PhD Thesis

Theory and Applications of
Parameterizing Triangulations

Kai Hormann

November 26, 2001

A team is a team is a team.
William Shakespeare (1564–1616)

Acknowledgements

First of all, I want to express my gratitude to my supervisor Prof. Günther Greiner for supporting me through all stages of this thesis and for giving me the scientific liberty to develop and pursue my own ideas. Working with him has always been a pleasure and the benefit of this collaboration invaluably contributed to the success of this work.

Additionally, I would like to thank Prof. Hans-Peter Seidel from the Max-Planck-Institute of Computer Science in Saarbrücken for his support and for reading and reviewing this thesis as well as Prof. Hans Strauß from the Institute of Applied Mathematics at the University of Erlangen, who kindly agreed to be a member of my examination board. I also wish to thank Prof. Heinrich Niemann from the Department of Computer Science at the University of Erlangen for leading this board and assisting me as the spokesman of the DFG project SFB 603 which financially supported my work.

I am grateful to Prof. Nira Dyn and Prof. David Levin from the Department of Applied Mathematics at the Tel Aviv University for their suggestions on optimizing triangulations. Furthermore, I am very much obliged to Prof. Michael S. Floater from the Norwegian Research Foundation SINTEF and the Department of Informatics at the University of Oslo for his inspiring ideas on parameterizations and for giving me the opportunity of a six-month research scholarship in Oslo which was supported by the European Union research project MINGLE.

I would also like to thank all my colleagues at the Computer Graphics Group in Erlangen and at Sintef in Oslo for the congenial and companionable atmosphere I had the pleasure to enjoy, especially Ulf Labsik for our fruitful collaborations on the subject of remeshing and my office mates Dr. Salvatore Spinello and Grzegorz Soza for cheering me up occasionally.

My deep respect is due to my secondary school teacher Thomas Müller who sparked my interest in mathematics and computer science and always encouraged me to go into scientific research.

Finally, and above all I want to express my deepest love and gratitude to my family, especially to my parents for supporting my career both ideologically and financially.

Contents

Introduction **1**

I Theory 4

1 Parameterizing Triangulations **5**
1.1 Related Work . . . 7
1.2 Linear Methods . . . 8
1.2.1 Univariate Spring Model . . . 9
1.2.2 Bivariate Spring Model . . . 11
1.2.3 Harmonic Maps . . . 13
1.2.4 Convex Combination Maps . . . 20
1.2.5 Parameterizing the Boundary . . . 23
1.2.6 Solving the Linear System . . . 27
1.3 Most Isometric Parameterizations . . . 34
1.3.1 Shape Deformation of Triangles . . . 34
1.3.2 Minimizing the MIPS Energy . . . 41
1.3.3 Hierarchical Optimization . . . 49
1.4 Arbitrary Topology . . . 51

II Applications 56

2 Triangulating Point Clouds **57**
2.1 Related Work . . . 57
2.2 Meshless Parameterization . . . 58
2.2.1 Local Neighbourhoods . . . 59
2.2.2 Patch Topology . . . 61
2.2.3 Spherical Topology . . . 63
2.3 Optimizing Triangulations . . . 68
2.3.1 Discrete Curvatures . . . 69
2.3.2 Minimizing Discrete Smoothness Functionals . . . 72

3 Remeshing Triangulations **76**
3.1 Related Work 77
3.2 Subdivision Connectivity Triangulations 78
3.2.1 Planar Remeshing 80
3.2.2 Spatial Remeshing 82
3.3 Regular Quadrilateral Meshes 87

4 Fitting Smooth Surfaces **90**
4.1 Related Work 91
4.2 Tensor Product B-Spline Surfaces 92
4.3 The Variational Approach 94
4.3.1 Parameter Correction 94
4.3.2 Interpolation and Least Squares Approximation 96
4.3.3 Smoothness Functionals 102
4.3.4 Hierachical Surface Fitting 107
4.3.5 Chebychev Approximation 110
4.3.6 Iteratively Reweighted Least Squares 113
4.4 Indirect Approximation 118

Conclusion **126**

Future Work **128**

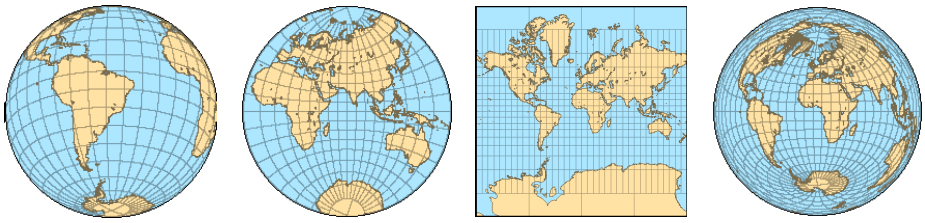
Bibliography **129**

L'universo è scritto in lingua matematica.
Galileo Galilei (1564–1642)

Introduction

What is the shape and the extent of our home, the Earth, and of the cosmos which contains it? Next to the questions about the meaning of life, the universe, and everything [1, 108], this is the oldest intellectual challenge facing the human mind and has been the province of religion, poetry, and myth. In the western scientific tradition this problem resolved itself into the twin enterprises of mapping the earth and the heavens. The Greek astronomer *Claudius Ptolemy* (100–168 A.D.) was the first known to produce the data for creating a map showing the inhabited world as it was known to the Greeks and Romans of about 100–150 A.D. In his work *Geography* he explains how to project a sphere onto a flat piece of paper using a system of gridlines—longitude and latitude.

As we all know from peeling oranges and trying to flatten the peels on a table, the sphere cannot be projected onto the plane without distortions and therefore certain compromises must be made. Figure 1 shows some examples. The orthographic projection (a), which was known to the Egyptians and Greeks more than 2 000 years ago, modifies areas and angles, but the directions from the center of projection are true. Probably the most widely used projection is the stereographic projection (b) usually attributed to *Hipparchus* (190–120 B.C.). It is a conformal projection which preserves shapes and angles at the expense of areas. It also preserves circles, no matter how large (great circles passing on the central point are mapped into straight lines), but a *loxodrome* is plotted as a logarithmic spiral. A loxodrome is a line of constant bearing and of vital importance in navigation. In 1569, the Flemish cartographer *Gerardus Merca-*



(a) (b) (c) (d)

Figure 1: Orthographic (a), stereographic (b), Mercator (c), and Lambert (d) projection of the Earth.

Figure 2: The constellations Sagittarius, Leo, Gemini, and Libra.

tor (1512–1594), whose goal it was to produce a map which sailors could use to determine courses, overcame this drawback by presenting the conformal cylindrical *Mercator projection* (c) which draws every loxodrome as a straight line. The mathematical basis of both projections was developed in 1851 by *Bernhard Riemann* (1826–1866) in his dissertation [111] and is known today as the Riemann Mapping Theorem. It says that a three-dimensional curved surface can be flattened while preserving the angular information. Finally, an equal area projection (d) was developed by *Johann Heinrich Lambert* (1728–1777) in 1772.

All these projections can be seen as functions that map a part of the sphere's surface $M \subset S^2$ to a planar domain Ω and the inverse of this mapping, $\phi : \Omega \to M$, is usually called a *parameterization* of M over Ω. The principles of parametric surfaces were developed by *Carl Friedrich Gauß* (1777–1855), mostly in [52].

This dissertation is about parameterizations of a special kind of surfaces, namely *triangulations*. Triangulations are piecewise linear surfaces and play an important role today in science, engineering, and entertainment as well. In fact, the representation of surfaces in a digital environment requires some kind of discretization and triangulations are the most common approach for two reasons. Firstly, they can be displayed very efficiently since modern graphics hardware is optimized for rendering triangles and secondly, they are capable of approximating a surface with arbitrary accuracy.

The first part of this thesis deals with the theory of parameterizing triangulations. Like projecting the sphere, this cannot be done without distortions in general and, analogously, certain compromises must be made. So, in Chapter 1 the requirements of a good parameterization and the advantages and disadvantages of several methods will be discussed.

In the second part of this thesis, the techniques of parameterizing triangulations will be utilized to reconstruct surfaces from three-dimensional point clouds. A major difficulty with point clouds is that they can be without any order and therefore are very hard to handle, a problem we are all familiar with. Anyone who has ever looked up to the starry skies at night knows how easily one can get lost in this biggest natural point cloud we call the Milky Way. Dating back to the Babylonians, men have always tried to organize the stars for a better orientation by grouping and connecting them with virtual lines. Figure 2 shows some examples of constellations originating from Greek mythology.

In Chapter 2 a similar concept is used to organize point clouds by connecting certain pairs of points with virtual lines. Some of the parameterization methods for triangulations also apply to point clouds with this more general kind of

connectivity structure and can be used to parameterize them over a planar domain. The correspondences between data points and associated parameter points obtained that way are essential for any parametric surface reconstruction method and the quality of a reconstructed surface usually depends on the quality of these correspondences. The better the arrangement of the parameter points in the parameter domain reflects the arrangement of the associated data points on the surface of the object from which they were sampled, the better the reconstructed surface approximates the shape of that object.

The simplest type of surface reconstruction is interpolation with piecewise linear surfaces, which yields in fact a triangulation of the data points. Due to the low quality of point cloud parameterizations, these triangulations usually do not represent the shape of the original surface very well, but after optimizing them with respect to some smoothness criterion they do. Once the data points are organized in such a smooth triangulation, the techniques from Chapter 1 can be applied to obtain parameterizations of high quality which are further used for other surface reconstruction methods.

Chapter 3 describes the approximation of data points with subdivision connectivity triangulations and quadrilateral meshes, a problem also known as *remeshing*. Due to its hierarchical structure, especially the former type of surface is of particular interest to people dealing with computer graphics and conversion of an arbitrary triangulation into one with subdivision connectivity can be looked upon as an important task. Chapter 4 discusses various aspects of surface reconstruction with tensor product B-splines. The main application of this process is *reverse engineering*, as it occurs, for example, when a designer's clay model shall be represented in terms of a mathematically described surface, so that it can further be processed by an engineer with a CAD software.

Finally, it seems appropriate to mention that the results of this thesis were partly published in [71, 69, 45, 46] (Chapter 1), [33] (Chapter 2), [86, 70, 72] (Chapter 3), and [58, 68] (Chapter 4).

Part I

Theory

Si les triangles faisaient un Dieu,
ils lui donneraient trois côtés.
Charles de Montesquieu (1689–1755)

Chapter 1

Parameterizing Triangulations

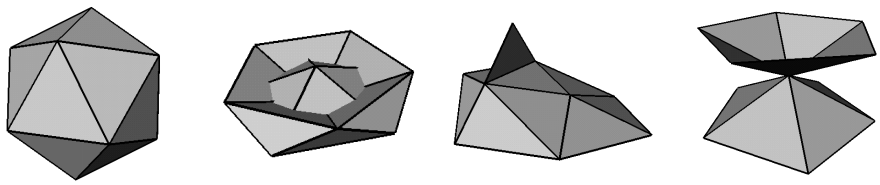
In this chapter we review several ways of constructing a parameterization of a triangulation, i.e. a surface which consist of triangles only. By a triangle T we understand the convex hull $T = [\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3]$ of three non-collinear points $\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3 \in \mathbb{R}^3$ and a triangulation is defined as follows.

Definition 1.1 *Let $\mathcal{T} = \{T_1, \ldots, T_n\}$ be a set of triangles in $\mathbb{R}^3$. We call $\mathcal{T}$ a* triangulation *if*

(i) *$T_i \cap T_j$, $i \neq j$, is either empty, a common vertex, or a common edge and*

(ii) *the union of the triangles $\Omega_{\mathcal{T}} = \bigcup_{i=1}^{n} T_i$ is an orientable 2-manifold.*

Figure 1.1 clarifies this definition by showing some examples. We call $\Omega_{\mathcal{T}}$ the *surface* of the triangulation $\mathcal{T}$ and further let $V = V(\mathcal{T})$ denote the set of *vertices* and $E = E(\mathcal{T})$ the set of *edges* in $\mathcal{T}$. If $\Omega_{\mathcal{T}}$ has a boundary we distinguish between the disjoint sets of *interior* and *boundary* vertices V_I and V_B. Two distinct vertices $\boldsymbol{v}, \boldsymbol{w} \in V$ are *neighbours* if they are the end points of an edge



(a) (b) (c) (d)

Figure 1.1: The icosahedron in (a) is a triangulation, but the object in (b) consists of triangles that intersect other than in a common vertex or edge and the surfaces in (c) and (d) are not 2-manifolds.

$e = [\boldsymbol{v}, \boldsymbol{w}] \in E$ and for $\boldsymbol{v} \in V$ we let

$$N_{\boldsymbol{v}} = \{\boldsymbol{w} \in V : [\boldsymbol{v}, \boldsymbol{w}] \in E\}$$

be the set of neighbours of $\boldsymbol{v}$. It will also be useful to define $S_k(\mathcal{T})$ to be the linear space of all continuous functions $s : \Omega_{\mathcal{T}} \to \mathbb{R}^k$ which are linear over each triangle $T \in \mathcal{T}$. Thus each of the k components of s belongs to $S_1^0(\mathcal{T})$, where $S_d^r(\mathcal{T})$ is the usual notation for the spline space of piecewise polynomials of degree d and smoothness C^r over $\Omega_{\mathcal{T}}$.

In general a *parameterization* ϕ of a triangulation $\mathcal{T}$ over a parameter domain $\Omega \subset \mathbb{R}^k$ is a homeomorphism between this domain and the surface of $\mathcal{T}$,

$$\phi : \Omega \to \Omega_{\mathcal{T}}.$$

From differential geometry we know that such a homeomorphism and the *inverse parameterization* $\psi = \phi^{-1}$ exist if and only if Ω and $\Omega_{\mathcal{T}}$ are *topologically equivalent*, i.e. Ω is a 2-manifold with the same number of *boundaries* and *handles* as $\Omega_{\mathcal{T}}$. The number of handles is also called the *genus* of a manifold. A sphere has genus zero, the genus of a torus is one, etc.

For triangulations the most natural class of parameterizations are piecewise linear functions and we want to discuss only these mappings in this chapter. Since the inverse of a bijective piecewise linear function is piecewise linear itself, we can say that we consider only those ϕ for which $\psi = \phi^{-1}$ is an injective element of $S_k(\mathcal{T})$. Due to the piecewise linearity, ψ induces a triangulation

$$\mathcal{S} = \{\psi(T) : T \in \mathcal{T}\},$$

with $\Omega_{\mathcal{S}} = \Omega$ which is equivalent to $\mathcal{T}$ in the sense that vertices, edges and triangles of $\mathcal{S}$ and $\mathcal{T}$ naturally correspond to each other,

$$\psi(V(\mathcal{T})) = V(\mathcal{S}), \qquad \psi(E(\mathcal{T})) = E(\mathcal{S}), \qquad \psi(\mathcal{T}) = \mathcal{S},$$

and

$$\phi(V(\mathcal{S})) = V(\mathcal{T}), \qquad \phi(E(\mathcal{S})) = E(\mathcal{T}), \qquad \phi(\mathcal{S}) = \mathcal{T}.$$

We call the triangles of $\mathcal{S}$ *parameter triangles* and $\mathcal{S}$ itself the *parameter triangulation*. Note that ϕ and ψ are uniquely determined by the images $\psi(\boldsymbol{v})$ which we call the *parameter points* or *parameter values* of the vertices $\boldsymbol{v} \in V$. Hence the task of parameterizing $\mathcal{T}$ is equivalent to finding parameter values $\psi(\boldsymbol{v}) \in \Omega$, one for each vertex $\boldsymbol{v} \in V$. Since we want ψ to be injective we have to assure the parameter points to be arranged such that the parameter triangles do not overlap and the parameter triangulation is valid in the sense of Definition 1.1.

After reviewing the related work in Section 1.1 we study the parameterization of *simple triangulations* with one boundary and genus zero in Sections 1.2 and 1.3. Since such triangulations are topologically equivalent to a disc they can be parameterized over a planar domain $\Omega \subset \mathbb{R}^2$. The usual way of solving the parameterization problem for this type of triangulations is to construct an injective function $\psi \in S_2(\mathcal{T})$, set $\Omega = \psi(\Omega_{\mathcal{T}})$, and finally let $\phi = \psi^{-1}$ be the parameterization of $\mathcal{T}$ over Ω.

If the surface $\Omega_{\mathcal{T}}$ of a triangulation $\mathcal{T}$ is the graph of a piecewise linear bivariate function f with vertices $\boldsymbol{v}_i = (x_i, y_i, f(x_i, y_i))$ then we can simply take as $\psi \in S_2(\mathcal{T})$ the orthogonal projection from $\mathbb{R}^3$ into the xy-plane, $\psi(x, y, z) = (x, y)$. Such triangulations frequently turn up in the fields of geology, meteorology, cartography, and others, but we are more interested in non-projectable triangulations $\mathcal{T}$ for which the projection into any plane will result in non-valid parameter triangulations. And even for projectable triangulations we may prefer a different parameterization if the projection leads to highly distorted triangles which are undesirable in many applications, e.g. texture mapping.

In Section 1.2 we show how to determine an injective $\psi \in S_2(\mathcal{T})$ by solving a linear system. This approach is motivated by a physical spring model which is also capable of explaining the usual curve parameterization techniques. The main drawback of these linear methods is that they require the parameter points of the boundary vertices to be specified in advance and it is not always clear how to choose them. In Section 1.3 we present a non-linear method which overcomes this drawback. It is based on the observation that ψ normally deforms the shape of the triangles of $\mathcal{T}$ and aims at minimizing these deformations.

The parameterization of more complicated triangulations with arbitrary topology is discussed in Section 1.4. The main idea to tackle this problem is to split the triangulation into several disjoint patches that are simple triangulations and parameterize each patch by using one of the methods discussed in Section 1.2. Special care has to be taken in order to guarantee continuity of the individual parameterizations along the boundaries of the patches.

1.1 Related Work

Especially in computer graphics the application of texture mapping soon became a driving force in the development of parameterization techniques for simple triangulations.

Bennis et al. [11] proposed a method based on differential geometry. They map isoparametric curves of the surface onto curves in the parameter domain so that the geodesic curvature at each point is preserved. The parameterization is then extended to both sides of that initial curve until some distortion threshold is reached. Since this requires the triangulation to be split into several independent regions they do not obtain one global parameterization but rather a collection of local ones, i.e. an atlas of parameterizations.

Maillot et al. [96] suggested to reduce the distortion between the shape of the triangles of $\mathcal{T}$ and the corresponding parameter triangles by minimizing the *Green-Lagrange deformation tensor* $\|\mathbf{I}_\phi - I\|^2$ that measures the distance of the first fundamental form of ϕ to the unit matrix in some 2×2 matrix norm. However, this leads to a highly non-linear functional and since they were interested in an interactive method they minimized a simplified version of the Green-Lagrange deformation tensor instead.

While Maillot et al. aimed at minimizing the distortion by using a functional that preserves lengths, Lévy and Mallet [92] presented a functional that pre-

serves perpendicularity and constant spacing of the isoparametric curves traced on the surface, and Sheffer and de Sturler [123] used a functional that preserves the angles of the triangles. The methods discussed in Sections 1.2.3 and 1.3 are also based on the idea of minimizing shape deformation.

Motivated by the problem of surface fitting as it occurs in several applications (see Chapter 4), Ma and Kruth [95] proposed to circumvent the problem of non-projectability to a plane by projecting the vertices $\boldsymbol{v} \in V$ of the triangulation onto a parametric base surface $S : \Omega \to \mathbb{R}^3$ instead and taking the parameter values of the projected vertices as $\psi(\boldsymbol{v})$, thereby exploiting the known parameterization of S. But since this method works only if the shape of the base surface is close to $\Omega_{\mathcal{T}}$ the problem of finding a suitable base surface for arbitrary triangulations becomes very difficult.

The problem of parameterizing triangulations with arbitrary topology has first been addressed by Eck et al. in [35]. By growing *Voronoi tiles* around a previously chosen set of *site faces* and constructing the dual *Delaunay Triangulation* they partition the given triangulation into several simple triangulations and use harmonic maps (see Section 1.2.3) for parameterizing the individual patches.

Lee et al. [90] solve this problem by simultaneously creating a hierarchy $\mathcal{T} = \mathcal{T}^n, \ldots, \mathcal{T}^0$ of triangulations, a process known as *mesh decimation*, and parameterizations $\phi^i : \Omega_{\mathcal{T}^{i-1}} \to \Omega_{\mathcal{T}^i}$. In each decimation step they remove one vertex from $V(\mathcal{T}^i)$ and the triangles adjoint to this vertex and retriangulate the hole. Therefore, $\mathcal{T}^i$ and $\mathcal{T}^{i-1}$ differ only locally and ϕ^i is not hard to find. Composition of the individual ϕ^i finally yields a parameterization $\phi : \Omega_{\mathcal{T}^0} \to \Omega_{\mathcal{T}}$, $\phi = \phi^n \circ \cdots \circ \phi^1$.

1.2 Linear Methods

In this section we concentrate on parameterization methods for simple triangulations $\mathcal{T}$ where the parameter values $\psi(\boldsymbol{v}) \in \mathbb{R}^2$ and thus the inverse parameterization $\psi \in S_2(\mathcal{T})$ are determined by solving a linear system. All these methods have the following outline in common:

1. Specify the parameter points of the boundary vertices $\boldsymbol{v} \in V_B$ by a certain method, e.g. by projection into a least squares plane.
2. Choose for each edge $[\boldsymbol{v}, \boldsymbol{w}] \in E$ a weight $\lambda_{\boldsymbol{vw}}$.
3. Use these weights to set up a linear system of equations and solve it twice in order to obtain the coordinates of the interior parameter values.

In Sections 1.2.1 to 1.2.4 we present different ways of choosing the weights $\lambda_{\boldsymbol{vw}}$. The first approach is motivated by a physical spring model and the univariate analogue (Sections 1.2.1 and 1.2.2). The inverse parameterization ψ obtained in this way is guaranteed to be injective if the parameter points of the boundary vertices form a convex polygon but fail to reproduce planar triangulations. In contrast, the harmonic parameterizations [104, 35] discussed in Section 1.2.3

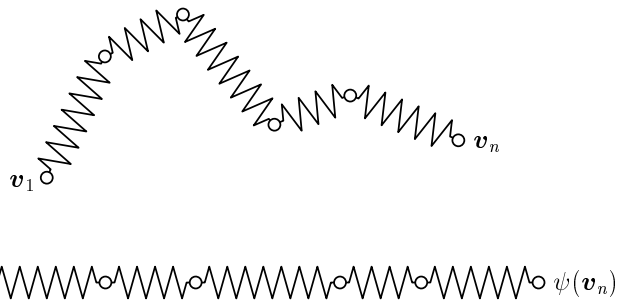


Figure 1.2: Parameterizing a sequence of n points with the spring model.

have this reproduction property but are not necessarily injective, i.e. the triangulation $\psi(\mathcal{T})$ may not be a valid triangulation. In Section 1.2.4 we study a choice of weights $\lambda_{\boldsymbol{vw}}$ that guarantees the resulting mapping ψ to have both properties [42]. Various ways of defining the boundary parameter values are reviewed in Section 1.2.5 and Section 1.2.6 finally discusses how to solve the linear system which will turn out to be regular and thus uniquely solvable for all three methods.

1.2.1 Univariate Spring Model

The first class of linear parameterization methods is motivated by a physical spring model. Let us first consider the case where a sequence of n points $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n \in \mathbb{R}^d$ shall be parameterized over an interval $[a, b] \subset \mathbb{R}$. By connecting each pair $(\boldsymbol{v}_i, \boldsymbol{v}_{i+1})$ of consecutive points with a spring we obtain a chain of $n-1$ springs. If we now take the two endpoints of this chain and pull them apart, the springs will relax to the stable equilibrium of minimal energy and the positions of the joints between the springs can be taken as parameter points (see Figure 1.2).

The potential energy of a spring is $E = \frac{1}{2}Ds^2$, where D is the (strictly positive) spring constant and s the length of the spring. If we let $\psi(\boldsymbol{v}_1)$ and $\psi(\boldsymbol{v}_n)$ be the two endpoints and $\psi(\boldsymbol{v}_2), \ldots, \psi(\boldsymbol{v}_{n-1})$ the interior joints of the chain of springs, the total energy is

$$E_S(\psi) = \sum_{i=1}^{n-1} \tfrac{1}{2} D_i (\psi(\boldsymbol{v}_{i+1}) - \psi(\boldsymbol{v}_i))^2. \tag{1.1}$$

Fixing the endpoints $\psi(\boldsymbol{v}_1) = a$ and $\psi(\boldsymbol{v}_n) = b$ and minimizing (1.1) with respect to the remaining unknowns $\psi(\boldsymbol{v}_2), \ldots, \psi(\boldsymbol{v}_{n-1})$ yields

$$\frac{\psi(\boldsymbol{v}_{i+1}) - \psi(\boldsymbol{v}_i)}{\psi(\boldsymbol{v}_i) - \psi(\boldsymbol{v}_{i-1})} = \frac{D_{i-1}}{D_i}$$

for $i = 2, \ldots, n-1$, resembling the well-known *uniform*, *centripetal*, and *chord-length* parameterization, if the spring constants $D_i = \|\boldsymbol{v}_{i+1} - \boldsymbol{v}_i\|^{-p}$ with $p = 0$,

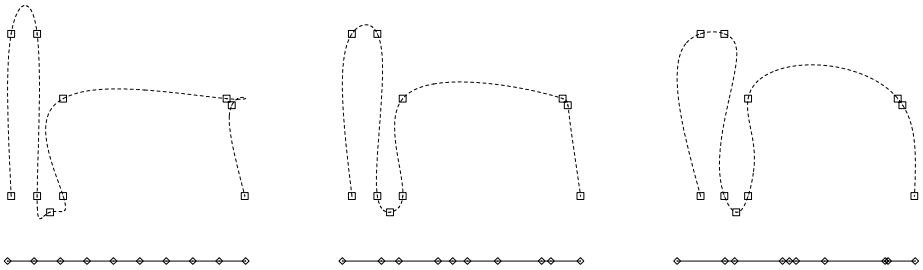


Figure 1.3: Interpolating cubic B-spline curve with uniform (a), centripetal (b), and chord-length (c) parameterization.

$p = 1/2$, and $p = 1$ are chosen. Figure 1.3 shows an example of an interpolating cubic B-spline curve for these three different parameterizations.

Note that the boundary conditions cannot be omitted, because the chain of springs would collapse to a single point $\psi(\boldsymbol{v}_1) = \ldots = \psi(\boldsymbol{v}_n)$ then, a solution with optimally low energy $E_S(\psi) = 0$. The same is true in the bivariate setting and will be explained more detailed in Section 1.2.5.

The following theorem [42, 67] summarizes the four most important characterizations of these *spring model parameterizations*.

Theorem 1.1 *For $t_1, \ldots, t_n \in \mathbb{R}$ and $D_1, \ldots, D_{n-1} \in \mathbb{R}_+$, the following statements are equivalent:*

(i) $\dfrac{t_{i+1} - t_i}{t_i - t_{i-1}} = \dfrac{D_{i-1}}{D_i}$ *for* $i = 2, \ldots, n-1$,

(ii) $t_2, \ldots, t_{n-1}$ *minimize the function* $E(s_2, \ldots, s_{n-1}) = \displaystyle\sum_{i=1}^{n-1} D_i (s_{i+1} - s_i)^2$ *with* $s_1 = t_1$ *and* $s_n = t_n$,

(iii) $t_i = (1 - \lambda_i) t_{i-1} + \lambda_i t_{i+1}$ *with* $\lambda_i = \dfrac{D_i}{D_{i-1} + D_i}$ *for* $i = 2, \ldots, n-1$,

(iv) $t_{i+1} = t_i + \dfrac{\mu}{D_i}$ *for* $i = 1, \ldots, n-1$ *with* $\mu = \dfrac{t_n - t_1}{\sum_{i=1}^{n-1} 1/D_i}$.

Statement (i) characterizes the parameterizations in terms of ratios, (ii) is the spring energy characterization that we are going to extend to triangulations in Sections 1.2.2 and 1.2.3, and (iii) shows that the interior parameter points can be expressed as *convex combinations* of their neighbours, an observation to be generalized in Section 1.2.4. From (iv) we can immediately derive an efficient algorithm for computing spring model parameterizations and an important property of the chord length parameterization.

Corollary 1.1 *If points* $\boldsymbol{v}_1 < \boldsymbol{v}_2 < \cdots < \boldsymbol{v}_n$, $\boldsymbol{v}_i \in \mathbb{R}$ *are given and the boundary conditions* $\psi(\boldsymbol{v}_1) = \boldsymbol{v}_1$ *and* $\psi(\boldsymbol{v}_n) = \boldsymbol{v}_n$ *are chosen, then the chord length parameterization with* $D_i = 1/(\boldsymbol{v}_{i+1} - \boldsymbol{v}_i)$ *for* $i = 1, \ldots, n-1$ *yields* $\psi(\boldsymbol{v}_i) = \boldsymbol{v}_i$ *for* $i = 2, \ldots, n-1$ *and thus* $\psi = \mathrm{Id}$.

