{\sqrt{3}}{2\sqrt{2\pi}}(u_0 + iu_1)$. Any three real valued linear combinations [1] of these, interpreted as Cartesian coordinates in the object space, will always describe an ellipsoid(see Figure 5.9 a). We rotate the object in parameter space so that the north pole ($\theta = 0$, on the u_2 axis) will be at one end of the shortest main axis of this first order ellipsoid and the point where the Greenwich meridian ($\phi = 0$) crosses the equator ($\theta = \frac{\pi}{2}$, on the u_0 axis) is at one end of the longest main axis.

This paragraph explains how I determine the main axes. At the three main axes, the length of the vector $\underline{x}_1(\theta, \phi)$ is stationary: it reaches a maximum, a saddle point, and a minimum, respectively. Measuring Euclidean lengths becomes simpler when we transform the component vectors to a Euclidean, real valued form. Applying the definitions of the

---

[1] The combination $\underline{x}_1$ is real if and only if $\left(\underline{c}_1^{-1}\right)^* = -\underline{c}_1^1$ and $\underline{c}_1^0 \in \mathbb{R}^3$

Y_1^m yields

$$\underline{x}_1(\underline{u}) \;=\; A\underline{u} \;=\; A\begin{pmatrix} u_0 \\ u_1 \\ u_2 \end{pmatrix} \;=\; \underline{a}_1\, u_0 + \underline{a}_2\, u_1 + \underline{a}_3\, u_2 \quad , \qquad (5.11)$$

where

$$A = (\underline{a}_1, \underline{a}_2, \underline{a}_3) \;=\; \frac{\sqrt{3}}{2\sqrt{2\pi}}\left(\underline{c}_1^{-1} - \underline{c}_1^1, i(\underline{c}_1^{-1} + \underline{c}_1^1), \sqrt{2}c_1^0\right) \quad .(5.12)$$

We are looking for the unit vectors $\hat{\underline{u}}_1$, $\hat{\underline{u}}_2$ and $\hat{\underline{u}}_3$ that maximize or minimize the length of the vector. The solutions are the eigenvectors of $A^T A$, with nonnegative eigenvalues $l_1^2 > l_2^2 > l_3^2$. Their roots l_1, l_2 and l_3 represent half the lengths of the main axes of the ellipsoid. At the middle eigenvector, $\hat{\underline{u}}_2$, $\|\underline{x}_1\|$ has a saddle-point rather than an extremum. The normalized eigenvectors form the rotation matrix $R_u^T = (\hat{\underline{u}}_1, \hat{\underline{u}}_2, \hat{\underline{u}}_3)$, which is applied to the parameters $\underline{u}[i]$ associated with each vertex i: $\underline{u}[i]' = R_u^T\, u_i$. This new parametrization results in new coefficients $\underline{c}_l^{m\prime}$ and hence in the new coefficient matrix $A' \;=\; A\, R_u$. Its three column vectors $\underline{a'}_1$, $\underline{a'}_2$ and $\underline{a'}_3$ are the main axes of the first order ellipsoid in object space.

All rotations are determined based on the values of $\underline{c}_1^{m'}$ $(m' = -1 \ldots 1)$ of the ellipsoid only, but they are applied to all components of the descriptor $\{\underline{c}_l^m\}$. The parameter space rotations result in a different description of the same object in the same position, just parametrized in a standard way.

Now, the ellipsoid is rotated in the object space to make its main axes coincide with the coordinate axes, putting the longest ellipsoid axis along x_0 and the shortest one along x_2. The object space rotations require only the matrix multiplication $\underline{c}_l^{m\prime\prime} \;=\; R_x\, \underline{c}_l^{m\prime}$. The object space rotation matrix is $R_x \;=\; \mathrm{diagonal}(\frac{1}{l_1}, \frac{1}{l_2}, \frac{1}{l_3}) \cdot A'^T$. It rotates the main axes of the ellipsoid into an axis-parallel position and makes the coefficient matrix $A'' \;=\; R_x\, A' \;=\; R_x\, A\, R_u$ diagonal. The elements of the diagonal are the lengths of the main axes of the ellipsoid.

Parameter Visualization

This subsection introduces a specific surface pattern. The pattern is gray valued and covers the whole surface of an object. It always stays fixed with relation to the parameter space Ω_3. This makes it possible to

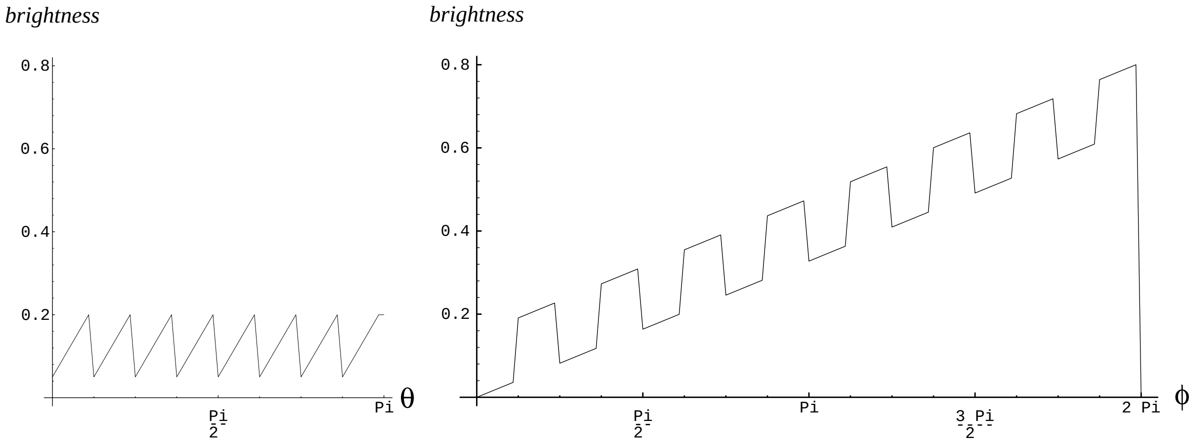


Figure 5.10: The components of the pattern are a sawtooth function of θ (*left*), and a ramp plus a rectangle function of ϕ (*right*).

estimate the parameter value associated with a location on the surface from the shading of is neighborhood. The pattern makes visible the parameter space rotations described above.

The brightness of the pattern varies with θ in a sawtooth fashion, cf. Figure 5.10a, and with ϕ in a ramp and rectangle function, cf. Figure 5.10b. The whole pattern is the additive superposition of the θ and ϕ contributions; Figure 5.11 presents it. The diagrams of Figure 5.10

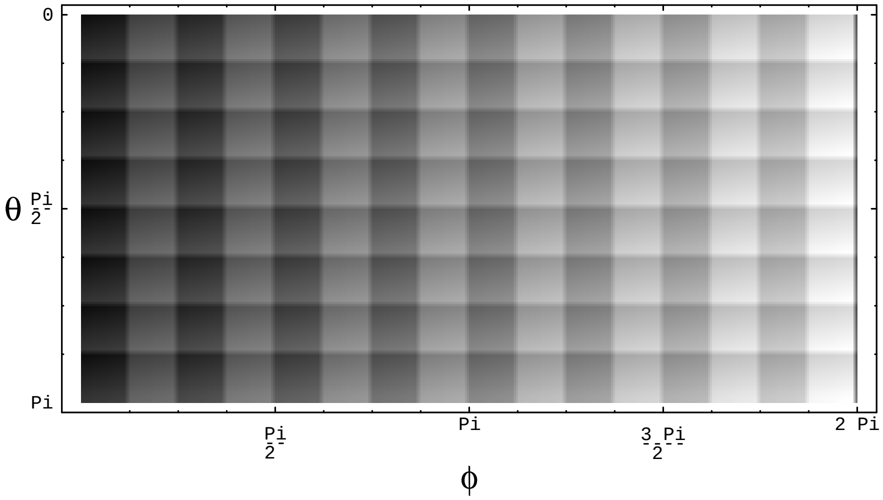


Figure 5.11: This pattern, when mapped to an object surface, reveals the pose of the parameter space. The θ axis points down to preserve a right handed coordinate system. The top border $\theta = 0$ corresponds to the north pole, the bottom border $\theta = \pi$ is the south pole.

illustrate that the functions interpolate linearly between sample points spaced $\frac{\pi}{64}$ apart and that there are no discontinuities (jumps). Even the descent in the wrap around for ϕ extends from $\frac{127\pi}{64}$ to $2\pi \equiv 0$ and is continuous.

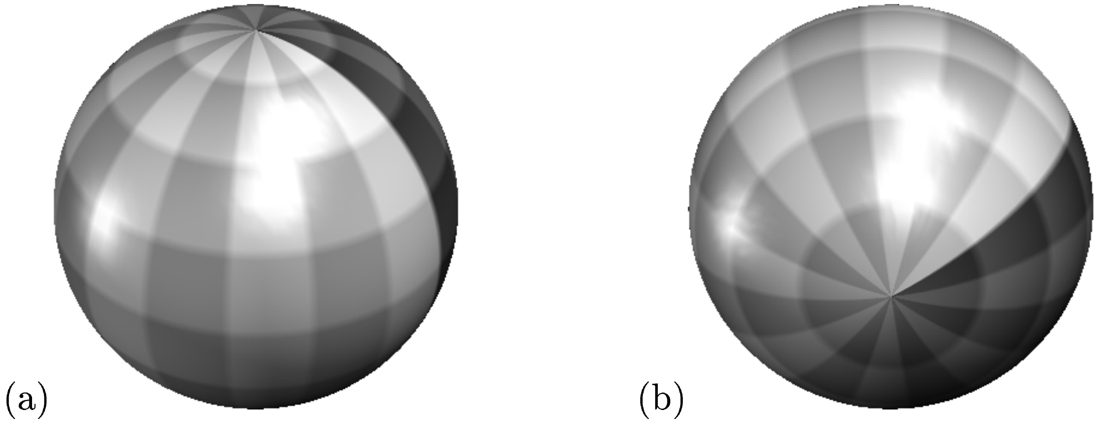

(a) (b)

Figure 5.12: The simplest object, a sphere, exemplifies the interpretation of the surface pattern. The specular highlights don't belong to the pattern but are added for a better 3D impression. *(a)* The north pole is visible from the standard view point. We perceive it as the center of a bright circular zone in spite of its dark center (the pole itself). *(b)* The region of the south pole, on the contrary, appears as a dark zone, although the pole has actually a light shade.

In Figure 5.12 the surface pattern covers the sphere $\underline{x} = \underline{u}$. This is the simplest possible object, because here only first degree terms contribute, and the object space coordinates are identical with the parameter space coordinates. Figure 5.12 might be called a "picture of the parameter space". It should help the interpretation of the following pictures that use the same pattern for parameter coding. The sphere is striped lengthwise with 16 sectors, each 8 lighter and darker ones. The stripes meet at the poles. The shading becomes slowly brighter as ϕ increases, i.e. towards east (cf Fig. 5.10b). With the normal (right handed) orientation of the parameter space, this gives a positive or counterclockwise increase of the brightness around the north pole and a negative or clockwise increase around the south pole. After one turn, the intensity leaps back from brightest to reach the darkest value on the Greenwich meridian. When moving southward, i.e. with increasing θ, the shading gradually becomes lighter, only to drop back to the same dark level every $\frac{\pi}{8}$ (cf Fig. 5.10a). The slow brightening from the north pole to the first maximum at $\pi/8$ is less visible than the rapid drop that follows, so that the north pole appears as the center of a bright disk (but this center is dark!). The opposite is true at the south pole, which is the bright center of circle with a dark border[2].

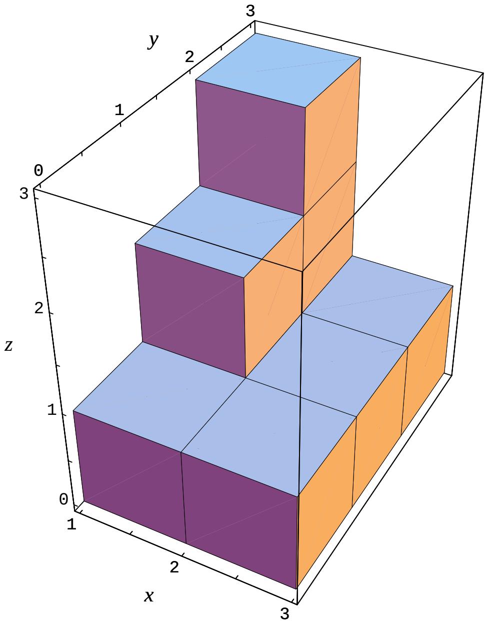


Figure 5.13: This nine voxel object will illustrate the standardization of descriptors and their comparison, including symmetry considerations. The object itself is completely asymmetric. Two steps of a stair sit on top of a 3 by 2 base plate. We call the object "stair".

To illustrate the descriptor and its rotations in parameter and object space, a test object should be very simple, but it must not have symmetries with respect to any plane, straight line, or point. Symmetries are frequent in few-voxel objects, but they would make the pose ambiguous and interfere with the symmetry discussions below in subsection 5.3.1. The nine voxel object in Figure 5.13 is fit for the purpose.

Parametrization of the surface and expansion into the spherical harmonic basis yields a descriptor for the object. Partial series of (5.5) define surfaces of increasing levels of detail. They visualize the descriptor in Figure 5.14, and they illustrate its hierarchical organization from coarse to fine. Thanks to the surface pattern, the relation to the parameter space Ω_3 is visible.

Figure 5.15 shows the descriptor $\underline{c}$ before any standardization. The

---

[2] The intensity drop that should happen exactly at the south pole is suppressed. The isolated dark spot would be distracting.

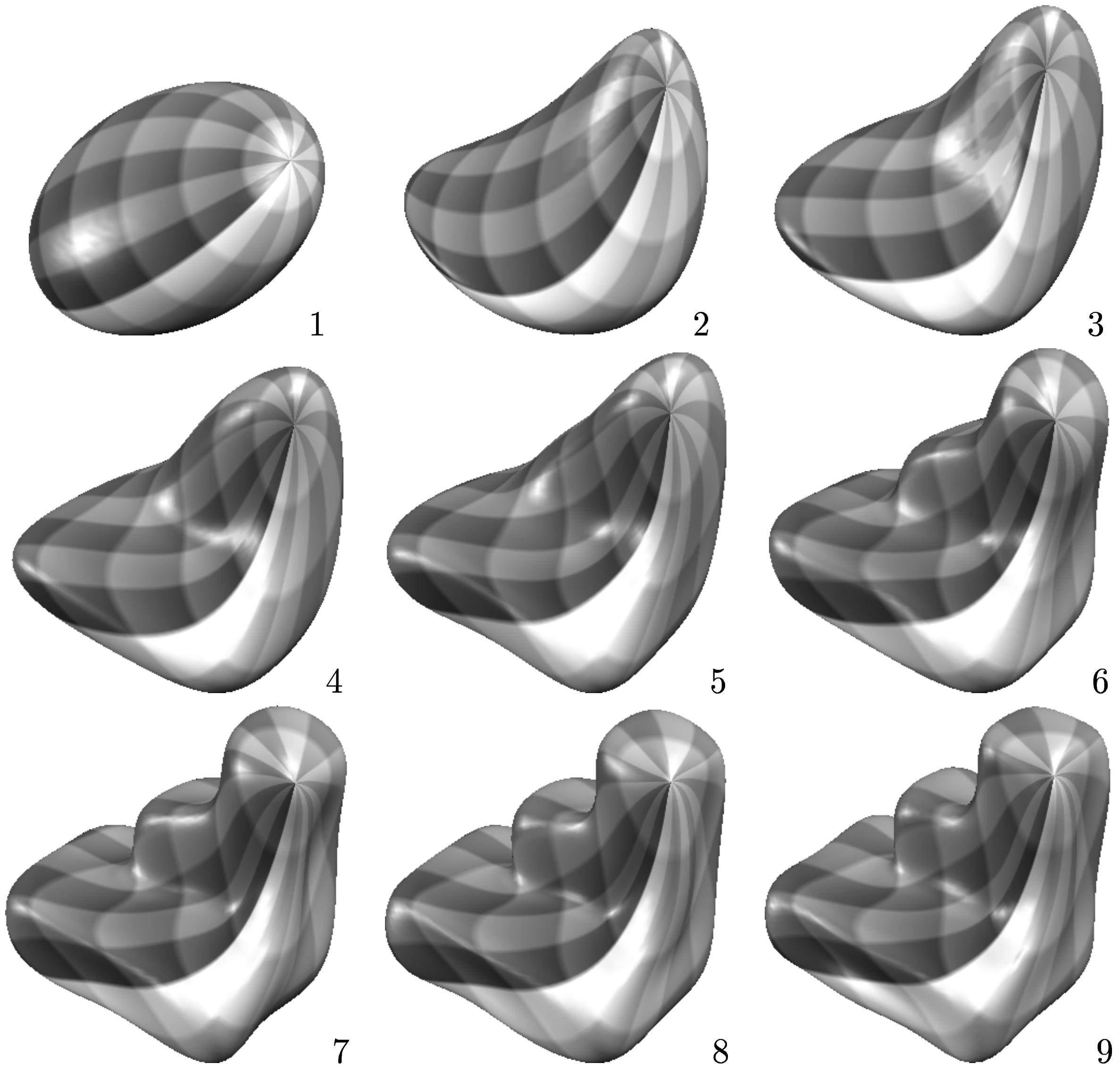


Figure 5.14: The "stair" object, reconstructed from its descriptor. The numbers indicate the maximum degree l in the partial sum, i.e. $n_l - 1$. As higher frequency components are added, more and more details show up. This is not the standard view: x_0 increases towards front and left, x_1 towards front and right, x_2 upwards. The new view point gives a better look at the north pole.

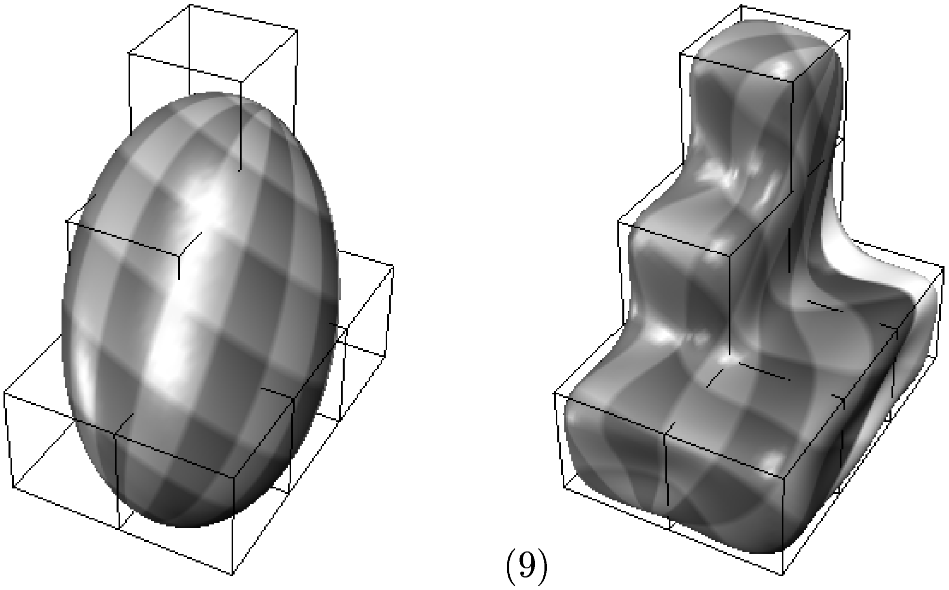

(1) (9)

Figure 5.15: The raw descriptor of the “stair” object. *Left:* The first degree ellipsoid. *Right:* Reconstruction up to degree 9 ($n_l = 10$). The edges of the original object give an impression of the accuracy of the partial sums. Standard view: x_0 increases towards the right and front, x_1 to the back and right, and x_2 increases upwards.

subsequent steps take as their reference the first degree ellipsoid, which appears in the left image. Their effects on both the ellipsoid and the full object ($n_l = 10$) are shown in each case.