Proof. We first observe $\sum_{i=1}^{n-1} 1/D_i = \boldsymbol{v}_n - \boldsymbol{v}_1$ and therefore $\mu = 1$. Assuming $\psi(\boldsymbol{v}_i) = \boldsymbol{v}_i$ we conclude by induction $\psi(\boldsymbol{v}_{i+1}) = \psi(\boldsymbol{v}_i) + (\boldsymbol{v}_{i+1} - \boldsymbol{v}_i) = \boldsymbol{v}_{i+1}$. □

This property is so natural that we would like to have an analogous property for triangulations.

Reproduction property: Whenever a given triangulation is planar, its parameterization should be the identity.

1.2.2 Bivariate Spring Model

Let us now go back to the problem of parameterizing a triangulation. In a first approach we will stick to the physical spring model and generalize the characterization (ii) of Theorem 1.1 to the bivariate setting.

By replacing each edge $[\boldsymbol{v}, \boldsymbol{w}] = [\boldsymbol{w}, \boldsymbol{v}] \in E$ of the triangulation with a spring, we obtain a network of $|E|$ springs with joints $\psi(\boldsymbol{v})$, $\boldsymbol{v} \in V$. The total energy of this system is, in analogy with the univariate case (1.1),

$$E_S(\psi) = \frac{1}{2} \sum_{[\boldsymbol{v},\boldsymbol{w}] \in E} \tfrac{1}{2} D_{\boldsymbol{v}\boldsymbol{w}} \|\psi(\boldsymbol{v}) - \psi(\boldsymbol{w})\|^2 \tag{1.2}$$

with certain (strictly positive) spring constants $D_{\boldsymbol{v}\boldsymbol{w}} = D_{\boldsymbol{w}\boldsymbol{v}}$, the additional factor $1/2$ occurring because each edge is summed up twice. The partial derivative of E_S with respect to $\psi(\boldsymbol{v})$ is

$$\frac{\partial E_S}{\partial \psi(\boldsymbol{v})}(\psi) = \sum_{\boldsymbol{w} \in N_{\boldsymbol{v}}} D_{\boldsymbol{v}\boldsymbol{w}} (\psi(\boldsymbol{v}) - \psi(\boldsymbol{w}))$$

and minimizing (1.2) subject to the boundary conditions of $\psi(\boldsymbol{v})$ being fixed for all $\boldsymbol{v} \in V_B$ is equivalent to solving the linear system of equations

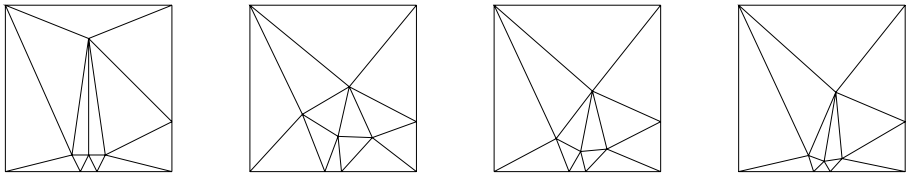
$$\psi(\boldsymbol{v}) \sum_{\boldsymbol{w} \in N_{\boldsymbol{v}}} D_{\boldsymbol{v}\boldsymbol{w}} = \sum_{\boldsymbol{w} \in N_{\boldsymbol{v}}} D_{\boldsymbol{v}\boldsymbol{w}} \psi(\boldsymbol{w}), \qquad \boldsymbol{v} \in V_I. \tag{1.3}$$

If we separate the interior and the boundary vertices in the sum on the right side of (1.3), we can rewrite this linear system as

$$\psi(\boldsymbol{v}) \sum_{\boldsymbol{w} \in N_{\boldsymbol{v}}} D_{\boldsymbol{v}\boldsymbol{w}} - \sum_{\boldsymbol{w} \in N_{\boldsymbol{v}} \cap V_I} D_{\boldsymbol{v}\boldsymbol{w}} \psi(\boldsymbol{w}) = \sum_{\boldsymbol{w} \in N_{\boldsymbol{v}} \cap V_B} D_{\boldsymbol{v}\boldsymbol{w}} \psi(\boldsymbol{w}), \qquad \boldsymbol{v} \in V_I, \tag{1.4}$$

or, more compact, as the matrix equation

$$A\boldsymbol{x} = \boldsymbol{b}, \tag{1.5}$$



(a) (b) (c) (d)

Figure 1.4: A planar triangulation (a) and its uniform (b), centripetal (c), and chord-length (d) parameterization.

where $\boldsymbol{x} = (\psi(\boldsymbol{v}))_{\boldsymbol{v}\in V_I}$ is the column vector of unknowns, $\boldsymbol{b}$ is the column vector whose elements are the right hand sides of (1.4), and the symmetric matrix $A = (a_{\boldsymbol{vw}})_{\boldsymbol{v},\boldsymbol{w}\in V_I}$ has dimension $|V_I| \times |V_I|$ and elements

$$a_{\boldsymbol{vw}} = \begin{cases} \sum_{\boldsymbol{u}\in N_{\boldsymbol{v}}} D_{\boldsymbol{vu}}, & \boldsymbol{w} = \boldsymbol{v}, \\ -D_{\boldsymbol{vw}}, & \boldsymbol{w} \in N_{\boldsymbol{v}}, \\ 0, & \text{otherwise.} \end{cases} \tag{1.6}$$

Theorem 1.2 *The matrix A is symmetric and positive definite.*

Proof. The symmetry of A follows immediately from $D_{\boldsymbol{vw}} = D_{\boldsymbol{wv}}$, and from the positivity of the spring constants we can conclude

$$\boldsymbol{x}^t A\boldsymbol{x} = \sum_{\substack{[\boldsymbol{v},\boldsymbol{w}]\in E \\ \boldsymbol{v},\boldsymbol{w}\in V_I}} \tfrac{1}{2} D_{\boldsymbol{vw}}(\boldsymbol{x}_{\boldsymbol{v}} - \boldsymbol{x}_{\boldsymbol{w}})^2 + \sum_{\substack{\boldsymbol{v}\in V_I \\ \boldsymbol{w}\in N_{\boldsymbol{v}}\cap V_B}} D_{\boldsymbol{vw}}(\boldsymbol{x}_{\boldsymbol{v}})^2 \geq 0.$$

If $\boldsymbol{x}^t A\boldsymbol{x} = 0$ then $\boldsymbol{x}_{\boldsymbol{v}} = 0$ for all $\boldsymbol{v} \in V_I$ with $N_{\boldsymbol{v}} \cap V_B \neq \emptyset$ and $\boldsymbol{x}_{\boldsymbol{v}} = \boldsymbol{x}_{\boldsymbol{w}}$ for all interior edges $[\boldsymbol{v}, \boldsymbol{w}]$. Since the interior edges of a triangulation form a simply-connected graph with all interior vertices as nodes we can conclude $\boldsymbol{x}_{\boldsymbol{v}} = 0$ for all $\boldsymbol{v} \in V_I$ and thus $\boldsymbol{x} = \boldsymbol{0}$. □

This does not only prove the existence and uniqueness of a solution to (1.3) but also that this solution is a minimum of the spring energy $E_S(\psi)$ because A is the *Hessian matrix* of E_S. Furthermore, Theorem 1.6 on page 21 states that the solution ψ is a bijection and thus $\phi = \psi^{-1}$ a valid parameterization, if the previously fixed parameter points $\psi(\boldsymbol{v})$ of the boundary vertices $\boldsymbol{v} \in V_B$ form a convex polygon.

In analogy with the univariate model we can choose the *distance weights*

$$D_{\boldsymbol{vw}} = \|\boldsymbol{v} - \boldsymbol{w}\|^{-p} \tag{1.7}$$

as spring constants with $p = 0$, $p = 1/2$, or $p = 1$ to obtain a uniform, centripetal, or chord length parameterization of the given triangulation.

The simple example in Figure 1.4 shows that neither choice gives the postulated reproduction property but in the next section we will discuss a different choice of spring constants $D_{\boldsymbol{vw}}$ that overcomes this drawback.

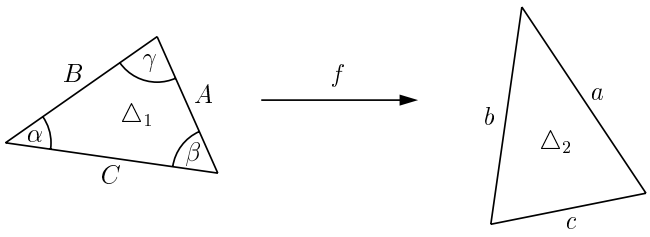


Figure 1.5: Linear map between two triangles.

1.2.3 Harmonic Maps

Another parameterization method that fits into the spring model framework was first proposed by Pinkall and Polthier as part of an algorithm for computing discrete minimal surfaces [104] and later by Eck et al. in [35]. It is based on

Dirichlet's boundary value problem: Given a two-manifold with boundary M, a simply-connected region $R \subset \mathbb{R}^2$, and a homeomorphism $g : \partial M \to \partial R$ between the boundaries of M and R, find a function $f : M \to R$ that agrees with g on ∂M and is harmonic, $\Delta f = 0$.

Variational calculus states that the solution to this problem minimizes the *Dirichlet energy* of f,

$$E_D(f) = \frac{1}{2} \int_M \|\nabla f\|^2, \tag{1.8}$$

subject to the same boundary condition. In our special setting, where the manifold M is a triangulation $\mathcal{T}$ and the boundary condition is given by the previously fixed parameter points of the boundary vertices, the Dirichlet energy of the piecewise linear mappings $\psi \in S_2(\mathcal{T})$ that we are interested in can be represented in the form (1.2). This is an immediate consequence of the following lemma [104].

Lemma 1.1 *Let f be the linear map between two triangles $\triangle_1$ and $\triangle_2$. Then the Dirichlet energy of f is*

$$E_D(f) = \frac{1}{4}(\cot\alpha\, a^2 + \cot\beta\, b^2 + \cot\gamma\, c^2),$$

where α, β, γ are the angles in $\triangle_1$ and a, b, c are the corresponding side lengths in $\triangle_2$ (see Figure 1.5).

Corollary 1.2 *The Dirichlet energy of a piecewise linear mapping $\psi \in S_2(\mathcal{T})$ is*

$$E_D(\psi) = \frac{1}{2} \sum_{[\boldsymbol{v},\boldsymbol{w}]\in E} \tfrac{1}{2} D_{\boldsymbol{vw}} \|\psi(\boldsymbol{v}) - \psi(\boldsymbol{w})\|^2 \tag{1.9}$$

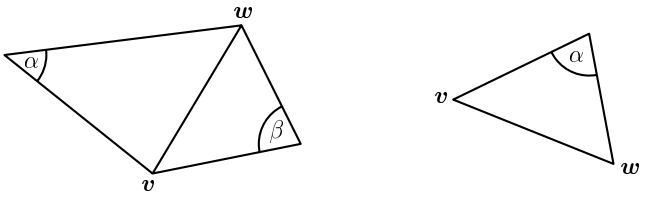


Figure 1.6: Interior edge with two and boundary edge with one adjacent triangle.

with the harmonic weights

$$D_{\boldsymbol{vw}} = \frac{1}{2}(\cot\alpha + \cot\beta) \qquad \text{and} \qquad D_{\boldsymbol{vw}} = \frac{1}{2}\cot\alpha \tag{1.10}$$

for interior and boundary edges respectively, where α and β are the angles opposite to $[\boldsymbol{v}, \boldsymbol{w}]$ in the adjacent triangles (see Figure 1.6).

Proof. By decomposing ψ into the linear maps $\psi_{|T}$, $T \in \mathcal{T}$ and considering

$$E_D(\psi) = \sum_{T \in \mathcal{T}} E_D(\psi_{|T}),$$

Equation 1.9 follows from Lemma 1.1 by rearranging the sum over the triangles $T \in \mathcal{T}$ to a sum over the edges $[\boldsymbol{v}, \boldsymbol{w}] \in E$ of the triangulation $\mathcal{T}$. □

If we now fix the parameter points $\psi(\boldsymbol{v})$ for the boundary vertices $\boldsymbol{v} \in V_B$ as we did for the spring model, minimizing the Dirichlet energy $E_D(\psi)$ with respect to the interior parameter points $\psi(\boldsymbol{v})$, $\boldsymbol{v} \in V_I$ gives a *harmonic map* ψ and the corresponding *harmonic parameterization* $\phi = \psi^{-1}$.

According to [37, 35], these maps also minimize *metric dispersion*, a measure of the extent to which a map stretches regions of small diameter in $\mathcal{T}$. Another approach that minimizes a different kind of metric distortion will be discussed in Section 1.3.

Like in Section 1.2.2, the problem of minimizing (1.9) leads to a linear system $A\boldsymbol{x} = \boldsymbol{b}$. Unfortunately, we can no longer use the proof of Theorem 1.2 to show the positive definiteness of A since the harmonic weights $D_{\boldsymbol{vw}}$ can be negative.

Proposition 1.1 *Let $[\boldsymbol{v}, \boldsymbol{w}]$ be an interior edge of a triangulation and $\triangle_1$, $\triangle_2$ the adjacent triangles. Rotating $\triangle_2$ around $[\boldsymbol{v}, \boldsymbol{w}]$ into the plane P which is defined by $\triangle_1$ gives a congruent triangle $\tilde{\triangle}_2 = [\boldsymbol{v}, \boldsymbol{w}, \boldsymbol{p}] \subset P$. The harmonic weight $D_{\boldsymbol{vw}}$ is positive, if and only if $\triangle_1$ and $\tilde{\triangle}_2$ have the* local Delaunay property*, i.e. $\boldsymbol{p}$ does not lie in the circumcircle of $\triangle_1$.*

Proof. In the notation of Figure 1.7 there exist for any points $\boldsymbol{p}_1$ outside and $\boldsymbol{p}_2$ inside the circumcircle C of $\triangle_1$ points $\boldsymbol{p}_1'$ and $\boldsymbol{p}_2'$ on C such that $\beta_1 < \beta_1'$ and $\beta_2 > \beta_2'$. From the theorem of quadrilaterals inscribed in a circle we know that $\alpha + \beta_1' = \alpha + \beta_2' = \pi$ so that we can conclude $\alpha + \beta_1 < \pi < \alpha + \beta_2$ and further

$$\cot\alpha + \cot\beta_1 = \frac{\sin(\alpha + \beta_1)}{\sin\alpha \sin\beta_1} > 0 > \frac{\sin(\alpha + \beta_2)}{\sin\alpha \sin\beta_2} = \cot\alpha + \cot\beta_2.$$

□

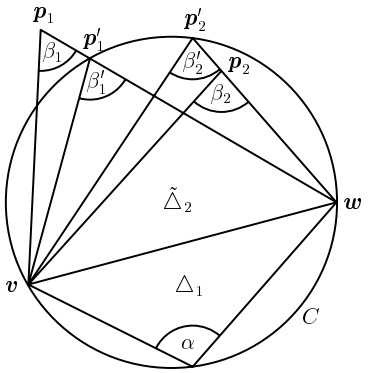


Figure 1.7: Notation for the proof of Proposition 1.1.

It is remarkable that the sum of the harmonic weights is always positive.

Proposition 1.2 *For any interior vertex $\boldsymbol{v} \in V_I$ of a triangulation, the sum of harmonic weights (1.10) corresponding to its neighbours $\boldsymbol{w} \in N_{\boldsymbol{v}}$ is positive.*

Proof. Let α and β be two angles in a non-degenerate triangle, then we have $0 < \alpha, \beta, \alpha + \beta < \pi$ and therefore

$$\cot\alpha + \cot\beta = \frac{\sin(\alpha+\beta)}{\sin\alpha \sin\beta} > 0. \tag{1.11}$$

If we now label the neighbours $\boldsymbol{w} \in N_{\boldsymbol{v}}$ of a vertex $\boldsymbol{v} \in V_I$ as in Figure 1.8, we can conclude

$$\sum_{i=1}^{n} D_{\boldsymbol{v}\boldsymbol{w}_i} = \sum_{i=1}^{n} \frac{1}{2}(\cot\alpha_i + \cot\beta_i) = \frac{1}{2}\sum_{i=1}^{n} \underbrace{\cot\alpha_{i+1} + \cot\beta_i}_{\overset{(1.11)}{>} 0} > 0.$$

□

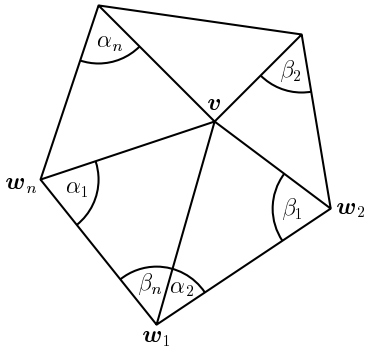


Figure 1.8: Notation for the neighbourhood of an interior vertex $\boldsymbol{v}$ with n neighbours $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_n$.

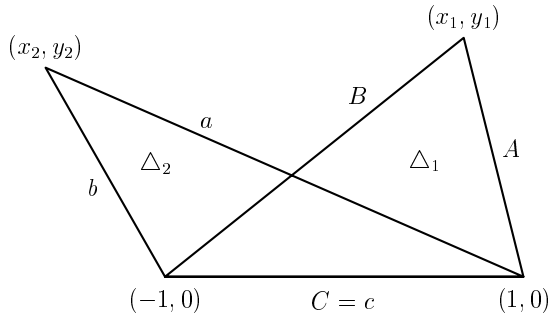


Figure 1.9: Notation for the proof of Lemma 1.2.

However, we can still show the positive definiteness of A and thus the existence and uniqueness of the harmonic map ψ.

Lemma 1.2 *For the Dirichlet energy of the linear map f between the two triangles $\triangle_1$ and $\triangle_2$,*

$$E_D(f) \geq \operatorname{area}(\triangle_2)$$

holds with equality if and only if $\triangle_1$ and $\triangle_2$ are similar or $\triangle_2$ is degenerated to a point.

Proof. The triangle $\triangle_2$ is degenerated to a point if and only if $a = b = c = 0$ and in this case $E_D(f) = 0 = \operatorname{area}(\triangle_2)$. Let us now assume $c > 0$. Since the angles α, β, γ of $\triangle_1$ do not depend on scalings and rotations we can assume without loss of generality $\triangle_1 = \left[\binom{-1}{0}, \binom{1}{0}, \binom{x_1}{y_1}\right]$ with $y_1 > 0$ and therefore $A^2 = (x_1 - 1)^2 + y_1^2$, $B^2 = (x_1 + 1)^2 + y_1^2$, $C^2 = 4$, and $\operatorname{area}(\triangle_1) = y_1$ (see Figure 1.9). Using the cosine rule we get

$$\cot\alpha = \frac{\cos\alpha}{\sin\alpha} = \frac{B^2 + C^2 - A^2}{2BC\sin\alpha} = \frac{B^2 + C^2 - A^2}{4\operatorname{area}(\triangle_1)} = \frac{1 + x_1}{y_1}$$

and similarly $\cot\beta = \frac{1-x_1}{y_1}$, $\cot\gamma = \frac{x_1^2+y_1^2-1}{2y_1}$. Moreover, $E_D(f)$ and $\operatorname{area}(\triangle_2)$ both depend quadratically on scalings of $\triangle_2$ and are invariant under rotations of $\triangle_2$, so that we can further assume $\triangle_2 = \left[\binom{-1}{0}, \binom{1}{0}, \binom{x_2}{y_2}\right]$. Therefore,

$$\begin{aligned}
4(E_D(f) - \operatorname{area}(\triangle_2))y_1 &= y_1(\cot\alpha\, a^2 + \cot\beta\, b^2 + \cot\gamma\, c^2 - 4\operatorname{area}(\triangle_2)) \\
&= (1 + x_1)((x_2 - 1)^2 + y_2^2) + \\
&\quad (1 - x_1)((x_2 + 1)^2 + y_2^2) + 2(x_1^2 + y_1^2 - 1) - 4y_1y_2 \\
&= 2x_2^2 + 2 - 4x_1x_2 + 2y_2^2 + 2x_1^2 + 2y_1^2 - 2 - 4y_1y_2 \\
&= 2(x_2 - x_1)^2 + 2(y_2 - y_1)^2 \\
&\geq 0
\end{aligned}$$

with equality if and only if $\binom{x_1}{y_1} = \binom{x_2}{y_2}$. The statement follows since $y_1 > 0$. □

Lemma 1.3 *For the Dirichlet energy of a piecewise linear mapping $\psi \in S_2(\mathcal{T})$*

$$E_D(\psi) \geq 0$$

holds with equality if and only if all $\psi(\boldsymbol{v})$, $\boldsymbol{v} \in V$ are the same, i.e. if ψ is a constant function.

Proof. As in the proof of Corollary 1.2 we decompose ψ into the linear maps $\psi_{|T}$, $T \in \mathcal{T}$ and conclude with Lemma 1.2

$$E_D(\psi) = \sum_{T \in \mathcal{T}} E_D(\psi_{|T}) \geq \sum_{T \in \mathcal{T}} \operatorname{area}(\psi(T)) \geq 0$$

with equality if and only if the areas of all triangles $\psi(T)$ vanish. □

Theorem 1.3 *The matrix A defined by the harmonic weights is symmetric and positive definite.*

Proof. The symmetry of A follows immediately from the symmetric definition of the harmonic weights, $D_{\boldsymbol{vw}} = D_{\boldsymbol{wv}}$. In order to show the positive definiteness we consider the symmetric $|V| \times |V|$ matrix $\bar{A} = (\bar{a}_{\boldsymbol{vw}})_{\boldsymbol{v},\boldsymbol{w} \in V}$ which is defined similarly to (1.6) by

$$\bar{a}_{\boldsymbol{vw}} = \begin{cases} \sum_{\boldsymbol{u} \in N_{\boldsymbol{v}}} D_{\boldsymbol{vu}}, & \boldsymbol{w} = \boldsymbol{v}, \\ -D_{\boldsymbol{vw}}, & \boldsymbol{w} \in N_{\boldsymbol{v}}, \\ 0, & \text{otherwise.} \end{cases}$$

with the rows and columns arranged such that A is the $|V_I| \times |V_I|$ upper left submatrix,

$$\bar{A} = \begin{pmatrix} A & * \\ * & * \end{pmatrix}.$$

We can now write the Dirichlet energy as $E_D(\psi) = \frac{1}{2} \bar{\boldsymbol{x}}^t \bar{A} \bar{\boldsymbol{x}}$ with $\bar{\boldsymbol{x}} = (\psi(\boldsymbol{v}))_{\boldsymbol{v} \in V}$ and from Lemma 1.3 we know

$$\bar{\boldsymbol{x}}^t \bar{A} \bar{\boldsymbol{x}} = 2E_D(\psi) \geq 0$$

with equality if and only if all triangles $\psi(T)$ are degenerated to the same point. Therefore, $\bar{A}$ is positive definite as long as the boundary of the parameterization is non-degenerate and so is A as an upper left submatrix of $\bar{A}$. □

Furthermore, harmonic maps have the postulated reproduction property.

Theorem 1.4 *For any planar triangulation $\mathcal{T}$, the harmonic map ψ, defined as the minimum of (1.9) subject to the boundary conditions $\psi(\boldsymbol{v}) = \boldsymbol{v}$ for $\boldsymbol{v} \in V_B$, is the identity, $\psi = \mathrm{Id}$.*

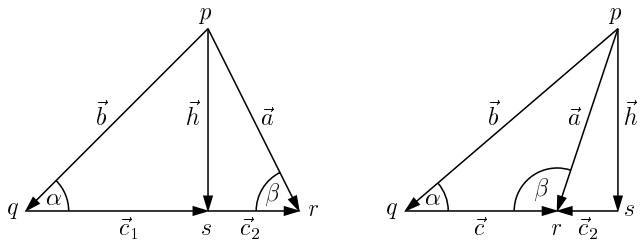


Figure 1.10: Notation for planar triangles. Let s be the orthogonal projection of p on the opposite side, $\vec{c}_1 = s - q$, $\vec{c}_2 = r - s$, and $\vec{c} = \vec{c}_1 + \vec{c}_2 = r - q$.

Proof. We know from the previous section that the minimum of (1.9) solves the equations (1.3). Since the uniqueness of this solution follows from Theorem 1.3, it is sufficient to show that $\psi(\boldsymbol{v}) = \boldsymbol{v}$ for $\boldsymbol{v} \in V_I$ satisfies (1.3), i.e.

$$\boldsymbol{v} \sum_{\boldsymbol{w} \in N_{\boldsymbol{v}}} D_{\boldsymbol{v}\boldsymbol{w}} = \sum_{\boldsymbol{w} \in N_{\boldsymbol{v}}} D_{\boldsymbol{v}\boldsymbol{w}} \boldsymbol{w}, \qquad \boldsymbol{v} \in V_I,$$

with the special choice of $D_{\boldsymbol{v}\boldsymbol{w}}$ in (1.10). For any planar triangle we have, using the notation in Figure 1.10, $\cot\alpha\,\vec{h} = R\,\vec{c}_1$ and $\cot\beta\,\vec{h} = R\,\vec{c}_2$, where $R = \left(\begin{smallmatrix} 0 & 1 \\ -1 & 0 \end{smallmatrix}\right)$ is the 90° clockwise rotation. From $\vec{c}_1 = \cot\alpha\, R^{-1}\vec{h}$ and $\vec{c}_2 = \cot\beta\, R^{-1}\vec{h}$ it follows $\cot\alpha\,\vec{c}_2 = \cot\beta\,\vec{c}_1$ and with $\vec{a} = \vec{h} + \vec{c}_2$, $\vec{b} = \vec{h} - \vec{c}_1$,

$$\cot\alpha\,\vec{a} + \cot\beta\,\vec{b} = R\,\vec{c}_1 + \cot\alpha\,\vec{c}_2 + R\,\vec{c}_2 - \cot\beta\,\vec{c}_1 = R\,\vec{c}. \tag{1.12}$$

In the notation of Figure 1.8 we have

$$\begin{aligned}
\sum_{i=1}^{n} D_{\boldsymbol{v}\boldsymbol{w}_i}(\boldsymbol{w}_i - \boldsymbol{v}) &= \sum_{i=1}^{n} \frac{1}{2}(\cot\alpha_i + \cot\beta_i)(\boldsymbol{w}_i - \boldsymbol{v}) \\
&= \frac{1}{2}\sum_{i=1}^{n} \cot\alpha_{i+1}(\boldsymbol{w}_{i+1} - \boldsymbol{v}) + \cot\beta_i(\boldsymbol{w}_i - \boldsymbol{v}) \\
&\overset{(1.12)}{=} \frac{1}{2}\sum_{i=1}^{n} R(\boldsymbol{w}_{i+1} - \boldsymbol{w}_i) = \frac{1}{2}R\underbrace{\sum_{i=1}^{n}(\boldsymbol{w}_{i+1} - \boldsymbol{w}_i)}_{=\mathbf{0}} \\
&= \mathbf{0}.
\end{aligned}$$

□

Despite this important property, harmonic maps have the serious disadvantage that ψ is not necessarily a bijection and thus $\psi(\mathcal{T})$ may not be a valid triangulation (see Figure 1.11). As we will later see in Section 1.2.4, the reason for this problem is that the harmonic weights can be negative. In fact, in the example in Figure 1.11 we have $D_{\boldsymbol{v}\boldsymbol{w}_1} = D_{\boldsymbol{v}\boldsymbol{w}_4} = \frac{1-\sqrt{5}}{2\sqrt{10}} < 0$.

We have now discussed two methods for choosing the spring constants of the spring model approach. The distance weights (1.7) were motivated by the

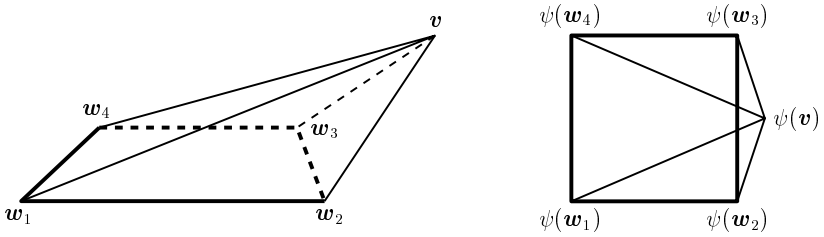


Figure 1.11: A triangulation with five vertices $\boldsymbol{v} = (2, 0, 1)$, $\boldsymbol{w}_1 = (-1, -1, 0)$, $\boldsymbol{w}_2 = (1, -1, 0)$, $\boldsymbol{w}_3 = (1, 1, 0)$, $\boldsymbol{w}_4 = (-1, 1, 0)$ and four triangles $[\boldsymbol{v}, \boldsymbol{w}_i, \boldsymbol{w}_{i+1}]$, $i = 1, 2, 3, 4$, and its harmonic parameterization, $\psi(\boldsymbol{w}_1) = (-1, -1)$, $\psi(\boldsymbol{w}_2) = (1, -1)$, $\psi(\boldsymbol{w}_3) = (1, 1)$, $\psi(\boldsymbol{w}_4) = (-1, 1)$, $\psi(\boldsymbol{v}) = \left(\frac{5\sqrt{5}-1}{3\sqrt{5}+1}, 0\right) = (1.320715, 0)$.

univariate analogue and yield valid parameterizations but fail to reproduce planar triangulations. On the other hand, the harmonic weights (1.10) have the reproduction property, but do not necessarily result in valid parameterizations. Whether there exists yet another choice of spring constants which gives both properties at the same time is currently unknown, but in the next section we will present a slightly different approach that gives an acceptable answer.

Let us wind up this section by giving an example that illustrates the behaviour of the different spring constants we have discussed. For the simple configuration in Figure 1.12 we have for the interior edge

$$D_{\text{distance}}(x) = \left(\frac{1}{2x}\right)^p \qquad \text{and} \qquad D_{\text{harmonic}}(x) = \frac{1}{x} - \frac{x}{4}.$$

All weights, except for the uniform ($p = 0$), are reciprocal to the edge length, i.e. the closer the end points are in the triangulation, the more they will attract each other in the spring model. Note that this effect is the stronger the larger

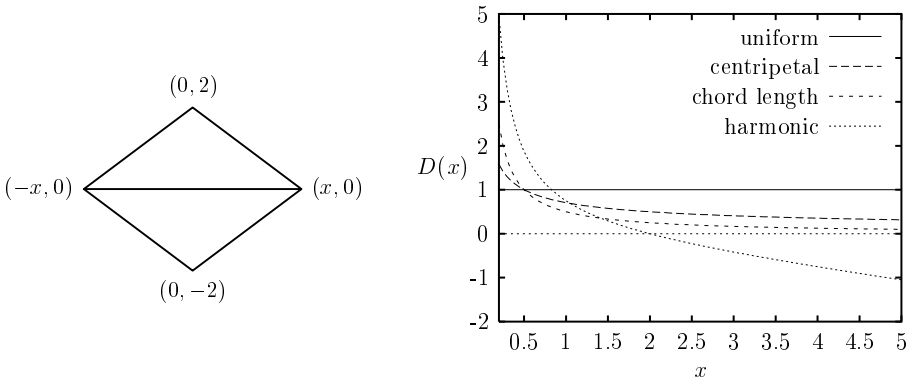


Figure 1.12: Behaviour of different spring constants for a simple configuration.

p is and strongest for the harmonic weight. In fact, if the edge length exceeds the distance between the two points opposite the edge in the adjacent triangles, then the harmonic weight becomes negative and acts as a repelling force in the spring model.

1.2.4 Convex Combination Maps

Another linear parameterization method was introduced by Floater in [42] and generalizes the third characterization of Theorem 1.1. As in the previous sections, the first step of that method is to specify the parameter points $\psi(\boldsymbol{v})$ of the boundary vertices $\boldsymbol{v} \in V_B$. Then, for each interior vertex $\boldsymbol{v} \in V_I$ a set of strictly positive *convex weights* $\lambda_{\boldsymbol{vw}}$, $\boldsymbol{w} \in N_{\boldsymbol{v}}$ with

$$\sum_{\boldsymbol{w} \in N_{\boldsymbol{v}}} \lambda_{\boldsymbol{vw}} = 1$$

is chosen and the remaining $\psi(\boldsymbol{v})$, $\boldsymbol{v} \in V_I$ are determined by solving the linear system of equations

$$\psi(\boldsymbol{v}) = \sum_{\boldsymbol{w} \in N_{\boldsymbol{v}}} \lambda_{\boldsymbol{vw}} \psi(\boldsymbol{w}), \qquad \boldsymbol{v} \in V_I. \tag{1.13}$$

Therefore, every interior parameter point is a *convex combination* of its neighbours. Like in Section 1.2.2, by separating interior and boundary vertices, (1.13) can be rewritten as

$$\psi(\boldsymbol{v}) - \sum_{\boldsymbol{w} \in N_{\boldsymbol{v}} \cap V_I} \lambda_{\boldsymbol{vw}} \psi(\boldsymbol{w}) = \sum_{\boldsymbol{w} \in N_{\boldsymbol{v}} \cap V_B} \lambda_{\boldsymbol{vw}} \psi(\boldsymbol{w}), \qquad \boldsymbol{v} \in V_I, \tag{1.14}$$

and further as

$$B\boldsymbol{x} = \boldsymbol{c}, \tag{1.15}$$

where $\boldsymbol{x} = (\psi(\boldsymbol{v}))_{\boldsymbol{v} \in V_I}$ is again the column vector of unknowns, $\boldsymbol{c}$ is the column vector whose elements are the right hand sides of (1.14), and the $|V_I| \times |V_I|$ matrix $B = (b_{\boldsymbol{vw}})_{\boldsymbol{v},\boldsymbol{w} \in V_I}$ has elements

$$b_{\boldsymbol{vw}} = \begin{cases} 1, & \boldsymbol{w} = \boldsymbol{v}, \\ -\lambda_{\boldsymbol{vw}}, & \boldsymbol{w} \in N_{\boldsymbol{v}}, \\ 0, & \text{otherwise.} \end{cases}$$

Note that in general $\lambda_{\boldsymbol{vw}} \neq \lambda_{\boldsymbol{wv}}$ and therefore B is usually not symmetric in contrast to A in (1.5). The following theorem guarantees the existence and uniqueness of a solution to (1.13).

Theorem 1.5 *The matrix B is regular.*

Proof. Since every internal node is the convex combination of its neighbours it is also a convex combination of the boundary vertices, i.e. it lies within their convex hull. Now, if $B\boldsymbol{x} = \boldsymbol{0}$, then all boundary vertices are $\boldsymbol{0}$ and their convex hull is $\{\boldsymbol{0}\}$. Therefore, $\boldsymbol{x} = \boldsymbol{0}$. □

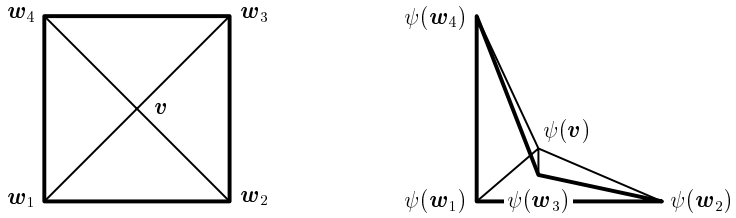


Figure 1.13: A planar triangulation with five vertices $\boldsymbol{v} = (\frac{1}{2}, \frac{1}{2})$, $\boldsymbol{w}_1 = (0,0)$, $\boldsymbol{w}_2 = (1,0)$, $\boldsymbol{w}_3 = (1,1)$, $\boldsymbol{w}_4 = (0,1)$, four triangles $[\boldsymbol{v}, \boldsymbol{w}_i, \boldsymbol{w}_{i+1}]$ and convex weights $\lambda_{\boldsymbol{v}\boldsymbol{w}_i} = \frac{1}{4}$, and its parameterization with a non-convex boundary, $\psi(\boldsymbol{w}_1) = (0,0)$, $\psi(\boldsymbol{w}_2) = (1,0)$, $\psi(\boldsymbol{w}_3) = (\frac{1}{3}, \frac{1}{7})$, $\psi(\boldsymbol{w}_4) = (0,1)$, $\psi(\boldsymbol{v}) = (\frac{1}{3}, \frac{2}{7})$.

The piecewise linear functions ψ that are defined by the parameter points $\psi(\boldsymbol{v})$ which solve (1.13) are called *convex combinations maps* and the following theorem [130, 42, 44] guarantees them to lead to proper parameterizations under certain conditions.

Theorem 1.6 *If the boundary polygon that is formed by the fixed parameter points $\psi(\boldsymbol{v})$, $\boldsymbol{v} \in V_B$ is strictly convex, then the convex combination map ψ is a bijection, i.e. the planar triangulation $\psi(\mathcal{T})$ is without self-intersections.*

This also proves the spring model parameterizations from Section 1.2.2 to be bijective because they are convex combination maps for the special choice of

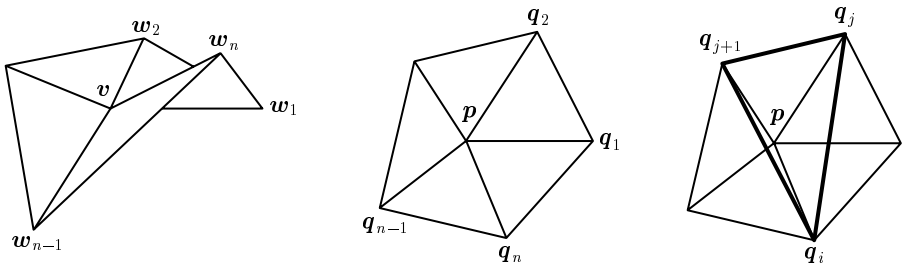
$$\lambda_{\boldsymbol{v}\boldsymbol{w}} = \frac{D_{\boldsymbol{v}\boldsymbol{w}}}{\sum_{\boldsymbol{u} \in N_{\boldsymbol{v}}} D_{\boldsymbol{v}\boldsymbol{u}}}, \qquad \boldsymbol{v} \in V_I, \quad \boldsymbol{w} \in N_{\boldsymbol{v}}. \tag{1.16}$$

Note that due to Proposition 1.2 these $\lambda_{\boldsymbol{v}\boldsymbol{w}}$'s are well-defined for the harmonic weights, too. But they are negative if and only if the corresponding harmonic weight $D_{\boldsymbol{v}\boldsymbol{w}}$ is negative and therefore harmonic maps are not *convex* but only *affine combination maps* and do not fulfill the requirements of Theorem 1.6.

Furthermore, the example in Figure 1.11 shows that the positivity of the convex weights is a necessary condition of Theorem 1.6 and the example in Figure 1.13 shows that we cannot do without the convexity condition of the boundary polygon either. However, here is a slightly different version of Theorem 1.6 with weaker assumptions [44].

Corollary 1.3 *If the boundary polygon is weakly convex and there is no triangle $[\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}] \in \mathcal{T}$ with $\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w} \in V_B$ such that $\psi(\boldsymbol{u}), \psi(\boldsymbol{v}), \psi(\boldsymbol{w})$ are collinear, then ψ is a bijection.*

Now that we know the convex combination maps to be unique and bijective the question is whether the weights $\lambda_{\boldsymbol{v}\boldsymbol{w}}$ can be chosen such that the reproduction property holds in addition. The positive answer was given by Floater in [42] and we will now briefly review his method of specifying the convex weights.



(a) (b) (c)

Figure 1.14: A local configuration (a) is flattened out into the plane by an exponential mapping (b) and the local weights are determined (c).

Let $\boldsymbol{v} \in V_I$ be an interior vertex with n neighbours $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_n$ as in Figure 1.14.a. In a first step this local configuration is "flattened out" into the plane by an exponential mapping, yielding temporary parameter points $\boldsymbol{p}$ and $\boldsymbol{q}_1, \ldots, \boldsymbol{q}_n$ (see Figure 1.14.b). The exponential mapping preserves the length of the edges $[\boldsymbol{v}, \boldsymbol{w}_i]$ and uniformly scales the angles $\gamma_i = \sphericalangle(\boldsymbol{w}_i, \boldsymbol{v}, \boldsymbol{w}_{i+1})$ of the triangles $[\boldsymbol{v}, \boldsymbol{w}_i, \boldsymbol{w}_{i+1}]$ at $\boldsymbol{v}$. One way to realize it is to set

$$\boldsymbol{p} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \qquad \text{and} \qquad \boldsymbol{q}_i = \|\boldsymbol{w}_i - \boldsymbol{v}\| \begin{pmatrix} \cos\delta_i \\ \sin\delta_i \end{pmatrix}$$

with

$$\delta_i = \rho \sum_{j=1}^{i-1} \gamma_j \qquad \text{and} \qquad \rho = \frac{2\pi}{\sum_{i=1}^{n} \gamma_i}.$$

In a second step we locate for each $\boldsymbol{q}_i$ the index j for which $\boldsymbol{p} \in [\boldsymbol{q}_i, \boldsymbol{q}_j, \boldsymbol{q}_{j+1}]$ (see Figure 1.14.c) and determine the barycentric coordinates of $\boldsymbol{p}$ with respect to that triangle,

$$\boldsymbol{p} = \tau_i^i \boldsymbol{q}_i + \tau_j^i \boldsymbol{q}_j + \tau_{j+1}^i \boldsymbol{q}_{j+1},$$

with $\tau_i^i + \tau_j^i + \tau_{j+1}^i = 1$. Letting all other $\tau_k^i = 0$, $k \neq i, j, j+1$, we finally take the *shape preserving weights* to be the averages of the local weights τ_i^j,

$$\lambda_{\boldsymbol{v}\boldsymbol{w}_i} = \frac{1}{n} \sum_{j=1}^{n} \tau_i^j. \tag{1.17}$$

These weights yield the desired reproduction property.

Theorem 1.7 *For any planar triangulation $\mathcal{T}$, the convex combination map ψ, defined by the solution to (1.13) with the shape preserving weights from (1.17) and $\psi(\boldsymbol{v}) = \boldsymbol{v}$, $\boldsymbol{v} \in V_B$, is the identity, $\psi = \mathrm{Id}$.*

Proof. We observe $\lambda_{\boldsymbol{v}\boldsymbol{w}_i} > 0$ because of the positivity of the barycentric coordinates and $\tau_i^i > 0$ and further

$$\sum_{i=1}^{n} \lambda_{\boldsymbol{v}\boldsymbol{w}_i} = \frac{1}{n} \sum_{j=1}^{n} \underbrace{\sum_{i=1}^{n} \tau_i^j}_{=1} = 1.$$

Therefore, the shape preserving weights are proper convex weights. In analogy to the proof of Theorem 1.4 it remains to show that $\psi(\boldsymbol{v}) = \boldsymbol{v}$ for $\boldsymbol{v} \in V_I$ solves (1.13). We know that

$$\sum_{i=1}^{n} \lambda_{\boldsymbol{v}\boldsymbol{w}_i} \boldsymbol{q}_i = \frac{1}{n} \sum_{j=1}^{n} \underbrace{\sum_{i=1}^{n} \tau_i^j \boldsymbol{q}_i}_{=\boldsymbol{p}} = \boldsymbol{p} \tag{1.18}$$

and since the described exponential mapping is a solid body transformation for planar triangulations and thus preserves convex combinations we can replace $\boldsymbol{p}$ and $\boldsymbol{q}_i$ in (1.18) with $\boldsymbol{v}$ and $\boldsymbol{w}_i$ respectively. □

The convex combination map ψ that corresponds to the shape preserving weights possesses all properties we were looking for. It is the unique solution to a linear system, has the reproduction property and is guaranteed to be a bijection if the parameter points of the boundary vertices were properly chosen. The inverse $\phi = \psi^{-1}$ of ψ is called a *shape preserving parameterization* [42].

1.2.5 Parameterizing the Boundary

Any of the previously discussed linear parameterization methods demands the boundary $\partial\mathcal{T}$ of the given triangulation $\mathcal{T}$ to be parameterized in advance, i.e. we need to specify a piecewise linear mapping $\psi : \partial\mathcal{T} \to \mathbb{R}^2$ which is uniquely determined by the values $\psi(\boldsymbol{v})$, $\boldsymbol{v} \in V_B$. We will now discuss various ways of carrying out this task. For this purpose we let $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ be the ordered boundary vertices $\boldsymbol{v}_i \in V_B$ and identify $\boldsymbol{v}_{n+1} = \boldsymbol{v}_1$ for the sake of convenience, so that $\bigcup_{i=1}^{n} [\boldsymbol{v}_i, \boldsymbol{v}_{i+1}] = \partial\mathcal{T}$.

The first method is motivated by Theorem 1.6 and the univariate parameterization methods discussed in Section 1.2.1. The bijectivity of the parameterization is assured if the parameter points of the boundary vertices form a convex polygon, which can be achieved, for example, by placing them on the unit circle and their distance should be chosen as in the spring model approach (see Figure 1.15.a). If we write the parameter points as

$$\psi(\boldsymbol{v}_i) = \begin{pmatrix} \cos\alpha_i \\ \sin\alpha_i \end{pmatrix}, \quad i = 1, \ldots, n+1,$$

a reasonable measure of the distance between $\psi(\boldsymbol{v}_i)$ and $\psi(\boldsymbol{v}_{i+1})$ is the difference $\alpha_{i+1} - \alpha_i$ as the length of the arc between those two points. The parameterization of the boundary can then be regarded as a univariate parameterization

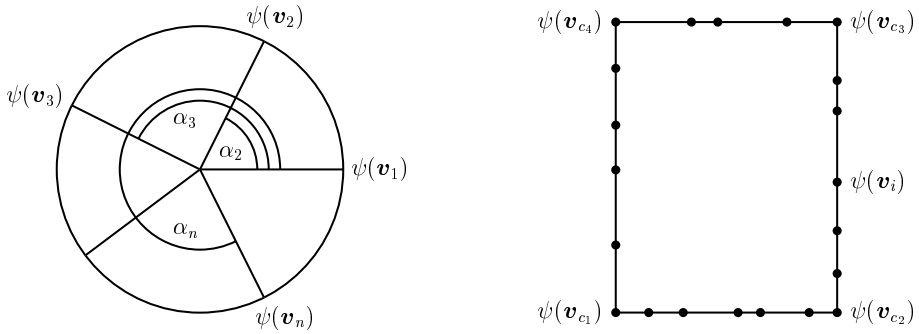


Figure 1.15: Parameterizing the boundary over a circle (a) or a rectangle (b).

problem with parameter points α_i and fixed endpoints $\alpha_0 = 0$, $\alpha_{n+1} = 2\pi$. From statement (iv) of Theorem 1.1 we know the solution to this problem to be

$$\alpha_{i+1} = \alpha_i + \frac{\mu}{D_i}, \quad i = 1, \ldots, n+1, \qquad \text{with} \qquad \mu = \frac{2\pi}{\sum_{i=1}^{n} 1/D_i},$$

using, for instance, the chord length weights $D_i = 1/\|\boldsymbol{v}_{i+1} - \boldsymbol{v}_i\|$.

Considering Corollary 1.3 we may also map the boundary of $\mathcal{T}$ to a weakly convex polygon. Especially in the context of texture mapping it is often desirable to have a square or rectangular parameter domain as many textures come in such a shape. This necessitates to identify four prominent *corner vertices* among the boundary vertices V_B which will be mapped to the corners of a rectangle (see Figure 1.15.b). If we let $c_1 < c_2 < c_3 < c_4$ be the indices of these corner vertices, we set

$$\psi(\boldsymbol{v}_{c_1}) = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad \psi(\boldsymbol{v}_{c_2}) = \begin{pmatrix} a \\ 0 \end{pmatrix}, \quad \psi(\boldsymbol{v}_{c_3}) = \begin{pmatrix} a \\ b \end{pmatrix}, \quad \psi(\boldsymbol{v}_{c_4}) = \begin{pmatrix} 0 \\ b \end{pmatrix},$$

and parameterize the remaining boundary vertices by chord length over the sides of the rectangle $[0, a] \times [0, b]$, e.g.

$$\psi(\boldsymbol{v}_i) = \begin{pmatrix} a \\ t_i \end{pmatrix}, \quad t_{i+1} = t_i + \frac{\mu}{D_i}, \quad i = c_2 + 1, \ldots, c_3 - 1,$$

with

$$\mu = \frac{b}{\sum_{i=c_2}^{c_3-1} 1/D_i} \qquad \text{and} \qquad t_{c_2} = 0.$$

Basically there are two ways of specifying the corner vertices. Firstly, we can take into account the angle $\sphericalangle(\boldsymbol{v}_{i-1}, \boldsymbol{v}_i, \boldsymbol{v}_{i+1})$ of the boundary polygon at $\boldsymbol{v}_i$ and take those four vertices with the smallest angles. This method works well as long as the boundary polygon has a squarish shape with four almost equally spaced vertices whose angles are significantly smaller than all other angles. Otherwise

we should define the corner vertices such that the distance $\sum_{j=c_i}^{c_{i+1}-1} \|\boldsymbol{v}_{j+1} - \boldsymbol{v}_j\|$ between them reflects the side lengths of the rectangle.

The third boundary parameterization method is based on the reproduction property in Theorems 1.4 and 1.7 which states that a bijective parameterization is obtained for a planar triangulation by parameterizing the boundary with the identity, regardless of whether $\partial\mathcal{T}$ is convex or not. Since the parameterization depends continuously on the coordinates of the vertices, this motivates the following procedure for parameterizing $\partial\mathcal{T}$. Note that it reproduces the identical mapping for planar triangulations.

We project the boundary vertices orthogonally into the plane P that approximates V_B best in a least squares sense and further map the projected vertices $\boldsymbol{v}'$ to $\mathbb{R}^2$ using a suitable solid body transformation in order to obtain the parameter points $\psi(\boldsymbol{v})$. The *least squares plane* of V_B is defined as the plane P that minimizes $\sum_{\boldsymbol{v}\in V_B} \operatorname{dist}(\boldsymbol{v}, P)^2$ and can be characterized as follows.

Proposition 1.3 *For a set of vertices V with centroid $\boldsymbol{c} = \frac{1}{|V|}\sum_{\boldsymbol{v}\in V} \boldsymbol{v}$ the plane $P = \{\boldsymbol{p} \in \mathbb{R}^3 : (\boldsymbol{p} - \boldsymbol{c}) \perp \boldsymbol{n}\}$ is the least squares plane of V if and only if $\boldsymbol{n}$ is the normalized eigenvector belonging to the smallest eigenvalue of the symmetric 3×3 matrix $A = \sum_{\boldsymbol{v}\in V}(\boldsymbol{v} - \boldsymbol{c})(\boldsymbol{v} - \boldsymbol{c})^t$.*

Proof. The distance between $\boldsymbol{v}$ and a plane $P_{\boldsymbol{s}} = \{\boldsymbol{p} \in \mathbb{R}^3 : (\boldsymbol{p} - \boldsymbol{s}) \perp \boldsymbol{n}\}$ is given by $\operatorname{dist}(\boldsymbol{v}, P_{\boldsymbol{s}}) = \langle\, \boldsymbol{v} - \boldsymbol{s} \,|\, \boldsymbol{n} \,\rangle$. For any plane $P_{\boldsymbol{s}}$ that does not contain $\boldsymbol{c}$ it follows $\operatorname{dist}(\boldsymbol{c}, P_{\boldsymbol{s}})^2 = \langle\, \boldsymbol{c} - \boldsymbol{s} \,|\, \boldsymbol{n} \,\rangle^2 > 0$ and

$$\begin{aligned}
\sum_{\boldsymbol{v}\in V} \operatorname{dist}(\boldsymbol{v}, P_{\boldsymbol{s}})^2 &= \sum_{\boldsymbol{v}\in V} \langle\, \boldsymbol{v} - \boldsymbol{s} \,|\, \boldsymbol{n} \,\rangle^2 = \sum_{\boldsymbol{v}\in V} \big(\langle\, \boldsymbol{v} - \boldsymbol{c} \,|\, \boldsymbol{n} \,\rangle + \langle\, \boldsymbol{c} - \boldsymbol{s} \,|\, \boldsymbol{n} \,\rangle\big)^2 \\
&= \sum_{\boldsymbol{v}\in V} \langle\, \boldsymbol{v} - \boldsymbol{c} \,|\, \boldsymbol{n} \,\rangle^2 + 2 \underbrace{\sum_{\boldsymbol{v}\in V} \langle\, \boldsymbol{v} - \boldsymbol{c} \,|\, \boldsymbol{n} \,\rangle \langle\, \boldsymbol{c} - \boldsymbol{s} \,|\, \boldsymbol{n} \,\rangle}_{=0} \\
&\quad + \sum_{\boldsymbol{v}\in V} \underbrace{\langle\, \boldsymbol{c} - \boldsymbol{s} \,|\, \boldsymbol{n} \,\rangle^2}_{>0} \\
&> \sum_{\boldsymbol{v}\in V} \operatorname{dist}(\boldsymbol{v}, P_{\boldsymbol{c}})^2,
\end{aligned}$$

proving $\boldsymbol{c}$ to lie in the least squares plane. Furthermore, we have

$$\sum_{\boldsymbol{v}\in V} \operatorname{dist}(\boldsymbol{v}, P_{\boldsymbol{c}})^2 = \sum_{\boldsymbol{v}\in V} \langle\, \boldsymbol{v} - \boldsymbol{c} \,|\, \boldsymbol{n} \,\rangle^2 = \sum_{\boldsymbol{v}\in V} \big[\boldsymbol{n}^t \underbrace{(\boldsymbol{v} - \boldsymbol{c})\big]\big[(\boldsymbol{v} - \boldsymbol{c})^t}_{=A_{\boldsymbol{v}}} \boldsymbol{n}\big] = \boldsymbol{n}^t A \boldsymbol{n}$$

with $A = \sum_{\boldsymbol{v}\in V} A_{\boldsymbol{v}}$. From real analysis we know that $f(\boldsymbol{n}) = \boldsymbol{n}^t A \boldsymbol{n}$ obtains its extrema with the condition $g(\boldsymbol{n}) = \|\boldsymbol{n}\|^2 = \boldsymbol{n}^t \boldsymbol{n} = 1$ if and only if there exists a $\lambda \in \mathbb{R}$ with

$$f'(\boldsymbol{n}) = \lambda g'(\boldsymbol{n}) \quad \Longleftrightarrow \quad A\boldsymbol{n} = \lambda \boldsymbol{n} \quad \Longleftrightarrow \quad \lambda \text{ is an eigenvalue of } A.$$

Due to $f(\boldsymbol{n}) = \boldsymbol{n}^t A \boldsymbol{n} = \boldsymbol{n}^t \lambda \boldsymbol{n} = \lambda \|\boldsymbol{n}\|^2 = \lambda$, f is minimal for the normalized eigenvector belonging to the smallest eigenvalue of A. □

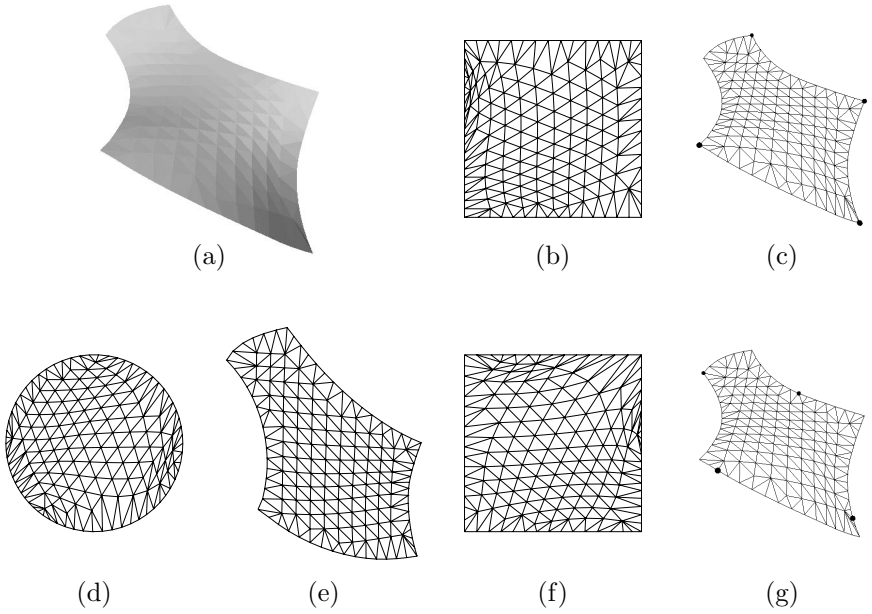


Figure 1.16: Different shape preserving parameterizations of the triangulation in (a) with $|V_I| = 100$, $|V_B| = 66$.

The projected vertices are then determined via $\boldsymbol{v}' = \boldsymbol{v} - \langle \boldsymbol{v} - \boldsymbol{c} \,|\, \boldsymbol{n} \rangle$ and the transformation to $\mathbb{R}^2$ can be realized, for example, by expressing $\boldsymbol{v}'$ in a local coordinate system that is perpendicular to $\boldsymbol{n}$.

Figure 1.16 gives an example of the impact the different boundary parameterization methods have on the shape preserving parameterization of the interior vertices. Distributing the boundary vertices by chord length on a circle gives the result shown in (d) and projecting them into the least squares plane leads to the parameterization in (e). If we specify four corner vertices using the smallest angle criterion (c) or by uniform distribution (g) and parameterize the boundary over a square we obtain the parameterizations in (b) and (f) respectively. Note that in this example the projection method gives the best result, namely the one with least distortion.

Another example is shown in Figure 1.17. Here the projection method (c) yields a non-bijective parameterization (one of the triangles on the boundary in the upper right part is flipped) and the rectangular boundary parameterization method (d) is to be preferred instead. In general it will depend on both the triangulation and the application which method is most appropriate and the choice should be left to the user.