Parameter space rotation takes us to Figure 5.16, showing $\underline{c}'$. The poles ($\theta = 0$ and $\theta = \pi$) end up on the shortest main axis, and the Greenwich meridian ($\phi = 0$) passes through the longest main axis. The resulting descriptor still represents exactly the same geometrical surface. The superimposed wireframe of the original “stair” object confirms that the object has not moved in object space. The object space rotation now leads to a descriptor $\underline{c}''$ in canonical position (Figure 5.17). The edges of the cube $\left[0, \sqrt{3/8\pi}\right]^3$ are overlaid as a reference of size and orientation. The main axes of the ellipsoid and of its parametrization line up with the coordinate axes. The final scaling makes the coefficient of $\mathrm{Re}(Y_1^1)$, which corresponds to the longest main axis, equal to 1. The half length of the longest axis becomes $-Y_1^1(\frac{\pi}{2}, 0) = -f_1^1((1,0,0)^T) = \sqrt{3/8\pi}$. For the descriptor of a reasonably sized object, this is a shrinking by several orders of magnitude.

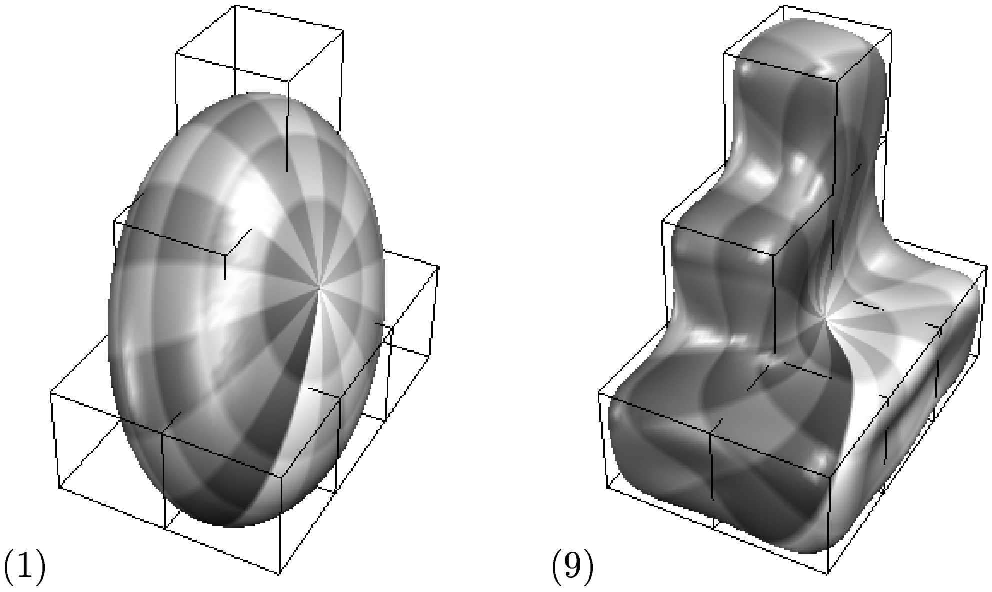

(1) (9)

Figure 5.16: The descriptor of the “stair” object after parameter space rotation. *Left:* The first degree ellipsoid. *Right:* Reconstruction up to degree 9 ($n_l = 10$). Standard view. The wireframe of the original object is overlaid again.

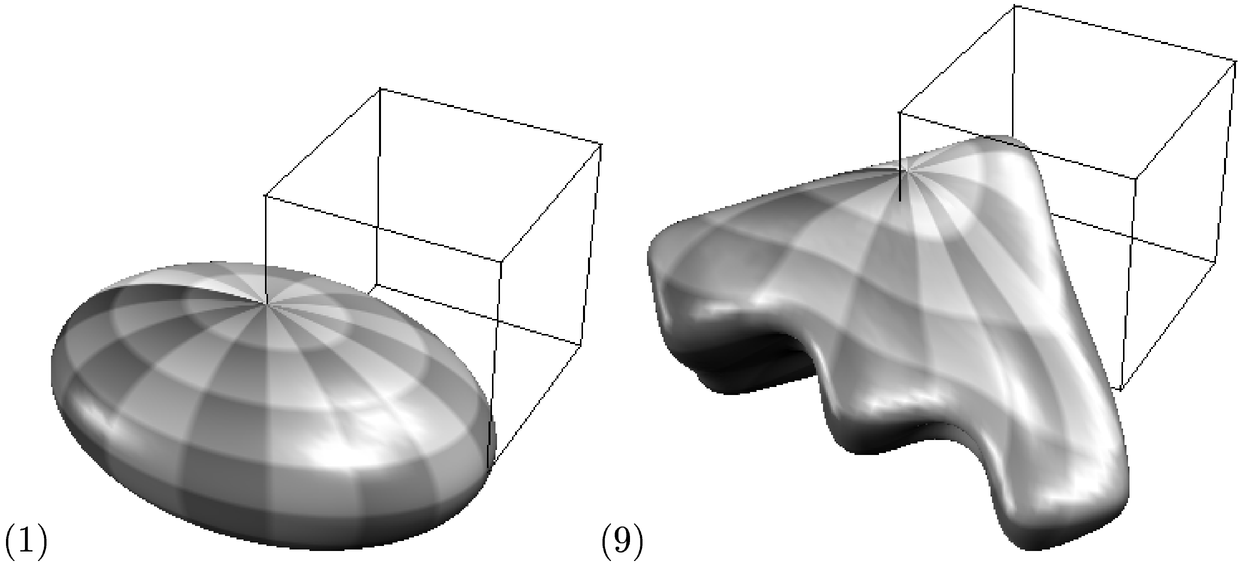

(1) (9)

Figure 5.17: The descriptor of the “stair” object in canonical position. *Left:* The first degree ellipsoid. *Right:* Reconstruction up to degree 9 ($n_l = 10$).

The descriptors $\underline{c}''$ are now invariant under rotation of the object, except mirrorings (rotations by π). Including information from higher degree coefficients could eventually disambiguate these cases. Subsection 5.3.1 presents an alternative approach, which considers all possible mirrorings. Ignoring $\underline{c}_0^0$ results in translation invariance. Scaling invariance can be achieved by dividing all descriptors by l_1, the length of the longest main axis.

5.2.3 Importance of uniform parametrization

This thesis thus far assumed that a homogeneous density and a minimal distortion of the parameter net would be important for shape characterization, especially for obtaining an invariant description. Similarly, the 2-D expansion of contours $s(t)$ into series of harmonics [23] was based on the model of tracing a curve with constant velocity, i.e. assigning same lengths ΔL to equivalent parameter steps Δt. A non-uniform distribution of parameters on an object surface, e.g. by clustering at certain locations, seems to be suboptimal with respect to a uniform representation of the whole surface. One would expect an over-representation of some parts at the expense of other regions, resulting in a distorted shape description.

The importance of a parametrization with minimal distortion can be demonstrated with an experiment. The expansion into a series of spherical harmonics is calculated for both the non-uniform initial parametrization (bypassing the optimization step for this part of the experiment) and the result after optimization. A manifestly non-star-shaped form was chosen: the original object consists of 11 voxels and is shaped like the character E. Its initial and optimized parametrizations are given in Figure 5.18 for comparison. The diagrams correspond to Figures 4.2a and 3.6a.

Figure 5.19 illustrates the expansion in a spherical harmonic series up to degree ten and the truncated reconstruction up to degree one (top), four (middle) and ten (bottom) for the initial (left) and optimized (right) parametrization. (A five-fold oversampling was applied to the surface to represent it accurately. This may be viewed as a rough form of numerical integration.) Comparing the expressive difference, one can conclude that a uniform parametrization is absolutely essential to obtain useful spherical harmonic descriptors. Even from the distorted initial parametrization, descriptors can be derived, that are necessarily “optimal” in the least squares sense, but the series of harmonics does not reflect the shape properties of the surface. Using the optimized parametrization, coeffi-

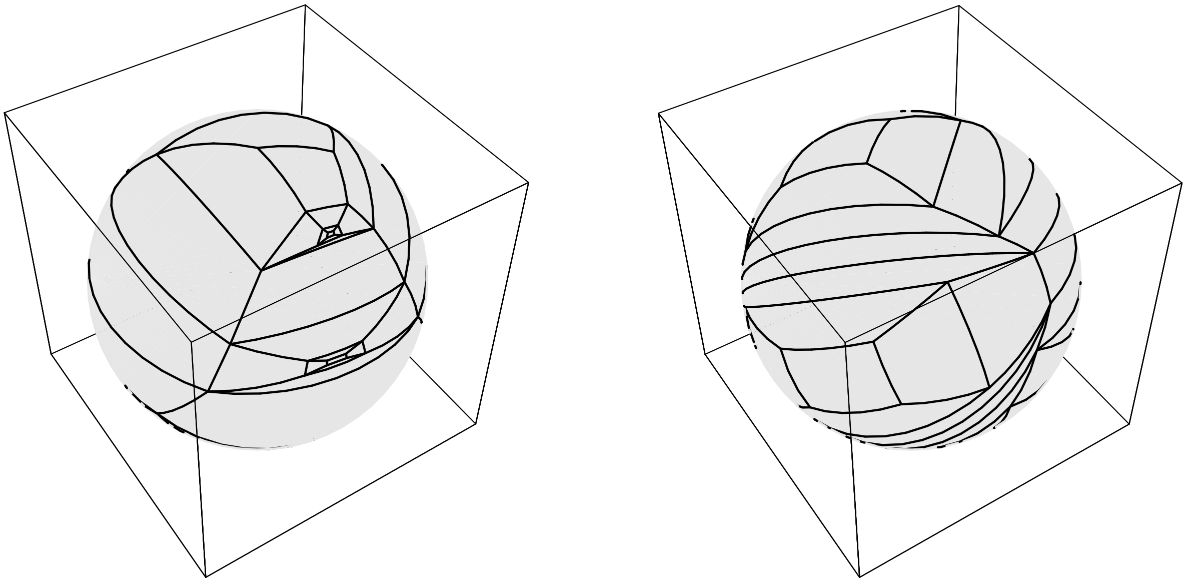


Figure 5.18: Two different parametrizations of the "E" object. *Left:* The initial parametrization is the starting point of the optimization. *Right:* The optimized parametrization.

cients of higher degree add information about details of higher spatial frequency; this is desirable. The first degree harmonic approximation (Figure 5.19b) covers the whole object and comprises information about the major size and elongation, whereas the three "legs" of the E-shape appear in the reconstruction using harmonics up to degree four (Figure 5.19d,f).

5.3 Comparison and Recognition of Shapes

The main purpose of invariant spherical harmonic descriptors is to transform the task of comparing the shape of two objects to the simpler comparison of their descriptors.

In the truncated series, l takes the n_l different values $0, 1, \ldots n_l - 1$. The descriptor presents itself as n_l^2 three-vectors or as one flat $3\,n_l^2$-vector. Thanks to the invariance properties of the descriptors, the similarity between the feature vectors of two objects measures the similarity between their shapes. One of the simplest dissimilarity measures is the Euclidean distance between the feature vectors. This is the same as the L_2 norm of the difference of the two vectors. The square of this quantity is the sum of the squared differences between corresponding entries in the two vectors.

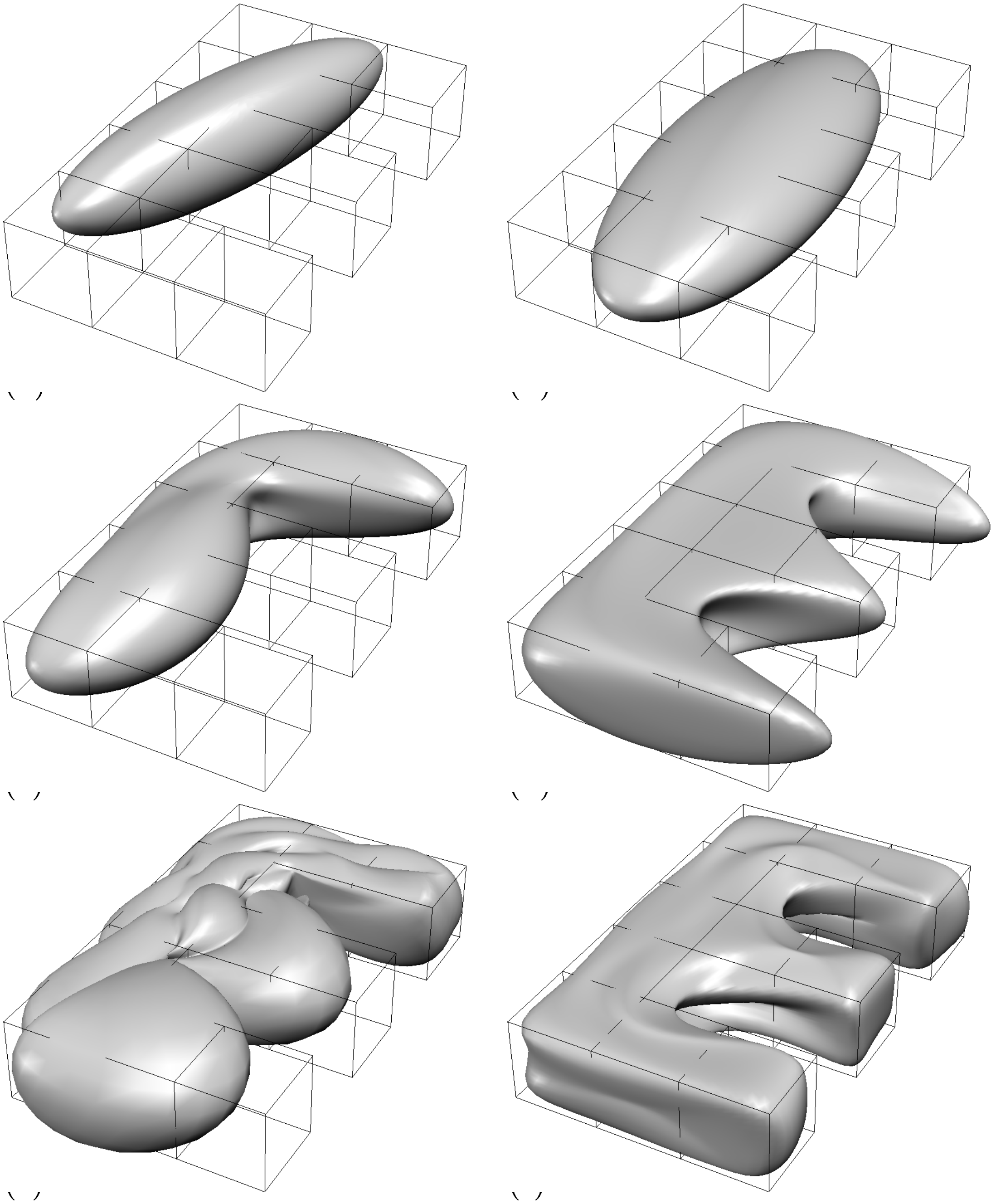


Figure 5.19: Homogeneous parameter distribution is important for shape description. The “E”-shaped object surface, indicated by a wireframe, is expanded into a series of spherical harmonics. The Shaded surfaces depict the reconstructions of the series up to degree 1 (a,b), 4 (c,d), and 10 (e,f). The initial, non-uniform parametrization yields a poor shape representation (a,c,e); its optimization achieves a significant improvement (b,d,f). $\|E\|_F$ measures the error quantitatively; $n_{\mathrm{vert}}^{-1/2}\,\|E\|_F$ gives the RMS distance in pixel units (a: 1.143, b: 0.884; c: 0.729, d: 0.250; e: 0.313, and f: 0.102).

5.3.1 Symmetries of the Ellipsoid

The symmetry of the first degree ellipsoid allows for four different standard orientations of the object. Each of them results from the other three by mirroring (rotating by π) about one of the three main axes. The four pose transformations form a commutative group with respect to concatenation. The group is known as Klein's four-group. The concatenation of two mirrorings results in identity if they are about the same axes, and in a mirroring about the third axis if they are different. An object can flip in this way in parameter space and in object space independently, which gives rise to sixteen combinations. In the comparison of two descriptors, the minimal distance resulting from any of the sixteen relative flips is relevant as the dissimilarity measure.

Any π flip leaves the magnitudes of all coefficients constant, however it changes the sign of some of the $c_{l\,k}^{m}$, where $0 \leq k < 3$ enumerates the object space coordinates. A mirroring (π flip) at the $x_{k'}$ axis in object space flips the sign of $c_{l\,k}^{m}$ if $k' \neq k$. For example, a flip around the x_0 axis changes the signs of all $c_{l\,1}^{m}$ and $c_{l\,2}^{m}$. The situation is less simple for flips around the parameter axes. The definition of the spherical harmonics Y_l^m (eqn. 5.3) is helpful to determine which coefficients change sign in response to parameter space flips. $({u_2}^2 - 1)^l$ is an even symmetric polynomial of degree $2l$ in u_2. Its $l+m$-th derivative is a polynomial of degree $2l - (l+m) = l - m$. This derivative is even symmetric when $l - m$ is even, and odd otherwise. Multiplying it with the even function $(1 - {u_2}^2)^{m/2}$ does not change its symmetry and yields a multiple of P_l^m. Constant factors are insignificant for the present symmetry considerations. Y_l^m (up to a constant factor) results from the multiplication by $e^{im\phi} = (u_0 + iu_1)^m$. The real part of the latter is m-symmetric in u_0 and even in u_1. The imaginary part is $(m-1)$-symmetric in u_0 and odd in u_1; it vanishes for $m = 0$. In terms of symmetry, Y_l^m is equivalent to $u_2^{l-m}\, u_0^m\, (1 + i\, u_0\, u_1)$ (only the parity of the exponents is relevant). Table 5.2 summarizes the sign changes.

The formulas in this table cause the sign change pattern that table 5.3 makes explicit. Any flip around one axis is the concatenation of the flips around the other two axes, and the condition for any one sign change is the exclusive disjunction of the other two.

Object recognition tries to match a given object with a similar one out of a collection of known model objects. If it is to use some form of descriptors, it has to describe all model objects and collect their descriptors ahead of time. The recognition then takes two steps. First it

A flip around axis	changes signs of	multiply $\mathrm{Re}(c_l^m)$ by	multiply $\mathrm{Im}(c_l^m)$ by
u_0	$u_1,\ u_2$	$(-1)^{l+m}$	$(-1)^{l+m+1}$
u_1	$u_0,\ u_2$	$(-1)^{l}$	$(-1)^{l+1}$
u_2	$u_0,\ u_1$	$(-1)^{m}$	$(-1)^{m}$

Table 5.2: The sign changes that parameter space flips cause.

l	$m = 0$	1		2		3		4	
		Re	Im	Re	Im	Re	Im	Re	Im
0	–								
1	01	12	02						
2	–	02	12	–	01				
3	01	12	02	01	–	12	02		
4	–	02	12	–	01	02	12	–	01

Table 5.3: The formulas in table 5.2 evaluate to this sign change pattern. A “0”, “1” or “2” entry means this coefficient changes sign when the object is mirrored on the u_0, u_1 or u_2 axis, respectively. The sign of an coefficient marked “–” will never change due to a π rotation about any parameter axis.

describes the unknown object. Then it compares that descriptor with all the descriptors of the models. If any distance is less than a preset dissimilarity threshold – or tolerance –, the unknown object is considered to match the corresponding model. A particular strategy might pick the closest match if several existed. For a large model database, indexing can give significant savings compared to the comparison with *all* model descriptors.