We conclude this section by emphasizing the necessity of fixing the parameter values of the boundary vertices V_B prior to parameterizing the interior vertices V_I. Taking a closer look at the matrices A and B in Equations 1.5 and 1.15 we observe the following. Each row of both A and B is weakly diagonal

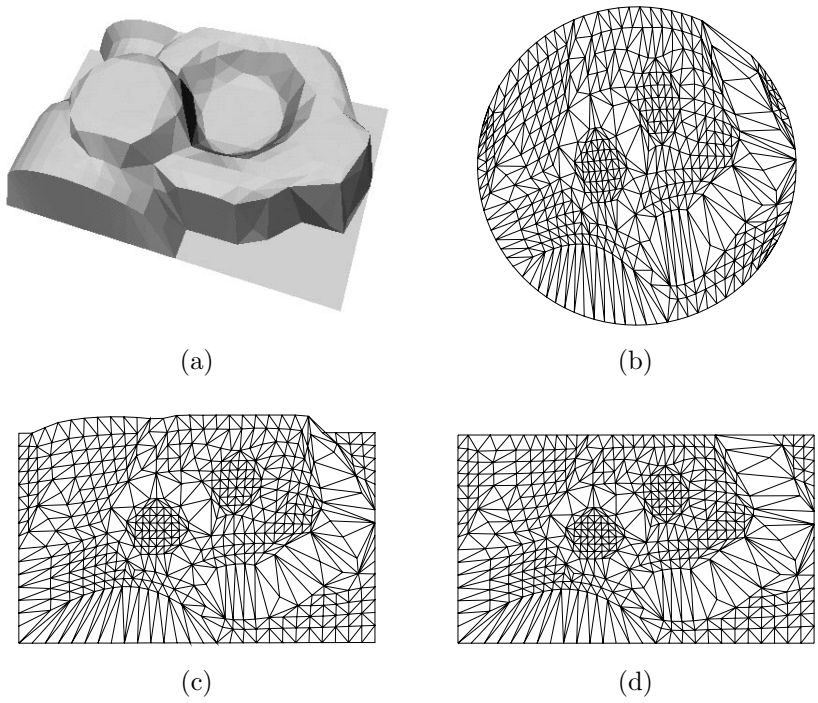


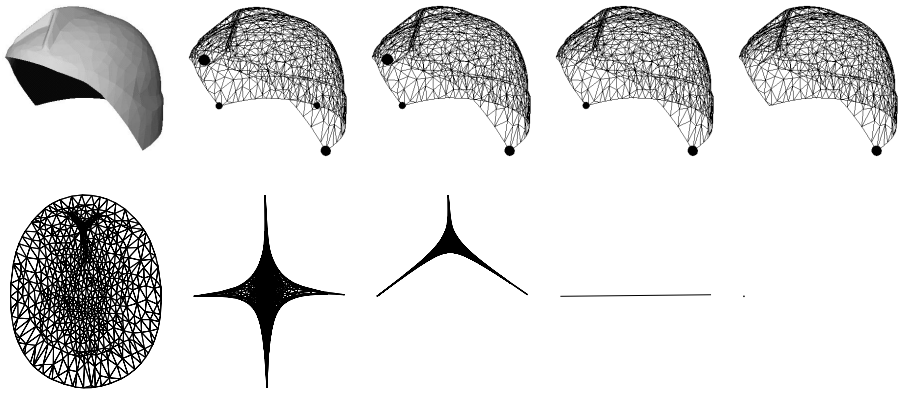
Figure 1.17: Different shape preserving parameterizations of the triangulation in (a) with $|V_I| = 390$, $|V_B| = 86$.

dominant and only those rows corresponding to an interior vertex with at least one neighbouring boundary vertex are strictly diagonal dominant. It is exactly these rather few rows that keep the matrices from being singular.

Figure 1.18 illustrates the effect of fixing the parameter points of a subset $V_B' \subset V_B$ only and treating the remaining $V_B \setminus V_B'$ like interior vertices instead. For $|V_B'| \geq 3$ the parameterization is star-shaped and highly distorted (b,c). Since we used chord length parameterization in this example which guarantees the interior parameter points $\psi(\boldsymbol{v})$, $\boldsymbol{v} \in V_I$ to lie within the convex hull of $\psi(\boldsymbol{v})$, $\boldsymbol{v} \in V_B'$, the parameterization is degenerated to a line or a point if $|V_B'| = 2$ (d) or $|V_B'| = 1$ (e). In the extreme case of $V_B' = \emptyset$ both matrices A and B become singular and for any $\boldsymbol{c} \in \mathbb{R}^2$ the constant parameterization $\psi(\boldsymbol{v}) = \boldsymbol{c}$, $\boldsymbol{v} \in V$ solves (1.5) as well as (1.15). Note that the proofs of the regularity of A and B in Theorems 1.2, 1.3, and 1.5 do not hold for $V_B = \emptyset$ accordingly.

1.2.6 Solving the Linear System

So far we have only discussed the existence and uniqueness of the linear parameterization methods as solutions to (1.5) or (1.15). We will now study how to efficiently solve these matrix equations. Although *direct methods* like LU



(a) (b) (c) (d) (e)

Figure 1.18: Chord length parameterization of a triangulation ($|V_I| = 636$, $|V_B| = 64$) with all (a) and only a few (b–e) boundary vertices fixed.

decomposition could be used, we consider *iterative methods* only since they are more appropriate if the matrix is large and sparse. Our matrices indeed have both properties because there can easily be as many as several thousands of interior vertices in a given triangulation and the number of non-zero entries per matrix row is $|N_{\boldsymbol{v}}| + 1$, i.e. 7 on average. This means that for a triangulation with 2000 interior vertices only 0.35 % of the matrix coefficients are non-zero.

The simplest class of iterative methods for solving matrix systems $Ax = b$ is based on a *splitting* $A = M - N$ of A such that the iteration step

$$Mx^{(k+1)} = Nx^{(k)} + b \tag{1.19}$$

is "easy" to solve. The convergence of the process depends on the *spectral radius*

$$\rho(G) = \max\{|\lambda| : \lambda \in \lambda(G)\},$$

i.e. the magnitude of the largest eigenvalue of the *iteration matrix* $G = M^{-1}N$. If $\rho(G) < 1$ then the iterates $x^{(k)}$ converge for any starting vector $x^{(0)}$ and the rate of convergence is the faster the smaller $\rho(G)$ is. The most prominent representatives of this method are the *Jacobi* and the *Gauss-Seidel iteration* (GS) with iteration matrices

$$G_J = -D^{-1}(L + U) \qquad \text{and} \qquad G_{GS} = -(D + L)^{-1}U$$

respectively, where $A = L + D + U$ is the usual splitting of A into its lower triangular, diagonal, and upper triangular part. For the kind of matrices that we consider for determining parameterizations in (1.5) and (1.15) some important theorems are known (see [134, 140, 127, 53] for details and proofs).

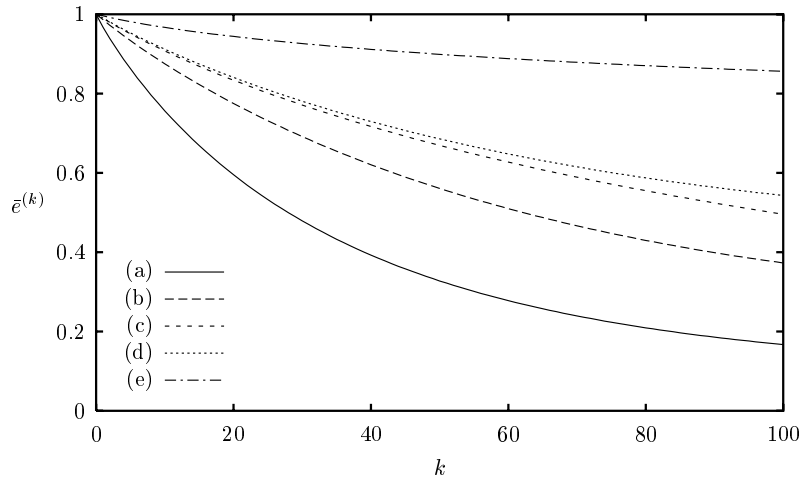


Figure 1.19: Convergence rates of the Gauss-Seidel iteration for the examples in Figure 1.18.

Theorem 1.8 *For a weakly diagonal dominant matrix A the spectral radius of the Jacobi iteration matrix is $\rho(G_J) \leq 1$. Moreover, if A is irreducible with positive D and non-positive $L+U$, then the following statements are equivalent:*

(i) $\rho(G_J) < 1$,

(ii) *at least one row of A is strictly diagonal dominant,*

(iii) *A is regular.*

These propositions also hold for the Gauss-Seidel iteration.

Note that the statements (ii) and (iii) of this theorem correspond with the previously mentioned fact that the parameterizations are well-defined only if the boundary vertices are fixed. While Theorem 1.8 guarantees both the Jacobi iteration and GS to converge the next theorem tells which one should be preferred.

Theorem 1.9 *(Stein-Rosenberg) If $L + U$ is non-positive, then one of the following relations holds:*

$$\begin{aligned}
&\text{(i)} && \rho(G_{GS}) = \rho(G_J) = 0 \\
&\text{(ii)} && 0 < \rho(G_{GS}) < \rho(G_J) < 1 \\
&\text{(iii)} && \rho(G_{GS}) = \rho(G_J) = 1 \\
&\text{(iv)} && \rho(G_{GS}) > \rho(G_J) > 1
\end{aligned}$$

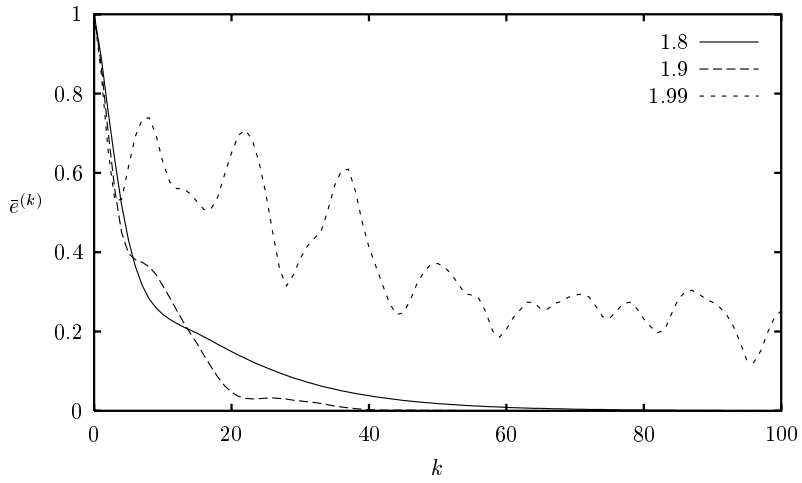


Figure 1.20: Convergence rates of successive over-relaxation with different relaxation parameters for the example in Figure 1.18.a.

Still we expect GS to be rather slow, especially for large triangulations where the number of rows of A that are strictly diagonal dominant is small compared to the size of the matrix and A is "almost" singular. Figure 1.19 affirms this assumption by showing the convergence rates of GS for the examples in Figure 1.18. The less boundary vertices are fixed and the less dominant the diagonal of A therefore is, the slower the method converges. In all cases we took the orthogonal projection of V_I into the least squares plane of V_B as initial vector $\psi^{(0)}(\boldsymbol{v})$ and plotted the *normalized residual error*

$$\bar{e}^{(k)} = \frac{e^{(k)}}{e^{(0)}} \qquad \text{with} \qquad e^{(k)} = \frac{1}{|V_I|} \sum_{\boldsymbol{v} \in V_I} \|\psi^{(k)}(\boldsymbol{v}) - \psi^*(\boldsymbol{v})\|,$$

where $\psi^{(k)}$ is the approximate solution after k iterations and ψ^* is the exact solution of the matrix equation.

It is possible to improve the convergence rate of GS with the method of *successive over-relaxation* (SOR) which is given by (1.19) with $M = D + \omega L$ and $N = (1-\omega)D - \omega U$. Note that this resembles GS for $\omega = 1$. The idea behind SOR is to choose the *relaxation parameter* ω such that the spectral radius of the iteration matrix

$$G_{SOR} = (D + \omega L)^{-1}((1-\omega)D - \omega U)$$

is minimized. The problem is that the optimal ω^* is usually hard to guess. Two of the few results known are that SOR can only converge for $0 < \omega < 2$ and that $1 \leq \omega^* < 2$ for irreducible matrices with non-positive $L+U$. Further statements can only be made for special kinds of matrices.

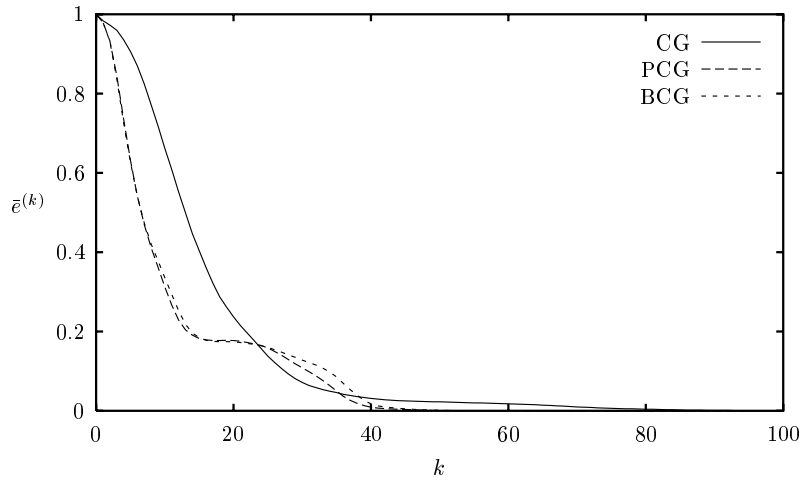


Figure 1.21: Convergence rates of various conjugate gradient methods for the example in Figure 1.18.a.

Theorem 1.10 *(Young, Varga) Let A be a* consistently ordered[1] *matrix. Further let all eigenvalues of G_J be real and $\rho(G_J) < 1$. Then*

$$\omega^* = \frac{2}{1 + \sqrt{1 - \rho(G_J)^2}}.$$

Hence ω^* is close to 2 if $\rho(G_J)$ is close to 1. Although our matrices A in (1.5) and B in (1.15) are not consistently ordered, $\omega = 2 - \varepsilon$ with small $\varepsilon > 0$ seems to be an appropriate choice and ε should be the smaller the larger $|V|$ is. Yet Figure 1.20 shows the sensitivity of SOR's convergence rate to ω.

We prefer to use the *method of conjugate gradients* (CG) instead which performed as well as SOR with optimal relaxation parameter ω^* in the examples tested. CG is guaranteed to converge for symmetric positive definite matrices and is therefore suitable for determining spring model as well as harmonic parameterizations. Moreover, CG can be speeded up considerably by slightly modifying the matrix equation. Instead of $Ax = b$ we solve

$$\tilde{A}\tilde{x} = \tilde{b} \qquad \text{with} \qquad \tilde{A} = C^{-1}AC^{-1}, \quad \tilde{x} = Cx, \quad \tilde{b} = C^{-1}b,$$

where C is symmetric positive definite. This method is called *preconditioned conjugate gradient method* (PCG) with *preconditioner* $M = C^2$. The idea behind is to choose C such that the condition number $\kappa(\tilde{A})$ of the new matrix is smaller than $\kappa(A)$ because the convergence rate of CG depends on this value. PCG turns out to be an effective technique, provided the system $My = z$ can easily be

[1] A matrix $A = U + D + L$ is consistently ordered if the eigenvalues of $D^{-1}(\alpha L + \alpha^{-1}U)$ are independent of $\alpha \neq 0$.

solved [53]. The following proposition motivates to use the simple preconditioner $M = \mathrm{diag}(A)$ which we found to perform quite well (see Figure 1.21) at negligible extra costs. Note that this always yields a matrix $\tilde{A}$ with unit diagonal.

Proposition 1.4 *For any* 2×2 *matrix*

$$A = \begin{pmatrix} a+\delta & -a \\ -a & a+\varepsilon \end{pmatrix}$$

with $a, \delta, \varepsilon \geq 0$ *and* $\det(A) = a(\delta+\varepsilon) + \delta\varepsilon > 0$ *we have*

$$\kappa_F(A) \geq \kappa_F(\tilde{A})$$

with $\tilde{A} = C^{-1}AC^{-1}$ *and* $C^2 = \mathrm{diag}(A)$.

Proof. For any 2×2 matrix $M = \left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right)$ we have $M^{-1} = \frac{1}{\det(M)}\left(\begin{smallmatrix} d & -b \\ -c & a \end{smallmatrix}\right)$ and conclude $\|M\|_F^2 = a^2+b^2+c^2+d^2$, $\|M^{-1}\|_F^2 = \frac{\|M\|_F^2}{\det(M)^2}$, and $\kappa_F(M) = \frac{\|M\|_F^2}{|\det(M)|}$. If we now consider the matrices

$$A = \begin{pmatrix} a+\delta & -a \\ -a & a+\varepsilon \end{pmatrix} \qquad \text{and} \qquad \tilde{A} = \begin{pmatrix} 1 & \frac{-a}{\sqrt{a+\delta}\sqrt{a+\varepsilon}} \\ \frac{-a}{\sqrt{a+\delta}\sqrt{a+\varepsilon}} & 1 \end{pmatrix},$$

we have $\det(\tilde{A}) = \frac{\det(A)}{(a+\delta)(a+\varepsilon)}$,

$$\kappa_F(A) = \frac{(a+\delta)^2 + (a+\varepsilon)^2 + 2a^2}{\det(A)}, \qquad \kappa_F(\tilde{A}) = \frac{2(a+\delta)(a+\varepsilon) + 2a^2}{\det(A)},$$

and finally

$$\kappa_F(A) - \kappa_F(\tilde{A}) = \frac{(\delta-\varepsilon)^2}{\det(A)} \geq 0.$$

□

Another way of preconditioning (1.5) and of obtaining a unit diagonal is to multiply the system by M^{-1} from left,

$$M^{-1}Ax = M^{-1}b.$$

This is exactly how a spring model parameterization problem (1.5) turns into a convex combination problem (1.15) since $B = M^{-1}A$ and $c = M^{-1}b$ with the convex weights chosen as in (1.16).

In this more general setting of Section 1.2.4 we loose symmetry and can no longer apply CG or PCG. Still we can use an iterative solver known as the *biconjugate gradient method* (BCG) which was developed to adapt the CG technique to non-symmetric linear systems [132]. Although few theoretical results are known about the convergence it is agreed on that BCG is performing well in practice as confirmed, for example, by Figure 1.21.

Another example is given in Figures 1.22 and 1.23 where the chord length parameterization of a triangulation with $|V| = 17,653$ has been computed. The

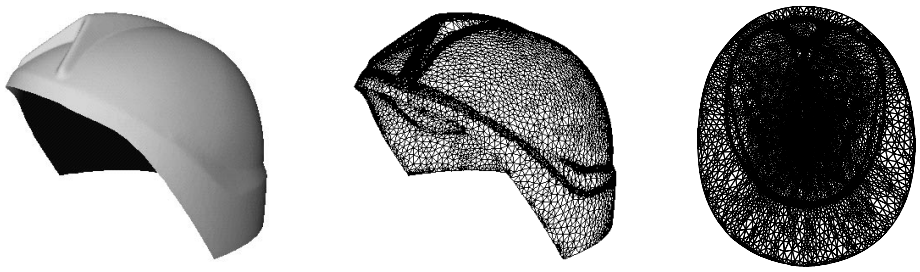


Figure 1.22: A triangulation with $|V_I| = 17,408$, $|V_B| = 245$ and its chord length parameterization.

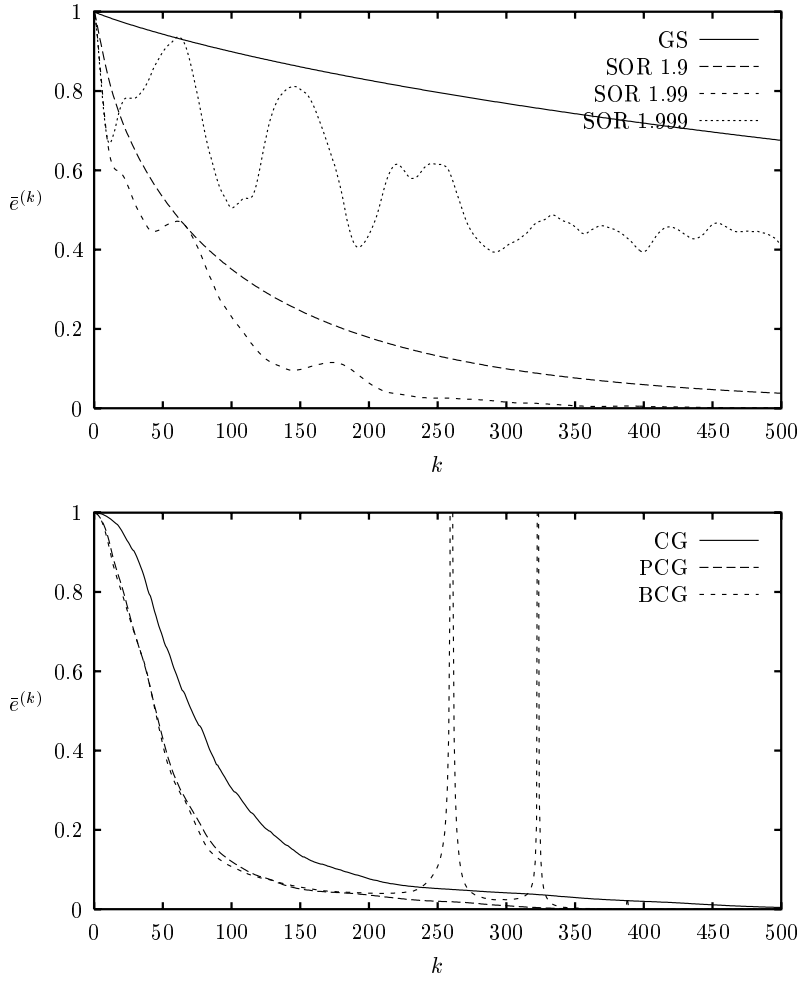


Figure 1.23: Convergence rates of different iterative solvers for the example in Figure 1.22.

convergence rates of the different iterative methods are similar to the previous example. GS converges very slowly but can be accelerated considerably by SOR if an adequate relaxation parameter is known. The performance of CG is comparable with optimal SOR and preconditioning (PCG) improves it by factor 2, approximately. BCG converges similarly to PCG except for the little irregularities at $k \approx 260$ and $k \approx 320$. We therefore strongly advise to use BCG to determine the shape preserving parameterization only and to apply PCG otherwise.

1.3 Most Isometric Parameterizations

The linear parameterization methods discussed in the previous sections clearly depend on the fixed parameter points $\psi(\boldsymbol{v})$ for $\boldsymbol{v} \in V_B$, i.e. on the choice of the boundary polygon $\psi(\partial\mathcal{T})$. In Section 1.2.5 we have presented various ways of determining $\psi(\partial\mathcal{T})$ but it is hard to automatically decide for a given triangulation which method produces the best results and the choice should be left to the user's expertise instead. The problem is that the linear methods are limited in the sense that they can be used to determine the interior parameter values only and do not give reasonable results if the boundary parameter points are included in the optimization process. In this section we introduce a *nonlinear* parameterization method that overcomes this limitation and yields parameter points $\psi(\boldsymbol{v})$ for all $\boldsymbol{v} \in V$.

1.3.1 Shape Deformation of Triangles

The nonlinear parameterization technique we are to discuss is based on the observation that ψ normally deforms the shape of the triangles of $\mathcal{T}$ and aims at minimizing these deformations. Consider the local configuration in Figure 1.14.a, which can be parameterized without distorting the triangles only if the interior angles γ_i sum to 2π, e.g. if $\boldsymbol{v}, \boldsymbol{w}_1, \ldots, \boldsymbol{w}_n$ lie in the same plane. This reflects a well-known fact from differential geometry [29] that only developable surfaces (like planar, cylindrical and conical surfaces) can be parameterized without any distortion. Such surfaces are *isometric* to the plane.

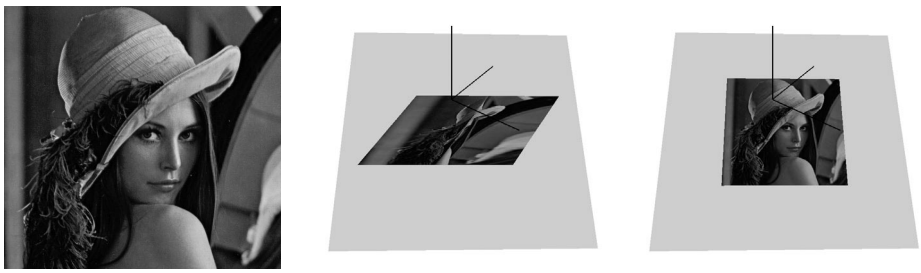
Definition 1.2 *Let $\Omega \subset \mathbb{R}^2$ and $f : \Omega \to \mathbb{R}^3$ be a differentiable and injective mapping and thus a parameterization of the surface $f(\Omega) \subset \mathbb{R}^3$. The symmetric 2×2 matrix*

$$\mathbf{I}_f = \nabla^t f \cdot \nabla f$$

is called the first fundamental form *of f. The parameterization f is called* isometric*, if*

$$\mathbf{I}_f = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = I$$

and a surface $S \subset \mathbb{R}^3$ is called isometric *to Ω if there exists an isometric parameterization f of S with $f(\Omega) = S$.*



(a) (b) (c)

Figure 1.24: Mapping a texture (a) to a surface using a non-isometric (b) and an isometric parameterization (c).

The first fundamental form $\mathbf{I}_f$ expresses how a surface inherits the metric from the parameter domain by f. The further $\mathbf{I}_f$ deviates from the identity matrix, the more the metric quantities (length, angle, and area) are distorted. Surfaces that are parameterized isometrically will therefore behave (locally) like planes which is very desirable in many applications. Consider, for example, the functions $f, g : \mathbb{R}^2 \to \mathbb{R}^3$,

$$f\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} 4u^3 + 1 \\ 3u^3 + 3v \\ v - 2 \end{pmatrix} \qquad \text{and} \qquad g = f \circ \pi_2 \circ \pi_1,$$

with $\pi_1, \pi_2 : \mathbb{R}^2 \to \mathbb{R}^2$,

$$\pi_1\begin{pmatrix} u \\ v \end{pmatrix} = \frac{1}{65}\begin{pmatrix} 13 & -9 \\ 0 & 25 \end{pmatrix}\begin{pmatrix} u \\ v \end{pmatrix} \qquad \text{and} \qquad \pi_2\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} \sqrt[3]{u} \\ v \end{pmatrix},$$

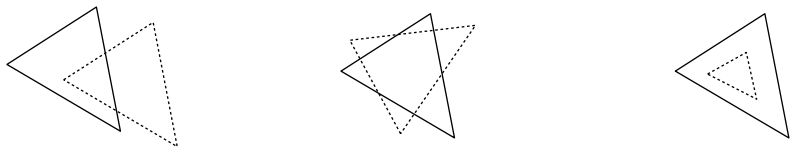
as two parameterizations of the surface $S = f(\mathbb{R}^2) = g(\mathbb{R}^2)$. We have

$$\nabla f\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} 12u^2 & 0 \\ 9u^2 & 3 \\ 0 & 1 \end{pmatrix}, \quad \mathbf{I}_f\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} 225u^4 & 27u^2 \\ 27u^2 & 10 \end{pmatrix},$$

and

$$g\begin{pmatrix} u \\ v \end{pmatrix} = \frac{1}{65}\begin{pmatrix} 52u - 36v + 65 \\ 39u + 48v \\ 25v - 130 \end{pmatrix}, \quad \nabla g\begin{pmatrix} u \\ v \end{pmatrix} = \frac{1}{65}\begin{pmatrix} 52 & -36 \\ 39 & 48 \\ 0 & 25 \end{pmatrix}, \quad \mathbf{I}_g\begin{pmatrix} u \\ v \end{pmatrix} = I,$$

thus g is isometric whereas f is not. Using f and g to paste the texture information (a) in Figure 1.24 onto S, we can see how f distorts the image (b) while the isometric parameterization g perfectly preserves the planar metric (c).



(a) (b) (c)

Figure 1.25: Translations (a), orthogonal transformations (b), and uniform scalings (c) do not modify the shape of a triangle.

Let us now go back to the problem of finding for a given triangulation $\mathcal{T}$ a piecewise linear mapping $\psi \in S_2(\mathcal{T})$ such that the distortion between the triangles $T \in \mathcal{T}$ and their images $\psi(T)$ is small. For each $T \in \mathcal{T}$ we can write the restriction $\psi_{|T}$ as a linear mapping

$$\psi(\boldsymbol{v}) = A\boldsymbol{v} + \boldsymbol{p}, \qquad \boldsymbol{v} \in T,$$

with $\boldsymbol{p} \in \mathbb{R}^2$. By introducing a local orthonormal coordinate system at T with the third axis perpendicular to T, we can assume $A \in \mathbb{R}^{2\times 2}$ and therefore

$$\psi_{|T} \in \Pi_1^2, \qquad T \in \mathcal{T},$$

where $\Pi_1^2 = \{f : \mathbb{R}^2 \to \mathbb{R}^2 : f \text{ linear}\}$ denotes the space of two-dimensional linear functions. The question now is how to measure the shape deformation between T and $\psi(T)$. Any such measure should be

- invariant with respect to translations, $(*1)$
- invariant with respect to orthogonal transformations, $(*2)$
- invariant with respect to uniform scalings, $(*3)$

since these are the linear operations that do not modify the shape of a triangle (see Figure 1.25). Let

$$\mathcal{E} = \{E : \Pi_1^2 \to \mathbb{R}, E \text{ fulfills } (*1), (*2), \text{ and } (*3)\}$$

be the set of functionals that measure the shape deformation of a linear function. Note that the Dirichlet energy (1.9) is not a shape deformation functional of this type, $E_D \notin \mathcal{E}$, since it clearly depends on uniform scalings.

Theorem 1.11 *For $f(\boldsymbol{x}) = A\boldsymbol{x} + \boldsymbol{b}$ and $E \in \mathcal{E}$, $E(f)$ depends on the ratio $\rho = \sigma_1/\sigma_2$ of the* singular values *of A only. Any such deformation functional can thus be seen as a function $E : \{\rho \in \mathbb{R} : \rho \geq 1\} \to \mathbb{R}$.*

Proof. Because of (∗1), $E(f)$ does not depend on the constant part $\boldsymbol{b}$ of f. If we then consider the *singular value decomposition* [53] of A which guarantees the existence of orthogonal matrices U and V such that

$$U^t A V = \Sigma = \begin{pmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{pmatrix},$$

where $\sigma_1 \geq \sigma_2 \geq 0$ are the singular values of A, it is clear that any E fulfilling (∗2) depends on σ_1 and σ_2 only. The singular values are the lengths of the semi-axes of the ellipse $\{A\boldsymbol{x} : \|\boldsymbol{x}\|_2 = 1\}$ and indicate the scaling of the linear mapping along the principal axes. Because of (∗3) the statement holds. □

The ratio $\rho = \sigma_1/\sigma_2$ is minimal ($\rho = 1$) for all linear mappings that do not deform shape ($\sigma_1 = \sigma_2$) and maximal ($\rho = \infty$) for all singular mappings ($\sigma_2 = 0$) that let triangles collapse to lines ($\sigma_1 > 0$) or points ($\sigma_1 = 0$). This motivates to consider only those functionals $E \in \mathcal{E}$, which are monotonically increasing as a function of ρ and to call them *proper shape deformation functionals*. For any proper shape deformation functional E we can then find a mapping ψ^* in $S_2(\mathcal{T})$ which minimizes

$$\sum_{T \in \mathcal{T}} E(\psi_{|T}) \,. \tag{1.20}$$

We will now look for a simple choice of such an E and find the concept of *condition numbers* of matrices [53] to provide an adequate tool. The condition number $\kappa(A) = \|A\| \, \|A^{-1}\|$ of a matrix A quantifies the sensitivity of the linear system $A\boldsymbol{x} = \boldsymbol{b}$ and measures the inverse distance of A to the set of singular matrices with respect to a certain matrix norm $\|\cdot\|$. A large value $\kappa(A)$ indicates that there exists a perturbation M with $\|M\|$ small such that $A+M$ is singular. By convention, $\kappa(A) = \infty$ for singular matrices.

Corollary 1.4 *For a linear function $f(\boldsymbol{x}) = A\boldsymbol{x} + \boldsymbol{b}$ the condition numbers κ_2 and κ_F of A and of the first fundamental form $\mathbf{I}_f = A^t A$ are proper shape deformation functionals.*

Proof. From linear algebra we know

$$\|A\|_2 = \sigma_1 \qquad \text{and} \qquad \|A\|_F^2 = \sigma_1^2 + \sigma_2^2$$

and that the singular values of $\mathbf{I}_f$ are σ_1^2 and σ_2^2. Therefore,

$$E_1(f) = \kappa_2(A) = \|A\|_2 \|A^{-1}\|_2 = \frac{\sigma_1}{\sigma_2} \,,$$

$$E_2(f) = \kappa_F(A) = \|A\|_F \|A^{-1}\|_F = \sqrt{\sigma_1^2 + \sigma_2^2}\sqrt{(\tfrac{1}{\sigma_1})^2 + (\tfrac{1}{\sigma_2})^2} = \frac{\sigma_1^2 + \sigma_2^2}{\sigma_1 \sigma_2} \,,$$

$$E_3(f) = \kappa_2(\mathbf{I}_f) = \kappa_2(A^t A) = \frac{\sigma_1^2}{\sigma_2^2} \,,$$

$$E_4(f) = \kappa_F(\mathbf{I}_f) = \kappa_F(A^t A) = \frac{\sigma_1^4 + \sigma_2^4}{\sigma_1^2 \sigma_2^2} \,,$$

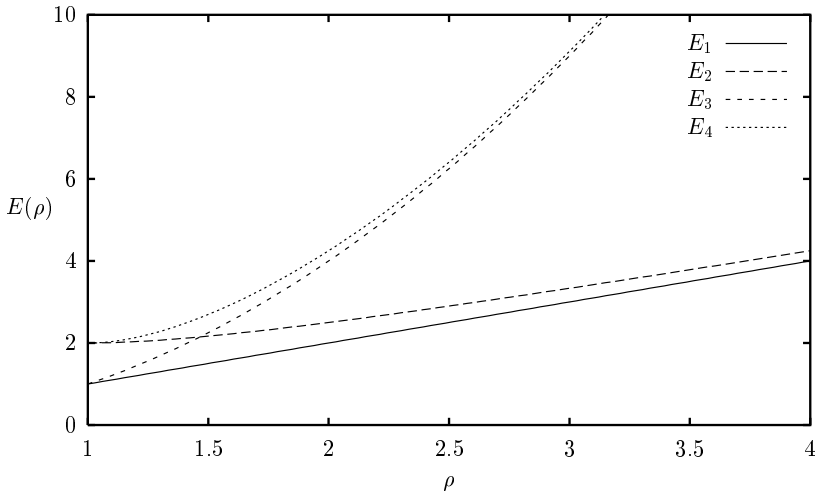


Figure 1.26: Comparison of different proper shape deformation functionals.

or, if seen as functions of $\rho = \sigma_1/\sigma_2$ (see Figure 1.26),

$$\begin{aligned} E_1(\rho) &= \rho, \\ E_2(\rho) &= \rho + \frac{1}{\rho}, \\ E_3(\rho) &= \rho^2, \\ E_4(\rho) &= \left(\rho + \frac{1}{\rho}\right)^2 - 2\,. \end{aligned}$$

□

In order be able to compute ψ efficiently by minimizing (1.20), we would further like to have a functional E that is a simple function of the unknown parameter points $\psi(\boldsymbol{v})$, $\boldsymbol{v} \in V$, which uniquely define ψ. Of course, we would prefer a quadratic dependence, since ψ could then be found by solving a linear system as in Section 1.2, but this automatically contradicts the postulated scaling invariance (∗3). However, we can show that E_2 is a rational quadratic function of $\psi(\boldsymbol{v})$, $\boldsymbol{v} \in V$.

Proposition 1.5 *For $\psi \in S_2(\mathcal{T})$ and with the notations of Figure 1.27 the proper shape deformation functional E_2 can be written as*

$$E_2(\psi_{|T}) = \frac{\cot\alpha\,\|\vec{a}\|^2 + \cot\beta\,\|\vec{b}\|^2 + \cot\gamma\,\|\vec{c}\|^2}{\|\vec{b}\times\vec{c}\|} \tag{1.21}$$

with $\vec{a} = \psi(\boldsymbol{v}) - \psi(\boldsymbol{w})$, $\vec{b} = \psi(\boldsymbol{w}) - \psi(\boldsymbol{u})$, and $\vec{c} = \psi(\boldsymbol{v}) - \psi(\boldsymbol{u})$.

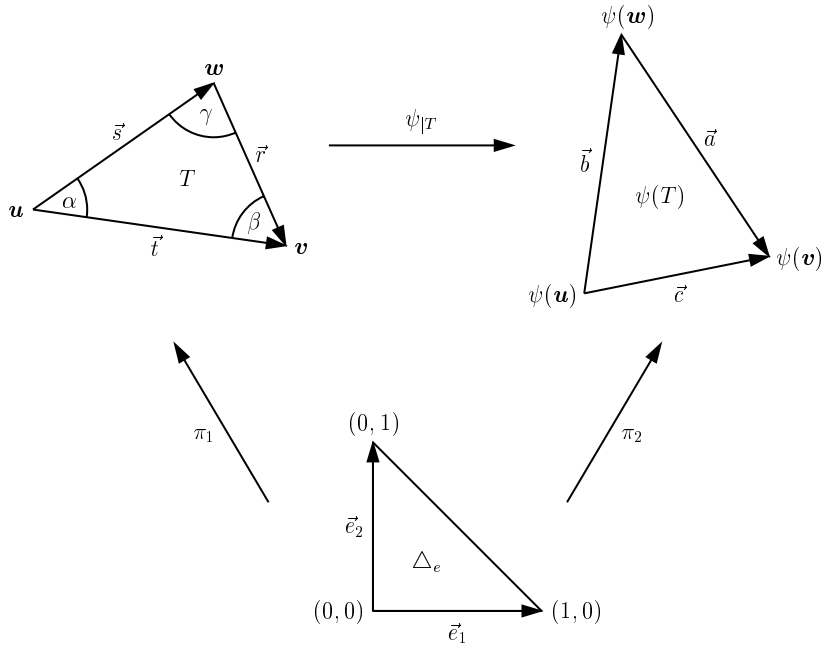


Figure 1.27: Decomposition of the linear map $\psi_{|T}$.

Proof. Let $\psi_{|T}$ be given by $\psi_{|T}(\boldsymbol{x}) = A\boldsymbol{x} + \boldsymbol{b}$. Then we know from Corollary 1.4

$$E_2(\psi_{|T}) = \kappa_F(A) = \frac{\sigma_1^2 + \sigma_2^2}{\sigma_1\sigma_2} = \frac{\operatorname{trace}(A^tA)}{|\det(A)|}.$$

Defining the two linear maps

$$\pi_1 : \triangle_e \to T,\ \boldsymbol{x} \mapsto A_1\boldsymbol{x} + \boldsymbol{u} \qquad \text{with} \qquad A_1 = (\vec{t}, \vec{s})$$

and

$$\pi_2 : \triangle_e \to \psi(T),\ \boldsymbol{x} \mapsto A_2\boldsymbol{x} + \psi(\boldsymbol{u}) \qquad \text{with} \qquad A_2 = (\vec{c}, \vec{b})$$

we can decompose $\psi_{|T}$ according to Figure 1.27 as $\psi_{|T} = \pi_2 \circ \pi_1^{-1}$ obtaining $A = A_2A_1^{-1}$ and $\boldsymbol{b} = \psi(\boldsymbol{u}) - A_2A_1^{-1}\boldsymbol{u}$. Considering

$$A_2^tA_2 = \begin{pmatrix} \langle\,\vec{c}\,|\,\vec{c}\,\rangle & \langle\,\vec{b}\,|\,\vec{c}\,\rangle \\ \langle\,\vec{b}\,|\,\vec{c}\,\rangle & \langle\,\vec{b}\,|\,\vec{b}\,\rangle \end{pmatrix}$$

and

$$A_1^{-1}A_1^{-t} = (A_1^tA_1)^{-1} = \frac{1}{\det(A_1)^2}\begin{pmatrix} \langle\,\vec{s}\,|\,\vec{s}\,\rangle & -\langle\,\vec{s}\,|\,\vec{t}\,\rangle \\ -\langle\,\vec{s}\,|\,\vec{t}\,\rangle & \langle\,\vec{t}\,|\,\vec{t}\,\rangle \end{pmatrix}$$

we get

$$\begin{aligned} E_2(\psi_{|T}) &= \frac{\operatorname{trace}(A^t A)}{|\det(A)|} = \frac{\operatorname{trace}(A_1^{-t} A_2^t A_2 A_1^{-1})}{|\det(A_2)|\,|\det(A_1^{-1})|} \\ &= \frac{|\det(A_1)|\operatorname{trace}(A_2^t A_2 A_1^{-1} A_1^{-t})}{|\det(A_2)|} \\ &= \frac{\langle\,\vec{c}\,|\,\vec{c}\,\rangle\langle\,\vec{s}\,|\,\vec{s}\,\rangle - 2\langle\,\vec{b}\,|\,\vec{c}\,\rangle\langle\,\vec{s}\,|\,\vec{t}\,\rangle + \langle\,\vec{b}\,|\,\vec{b}\,\rangle\langle\,\vec{t}\,|\,\vec{t}\,\rangle}{|\det(A_1)|\,|\det(A_2)|}. \end{aligned}$$

With $\vec{a} = \vec{c} - \vec{b}$ and $-2\langle\,\vec{b}\,|\,\vec{c}\,\rangle = \langle\,\vec{a}\,|\,\vec{a}\,\rangle - \langle\,\vec{b}\,|\,\vec{b}\,\rangle - \langle\,\vec{c}\,|\,\vec{c}\,\rangle$ we further have

$$\begin{aligned} E_2(\psi_{|T}) &= \frac{\langle\,\vec{a}\,|\,\vec{a}\,\rangle\langle\,\vec{s}\,|\,\vec{t}\,\rangle + \langle\,\vec{b}\,|\,\vec{b}\,\rangle\langle\,\vec{t}-\vec{s}\,|\,\vec{t}\,\rangle + \langle\,\vec{c}\,|\,\vec{c}\,\rangle\langle\,\vec{s}\,|\,\vec{s}-\vec{t}\,\rangle}{|\det(A_1)|\,|\det(A_2)|} \\ &= \frac{\|\vec{a}\|^2\langle\,\vec{s}\,|\,\vec{t}\,\rangle + \|\vec{b}\|^2\langle -\vec{r}\,|\,-\vec{t}\,\rangle + \|\vec{c}\|^2\langle -\vec{s}\,|\,\vec{r}\,\rangle}{|\det(A_1)|\,|\det(A_2)|}. \end{aligned}$$

Recalling $|\det(A_1)| = \|\vec{s}\times\vec{t}\| = 2\operatorname{area}(T)$,

$$\cos(\alpha) = \frac{\langle\,\vec{s}\,|\,\vec{t}\,\rangle}{\|\vec{s}\|\,\|\vec{t}\|}, \quad \sin(\alpha) = \frac{\|\vec{s}\times\vec{t}\|}{\|\vec{s}\|\,\|\vec{t}\|} \quad\Longrightarrow\quad \cot(\alpha) = \frac{\langle\,\vec{s}\,|\,\vec{t}\,\rangle}{2\operatorname{area}(T)}$$

and analogously for $\cot(\beta)$ and $\cot(\gamma)$ we conclude

$$E_2(\psi_{|T}) = \frac{\|\vec{a}\|^2\cot(\alpha) + \|\vec{b}\|^2\cot(\beta) + \|\vec{c}\|^2\cot(\gamma)}{|\det(A_2)|}.$$

The statement finally follows with $|\det(A_2)| = \|\vec{b}\times\vec{c}\|$. □

Interestingly, this reveals a close connection between the Dirichlet energy E_D (1.9) and E_2,

$$2E_D(\psi_{|T}) = E_2(\psi_{|T})\operatorname{area}(\psi(T)), \tag{1.22}$$

confirming an observation we have made before. While E_2 is unaffected by uniform scalings of $\psi(T)$, the Dirichlet energy favours small parameter triangles which is the reason why the parameter values of the boundary vertices had to be fixed in Section 1.2.3.

Since the shape deformation measured by E_2 is minimal for isometric mappings, minimizing

$$E_M(\psi) = \sum_{T\in\mathcal{T}} E_2(\psi_{|T}) \tag{1.23}$$

gives parameterizations $\phi = \psi^{-1}$ that are "as isometric as possible". This has led to the term MIPS [69], an acronym of most isometric parameterizations. We will also call E_M the *MIPS energy*.

1.3.2 Minimizing the MIPS Energy

When it comes to minimizing the MIPS energy E_M (1.23) we have to consider the following. As stated in Proposition 1.5, E_M is a quadratic rational function and it is possible to derive E_M analytically. The actual computation of ∇E_M is furthermore quite efficient (see Proposition 1.6) and suggests to use either a *multivariate conjugate gradient* method or the *Davidon-Fletcher-Powell* algorithm [105, 107]. The problem with both methods is that they require as a subtask to minimize E_M along a line,

$$\min_{t\in\mathbb{R}} E_M(\psi^0 + t\psi).$$

Albeit simple in principle this one-dimensional search is critical if we accept bijective mappings only. Assuming ψ^0 to be a bijection, the continuity of E_M guarantees an interval $I =]a, b[$ with $a < 0 < b$ to exist such that $\psi^0 + t\psi$ is bijective for all $t \in I$. However, finding I in order to restrict the line minimization problem to this interval requires to carry out quite often the rather costly operation of deciding whether a triangulation $\psi(\mathcal{T})$ self-intersects or not.

Instead, we use a Gauss-Seidel-like optimization approach and minimize the MIPS energy successively with respect to one parameter value at a time:

procedure **optimize** (ψ)
  repeat
    choose $\boldsymbol{v} \in V$ at random
    minimize E_M with respect to $\psi(\boldsymbol{v})$ $\quad(*)$
  until numerical convergence

The advantage of this search method is that the *local* minimization problem $(*)$ is convex and thus has a unique solution. Moreover, if the initial mapping ψ is bijective then this property is automatically preserved by each local optimization step. Before proving these statements let us establish some useful lemmas.

Lemma 1.4 *Let $a, b, c, d, e \in \mathbb{R}$ with $a \neq 0$ and $d \neq 0$. Then*

$$f : \mathbb{R} \setminus \{\tfrac{-e}{d}\} \to \mathbb{R},\ x \mapsto \frac{ax^2 + bx + c}{|dx + e|}$$

is either piecewise convex or piecewise concave and the positivity of f is sufficient for its convexity.

Proof. For any univariate $g(x) = \frac{u(x)}{|v(x)|}$ we have

$$g' = \frac{u'|v| - uv'\mathrm{sgn}(v)}{|v|^2} = \frac{u'}{|v|} - \frac{u}{|v|}\frac{v'}{v} = \frac{u'}{|v|} - g\frac{v'}{v}$$

and

$$\begin{aligned} g'' &= \frac{u''}{|v|} - \frac{u'}{|v|}\frac{v'}{v} - g'\frac{v'}{v} - g\frac{v''}{v} + g\frac{v'}{v}\frac{v'}{v} = \frac{u''}{|v|} - 2g'\frac{v'}{v} - g\frac{v''}{v} \\ &= \frac{u''v^2 - 2u'vv' + 2u(v')^2 - uvv''}{|v|v^2}, \end{aligned}$$

so that we conclude for f:

$$\begin{aligned} f''(x) &= \frac{2a(dx+e)^2 - 2(2ax+b)(dx+e)d + 2(ax^2+bx+c)d^2}{|dx+e|(dx+e)^2} \\ &= 2\frac{ae^2 - bde + cd^2}{|dx+e|^3}. \end{aligned}$$

Depending on the sign of the numerator, $f''(x)$ is either positive or negative for all x and therefore f is convex or concave on both intervals $]-\infty, \frac{-e}{d}[$ and $]\frac{-e}{d}, \infty[$. Rewriting the numerator of $f(x)$ and $f''(x)$ as

$$a\big(x+\tfrac{b}{2a}\big)^2 + c - \tfrac{b^2}{4a} \qquad \text{and} \qquad a\big(e-\tfrac{b}{2a}d\big)^2 + d^2\big(c-\tfrac{b^2}{4a}\big)$$

immediately shows that the positivity of f implies f'' to be positive, too. □

Lemma 1.5 *Using the same notation as in Proposition 1.5, the function*

$$F_T : \psi(\boldsymbol{u}) \mapsto E_2(\psi_{|T}). \tag{1.24}$$

is convex on both half-planes H^+ and H^- defined by the line $L = \overline{\psi(\boldsymbol{w})\psi(\boldsymbol{v})}$.

Proof. We prove this statement by first showing $F_T \circ g$ to be piecewise convex for any line $g : t \mapsto \psi^0(\boldsymbol{u}) + t\psi(\boldsymbol{u})$ with $\psi(\boldsymbol{u}) \neq \boldsymbol{0}$. Substituting $a = \cot\alpha$, $b = \cot\beta$, $c = \cot\gamma$, $\boldsymbol{x} = \psi(\boldsymbol{u})$, $\boldsymbol{y} = \psi(\boldsymbol{v}) - \psi^0(\boldsymbol{u})$, $\boldsymbol{z} = \psi(\boldsymbol{w}) - \psi^0(\boldsymbol{u})$ in (1.21) we obtain

$$\begin{aligned} (F_T \circ g)(t) &= \frac{a\,\|\boldsymbol{y}-\boldsymbol{z}\|^2 + b\,\|\boldsymbol{z}-t\boldsymbol{x}\|^2 + c\,\|\boldsymbol{y}-t\boldsymbol{x}\|^2}{\|(\boldsymbol{z}-t\boldsymbol{x})\times(\boldsymbol{y}-t\boldsymbol{x})\|} \\ &= \frac{t^2(b+c)\|\boldsymbol{x}\|^2 - 2t\langle\, b\boldsymbol{z}+c\boldsymbol{y} \mid \boldsymbol{x}\,\rangle + a\,\|\boldsymbol{y}-\boldsymbol{z}\|^2 + b\,\|\boldsymbol{z}\|^2 + c\,\|\boldsymbol{y}\|^2}{\|\,t\,\boldsymbol{x}\times(\boldsymbol{z}-\boldsymbol{y}) + \boldsymbol{z}\times\boldsymbol{y}\|}. \end{aligned}$$

From (1.11) we know $(b+c)\|\boldsymbol{x}\|^2 > 0$. If $\boldsymbol{x}$ is parallel to L, $\boldsymbol{x} = \mu(\boldsymbol{z}-\boldsymbol{y})$, the denominator of $(F_T \circ g)(t)$ simplifies to a positive constant and $F_T \circ g$ is a convex quadratic function. Otherwise, $\boldsymbol{x}\times(\boldsymbol{z}-\boldsymbol{y}) \neq \boldsymbol{0}$ and Lemma 1.4 proves $F_T \circ g$ to be piecewise convex. Note that the denominator vanishes if $t\boldsymbol{x} = \lambda\boldsymbol{z} + (1-\lambda)\boldsymbol{y}$,

$$\begin{aligned} t\,\boldsymbol{x}\times(\boldsymbol{z}-\boldsymbol{y}) + \boldsymbol{z}\times\boldsymbol{y} &= (\lambda\boldsymbol{z} + (1-\lambda)\boldsymbol{y})\times(\boldsymbol{z}-\boldsymbol{y}) + \boldsymbol{z}\times\boldsymbol{y} \\ &= (-\lambda - (1-\lambda) + 1)\boldsymbol{z}\times\boldsymbol{y} \\ &= \boldsymbol{0}, \end{aligned}$$

which is equivalent to $g(t) \in L$, and we conclude $F_{T|g(\mathbb{R})\cap H^+}$ and $F_{T|g(\mathbb{R})\cap H^-}$ to be convex. □

An example is shown in Figure 1.28 where we chose $\alpha = 90°$, $\beta = \gamma = 45°$, $\psi(\boldsymbol{v}) = \binom{0}{-1}$, $\psi(\boldsymbol{w}) = \binom{0}{1}$. With $\psi(\boldsymbol{u}) = \binom{x}{y}$, F_T in (1.24) becomes

$$F_T\begin{pmatrix} x \\ y \end{pmatrix} = \frac{x^2+y^2+1}{|x|}$$

with two minima at $\binom{1}{0}$ and $\binom{-1}{0}$.

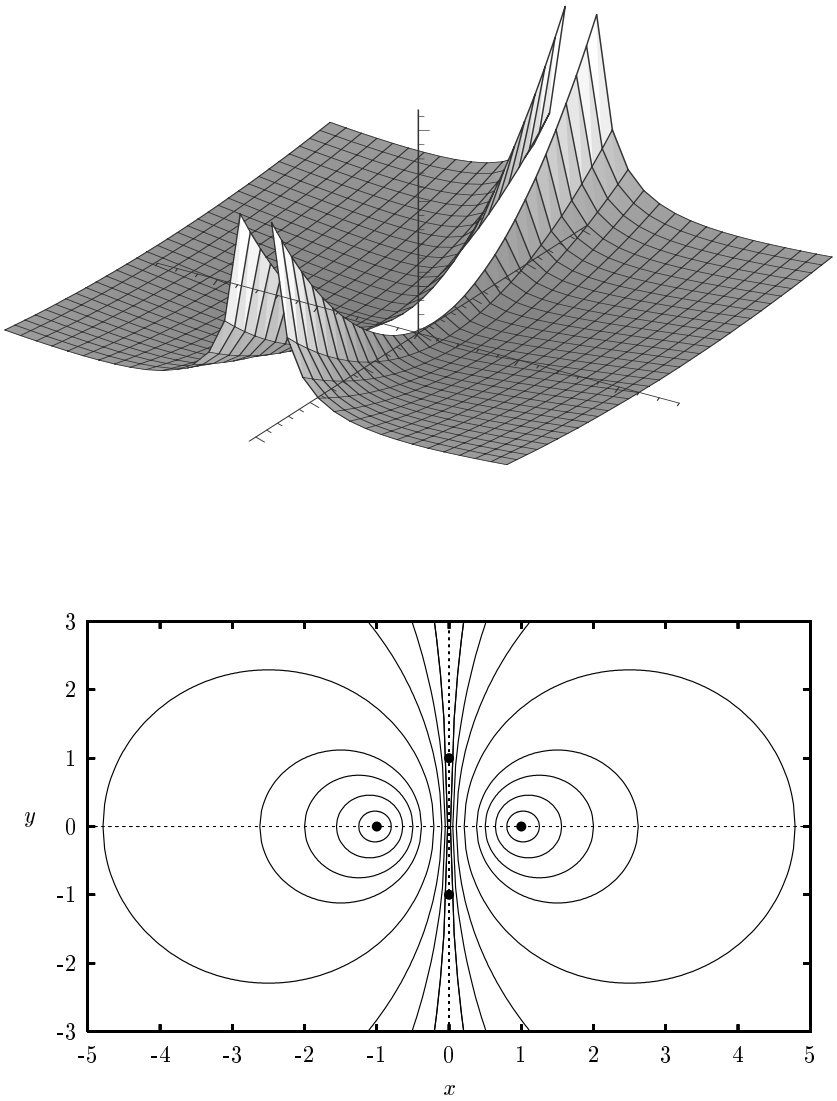
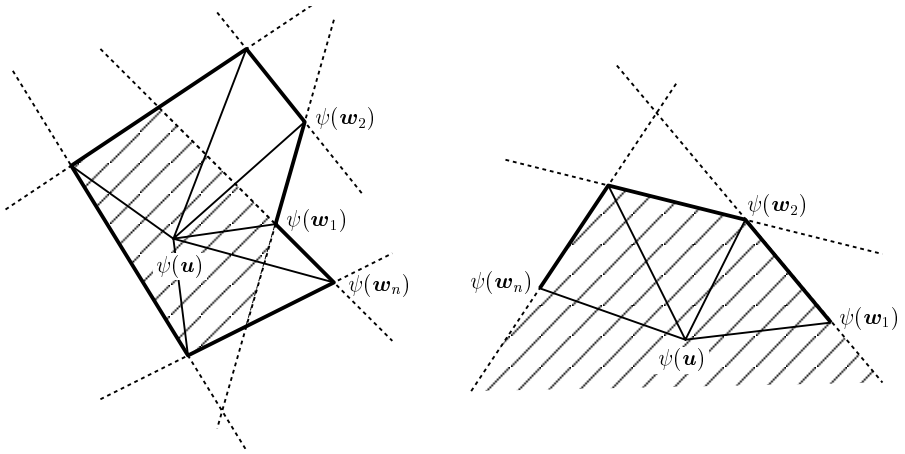


Figure 1.28: The graph of the function $f(x, y) = (x^2 + y^2 + 1)/|x|$ (top) and the isolines for $f(x, y) \in \{2.05, 2.2, 2.5, 3, 5, 10, 20, 50\}$ (bottom).



(a) (b)

Figure 1.29: Let $\boldsymbol{u} \in V_I$ (a) or $\boldsymbol{u} \in V_B$ (b). Then the triangulation $\psi(\mathcal{T})$ is locally without self-intersections as long as $\psi(\boldsymbol{u})$ lies within the dashed region, which is the intersection of all half-planes left to the lines connecting the neighbour vertices $\psi(\boldsymbol{w}_1), \ldots, \psi(\boldsymbol{w}_n)$ of $\psi(\boldsymbol{u})$.

Theorem 1.12 *Let $\boldsymbol{u} \in V$. Then the local minimization problem*

$$\min_{\psi(\boldsymbol{u})} E_M(\psi)$$

is convex on the space of bijective mappings $\psi \in S_2(\mathcal{T})$.