When the task is not only to recognize an object but also to estimate its pose, the parameters of the transformation to standard position, orientation and size must be stored along with each descriptor. The transformation has 7 degrees of freedom, divided in 3 for translation (in $\underline{c}_0^0$), 3 for rotation, and one for scale.

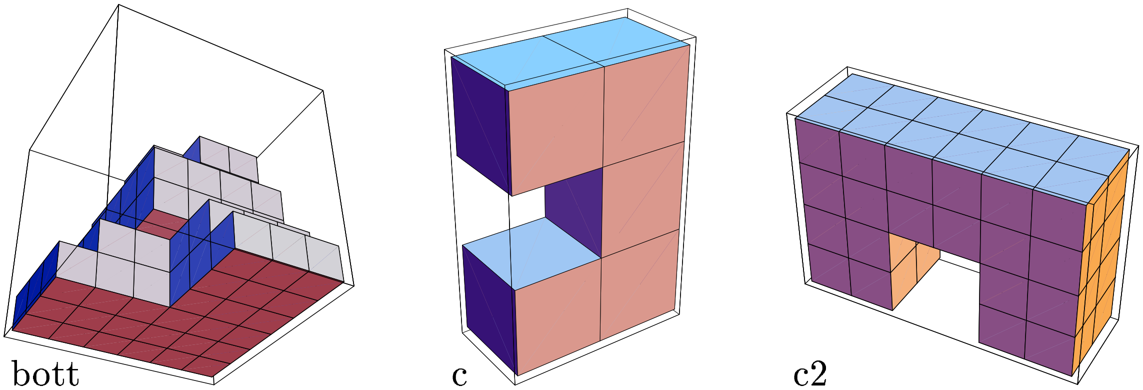
5.3.2 Limitations

A word of caution is in order. The descriptors introduced above are prone to serious quantization artefacts. These can lead to the undesirable situation where two copies of the same object produce different descriptors, although they are only rotated, translated and/or scaled with respect to each other. When the object is too small, i.e. of the order of a few voxel units, a sampled voxel version cannot represent it adequately. Sampling the object thus misses shape aspects that cannot be recovered from the voxel collection. This leads one kind of quantization artefacts which can be corrected by sampling the object at sufficient resolution.

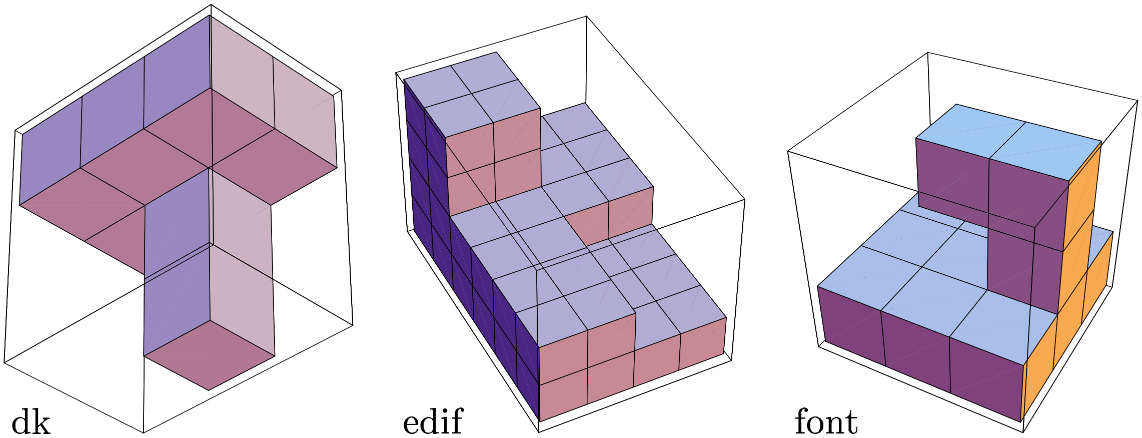
The parametrization by its construction assigns exactly the same amount of parameter space to each surface facet. But there are situations that do not justify this. The same planar area on the surface of an ideal object corresponds to a varying number of facets, depending on its orientation. When the planar surface is parallel to one of the coordinate planes and is sufficiently large, it is discretized into a number of facets approximately equal to its true area, measured in square grid units. When we rotate the surface by $\pi/4$ around one of the two coordinate directions it is parallel to, its quantization turns into a stair, and the number of facets for the same area multiplies by $\sqrt{2}$. When the surface is orthogonal to $(1,1,1)^T$ (diagonal in space), the number of facets is even $\sqrt{3}$ times the true area. The various regions of an object's surface can claim different amounts of parameter space with respect to each other, depending on the rotational position of the object. This leads to a different parametrization and hence to different descriptors, which cannot made invariant. If there is an object that presents this problem, the problem will persist, no matter how fine the discretization is.

5.3.3 Experimental results

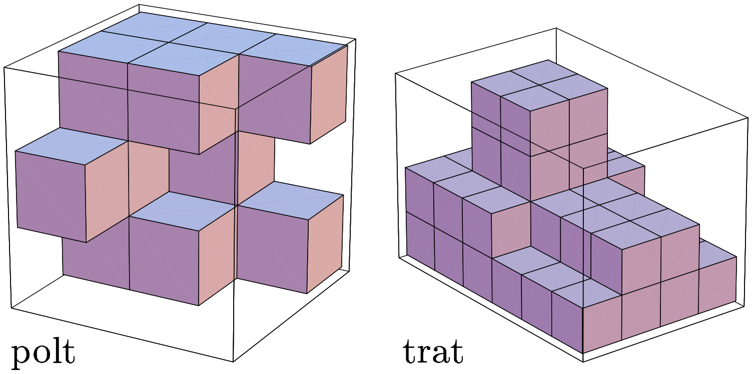
In a student project, Matteo Frapolli compared the spherical shape descriptors for a number of small test objects. Table 5.4 presents the objects he prepared for these experiments. He calculated invariant descriptors for all the test objects. Table 5.5 lists all the distances between the descriptors of any two shapes. Spherical harmonics up to degree 7 have been used, i.e. $n_l = 8$. As a metric, this matrix must be symmetric. The most similar objects are “c” and “c2”, as expected. Discretization artefacts cause the remaining differences; the small objects are at the limit of the resolution.



bott c c2



dk edif font



polt trat

Table 5.4: A collection of simple test objects for shape comparison, together with their working names. Object “c2” is a copy of “c”, enlarged by voxel replication.

	bott	c	c2	dk	edif	font	polt	trat
bott	0	0.3957	0.3910	0.1553	0.2064	0.2588	0.1759	0.2079
c	0.3957	0	0.0159	0.5251	0.1811	0.5255	0.5381	0.2388
c2	0.3910	0.0159	0	0.5296	0.1824	0.5386	0.5094	0.2412
dk	0.1553	0.5251	0.5291	0	0.3133	0.2071	0.2779	0.2884
edif	0.2064	0.1811	0.1824	0.3133	0	0.3237	0.2830	0.1132
font	0.2588	0.5255	0.5386	0.2071	0.3237	0	0.3364	0.3267
polt	0.1759	0.5381	0.5094	0.2779	0.2830	0.3364	0	0.3239
trat	0.2079	0.2388	0.2412	0.2884	0.1132	0.3267	0.3239	0

Table 5.5: Dissimilarity between the descriptors and hence between the shapes of any two of the small test objects.

Larger Objects

The surfaces of the following test objects are parametrized, and their shapes are expressed with spherical harmonic descriptors.

- *c4* A "c"-shaped polyhedron made up from five 4x4x4 voxel cubes.
- *c8* The same object, magnified by a factor of two in all coordinate directions.
- *box A* The voxels fill a rectangular box, which is not aligned with the coordinate axes.
- *box B* Rotating the box A results in a completely different sampling.
- *patella* The patella was extracted from a segmented CT scan data set of a human knee.
- *ventricle* The ventricular system has been segmented from a MRI data set of a hydrocephalus patient[3]. We selected one lateral ventricle. The data has been interpolated to compensate for the aspect ratio of 1:1:6.4 of the original data.

For each object, Figures 5.20, 5.21, and 5.22 present the cuberille interpretation of the input data, a spread-out graph of the parametrization, and the reconstruction from spherical harmonic descriptors. The cylindrical projection we chose for drawing the parametrization shows the

---

[3] Data courtesy Ron Kikinis, M. D., Surgical Planning Lab, Department of Radiology, Brigham and Women's hospital and Harvard Medical School, Boston

true area ratios. The smooth surface of the reconstruction is shaded in a pattern that allows the estimation of the parameter values. These latter parameter values do not coincide with the ones in the middle diagram. They rather differ by the rotation in parameter space that makes the descriptors rotation invariant. The object space rotation is suppressed in the diagrams to show the spatial relation of the original data – shown as a wireframe – with the reconstruction from the descriptor, up to degree 8. In the case of the ventricle, this shows an insufficient degree of detail, but the same value was chosen for comparability.

Table 5.6 summarizes the sizes and differences of the various test objects. Virtually all of the processing time for an object is spent in the optimization. The figure for computation time must be interpreted with caution; it qualifies only the optimization program, which is not necessarily as efficient as possible, and which might be substituted with an out-of-the box optimizer. Times are measured on a HP 9000/735. The number of vertices, n_{vert}, indicates the size of the problem: the optimization has $3\,n_{\mathrm{vert}}$ variables, $2n_{\mathrm{vert}} - 3$ equality constraints and $4\,n_{\mathrm{vert}} - 8$ inequality constraints.

The distance between the descriptors appears to be a valid rough measure of shape dissimilarity. The matrix of distances is symmetric by definition. The two “c”s are most similar to each other. The two boxes are also quite similar. Both these examples illustrate the translation, rotation and scale invariance of the descriptors. The patella is more similar to a box than to a “c”, whereas the ventricle is more similar to a “c” than to any of the other objects.