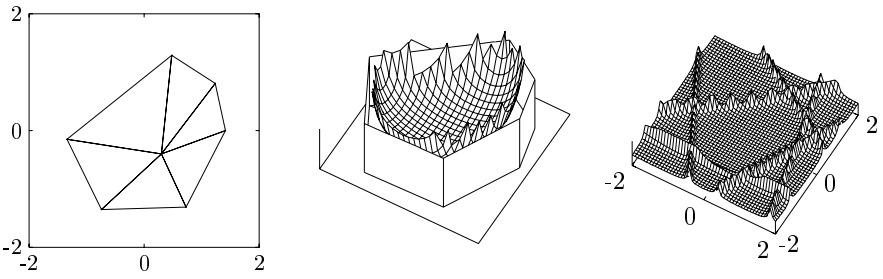
Proof. Let $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_n$ be the neighbour vertices of $\boldsymbol{u}$ ordered counterclockwise and $T_i = [\boldsymbol{u}, \boldsymbol{w}_i, \boldsymbol{w}_{i+1}]$, $i \in I$ be the triangles that contain $\boldsymbol{u}$ with $I = \{1, \ldots, n\}$ if $\boldsymbol{u} \in V_I$ and $I = \{1, \ldots, n-1\}$ if $\boldsymbol{u} \in V_B$. Since $E_2(\psi_{|T})$ does not depend on $\psi(\boldsymbol{u})$ for $T \in \mathcal{T}$ with $\boldsymbol{u} \notin T$, we have

$$\min_{\psi(\boldsymbol{u})} E_M(\psi) \overset{(1.23)}{=} \min_{\psi(\boldsymbol{u})} \sum_{T \in \mathcal{T}} E_2(\psi_{|T}) = \min_{\psi(\boldsymbol{u})} \sum_{i \in I} E_2(\psi_{|T_i}) \overset{(1.24)}{=} \min_{\psi(\boldsymbol{u})} E_{\boldsymbol{u}}(\psi(\boldsymbol{u})),$$

where $E_{\boldsymbol{u}} = \sum_{i \in I} F_{T_i}$ denotes the *local* MIPS energy at $\psi(\boldsymbol{u})$. Now let H_i^+ and H_i^- be the half-planes to the right and the left of the line $\overline{\psi(\boldsymbol{w}_i)\psi(\boldsymbol{w}_{i+1})}$. According to Lemma 1.5, $E_{\boldsymbol{u}}$ is convex on any region $R \in \mathcal{R}$ with

$$\mathcal{R} = \left\{ \bigcap\nolimits_{i \in I} H_i : H_i \in \{H_i^+, H_i^-\} \right\}.$$

The statement follows because ψ is bijective if and only if $\psi(\boldsymbol{u}) \in \bigcap_{i \in I} H^-$ (see Figure 1.29). □



(a) (b) (c)

Figure 1.30: A triangulation $\psi(\mathcal{T})$ (a) and the graph of the local MIPS energy, restricted to the region that guarantees bijectivity (b) and for all regions (c).

Note that this statement is valid for all vertices V and not restricted to the interior ones V_I which enables us to optimize the parameter values of the boundary vertices, too. Figure 1.30 shows an example of the local MIPS energy $E_{\boldsymbol{u}}$.

In order to find the unique minimum of $E_{\boldsymbol{u}}$ we apply *Newton's method* [107],

$$\psi(\boldsymbol{u})^{(k+1)} = \psi(\boldsymbol{u})^{(k)} - \mathbf{H}_{E_{\boldsymbol{u}}}^{-1}(\psi(\boldsymbol{u})^{(k)})\nabla E_{\boldsymbol{u}}(\psi(\boldsymbol{u})^{(k)}),$$

which is known to converge quadratically and because $E_{\boldsymbol{u}}$ is convex this method converges for any initial guess $\psi(\boldsymbol{u})^{(0)}$. Moreover, since $E_{\boldsymbol{u}}$ is a sum of functions F_{T_i}, the gradient $\nabla E_{\boldsymbol{u}}$ as well as the Hessian matrix $\mathbf{H}_{E_{\boldsymbol{u}}}$ can be computed efficiently using the following formulas.

Proposition 1.6 *The function value, the gradient, and the Hessian matrix of* F_T *at* $\psi(\boldsymbol{u}) = \mathbf{0}$ *are*

$$F_T(\mathbf{0}) = m_a(d_x^2 + d_y^2) + m_b(w_x^2 + w_y^2) + m_c(v_x^2 + v_y^2) = e,$$

$$\nabla F_T(\mathbf{0}) = \big(-2(m_b w_x + m_c v_x) + m_y e, -2(m_b w_y + m_c v_y) - m_x e\big) = (e_x, e_y),$$

$$\mathbf{H}_{F_T}(\mathbf{0}) = \begin{pmatrix} 2(m_b + m_c + m_y e_x) & m_y e_y - m_x e_x \\ m_y e_y - m_x e_x & 2(m_b + m_c - m_x e_y) \end{pmatrix},$$

where $\binom{v_x}{v_y} = \psi(\boldsymbol{v})$, $\binom{w_x}{w_y} = \psi(\boldsymbol{w})$, $d_x = v_x - w_x$, $d_y = v_y - w_y$, $m = w_x v_y - w_y v_x$, $m_a = \frac{\cot\alpha}{|m|}$, $m_b = \frac{\cot\beta}{|m|}$, $m_c = \frac{\cot\gamma}{|m|}$, $m_x = \frac{d_x}{m}$, $m_y = \frac{d_y}{m}$.

Proof. In analogy to the discussion on the derivatives of a univariate function $g(x) = \frac{u(x)}{|v(x)|}$ in Lemma 1.4 we have for a multivariate $g(\boldsymbol{x}) = \frac{u(\boldsymbol{x})}{|v(\boldsymbol{x})|}$

$$\nabla g = \frac{\nabla u}{|v|} - g\frac{\nabla v}{v} \qquad \text{and} \qquad \nabla\nabla^t g = \mathbf{H}_g = \frac{\mathbf{H}_u}{|v|} - \frac{\nabla^t v \nabla g + \nabla^t g \nabla v}{v} - g\frac{\mathbf{H}_v}{v}.$$

Considering the numerator and the denominator of F_T we find with $\psi(\boldsymbol{u}) = \binom{u_x}{u_y}$,

$$\begin{aligned} u\tbinom{u_x}{u_y} &= \cot\alpha\big[(v_x-w_x)^2+(v_y-w_y)^2\big] + \\ &\quad \cot\beta\big[(w_x-u_x)^2+(w_y-u_y)^2\big] + \\ &\quad \cot\gamma\big[(v_x-u_x)^2+(v_y-u_y)^2\big], \end{aligned}$$

$$\begin{aligned} \nabla u\tbinom{u_x}{u_y} &= \big(-2\cot\beta(w_x-u_x)-2\cot\gamma(v_x-u_x), \\ &\quad -2\cot\beta(w_y-u_y)-2\cot\gamma(v_y-u_y)\big), \end{aligned}$$

$$\mathbf{H}_u\tbinom{u_x}{u_y} = \begin{pmatrix} 2\cot\beta+2\cot\gamma & 0 \\ 0 & 2\cot\beta+2\cot\gamma \end{pmatrix}$$

and

$$v\tbinom{u_x}{u_y} = w_xv_y - w_yv_x - u_x(v_y-w_y) + u_y(v_x-w_x),$$

$$\nabla v\tbinom{u_x}{u_y} = (w_y-v_y, v_x-w_x),$$

$$\mathbf{H}_v\tbinom{u_x}{u_y} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix},$$

and for $\psi(\boldsymbol{u}) = \mathbf{0}$,

$$u(\mathbf{0}) = |m|\big(m_a(d_x^2+d_y^2)+m_b(w_x^2+w_y^2)+m_c(v_x^2+v_y^2)\big)$$

$$\nabla u(\mathbf{0}) = |m|\big(-2(m_bw_x+m_cv_x), -2(m_bw_y+m_cv_y)\big)$$

$$\mathbf{H}_u(\mathbf{0}) = |m|\begin{pmatrix} 2(m_b+m_c) & 0 \\ 0 & 2(m_b+m_c) \end{pmatrix}$$

and

$$v(\mathbf{0}) = m, \qquad \nabla v(\mathbf{0}) = (-d_y, d_x), \qquad \mathbf{H}_v(\mathbf{0}) = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix},$$

leading directly to the formulas stated above. □

Note that it suffices to evaluate $\nabla E_{\boldsymbol{u}}$ and $\mathbf{H}_{E_{\boldsymbol{u}}}$ at the origin if we keep shifting the neighbour vertices $\psi(\boldsymbol{w}_i)$, $i \in I$ instead of $\psi(\boldsymbol{u})$,

$$\psi(\boldsymbol{w}_i)^{(0)} = \psi(\boldsymbol{w}_i) - \psi(\boldsymbol{u}),$$

$$\psi(\boldsymbol{w}_i)^{(k+1)} = \psi(\boldsymbol{w}_i)^{(k)} + \mathbf{H}_{E_{\boldsymbol{u}}}^{-1}(\mathbf{0})\nabla E_{\boldsymbol{u}}(\mathbf{0}).$$

In all the examples tested we found this process to converge numerically after just a few iterations ($k \approx 5$), thus confirming the theoretical convergence rate.

Let us now summarize why the procedure **optimize** is suited for determining MIPS. Firstly, Theorem 1.12 assures the local minimization problem $(*)$ to have a unique solution which can be determined efficiently with the formulas given in Proposition 1.6. Secondly, Theorem 1.12 also guarantees that once we start with a bijective initial parameterization ψ, we are not going to lose the bijectivity by successively performing the local optimization steps. Note that such an initial parameterization can always be found by using one of the linear methods discussed in Section 1.2. Thirdly, the algorithm always terminates since E_M as a positive function is bounded from below and reduced in each local minimization step.

It remains to be seen whether this approach finds the global minimum of E_M or only a local one. First of all, let us recall that E_2 is invariant with respect to translations, rotations, reflections, and scalings and so is E_M. Thus any minimum of E_M is only unique up to these transformations. However, by fixing the parameter values of any two vertices $\boldsymbol{v}_1, \boldsymbol{v}_2 \in V$ we eliminate the diversity due to translations, rotations, and scalings. The remaining two triangulations $\psi(\mathcal{T})$ and $\psi'(\mathcal{T})$ which correspond to the same minimum differ from each other by reflection along the line $\overline{\psi(\boldsymbol{v}_1)\psi(\boldsymbol{v}_2)}$ and by considering only those parameterizations that keep the orientation of the triangles we exclude one of them. On the subspace of parameterizations defined by these restrictions any minimum of E_M is unique.

Motivated by the local convexity as proved in Theorem 1.12 we may expect the MIPS energy to be globally convex as well. We could then be sure that one and only one minimum exists at which the procedure **optimize** necessarily arrives. But a simple example dashes hopes.

Consider the triangulation $\mathcal{T} = \{[\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3], [\boldsymbol{v}_4, \boldsymbol{v}_3, \boldsymbol{v}_2]\}$ with $\boldsymbol{v}_1 = \binom{1}{-1}$, $\boldsymbol{v}_2 = \binom{2}{0}$, $\boldsymbol{v}_3 = \binom{0}{0}$, $\boldsymbol{v}_4 = \binom{0}{2}$ and the two parameterizations $\psi^0, \psi \in S_2(\mathcal{T})$ defined by

$$\psi^0(\boldsymbol{v}_1) = \tbinom{-1}{-2}, \quad \psi^0(\boldsymbol{v}_2) = \tbinom{1}{0}, \quad \psi^0(\boldsymbol{v}_3) = \tbinom{-1}{2}, \quad \psi^0(\boldsymbol{v}_4) = \tbinom{1}{1}$$

and

$$\psi(\boldsymbol{v}_1) = \tbinom{1}{-2}, \quad \psi(\boldsymbol{v}_2) = \tbinom{2}{1}, \quad \psi(\boldsymbol{v}_3) = \tbinom{1}{0}, \quad \psi(\boldsymbol{v}_4) = \tbinom{1}{2}.$$

Along the line $g : t \mapsto \psi^0 + t\psi$ we have

$$(E_M \circ g)(t) = \frac{7t^2 + 16t + 12}{|t+2|^2} + \frac{6t^2 - 4t + 13}{|2t^2 + t + 2|}, \tag{1.25}$$

which is clearly not convex (see Figure 1.31).

However, this does not contradict the existence of a unique global minimum. In all the examples tested the optimization method **optimize** yielded the same minimum for various initial values, affirming the assumption that E_M has indeed no local minima, but a proof needs yet to be found.

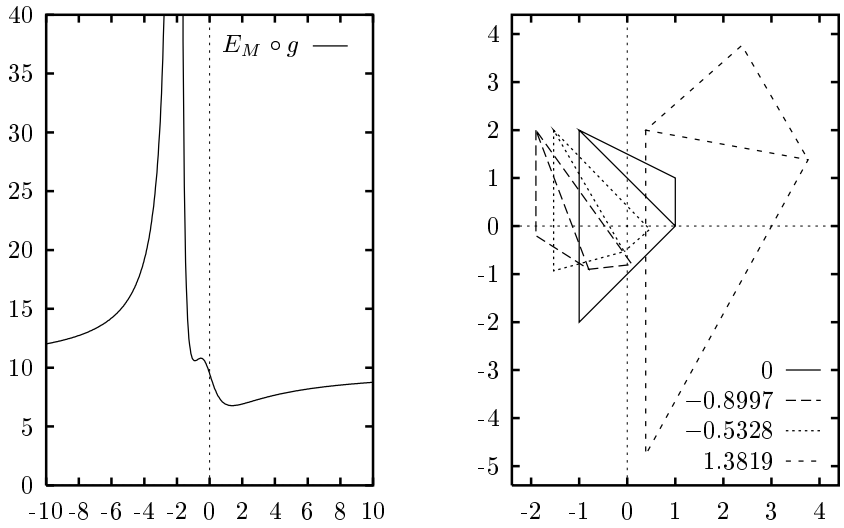
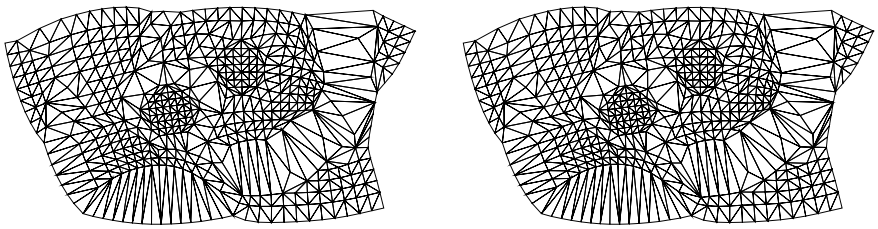


Figure 1.31: Graph of the function in (1.25) and the triangulations $(\psi^0 + t\psi)(\mathcal{T})$ for $t = 0$, at the local minima $t = -0.8997$, $t = 1.3819$, and at the local maximum $t = -0.5328$.

The main drawback of this minimization approach is its very slow convergence. We have already seen Gauss-Seidel iterations to be most inapt for solving the linear problems in Section 1.2.6 and the performance is even worse when applied to the MIPS energy. One way of accelerating the process by exploiting hierarchies will be discussed in the next section, another one is inspired by the following relation between the MIPS and the Dirichlet energy (compare Equation 1.22):

$$E_M(\psi) = 2 \sum_{T \in \mathcal{T}} \frac{E_D(\psi_{|T})}{\text{area}(\psi(T))}.$$

If we assume $\text{area}(\psi(T))$ to be approximately the same for all $T \in \mathcal{T}$ the parameterizations obtained by minimizing both energies should be quite similar. And in fact, if we take the most isometric parameterization of the triangulation in Figure 1.17.a, fix the parameter values of the boundary vertices, and minimize the Dirichlet energy with respect to the interior parameter points, the result is almost indistinguishable from the most isometric parameterization (see Figure 1.32). This observation suggests to use the nonlinear and costly minimization of the MIPS energy for determining the parameter values at the boundary only while applying one of the linear methods to the interior vertices.



(a) (b)

Figure 1.32: The most isometric parameterization (a) of the triangulation in Figure 1.17.a has a MIPS energy of 1 735.98 while the harmonic parameterization with the same boundary (b) has MIPS energy 1 737.89.

1.3.3 Hierarchical Optimization

As explained in the previous section, finding the most isometric parameterization especially of large triangulations can be quite tedious due to the slow convergence rate of the minimization algorithm. Nevertheless, if the triangulation consists of a few triangles only or if good estimates of the optimal parameter values $\psi(\boldsymbol{v})$ are known, it does perform quite well. We will now present a method that utilizes this observation and accelerates the parameterization process considerably. It exploits a hierarchical representation of the triangulation and is thus comparable to multigrid methods [15] that are frequently used to efficiently solve linear systems.

The basic idea of this hierarchical optimization approach is to generate a hierarchy $\mathcal{T} = \mathcal{T}^n, \ldots, \mathcal{T}^0$ of triangulations and to determine the parameterization ϕ^0 of the coarsest level $\mathcal{T}^0$ first. Since this involves only a very limited number of triangles, it poses no problem to the minimization method. Furthermore, ϕ^0 is a good approximation to the parameterization ϕ^1 of the next level $\mathcal{T}^1$ so that computing ϕ^1 is not too costly either. Iterating this principle finally gives the parameterization $\phi^n = \phi$ of $\mathcal{T}$.

The process of creating a hierarchical representation of a triangulation is also known as *mesh decimation* or *mesh simplification* and surveys of the large variety of techniques can be found in [64, 83]. The major class of these algorithms modifies a given triangulation by iteratively applying a topological operator such as *vertex removal* or *edge collapse* that decreases the number of triangles by two in each step without changing the topology of the triangulation's surface. The algorithms differ mainly by the strategy that is chosen to decide in which order the simplification steps are to be carried out.

We decided to use the simplest type of an edge collapse operation, namely the *half edge collapse*, which collapses a vertex $\boldsymbol{v}$ into a neighboring vertex $\boldsymbol{w}_1$ (see Figure 1.33). We can further apply these simplification operations in random order since our hierarchical parameterization method does not depend on the accuracy and the quality of the triangulations at coarser levels. In detail, we proceed as follows.

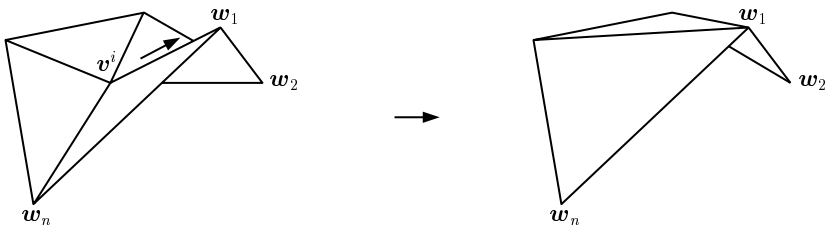


Figure 1.33: Half edge collapse of $\boldsymbol{v}^i$ into $\boldsymbol{w}_1$.

Given a triangulation $\mathcal{T}^i$ we randomly choose a vertex $\boldsymbol{v}^i \in V(\mathcal{T}^i)$ with n neighbours $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_n$ to be removed by collapsing it into $\boldsymbol{w}_1$ as shown in Figure 1.33 in order to create the coarser triangulation $\mathcal{T}^{i-1}$. This half edge collapse removes not only the vertex $\boldsymbol{v}^i$, so that $V(\mathcal{T}^{i-1}) = V(\mathcal{T}^i) \setminus \boldsymbol{v}^i$, but also the two triangles $[\boldsymbol{v}^i, \boldsymbol{w}_1, \boldsymbol{w}_2]$ and $[\boldsymbol{v}^i, \boldsymbol{w}_1, \boldsymbol{w}_n]$. It also replaces the $n-2$ triangles $[\boldsymbol{v}^i, \boldsymbol{w}_j, \boldsymbol{w}_{j+1}]$, $j = 2, \ldots, n-1$, by $[\boldsymbol{w}_1, \boldsymbol{w}_j, \boldsymbol{w}_{j+1}]$, so that $\mathcal{T}^{i+1}$ can be written as

$$\mathcal{T}^{i-1} = \mathcal{T} \setminus \bigcup_{j=1}^{n} [\boldsymbol{v}^i, \boldsymbol{w}_j, \boldsymbol{w}_{j+1}] \cup \bigcup_{j=2}^{n-1} [\boldsymbol{w}_1, \boldsymbol{w}_j, \boldsymbol{w}_{j+1}],$$

where $\boldsymbol{w}_{n+1} = \boldsymbol{w}_1$ by definition. In order to be able to reverse the half edge collapse later, we also determine the *parametric coordinates* of $\boldsymbol{v}^i$. To this end, we compute the exponential mapping of the local configuration around $\boldsymbol{v}^i$, yielding the planar points $\boldsymbol{p}$ and $\boldsymbol{q}_1, \ldots, \boldsymbol{q}_n$ as described in Section 1.2.4 on page 22 and consider the corresponding half edge collapse in the plane (see Figure 1.34). We then locate the index j for which $\boldsymbol{p} \in [\boldsymbol{q}_1, \boldsymbol{q}_j, \boldsymbol{q}_{j+1}]$ and determine the barycentric coordinates of $\boldsymbol{p}$ with respect to that triangle,

$$\boldsymbol{p} = \tau_0 \boldsymbol{q}_1 + \tau_1 \boldsymbol{q}_j + \tau_2 \boldsymbol{q}_{j+1},$$

with $\tau_0 + \tau_1 + \tau_2 = 1$. After having computed the inverse parameterization ψ^{i-1} of $\mathcal{T}^{i-1}$ which is uniquely determined by the parameter values $\psi^{i-1}(\boldsymbol{v})$, $\boldsymbol{v} \in V(\mathcal{T}^{i-1})$, we set

$$\psi^i(\boldsymbol{v}) = \psi^{i-1}(\boldsymbol{v}), \qquad \boldsymbol{v} \in V(\mathcal{T}^{i-1})$$

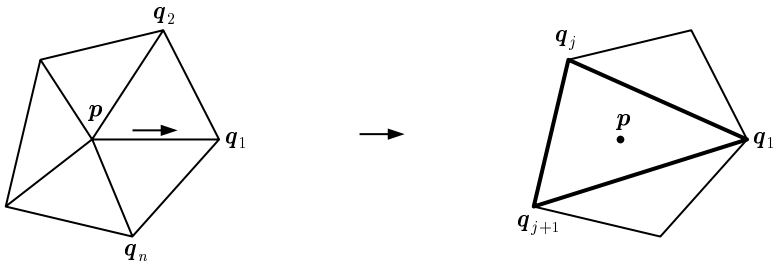


Figure 1.34: Determining the parametric coordinates of a half edge collapse.

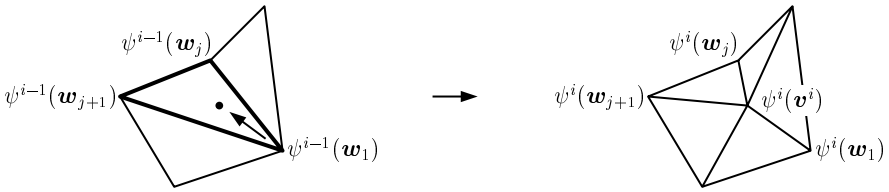


Figure 1.35: Reversing a half edge collapse in the parameter domain.

and

$$\psi^i(\boldsymbol{v}^i) = \tau_0\psi^i(\boldsymbol{w}_1) + \tau_1\psi^i(\boldsymbol{w}_j) + \tau_2\psi^i(\boldsymbol{w}_{j+1})$$

as shown in Figure 1.35, thereby defining an initial ψ^i which is further optimized by the minimization algorithm.

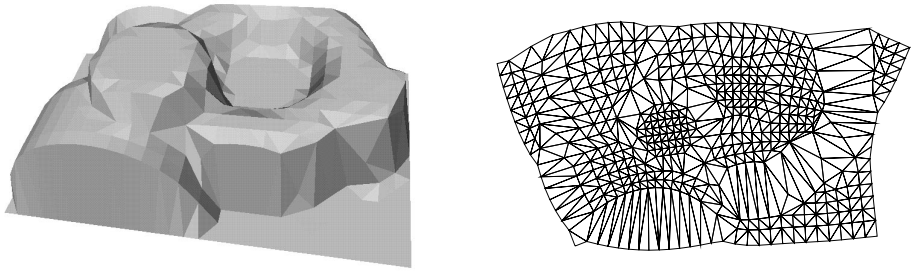
In order to speed up this hierarchical approach it is useful to collect several successive half edge collapses and consider a coarser hierarchy of triangulations. Such a coarser hierarchy is determined as follows. Given the triangulation $\mathcal{T}^i$ at level i, we first mark all vertices as "selectable". Then we successively choose selectable vertices for a half edge collapse and mark all neighbour vertices as "non selectable" until no selectable vertices are left. The resulting triangulation is finally defined as the triangulation $\mathcal{T}^{i-1}$ at the next level $i-1$. In this way, the number of triangles is reduced by approximately 25%, $|\mathcal{T}^i| \approx \frac{4}{3}|\mathcal{T}^{i-1}|$.

As an example we have parameterized the triangulation from Figure 1.17.a with this hierarchical approach. It was decomposed into 10 coarse hierarchy levels $\mathcal{T} = \mathcal{T}^9, \ldots, \mathcal{T}^0$. Figure 1.36 shows the triangulations of level 9 (a) with $|\mathcal{T}^9| = 476$, level 6 (b) with $|\mathcal{T}^6| = 198$, level 3 (c) with $|\mathcal{T}^3| = 76$, and level 0 (d) with $|\mathcal{T}^0| = 32$ and the corresponding most isometric parameterizations at those levels. Overall, the hierarchical sequence and the parameterizations of the triangulations on all levels were computed in 20 seconds while it took the non-hierarchical optimization method more than 20 minutes to find the parameterization of $\mathcal{T}$, taking the shape preserving parameterization as start value.

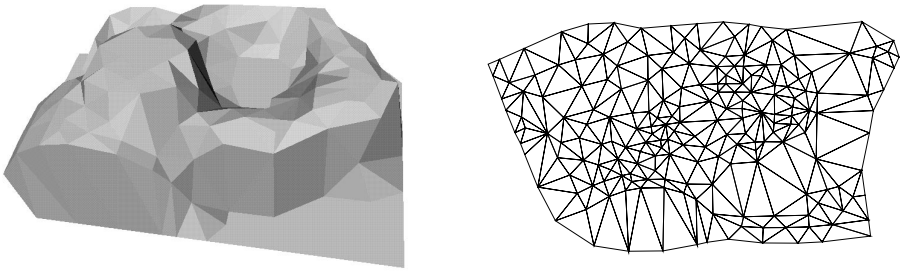
1.4 Arbitrary Topology

So far we have only considered the parameterization of simple triangulations, but what has to be done if the triangulation is more complicated, e.g. homeomorphic to a sphere or a torus? A triangulation $\mathcal{T}$ with arbitrary topology can no longer be parameterized over a planar parameter domain as in the previous sections and we rather need a spatial domain that is topologically equivalent to $\Omega_{\mathcal{T}}$.

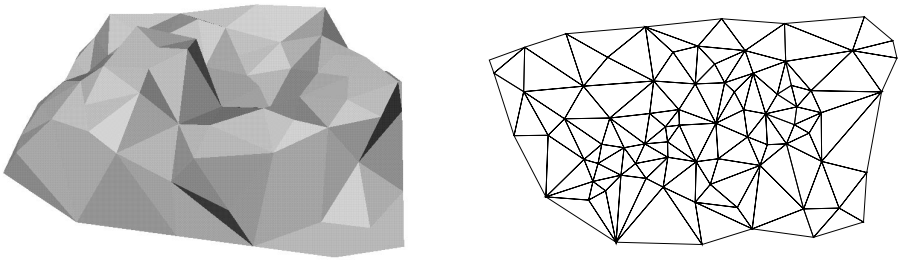
One way of generating a suitable spatial parameter domain is by one of the various mesh decimation or mesh simplification algorithms mentioned in Section 1.3.3. All these algorithms result in a triangulation $\mathcal{T}^0$ that approximates the shape of the initial triangulation $\mathcal{T}$ in some way, $\Omega_{\mathcal{T}^0} \approx \Omega_{\mathcal{T}}$. An example is



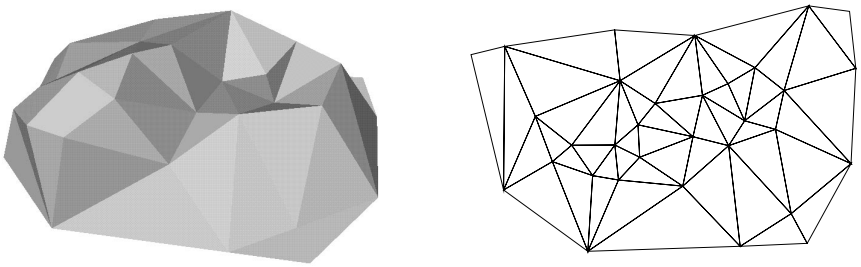
(a)



(b)

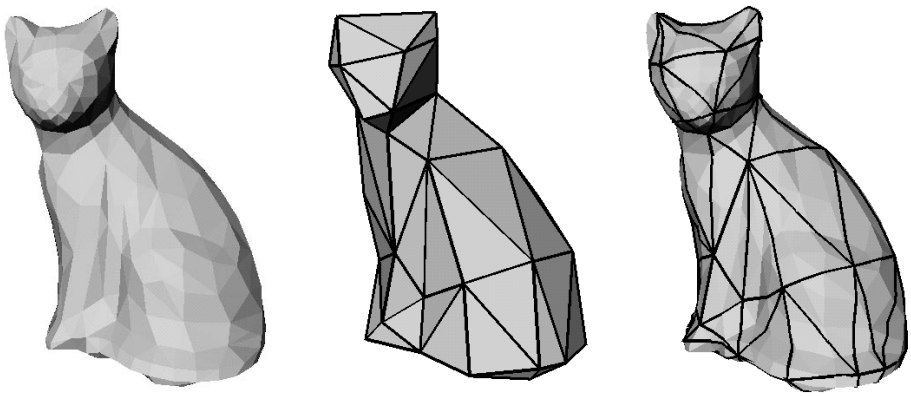


(c)



(d)

Figure 1.36: Computing the most isometric parameterization of the data set in Figure 1.17.a by using hierarchical optimization.