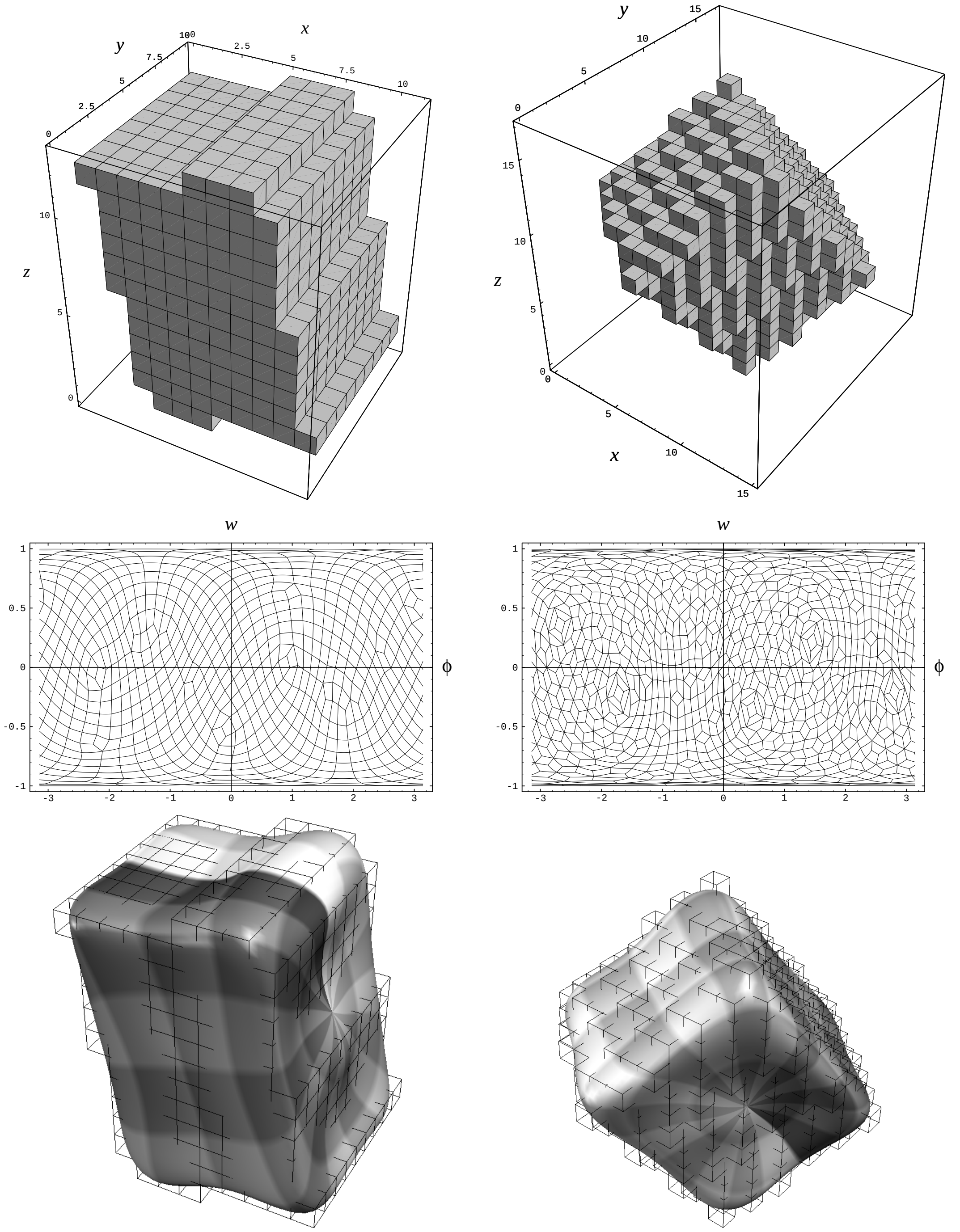


Figure 5.20: Boxes as test objects. Left column: object “box A”; right column: “box B”. From top to bottom in each column: The cuberille interpretation of the input data, the parametrization drawn as a flat net spread out in $(\phi, w = \cos\theta)$ parameter space, and the reconstruction overlayed with a wireframe of the original data.

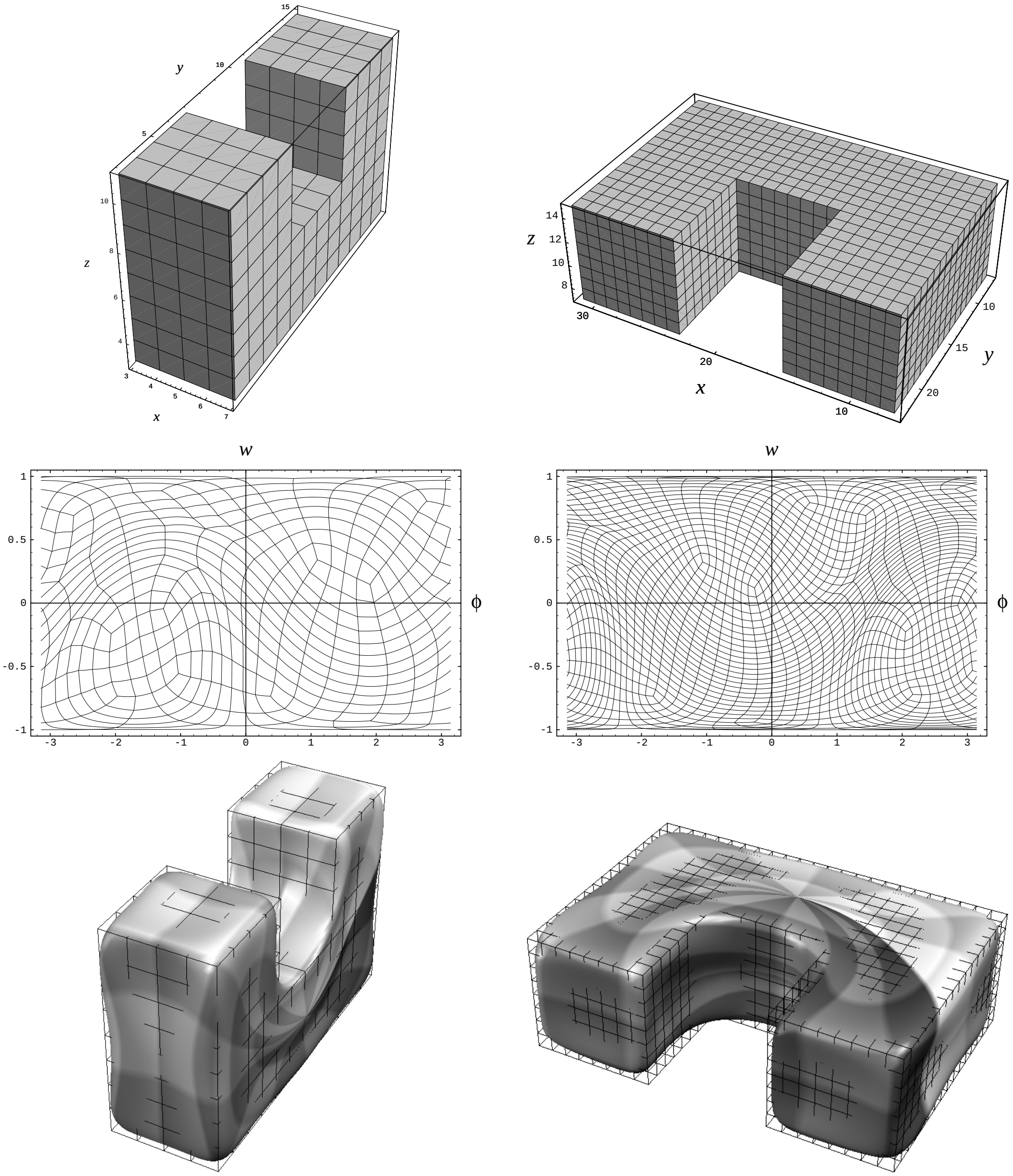


Figure 5.21: "C" letters as test objects. Left column: object "c4"; right column: "c8". From top to bottom in each column: The cuberille interpretation of the input data, the parametrization drawn as a flat net spread out in $(\phi, w = \cos\theta)$ parameter space, and the reconstruction overlayed with a wireframe of the original data.

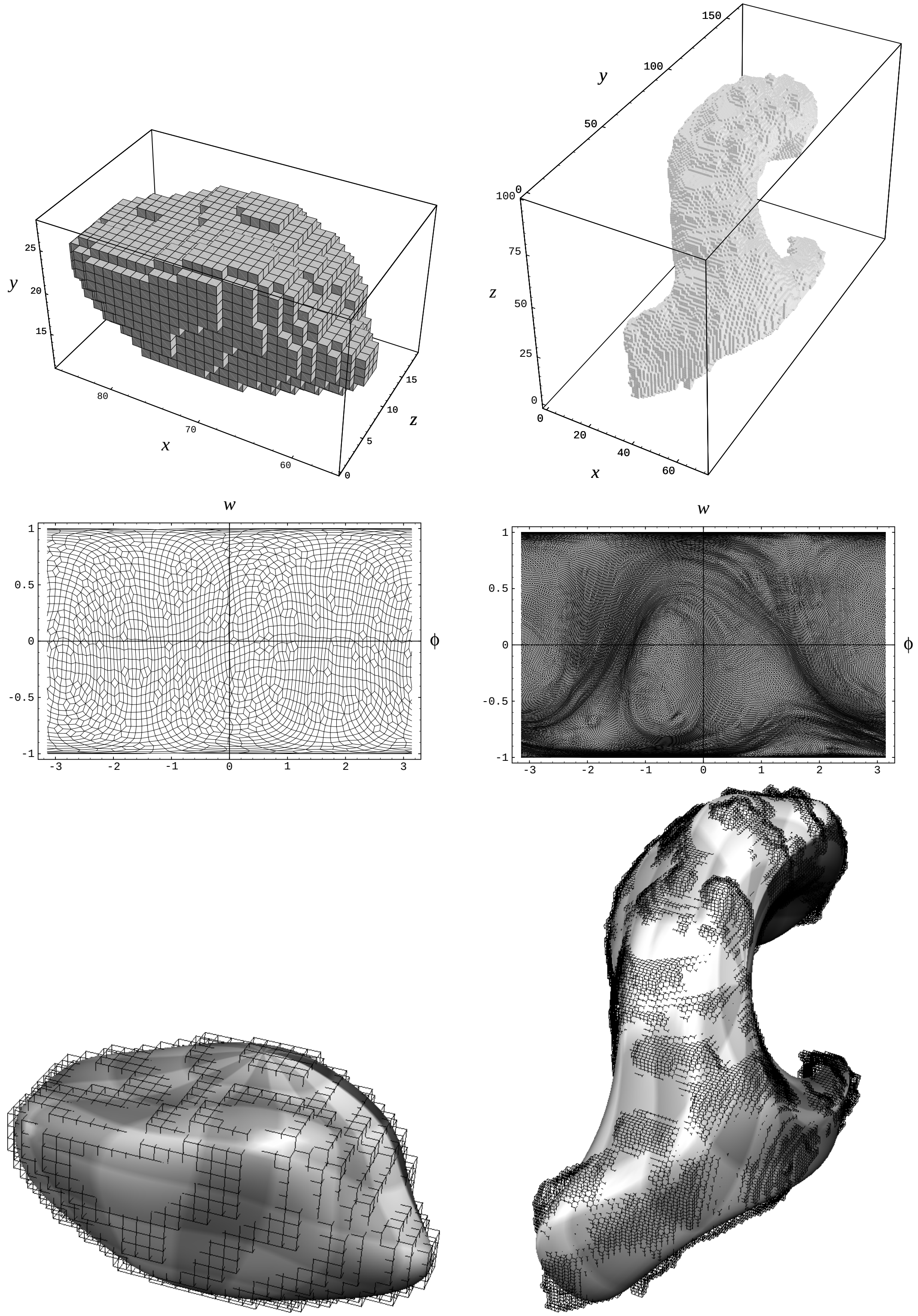


Figure 5.22: Medical test objects.Left column: "patella" object; right column: "ventricle". From top to bottom in each column: The cuberille interpretation of the input data, the parametrization drawn as a flat net spread out in $(\phi, w = \cos\theta)$ parameter space, and the reconstruction overlayed with a wireframe of the original data.

			name			
	box A	box B	c4	c8	patella	ventricle
n_{vert}	628	902	354	1410	2182	37654
time	33 s	267 s	26 s	338 s	536 s	28 h
distance to						
box A	0	0.0241	0.2370	0.2378	0.0673	0.3850
box B	0.0241	0	0.2796	0.2808	0.0859	0.4555
c4	0.2370	0.2796	0	0.0002	0.2309	0.2175
c8	0.2378	0.2808	0.0002	0	0.2299	0.2143
patella	0.0673	0.0859	0.2309	0.2299	0	0.2623
ventricle	0.3850	0.4555	0.2175	0.2143	0.2623	0

Table 5.6: Comparison of the six test objects, including the squared Euclidean distances between their descriptors.

Chapter 6

Conclusions

This thesis presents new techniques to generate explicit parametric representations of *convoluted* object surfaces with minimal distortion. The new techniques allow characterizing 3-D surfaces by *invariant* spherical harmonic shape descriptors. We have overcome traditional limitations of expressing an object surface by explicit parametric representations. The limitations were the restriction of star-shaped objects, the non-uniform spacing of parameters on the object surface, and often a specific choice of the parameter coordinate system with respect to the object geometry. The parametrization technique is described in detail for closed surfaces of simply connected objects, but it generalizes naturally to the unfolding or flattening of complex surface patches (open surfaces with one more edges, including tubes). Other classes of simple 3-D surfaces with different topology, for example tori and pretzels, are not considered.

The following paragraphs summarize the properties of the novel surface analysis techniques.

Parametrization: The unfolding and optimization procedure maps nodes of the surface net onto a sphere. Each node is associated with a voxel vertex, and its position on the sphere can be expressed with two parameters. The procedure imposes no restrictions regarding the geometry of objects and is suitable for surfaces of arbitrary complexity. An initial diffusion of "temperature" on the object's surface achieves a continuous mapping by assigning co-latitude and longitude to each surface vertex. The position of the poles and the geometry of the object produce a clustering of parameters at certain regions of the object surface. This non-uniform distribution of parameter density on the object surface is corrected with a nonlinear optimization technique, which restores the area ratios of the original surface elements in parameter space and minimizes their distortion. The latter is formulated as the goal function of the nonlinear optimization problem and the former as its constraints. The resulting arrangement of the vertex nodes on the sphere (in parameter space) reflects the geometry of the original shape and achieves similar parametrizations for similar shapes, however, it is free to rotate around any axis as no surface points are kept fixed. It must be pointed out that the polar coordinate system for spherical surfaces is only used for the sake of visualization. This coordinate system itself determines a non-uniform tessellation of the sphere and may give a misleading visual impression, making parameters seem denser than they are the poles (see e.g. Figure 3.6). An alternative displaying technique is found by tessellating the sphere into a different pattern of small cells, for example with regular or semiregular polyhedra (Figure 2.9).

The parametrization of object surfaces forms an intermediate representation with the following properties:

- The surface of arbitrarily shaped (but simply connected) objects can be parametrized. Objects are not restricted to a limited family of shapes; even protrusions and intrusions are dealt with appropriately.

- The surface is represented explicitly by the variation of two parameters, expressing the properties of local surface neighborhoods as well as of the global shape.

- The parametrization results in a continuous, one-to-one mapping of surface vertices to a sphere. While varying the vector $\underline{u}$ (or the two parameters θ and ϕ) over the parameter range, each point of the surface is visited exactly once.

- The optimization yields a unique, reproducible solution (except for rotation).

- The parametrization preserves areas exactly and minimizes local distortions which cannot be avoided when mapping an object with corners to a sphere (see location marked by a black dot in Figures 3.6 a, b and c). The uniformity of the parametrization is important for a subsequent shape description, as illustrated in Figure 5.19.

The parametrization technique is potentially interesting for applications where a mapping of convoluted object surfaces to a surface of minimal curvature is required. The unfolding or flattening process with minimization of distortions generates a representation which could serve as a useful intermediate surface description for many structure analysis processes. The only restriction, i.e. the presence of the closed surface of a simply connected object, highlights the generality of the approach.

As discussed previously, the unfolding is in principle not restricted to closed surfaces. Ongoing developments focus on a similar technique for flattening parts of surfaces onto planar charts. This procedure could be interesting for the comparative analysis and description of convoluted surface patches. Practical applications can be found in brain research, for example, where regional cortical patterns of the human brain are qualitatively and quantitatively analyzed.

Applications are still constrained by the efficiency of the nonlinear optimization. Although the method itself poses no restriction on the maximum number of object vertices, the commonly available optimization routines cannot be applied for a large number of vertices (exceeding several hundreds). In real applications, e.g. the analysis of volume data in medicine, one can expect to deal with object surfaces with up to one million voxel vertices. We are presently developing an optimization technique which takes into account the sparsity of the problem and the specific nature of the local constraints.

Shape description: The new parametrization allows representation of object surfaces of arbitrary complexity. As one possible approach to global shape analysis, it enables us to expand an object surface into a series of spherical harmonic functions. The numerical coefficients in the Fourier series represent an object-centered, surface-oriented descriptor of the object's form. Surface description with harmonic descriptors is no longer restricted to star-shaped objects but can now be applied to a broad class of shapes. Invariance has to be considered as one of the most important properties of shape description, as it only allows a comparative analysis between different objects or a match between objects and models. With the development of new scale and rotation independent descriptors we obtain a *global, object-centered shape description* which is invariant to standard transformations (rotation, translation and scaling). The invariant positioning of the object and of the parameter net are based on the analysis of harmonic descriptors up to the first degree, defining the three main axes of the ellipsoid. The symmetry of this low frequency representation determines a general 3-D object only up to four different positions. Including coefficients of higher degree could disambiguate the different cases and avoid a matching using four different object descriptions. For example, the second degree contribution can be evaluated at the extremal points of the first degree ellipsoid.

Applications of global object representation and description in computer vision and image analysis are imminent. Generality with respect to object complexity, invariance to standard transformations, and descriptive power in terms of object geometry are the critical issues for shape-based categorization and comparison of 3-D objects. For instance, robot vision and medical image analysis are dealing with recovering the global shape characteristics of objects. Whereas the former most often deals with a small number of views of objects and hence only a partial surface description, modern scanning techniques in medicine can provide full 3-D images. Mapping of convoluted surface structures and high level 3-D shape descriptions of anatomical objects (e.g. the heart cavities, the ventricular system or cortical substructures of the brain) will play a significant role in the analysis of shape dissimilarities, morphological deformations and in the comparison of malformed with "normal" shape structures. The overall shape is captured by a small number of parameters, expressing structural details at various scales with coefficients of different degrees. The continuous analytical description of the approximated surface permits to compute local differential characteristics, e.g. principal curvature [33]. Inferring the differential structure would result in a characterization of important landmarks, e.g., for registration of different 3-D objects.

Acknowledgements

In the first place I would like to thank Prof. Dr. Guido Gerig, my supervisor, for being supportive throughout my work and for the many fruitful discussions we had. He gave me freedom to explore new ideas. He carefully read drafts of this thesis and gave me invaluable suggestions for improvement.

I am greatly indebted to Prof. Dr. Olaf Kübler for the open and friendly atmosphere he created in the computer vision group. He provided a very inspiring research environment, and he was always helpful when I had questions about mathematics or physics.

I am grateful to Prof. Dr. Nicholas Ayache for his spontaneous willingness to co-examine the thesis.

Special thanks are due to the colleagues Dr. Martin Peter, Martin Rutishauser, Dr. Robert L. Ogniewicz, and Manuel Sturm for taking care of the lab's computing infrastructure and for all the help they gave me whenever I had questions.