(a) (b) (c)

Figure 1.37: A triangulation with sphere-like topology (a), its simplified triangulation (b), and its surface triangulation (c).

shown in Figure 1.37 where a triangulation with 768 triangles (a) was simplified to one with 100 triangles (b) using the method of Campagna [16].

We solve the parameterization problem for triangulations with arbitrary topology by partitioning the triangulation $\mathcal{T}$ into $M_0 = |\mathcal{T}^0|$ disjoint subsets $\mathcal{S}^1, \mathcal{S}^2, \ldots, \mathcal{S}^{M_0}$ such that each $\mathcal{S}^i$ is a simple triangulation and corresponds to one of the triangles $T_i \in \mathcal{T}^0$ in the spatial parameter domain. We call each $\mathcal{S}^i$ a *surface triangle* and the collection $\mathcal{S} = \{\mathcal{S}^i\}_{i=1,\ldots,M_0}$ a *surface triangulation*. Figure 1.37.c shows an example.

We can then use any of the simple parameterization methods for constructing a parameterization ϕ_i of $\mathcal{S}^i$ over T_i and combine them in the end to define the parameterization ϕ by letting

$$\phi_{|T_i} = \phi_i, \qquad i = 1, \ldots, M_0.$$

In order to assure ϕ to be a homeomorphism we require the individual parameterizations ϕ_i to be continuous across the edges of $\mathcal{T}^0$. This is achieved by parameterizing the edges first and then fixing these parameter values before parameterizing the interior of each surface triangle.

In practice the whole procedure is performed in three steps. Firstly, ϕ is defined for the vertices V^0 of $\mathcal{T}^0$. If $\mathcal{T}^0$ was constructed by mesh simplification we always have $V^0 \subset V$ and simply set $\phi(\boldsymbol{v}) = \boldsymbol{v}$, $\boldsymbol{v} \in V^0$. If the vertex correspondences are chosen by some other method, e.g. interactively, $\phi(\boldsymbol{v})$ does not necessarily need to be a vertex of $\mathcal{T}$. In this case we refine $\mathcal{T}$ by an edge or a one-to-three triangle split to a triangle mesh $\mathcal{T}'$ such that $\phi(\boldsymbol{v})$ is a vertex of $\mathcal{T}'$ (see Figure 1.38). Note that this does not change the geometry of the given triangulation, i.e. $\Omega_{\mathcal{T}} = \Omega_{\mathcal{T}'}$, but only its connectivity. We therefore assume without loss of generality $\phi(V^0) \subset V$.

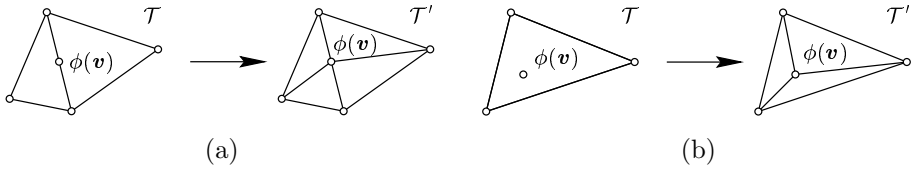


Figure 1.38: Refining $\mathcal{T}$ by an edge split (a) or a one-to-three triangle split (b).

Secondly, ϕ is extended to the edges E^0 of $\mathcal{T}^0$. This is done by finding for each edge $e = [\boldsymbol{v}, \boldsymbol{w}] \in E^0$ the shortest path $\bar{e}$ on $\Omega_{\mathcal{T}}$ between $\phi(\boldsymbol{v})$ and $\phi(\boldsymbol{w})$, e.g. with one of the algorithms in [20, 79, 77]. We refer to this shortest path as a *surface edge*. Since the shortest path across a triangle is a straight line, shortest paths on triangulations are polygonal curves with the vertices lying on the edges of the triangulation. Thus we may write $\bar{e}$ as a set of line segments

$$\bar{e} = [\boldsymbol{u}_1, \boldsymbol{u}_2] \cup [\boldsymbol{u}_2, \boldsymbol{u}_3] \cup \cdots \cup [\boldsymbol{u}_{r-1}, \boldsymbol{u}_r],$$

where $\boldsymbol{u}_1 = \phi(\boldsymbol{v})$, $\boldsymbol{u}_r = \phi(\boldsymbol{w})$ and the $\boldsymbol{u}_i$ are located on the edges of $\mathcal{T}$. Since we can always refine $\mathcal{T}$ to a triangle mesh $\mathcal{T}'$ as shown in Figure 1.39, we assume without loss of generality $\boldsymbol{u}_i \in V$ and $[\boldsymbol{u}_i, \boldsymbol{u}_{i+1}] \in E$. We can now parameterize $\bar{e}$ over e by a standard line parameterization technique, for example, chord length parameterization (see Section 1.2.1).

Thirdly, we further extend ϕ to the triangles of $\mathcal{T}^0$. For each triangle $T_i = [\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}] \in \mathcal{T}^0$ we let the surface triangle $\mathcal{S}^i$ be that region of $\mathcal{T}$ which is bounded by the three surface edges $\phi([\boldsymbol{u}, \boldsymbol{v}])$, $\phi([\boldsymbol{v}, \boldsymbol{w}])$, $\phi([\boldsymbol{w}, \boldsymbol{u}])$. As we have already defined the parameter points of the boundary vertices of $\mathcal{S}^i$ in the previous step we can now apply shape preserving parameterization (see Section 1.2.4), for example, to determine the parameter points of the interior vertices. In this way we obtain for each $i = 1, \ldots, M_0$ the individual parameterization $\phi_i : T_i \to \Omega_{\mathcal{S}^i}$ which are then combined to give $\phi : \Omega_{\mathcal{T}^0} \to \Omega_{\mathcal{T}}$. Note that the use of shortest paths as surface edges and Corollary 1.3 always guarantee each ϕ_i and thus ϕ to be bijective.

Though this parameterization method applies to triangulations of arbitrary topology, we simply illustrate with the triangulation $\mathcal{T}$ in Figure 1.40.a which is homeomorphic to a sphere. We first computed a simplified triangulation $\mathcal{T}^0$ by mesh decimation as described in [16], shown in Figure 1.40.b. For the 171 edges in $\mathcal{T}^0$ we then computed the corresponding surface edges in $\Omega_{\mathcal{T}}$ as shown in Figure 1.41.a and parameterized them using chord length parameterization.

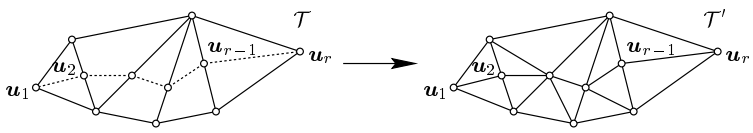
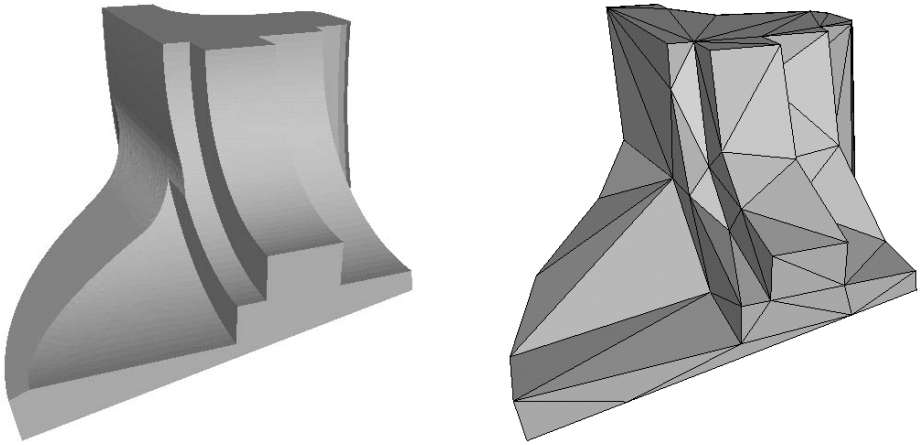
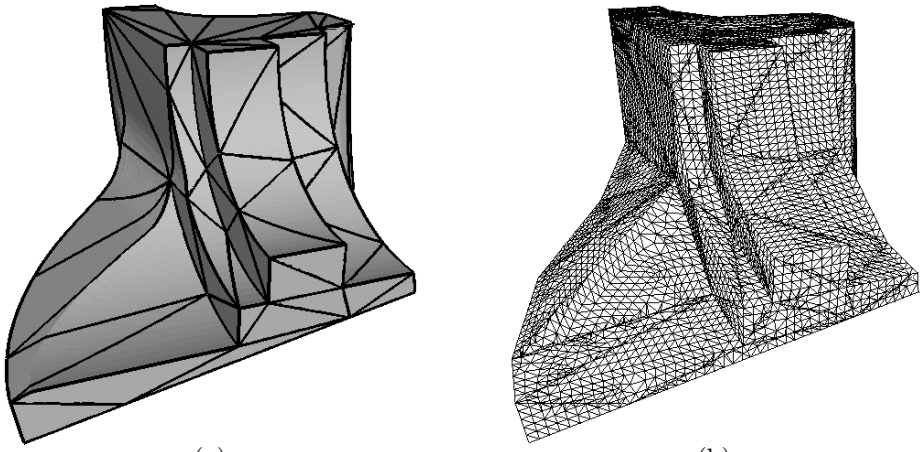


Figure 1.39: Refining $\mathcal{T}$ in order to embed a surface edge $\bar{e}$.



(a) (b)

Figure 1.40: Triangulation of a fandisk with 12 946 triangles (a) and simplified triangulation $\mathcal{T}^0$ with 114 triangles (b).



(a) (b)

Figure 1.41: Surface edges on $\Omega_{\mathcal{T}}$ (a) and parameterization of $\Omega_{\mathcal{T}}$ over $\Omega_{\mathcal{T}^0}$ (b).

For each surface triangle $\mathcal{S}^i$ of $\mathcal{T}$ we then computed the shape preserving parameterization over the corresponding triangle $T_i \in \mathcal{T}^0$, the result being shown in Figure 1.41.b.

Part II

Applications

Es gibt keine Ordnung der Dinge a priori.
Ludwig Wittgenstein (1889–1951)

Chapter 2

Triangulating Point Clouds

The problem of reconstructing surfaces from a set of scattered data points arises in many practical situations. In this chapter we shall focus on the task of triangulating such point clouds, while the reconstruction of smooth surfaces from scattered data will be discussed in Chapter 4. The problem we consider here can be stated as follows: given a set $V = \{\boldsymbol{v}_i\}_i$ of data points $\boldsymbol{v}_i \in \mathbb{R}^3$, find a triangulation $\mathcal{T}$ with these data points as vertices, $V(\mathcal{T}) = V$.

Several methods for solving this problem have been developed in recent years, based on different ideas. We give a brief survey of the most important work in Section 2.1 but also refer to [98] as a detailed survey paper. In Section 2.2 we discuss a recently presented technique for triangulating unorganized points that is based on parameterizing the data points [47]. This method is limited to reconstructing surfaces that are homeomorphic to a disc but in Section 2.2.3 we show how it can be extended to handle spherical topology as well. In Section 2.3 we finally present an algorithm for optimizing the shape of a triangulation without changing the position or number of vertices but only by modifying the way in which they are connected by triangles.

2.1 Related Work

A common approach is to start with the *Delaunay tetrahedrization* [106] of the data points. A tetrahedrization of a set V of three-dimensional points is a decomposition of the convex hull of V into tetrahedra such that the vertices of all tetrahedra are the points in V. It is called a Delaunay tetrahedrization if the circumsphere of each tetrahedron does not contain points in V. In a second step certain tetrahedra are removed from the tetrahedrization and finally the boundary triangles of the remaining tetrahedra are taken as the reconstructed surface. The algorithms in [14, 36, 135, 75] follow this idea and differ by the strategy chosen for removing the tetrahedra.

The methods in [116, 5] are similar but use properties of the *Voronoi diagram*. The Voronoi diagram of a point set V is a partition of the space into

regions of nearest neighbourhood. For each point in V the corresponding region in the Voronoi diagram consists of all points in space that are closer to this point than to any other point in V. The Voronoi diagram is dual to the Delaunay tetrahedrization in the sense that each vertex in the Voronoi diagram corresponds to the center of a tetrahedron in the Delaunay tetrahedrization and an edge in the Voronoi diagram corresponds to the triangle which the tetrahedra dual to the vertices connected by the edge have in common.

Incremental algorithms for triangulating point clouds were presented in [14, 97, 12, 54]. They start with a single edge or triangle connecting two or three of the given data points and successively add more points by generating additional edges and triangles. The process stops when all data points have been connected.

Another class of methods [65, 66, 7, 4] does not create a triangulation that interpolates the points but only approximates the data set. The common strategy of these methods is to first decompose the space into either cubic or tetrahedral cells. In a second step those cells that are traversed by the surface, especially those containing at least one data point, are determined and finally a surface is extracted from the selected cells.

2.2 Meshless Parameterization

A totally different approach for triangulating three-dimensional point clouds was proposed by Floater and Reimers in [47]. They observed that the usual linear parameterization methods (see Section 1.2) do not require the data points to be organized in a globally consistent triangulation. The only information needed for setting up the linear system in the case of harmonic maps or shape preserving parameterizations is the triangle fan around each vertex, in other words an *ordered* set of neighbours $N_{\boldsymbol{v}}$ for each $\boldsymbol{v} \in V$. And computing the distance weights for the spring model parameterizations does not even require this ordering but an *unordered* neighbourhood $N_{\boldsymbol{v}}$ only. Several ways of defining neighbourhoods of scattered data points will be explained in Section 2.2.1.

Once these neighbourhoods are specified, we can apply one of the linear parameterization methods to determine parameter points $\psi(\boldsymbol{v})$, one for each $\boldsymbol{v} \in V$. Note that in contrast to Chapter 1, ψ is not a piecewise linear function over some triangulation but just a discrete mapping $\psi : V \to \Omega$ with $\Omega \subset \mathbb{R}^2$. We can then use one of the standard methods for triangulating two-dimensional data points, e.g. the *Delaunay triangulation*, to find a triangulation $\mathcal{S}$ of the parameter points $\psi(\boldsymbol{v})$. Finally, we create a triangulation $\mathcal{T}$ of the given point cloud by collecting all triangles $[\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}]$ for which $[\psi(\boldsymbol{u}), \psi(\boldsymbol{v}), \psi(\boldsymbol{w})]$ is a triangle in $\mathcal{S}$. We shall review the details of this approach in Section 2.2.2.

The drawback of this method is that it can only handle point sets which are assumed to be sampled from a single surface patch, i.e. a surface with genus zero and one boundary. However, more complex point sets can be split into several patches and each patch triangulated separately. In Section 2.2.3 we show how this can be done for points sampled from a surface homeomorphic to a sphere.

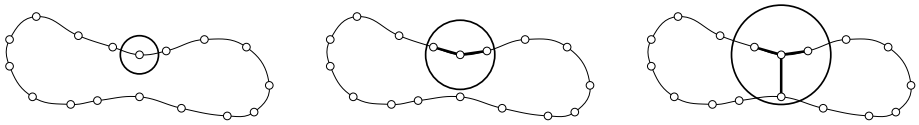


Figure 2.1: Choosing different radii for the ball neighbourhood.

2.2.1 Local Neighbourhoods

The simplest way of defining a set of unordered neighbours $N_{\boldsymbol{v}}$ for a point $\boldsymbol{v}$ from a point cloud V is motivated by discretizing the concept of neighbourhood on the surface from which the data was sampled. Suppose the underlying surface is a two-manifold M and let $d(\boldsymbol{v}, \boldsymbol{w})$ be the *geodesic distance*, i.e. the Euclidean length of the shortest path in M connecting $\boldsymbol{v}$ and $\boldsymbol{w}$. Then we can define for each $\boldsymbol{v} \in M$ the (open) r-neighbourhood

$$N_{\boldsymbol{v}}^{r,M} = \{\boldsymbol{w} \in M : 0 < d(\boldsymbol{v}, \boldsymbol{w}) < r\}$$

for some radius $r > 0$ and the discrete version

$$\hat{N}_{\boldsymbol{v}}^{r} = N_{\boldsymbol{v}}^{r,M} \cap V$$

for the data set $V \subset M$. Of course, M is unknown and we cannot determine $\hat{N}_{\boldsymbol{v}}^{r}$ this way, but the *ball neighbourhood*

$$N_{\boldsymbol{v}}^{r} = \{\boldsymbol{w} \in V : 0 < \|\boldsymbol{v} - \boldsymbol{w}\| < r\}.$$

is a good approximation as long as the ball radius r is adequately chosen. If r is too large then it may happen that the neighbourhood of a vertex contains points that were sampled from a different part of the surface and if r is too small then the set of neighbours can be empty for some data points (see Figure 2.1).

A data set $V \subset M$ is said to be ρ-*dense* [65] if ρ is the smallest value such that any sphere with radius ρ and center in M contains at least one point in V. For a ρ-dense data set we can assure $N_{\boldsymbol{v}}^{r} \neq \emptyset$ for all $\boldsymbol{v} \in V$ if $r > 2\rho$ and if the distribution of the points is reasonably uniform in addition, $r = 2\rho$ usually is a good choice. However, determining ρ remains a rather computationally expensive task and it is simpler to choose r interactively.

If the density of the point set V varies, it might be preferable to adapt r to the local density of V in order to avoid big differences in the size of the neighbourhoods. Another strategy is to fix the size of $N_{\boldsymbol{v}}$ and let each neighbourhood consist of the k nearest points to $\boldsymbol{v}$. In practice, $k \approx 10$ has proven to give good results.

Figure 2.2 illustrates the effect of choosing different values for r and k by showing the *connectivity graph* of the data set which consists of all edges

$$E = \{[\boldsymbol{v}, \boldsymbol{w}] : \boldsymbol{w} \in N_{\boldsymbol{v}} \text{ or } \boldsymbol{v} \in N_{\boldsymbol{w}}\}.$$

Note that $\boldsymbol{w} \in N_{\boldsymbol{v}}$ implies $\boldsymbol{v} \in N_{\boldsymbol{w}}$ only for the ball neighbourhood.

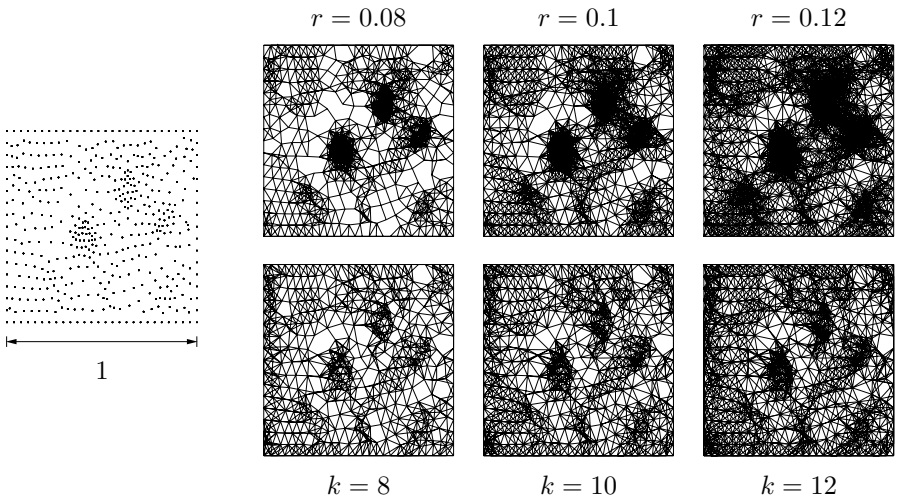


Figure 2.2: Connectivity graph of the data set to the left for different ball neighbourhoods (top) and different numbers of nearest neighbours (bottom).

The ball neighbourhood as well as the neighbourhood defined by the k nearest points are unordered and allow to use the parameterization methods based on the distance weights. But computing a harmonic or a shape preserving parameterization further requires the neighbours of each point to be ordered. This can be done as follows. Let $N_{\boldsymbol{v}} = \{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n\}$ be the unordered neighbours of a point $\boldsymbol{v} \in V$. Then we first compute the least squares plane $P_{\boldsymbol{v}}$ of $\boldsymbol{v} \cup N_{\boldsymbol{v}}$ and project $\boldsymbol{v}$ and $N_{\boldsymbol{v}}$ orthogonally into $P_{\boldsymbol{v}}$, yielding new vertices $\boldsymbol{v}'$ and $\boldsymbol{v}'_1, \ldots, \boldsymbol{v}'_n$ (compare Section 1.2.5). A slightly different approach that rotates the vertices in $N_{\boldsymbol{v}}$ into the least squares plane of $N_{\boldsymbol{v}}$ through $\boldsymbol{v}$ was proposed in [54]. However, in both cases we proceed by computing the Delaunay triangulation $\mathcal{T}'$ of the new vertices and defining the *Delaunay neighbourhood*

$$N'_{\boldsymbol{v}} = \{\boldsymbol{v}_i \in N_{\boldsymbol{v}} : [\boldsymbol{v}', \boldsymbol{v}'_i] \in E(\mathcal{T}')\}$$

and order the vertices $\boldsymbol{w}$ in $N'_{\boldsymbol{v}}$ in the same way as the corresponding vertices $\boldsymbol{w}'$ around $\boldsymbol{v}'$ in $\mathcal{T}'$. The whole process is shown in Figure 2.3.

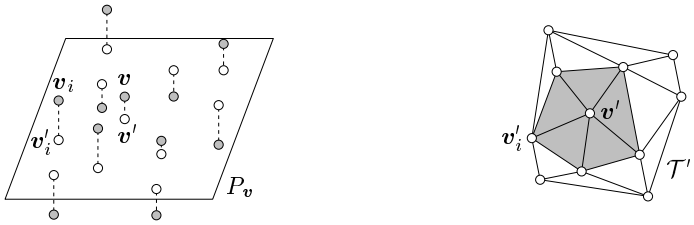


Figure 2.3: Projecting the unordered neighbours into the least squares plane (left) and computing the Delaunay neighbourhood (right).

2.2.2 Patch Topology

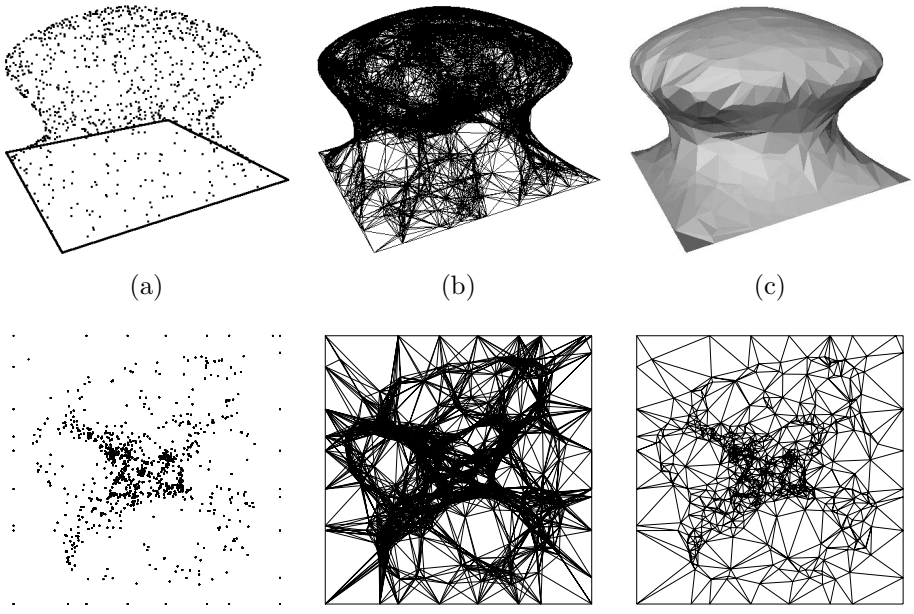
In order to parameterize a point cloud V which is assumed to be sampled from a surface patch we proceed in the same way as for the parameterization of a simple triangulation. We first specify parameter points for the boundary vertices V_B and then find the parameter points of the interior vertices $V_I = V \setminus V_B$ by solving a linear system. Therefore, we have to identify a set $V_B \subset V$ containing those vertices in the data set V that were sampled from the boundary of the underlying surface. This can be done interactively or by means of the method proposed by Floater and Reimers in [47].

They suggest to let V_B consist of those vertices $\boldsymbol{v} \in V$ for which the projected point $\boldsymbol{v}'$ is a boundary vertex in the local Delaunay triangulation $\mathcal{T}'$. Since according to Section 1.2.5 we also require the boundary vertices to be ordered, they further divide V_B into two halves V_B^1, V_B^2 using a suitable dividing plane and apply a univariate analogue of the meshless parameterization method to determine the order of the vertices in V_B^1 and V_B^2.

After specifying the ordered set of boundary vertices we can compute corresponding parameter points $\psi(\boldsymbol{v})$, $\boldsymbol{v} \in V_B$ with any of the methods from Section 1.2.5. The remaining $\psi(\boldsymbol{v})$, $\boldsymbol{v} \in V_I$ are then determined by solving the linear system (1.15) with either distance, harmonic, or shape preserving weights (see Section 1.2). With the arguments from the proof of Theorem 1.5 it can be shown that the linear system for the unorganized points is uniquely solvable as long as the connectivity graph is simply connected. Whether the resulting parameter points $\psi(\boldsymbol{v})$ are always distinct remains an open problem but in the numerical examples this only happened for uniform parameterizations. In fact, it has been shown in [47] that if the neighbourhoods of two interior vertices $\boldsymbol{v}$, $\boldsymbol{w}$ have the property that $\boldsymbol{v} \cup N_{\boldsymbol{v}} = \boldsymbol{w} \cup N_{\boldsymbol{w}}$ and uniform weights are used to determine the parameter points, then $\psi(\boldsymbol{v}) = \psi(\boldsymbol{w})$.

Once the parameter points $\psi(\boldsymbol{v})$, $\boldsymbol{v} \in V$ have been computed, we can use a standard method for finding a triangulation $\mathcal{S}$ with $V(\mathcal{S}) = \psi(V)$. One particular choice is the *Delaunay triangulation* which can be characterized in many ways, e.g. by the fact that the circumcircle of each triangle in the Delaunay triangulation does not contain any vertices (local circle criterion) or that the Delaunay triangulation is among all possible triangulations of a data set the one that maximizes the minimal angle (max-min angle criterion). For other properties of the Delaunay triangulation and efficient algorithms for its determination we refer to [24, 126, 89, 50, 120, 60]. Finally, we let $\mathcal{T}$ be the triangulation of V corresponding to $\mathcal{S}$, in other words, we take $\mathcal{T}$ to be the set of triangles $[\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}]$ for which $[\psi(\boldsymbol{u}), \psi(\boldsymbol{v}), \psi(\boldsymbol{w})]$ is a triangle in $\mathcal{S}$.

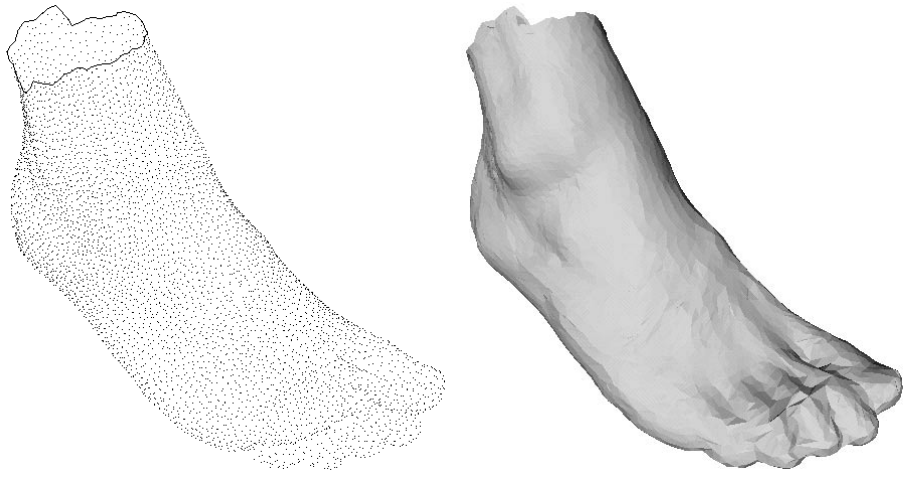
Figures 2.4–2.6 show some examples. In the saltdome example, the ordered boundary was already part of the data set and we chose the projection method for parameterizing it. For parameterizing the interior vertices we used local ball neighbourhoods with radius $r = 0.6$. The boundary of the foot data set was specified interactively and parameterized by chord length onto a unit circle. In this example we computed the $k = 10$ nearest neighbours for each data point. For the bunny example, the boundary was determined by univariate



(a) (b) (c)

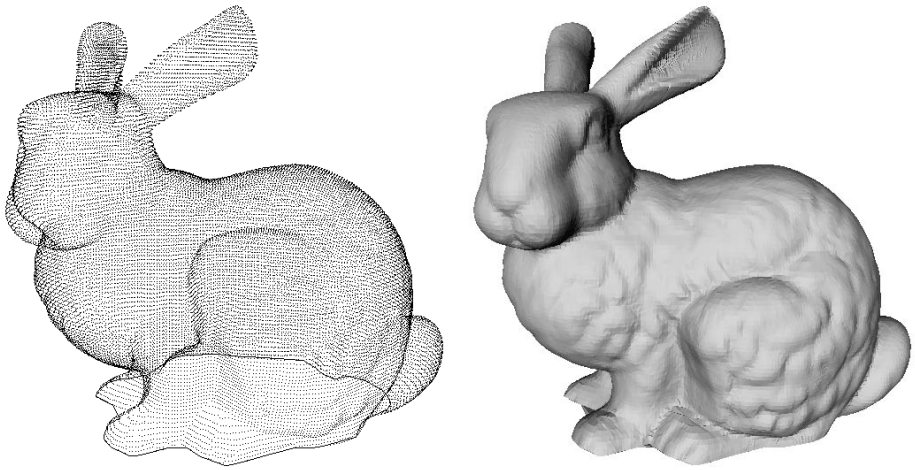
(d) (e) (f)

Figure 2.4: Point cloud of a saltdome with 1 019 data points (a), connectivity graph for the ball neighbourhoods with $r = 0.6$ (b), and reconstructed triangulation with 2 006 triangles (c). The bottom row shows the corresponding parameter points (d), their connectivity graph (e), and the Delaunay triangulation (f).