I am very thankful to Dr. Gladys Monagan, who critically and very carefully reviewed the manuscript. I greatly appreciate both her suggestions for improving clarity and her help around the difficulties I experienced with the English language.

Chapter 7

Appendix: Polar Coordinates for Ω_2 and Ω_3

The generalized sphere Ω_{m+1} in $I\!R^{m+1}$ is an m-dimensional manifold (hypersurface). It is defined as

$$\Omega_{m+1} \quad = \quad \{x = (x_0, \dots x_m)^T \,\big|\, |x|^2 = x_0^2 + \dots x_m^2 = 1\} \, . \tag{7.1}$$

A location on Ω_{m+1} is identified by $m + 1$ Cartesian or m polar coordinates. The cases $m = 1$ (the circle) and $m = 2$ (the sphere) are important in this thesis. The Cartesian coordinates u_0 and u_1 (and u_2 in 3D) are related to the polar ϕ (and θ in 3D) by the relations

$$\begin{aligned} u_0 &= \cos\phi \\ u_1 &= \sin\phi \end{aligned} \tag{7.2}$$

in 2D and

$$\begin{aligned} u_0 &= \sin\theta\,\cos\phi \\ u_1 &= \sin\theta\,\sin\phi \\ u_2 &= \cos\theta \end{aligned} \tag{7.3}$$

in 3D.

Bibliography

[1] D. H. Ballard and Ch. M. Brown. *Computer Vision.* Prentice-Hall Inc, Englewood Cliffs, New Jersey, 1982.

[2] Richard H. Bartels, John C. Beatty, and Brian A. Barsky. *An Introduction to Splines for use in Computer Graphics and Geometric Modelling.* Morgan Kaufmann Publishers, Inc., 95 First Street, Los Altos, California 94022, 1987.

[3] Pierre E. Bézier. *Emploi des machines à commande numérique.* Masson et Cie, Paris, 1970. Translated by A. Robin Forrest and Anne F. Pankhurst (1972) as *Numberical Control – Mathematics and Applications*, John Wiley & Sons, New York.

[4] Pierre E. Bézier. Mathematical and opractical possibilities of unisurf. In Robert E. Barnhill and Richard F. Riesenfeld, editors, *Computer Aided Geometric Design*, pages 127–152. Academic Press, New York, 1974.

[5] Pierre E. Bézier. *Essai de Définition Numérique des Courbes et des Surfaces Expèrimentales.* PhD thesis, l' Université Pierre et Mearie Curie, Paris, February 1977.

[6] H. Blum. A transformation for extracting new descriptors of shape. In W. Walthen-Dunn, editor, *Models for the Perception of Speech and Visual Form.* MIT Press, Cambridge, MA, 1967.

[7] Christian Brechbühler, Guido Gerig, and Olaf Kübler. Surface parametrization and shape description. In Richard A. Robb, editor, *Visualization in biomedical computing 1992*, volume Proc. SPIE 1808, pages 80–89, 1992.

[8] Christian Brechbühler, Guido Gerig, and Olaf Kübler. Towards representation of 3d shape: Global surface parametrization. In C. Arcelli, L. P. Cordella, and G. Sanniti di Baya, editors, *Visual Form: Analysis and Recognition*, pages 79–88, New York and London, 1992. Plenum Press.

[9] Hervé Delingette. Simplex meshes: a general representation for 3d shape reconstruction. Technical Report 2214, Unité de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 Sophia-Antipolis Cedex (France), March 1994.

[10] Hervé Delingette, G. Subsol, S. Cotin, and J. Pignon. A craniofacial surgery simulation testbed. In Richard A. Robb, editor, *Visualization in Biomedical Computing 1994*, volume 2359 of *Proc. SPIE*, pages 607–618, Rochester, MN, October 1994. SPIE.

[11] Tracy L. Faber and Ernest M. Stokely. Orientation of 3-d structures in medical images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(5):626–633, September 1980.

[12] R. Fletcher. *Practical Methods of Oprimization.* John Wiley & Sons, Chichester, New York, Brisbane, Toronto, Singapore, 2nd edition, 1987. “A Wiley-Interscience Publication”.

[13] H. Freeman. *Picture Processing and Psychopictorics*, chapter Boundary encoding and processing, pages 241–306. Academic Press, NewYork, 1970.

[14] Jörg Friedrich. *Formanalyse von Partikelkollektiven.* Fortschrittsberichte vdi, reihe 3: Verfahrenstechnik, nr. 268, Universität Karlsruhe; Institut für Mechanische Verfahrenstechnik und Mechanik, 1991.

[15] W. Gellert, Dr. H. Küstner, Dr. M. Hellwich, and H Kästner, editors. *Grosses Handbuch der Mathematik.* Buch und Zeit Verlagsges. m. b. H. Köln, 1969.

[16] P. E. Gill, W. Murray, and M. H. Wright. *Practical optimization.* Academic Press, London a.o., 1981.

[17] D. Gordon and J. K. Udupa. Fast surface tracking in three-dimensional binary images. *Computer Vision, Graphics, and Image Processing*, 45(2):196–214, February 1989.

[18] W. Greiner and H. Diehl. *Theoretische Physik - Ein Lehr- und Übungsbuch für Anfangssemester*, volume 3: Elektrodynamik. Verlag Harri Deutsch, Zürich und Frankfurt am Main, 1986.

[19] Gabor T. Herman and Hsun Kao Liu. Three-dimensional display of human organs from computer tomograms. *Computer Graphics and Image Processing*, 9(1):1–21, January 1979.

[20] M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *J. Res. Nat. Bur. Stand.*, 49:409–436, 1952.

[21] M.-K. Hu. Visual pattern recognition by moment invariants. *IRE Transactions on Information Theory*, IT-8:179–187, Feb 1962.

[22] D. N. Kennedy, J. Sacks, P. A. Filipek, and V. S. Caviness. Three-dimensional fourier shape analysis in magnetic resonance imaging. In Peder C. Pedersen and Banu Onaral, editors, *Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, volume 12/1, pages 78–79, Philadelphia, PA, USA, October 1990.

[23] F. P. Kuhl and Ch. R. Giardina. Elliptic fourier features of a closed contour. *Computer Graphics and Image Processing*, 18(3):236–258, March 1982.

[24] Chung Nim Lee, Timothy Poston, and Azriel Rosenfeld. Holes and genus of 2d and 3d digital images. *Graphical Models and Image Processing*, 55(1):20–47, 1993.

[25] Chong-Huah Lo and Hon-Son Don. 3-d moment forms: Their construction an application to object identification and positioning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(10):1053–1064, October 1989.

[26] L. Nirenbarg. A strong maximum principle for parabolic equations. *Commun. Pure Appl. Math., vol. VI*, pages 167–177, 1953.

[27] Robert Obniewicz, Gábor Székely, Markus Näf, and Olaf Kübler. Medial manifolds and hierarchical description of 2d and 3d objects with applications to mri data of the human brain. In *Proceedings SCIA, 8th Scandinavian Conference on Image Analysis*, pages 875–883, Tromso, Norway, May 1993.

[28] Robert Ogniewicz. *Discrete Voronoi Skeletons.* PhD thesis, ETH Zürich, 1993.

[29] E. Persoon and K.S. Fu. Shape discrimination using fourier descriptors. *IEEE Trans. Systems, Man and Cybernetics SMC*, 7(3):170–179, March 1977.

[30] C. Pommerell and W. Fichtner. PILS: An iterative linear solver package for ill-conditioned systems. In *Proceedings of the Supercomputing '91, Albuquerque, New Mexico, November 18-22*, pages 588–599, 1991.

[31] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. Vetterling. *Numerical recipes in C - The art of scientific computing.* Cambridge University Press, Cambridge, 1988.

[32] Firooz A. Sadjadi and Ernest L. Hall. Three-dimensional moment invariants. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(2):127–136, March 1980.

[33] P. T. Sander and S. W. Zucker. Inferring surface trace and differential structure from 3-d images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(9):833–854, September 1990.

[34] Robert Schweikert. Kartierung der Hirnoberfläche. Diplomarbeit IIIB, Swiss Federal Institue of Technology (ETH), Zürich, IKT / Image Science, March 1995.

[35] Jonathan Richard Shewchuk. An introduction to the conjugate gradient method without the agonizing pain. An electronic copy of this article is available by anonymous FTP to `WARP.CS.CMU.EDU` (IP address 128.2.209.103) under the filename `quake-papers/painless-conjugate-gradient.ps`, August 1994. School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213.

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

[37] Lawrence H. Staib and James S. Duncan. Deformable fourier models for surface finding in 3d images. In Richard A. Robb, editor, *Visualization in biomedical computing 1992*, volume Proc. SPIE 1808, pages 90–104, 1992.

[38] M. R. Teague. Image analysis via the general theory of moments. *J. Opt. Soc. Amer.*, 70:920–930, August 1980.

[39] Jean-Philippe Thirion. The extremal mesh and the understanding of 3d surfaces. Technical report, Unité de recherche INRIA Sophia-Antipolis, 1993.

[40] Jayaram K. Udupa and Venkatramana G. Ajjangadde. Boundary and object labelling in three-dimensional images. *Computer Vision, Graphics, and Image Processing*, 51(3):355–369, September 1990.

Curriculum Vitae

Christian Michael Brechbühler-Miškuv

born on the 13th of March 1964 in Baden, Switzerland
married since May 1995.

Schools:

1971-1976: Primarschule Brugg - Lauffohr
1976-1980: Bezirksschule Brugg
1980-1983: Kantonsschule Aarau
Maturität Typ C in the fall of 1983
1983-1988: ETH Zürich (Department IIIC, Computer Science)
Diploma in the spring of 1988
1995, July: PhD in Technical Sciences

Occupation:

1988-1994: Assistant at the Swiss Federal Institute of Technology (ETH) Zürich, Communication Technology Laboratory
Image Science Group
1995: Postdoc (Oberassistent) in the Image Science Group.