(a) (b)

Figure 2.5: Point cloud of a foot with 5 092 data points (a) and reconstructed triangulation with 10 121 triangles (b).



(a) (b)

Figure 2.6: Point cloud of a bunny with 30 571 data points (a) and reconstructed triangulation with 61 050 triangles (b).

meshless parameterization and mapped onto a circle. The radius of the ball neighbourhoods was set to $r = 0.003$. In all three examples we employed chord length parameterization for the interior vertices and Delaunay triangulation for triangulating the parameter points.

2.2.3 Spherical Topology

For parameterizing a point cloud V that was sampled from an object which is topologically more complicated, e.g. homeomorphic to a sphere or a torus, we can adapt the ideas of parameterizing triangulations with arbitrary topology from Section 1.4. That is, we start with a coarse base mesh $\mathcal{T}^0$ whose vertices $V(\mathcal{T}^0)$ are a subset of V and which has the same topology as the underlying object. Then we partition the point cloud V into $M_0 = |\mathcal{T}^0|$ subsets $V^1, \ldots, V^{M_0}$ such that each V^i corresponds to one of the triangles $T_i \in \mathcal{T}^0$.

Like in Section 1.4 this partitioning of V is found as follows. For each edge $e = [\boldsymbol{v}, \boldsymbol{w}] \in E^0$ in the base mesh we find the shortest path $\bar{e}$ in the connectivity graph E of V connecting $\boldsymbol{v}$ and $\boldsymbol{w}$, say

$$\bar{e} = [\boldsymbol{u}_1, \boldsymbol{u}_2] \cup [\boldsymbol{u}_2, \boldsymbol{u}_3] \cup \cdots \cup [\boldsymbol{u}_{r-1}, \boldsymbol{u}_r],$$

where $\boldsymbol{u}_1 = \boldsymbol{v}$, $\boldsymbol{u}_r = \boldsymbol{w}$ and $[\boldsymbol{u}_i, \boldsymbol{u}_{i+1}] \in E$. A common algorithm for determining shortest paths in a graph is *Dijkstra's algorithm* [28]. We further let $V_{\boldsymbol{vw}} \subset V$ be the set of vertices traversed by this path, $V_{\boldsymbol{vw}} = \bigcup_{i=1}^{r} \boldsymbol{u}_i$. The remaining *interior* vertices

$$V_I = V \setminus \bigcup_{[\boldsymbol{v},\boldsymbol{w}] \in E^0} V_{\boldsymbol{vw}}$$

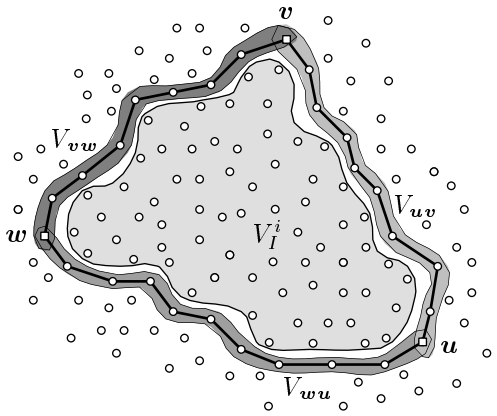


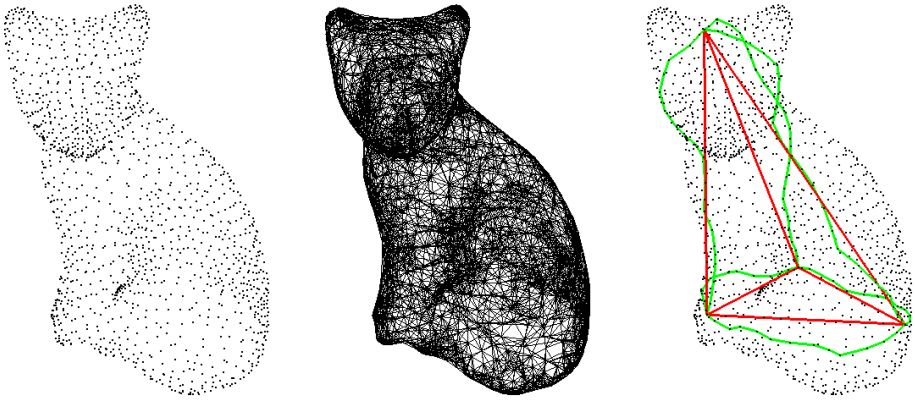
Figure 2.7: The vertices traversed by the shortest paths between $\boldsymbol{u}$, $\boldsymbol{v}$, and $\boldsymbol{w}$ are collected in the sets $V_{\boldsymbol{uv}}$, $V_{\boldsymbol{vw}}$, and $V_{\boldsymbol{wu}}$ and the vertices that are surrounded by these boundary vertices are defined as the interior vertices V_I^i.

are then split into disjoint subsets $V_I^1, \ldots, V_I^{M_0}$ such that for each base mesh triangle $T_i = [\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}] \in \mathcal{T}^0$ the corresponding set of interior vertices V_I^i is bounded by the shortest paths connecting $\boldsymbol{u}$, $\boldsymbol{v}$, and $\boldsymbol{w}$ (see Figure 2.7). We collect all vertices traversed by these three paths in the set of *boundary* vertices $V_B^i = V_{\boldsymbol{uv}} \cup V_{\boldsymbol{vw}} \cup V_{\boldsymbol{wu}}$ and finally let $V^i = V_B^i \cup V_I^i$ for all $i = 1, \ldots, M_0$.

We can now apply the triangulation method of Section 2.2.2 to each V^i, yielding triangulations $\mathcal{T}^i$ whose union $\mathcal{T} = \mathcal{T}^1 \cup \cdots \cup \mathcal{T}^{M_0}$ gives a triangulation of the whole point cloud V. Note that the order of the boundary vertices V_B^i as required by the meshless parameterization method is given by the sequence of vertices in the shortest paths. The reason for $\mathcal{T}$ to be a proper triangulation is that neighbouring triangulations $\mathcal{T}^i$ and $\mathcal{T}^j$ match nicely. Consider two neighbouring triangles T_i, T_j in $\mathcal{T}^0$ and their common edge $e = [\boldsymbol{v}, \boldsymbol{w}]$. Then the shortest path $\bar{e}$ between $\boldsymbol{v}$ and $\boldsymbol{w}$ in E is part of both the boundaries of $\mathcal{T}^i$ and $\mathcal{T}^j$. In fact, $\bar{e} = \Omega_{\mathcal{T}^i} \cap \Omega_{\mathcal{T}^j}$, and therefore $\mathcal{T}^i \cup \mathcal{T}^j$ is a proper triangulation.

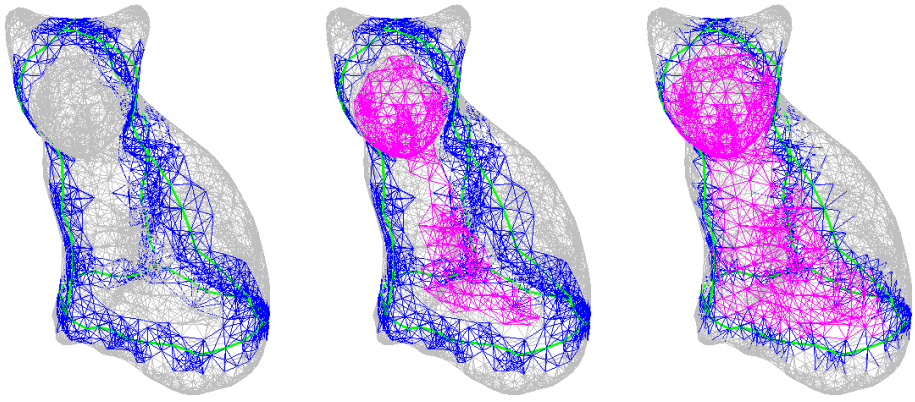
For a better understanding of this algorithm the following example may be helpful. The point cloud V in Figure 2.8 was sampled from an object homeomorphic to a sphere and we therefore chose the simplest base mesh representing this topology, namely a tetrahedron $\mathcal{T}^0$, whose vertices were set interactively be the user. Figure 2.8.c shows this base mesh as well as the shortest paths connecting its vertices in the connectivity graph E of V. In the next step of the algorithm we have to split the interior vertices into four subsets $V_I^1, \ldots, V_I^4$, each corresponding to one of the four triangles in $\mathcal{T}^0$. Figures 2.9 and 2.10 illustrate how this is done. Firstly, we temporarily enlarge the set of boundary vertices by adding all neighbouring vertices,

$$V_B' = V_B \cup \{\boldsymbol{v} \in V_I : \exists\ \boldsymbol{w} \in V_B \text{ with } [\boldsymbol{v}, \boldsymbol{w}] \in E\}.$$



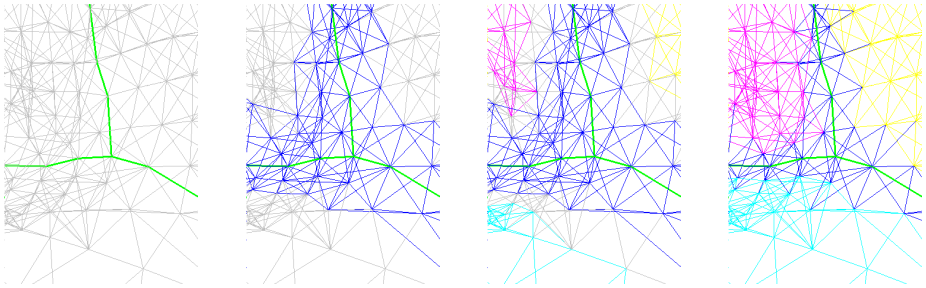
(a) (b) (c)

Figure 2.8: Point cloud of a cat with 1 458 data points (a), connectivity graph using the 12 nearest neighbours (b), tetrahedron base mesh $\mathcal{T}^0$, and shortest paths corresponding to the edges of $\mathcal{T}^0$ (c).



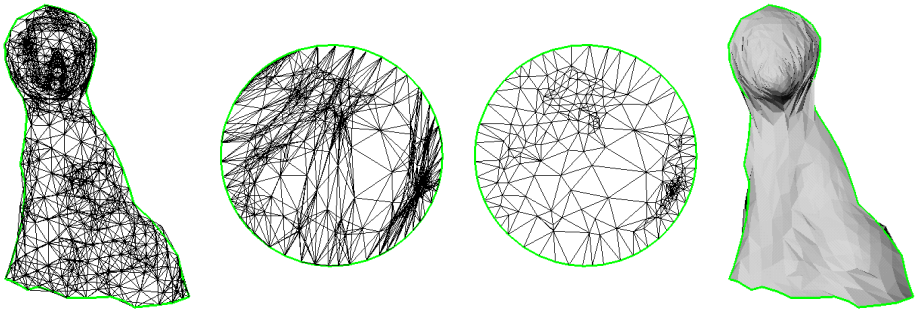
(a) (b) (c)

Figure 2.9: Detecting interior vertices by first growing the shortest paths (a), then finding a connected component in the connectivity graph (b), and finally growing that region (c).



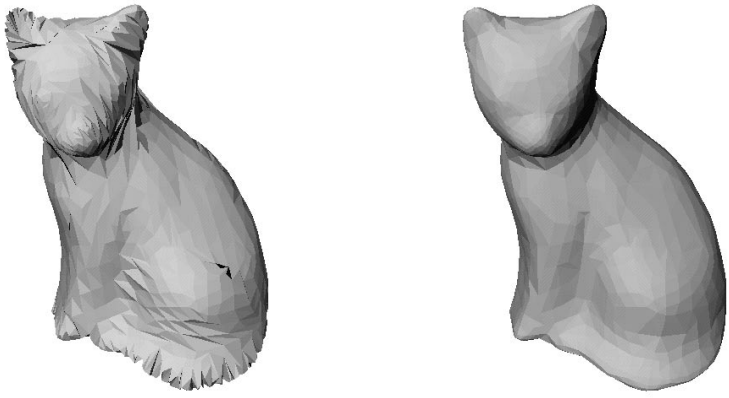
(a) (b) (c) (d)

Figure 2.10: Close-up view to the connectivity graph with shortest path (a). Interior vertices are found by growing the shortest paths (b), finding connected components in the connectivity graph (c), and growing these regions (d).



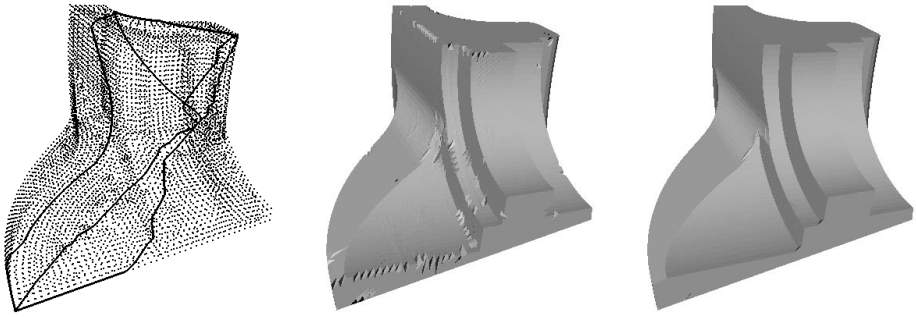
(a) (b) (c) (d)

Figure 2.11: Reconstructing one of the subsets V^i. Connectivity graph (a), parameterization (b), Delaunay triangulation (c), and triangulation $\mathcal{T}^i$.



(a) (b)

Figure 2.12: Final reconstruction $\mathcal{T}$ with 2 912 triangles before (a), and after optimization (b).



(a) (b) (c)

Figure 2.13: Point cloud of a fandisk with 6 475 data points and shortest paths (a), reconstructed triangulation with 12 946 triangles before (b) and after optimization (c).

For clarity, we have coloured all edges $[\boldsymbol{v}, \boldsymbol{w}]$ with $\boldsymbol{v}, \boldsymbol{w} \in V'_B$ blue. We then remove all edges incident to one of the vertices in V'_B from the connectivity graph,

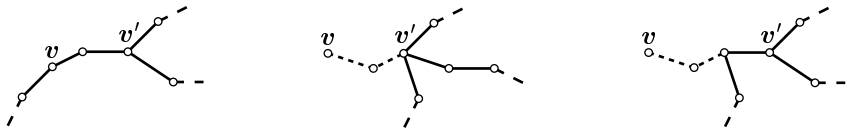
$$E' = E \setminus \big\{[\boldsymbol{v}, \boldsymbol{w}] : \boldsymbol{v} \in V'_B\big\},$$

and determine the connected components in E' as shown in Figures 2.9.b and 2.10.c. In this example, the number of connected components equals the number of triangles in the base mesh but if the connectivity graph is very dense, it may happen that two or more regions that should be disjoint are still connected by an edge, resulting in a smaller number of connected components. Then we simply enlarge V'_B again and remove further edges from the connectivity graph until E' decomposes into the desired number of components E^i.

Finally, the components E^i are enlarged again by adding the "missing" interior vertices. For each $\boldsymbol{v} \in V'_B \setminus V_B$ we determine the closest neighbour $\boldsymbol{w} \in N_{\boldsymbol{v}}$ that is not in V'_B. If such a $\boldsymbol{w}$ exists, it clearly belongs to one of the components E^i. In this case we add all edges $[\boldsymbol{v}, \boldsymbol{w}] \in E$ with $\boldsymbol{w} \in E^i$ to E^i and remove $\boldsymbol{v}$ from V'_B. If all neighbours of $\boldsymbol{v}$ are in V'_B then we proceed with another vertex and come back to $\boldsymbol{v}$ later. This growing procedure stops when all vertices in $V'_B \setminus V_B$ have been added to one of the components (see Figures 2.9.c and 2.10.d) and we can now define the disjoint sets of interior vertices

$$V_I^i = \{\boldsymbol{v} \in V_I : \boldsymbol{v} \in E^i\}.$$

We then reconstruct each subset V^i with the method from Section 2.2.2, yielding triangulations $\mathcal{T}^i$ (see Figure 2.11) which are finally merged to the triangulation $\mathcal{T}$ of V shown in Figure 2.12.a. Although it looks a bit crinkly it is a topologically correct triangulation which can be further optimized to give the result in Figure 2.12.b. We will explain how to perform this optimization in the next section. Figure 2.13 shows another example.



(a) (b) (c)

Figure 2.14: Avoiding coincidence of three shortest paths emanating from $\boldsymbol{v}$.

Let us wind up this section by mentioning that although we have only tested this method for spherical topology it can also be applied to arbitrary topology, but it might be hard to interactively set the vertices of the base mesh in extreme cases. We also realize that the base mesh does not necessarily have to consist of triangles but can be a more general polyhedral object since we only need the base mesh as a blueprint of a patch-layout. With general n-gons as faces we are also able to create base meshes of any topology where all vertices have exactly three neighbours. This is of particular interest since we can avoid the problem of coincident shortest paths for such vertices which may lead to degeneracies otherwise. As we compute the shortest paths in the connectivity graph emanating from a vertex $\boldsymbol{v}$ it can happen that the first segments of two or more paths are the same. However, if there are exactly three paths then we can always find a new vertex $\boldsymbol{v}'$ with three different outgoing paths. Figure 2.14 shows how to find $\boldsymbol{v}'$. If two of the paths from $\boldsymbol{v}$ coincide we simply follow this common path until we reach the branching point $\boldsymbol{v}'$ where the two paths diverge (a). Should all three paths have a common starting sequence we omit these edges and choose $\boldsymbol{v}'$ as in Figure 2.14.b or 2.14.c.

2.3 Optimizing Triangulations

Over the past few years the problem of *fairing* (or *smoothing*) triangulations has received a lot of attention. The need for these methods ranges from technical applications, where the noise that is due to measurement errors has to be removed from measured data, to entertainment applications, that require triangulated three-dimensional models with a pleasing visual appearance. The usual approach in fairing triangulations is to move the vertices of the triangulation such that a certain energy functional is minimized [129, 81, 25] or to apply a subdivision scheme [17, 31, 93, 32, 141, 82, 85] that refines a given triangulation by inserting additional vertices and places the new vertices such that the refined triangulation is smooth.

However, these methods cannot be applied whenever the position and the number of the original data points must not be changed, which is exactly the situation at which we arrive in Section 2.2 after having triangulated a point cloud V. Of course, there exist many triangulations with V as vertices which differ by the way in which the points are connected by triangles and in this section we investigate how to obtain an optimal triangulation of V.

While the optimization of two-dimensional triangulations has been studied thoroughly in the early 1990ies [34, 109, 10, 121] little is known so far of the three-dimensional case [131, 2, 3]. Assuming the surface from which the data points were sampled to be smooth we claim an optimal triangulation to be smooth, too. But how to define the smoothness of a triangulation?

In Section 4.3.3 we shall see that the smoothness of (twice differentiable) surfaces can be expressed in terms of curvature. Since the concept of curvature carries over to triangulations as continuous, piecewise linear surfaces, yielding *discrete curvatures* as explained in Section 2.3.1, we can use discrete variants of the smoothness functionals from Section 4.3.3 to rate the quality of a triangulation.

Then a simple algorithm can be used to find the optimal triangulation in the sense that it minimizes a discrete smoothness functional. Starting with an arbitrary initial triangulation of V we successively swap edges in a greedy way so as to reduce the functional at each step to the highest possible degree. We shall explain this algorithm that is guaranteed to terminate at a (local) minimum of the functional in Section 2.3.2 and give some examples.

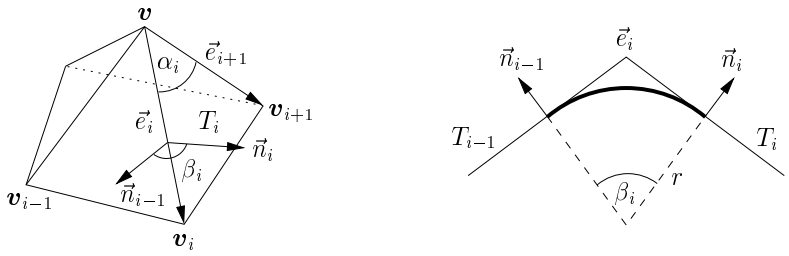
2.3.1 Discrete Curvatures

From a theoretical point of view, a triangulation $\mathcal{T}$ does not have any curvature at all, since all triangles are flat and the curvature is not properly defined along edges and at vertices because the surface $\Omega_{\mathcal{T}}$ is not C^2-differentiable there. But thinking of a triangulation as a piecewise linear approximation of an unknown smooth surface one can try to estimate the curvatures of that unknown surface by using only the information that is given by the triangulation itself. We are particularly interested in computing the *Gaussian curvature* K and the absolute *mean curvature* $|H|$ at the vertices of the triangulation, since we base the discrete smoothness functionals to be minimized in the optimization process on these values. But let us first fix the notation before explaining how to derive K and $|H|$ from the given data.

Let $\boldsymbol{v} \in V_I$ be an interior vertex of a triangulation $\mathcal{T}$ and let $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ be the ordered neighbouring vertices of $\boldsymbol{v}$ as in Figure 2.15.a. We define the (directed) edges $\vec{e}_i = \boldsymbol{v}_i - \boldsymbol{v}$, the angles $\alpha_i = \sphericalangle(\vec{e}_i, \vec{e}_{i+1})$ between two successive edges, and the normal $\vec{n}_i = \frac{\vec{e}_i \times \vec{e}_{i+1}}{\|\vec{e}_i \times \vec{e}_{i+1}\|}$ of the triangle $T_i = [\boldsymbol{v}, \boldsymbol{v}_i, \boldsymbol{v}_{i+1}]$. The *dihedral angle* $\beta_i = \sphericalangle(\vec{n}_{i-1}, \vec{n}_i)$ at an edge $\vec{e}_i$ is the angle between the normals of the adjacent triangles. Note that in these definitions we identify the index 0 with n and the index $n+1$ with 1.

Now we can define the integral Gaussian curvature $\bar{K} = \bar{K}_{\boldsymbol{v}}$ and the integral absolute mean curvature $|\bar{H}| = |\bar{H}_{\boldsymbol{v}}|$ with respect to the region $S = S_{\boldsymbol{v}}$ attributed to $\boldsymbol{v}$ by

$$\bar{K} = \int_S K = 2\pi - \sum_{i=1}^{n} \alpha_i \qquad \text{and} \qquad |\bar{H}| = \int_S |H| = \frac{1}{4} \sum_{i=1}^{n} \|\vec{e}_i\| \, |\beta_i|. \tag{2.1}$$



(a) (b)

Figure 2.15: The local configuration around an interior vertex $\boldsymbol{v}$ (a) and the blending cylinder along $\vec{e}_i$ between triangles T_{i-1} and T_i, seen from the side (b).

These formulas are also used by other authors [129, 131, 2, 3] and can be understood in the following way. Suppose we replace each edge by a small cylinder of radius r that joins the adjacent triangles tangentially (see Figure 2.15.b) and blend these cylinders smoothly at the vertices in a C^2 manner. Now the triangulation is approximated by a smooth surface, K and $|H|$ are integrable functions on it, and we can apply well-known theorems from differential geometry [29]. A straightforward computation [131, 2] finally results in formulas (2.1) that depend neither on the choice of r nor on the specific blending method at the vertices.

To derive the curvatures at the vertex $\boldsymbol{v}$ from these integral values, we assume the curvatures to be uniformly distributed around the vertex, and simply normalize by the area of the region S:

$$K = \frac{\bar{K}}{\text{area}(S)} \qquad \text{and} \qquad |H| = \frac{|\bar{H}|}{\text{area}(S)}.$$

Of course there are different ways of defining the region $S_{\boldsymbol{v}}$ attributed to a vertex $\boldsymbol{v}$, which result in different curvature values. We confine ourselves to those methods for which the regions around all vertices sum up to the surface of the triangulation $\mathcal{T}$, i.e., $\sum_{\boldsymbol{v}\in V} S_{\boldsymbol{v}} = \Omega_{\mathcal{T}}$, since this enables us to write an integral over $\Omega_{\mathcal{T}}$ as the sum of integrals over the single regions, e.g. $\int_{\Omega_{\mathcal{T}}} K = \sum_{\boldsymbol{v}\in V} \int_{S_{\boldsymbol{v}}} K$.

The region that is most commonly used in literature is the *barycentric region* S^B which is one third of the area of the triangles adjacent to $\boldsymbol{v}$, and can be constructed by connecting the edge midpoints with the barycenters of the adjacent triangles (see Figure 2.16.a). However, we decided to use the *Voronoi region* S^V instead, which sums up the local Voronoi cells of $\boldsymbol{v}$ restricted to the triangles adjacent to $\boldsymbol{v}$, according to the Euclidean distance to the vertices of the triangulation (see Figure 2.16.b).

In order to compute the areas of the local Voronoi cells restricted to a triangle $\triangle = [A, B, C]$, we have to distinguish between obtuse and non-obtuse triangles

(a) (b)

Figure 2.16: Barycentric region S^B (a) and Voronoi region S^V (b) around a vertex.

(see Figure 2.17). In the latter case they are given by

$$\operatorname{area}\big(S_A^V\big) = \frac{1}{8}\left(b^2\cot(\beta) + c^2\cot(\gamma)\right),$$

and likewise for S_B^V and S_C^V. For obtuse triangles,

$$\operatorname{area}\big(S_B^V\big) = \tfrac{1}{8}\,c^2\tan(\beta), \qquad \operatorname{area}\big(S_C^V\big) = \tfrac{1}{8}\,b^2\tan(\gamma),$$
$$\operatorname{area}\big(S_A^V\big) = \operatorname{area}(\triangle) - \operatorname{area}\big(S_B^V\big) - \operatorname{area}\big(S_C^V\big).$$

Besides the Gaussian and the absolute mean curvature, we are also interested in the sum of the absolute principal curvatures $|\kappa_1|$ and $|\kappa_2|$. From the relations $K = \kappa_1\kappa_2$ and $H = \frac{1}{2}(\kappa_1 + \kappa_2)$, we get $\kappa_{1,2} = H \pm \sqrt{H^2 - K}$. Moreover, we get the sum of the absolute principal curvatures without knowing H but only $|H|$:

$$|\kappa_1| + |\kappa_2| = \begin{cases} 2\,|H|, & \text{if } K \geq 0, \\ 2\,\sqrt{|H|^2 - K}, & \text{otherwise.} \end{cases}$$

Note that $|\kappa_1| + |\kappa_2|$ is always a real number, even if $|H|^2 = H^2 < K$, which corresponds to complex principal curvature values. Of course, H^2 is always greater or equal to K for smooth surfaces, but since we are dealing with discrete curvatures, it may be smaller than K at some vertices.

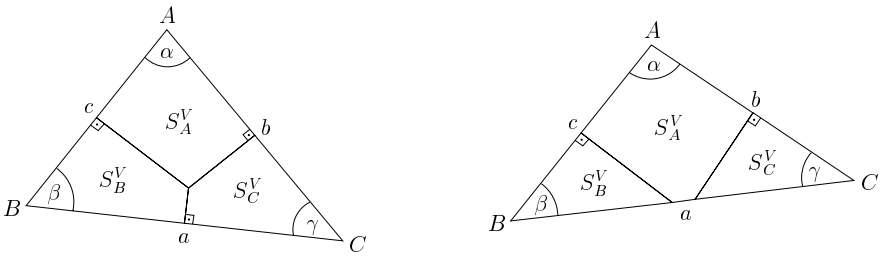


Figure 2.17: Local Voronoi cells restricted to a triangle for a non-obtuse (left) and an obtuse triangle (right).

2.3.2 Minimizing Discrete Smoothness Functionals

One of the energy functionals that has often been used for fairing triangulations as well as for fairing continuous surfaces is the thin plate energy

$$F_{\mathrm{TP}} = \int_{\Omega_{\mathcal{T}}} 4a\,H^2 + 2\,(1-a-b)\,K,$$

with certain parameters $a, b \in \mathbb{R}$. However, for closed surfaces or surfaces with a fixed boundary this expression can be simplified to $F_{\mathrm{TP}} = 4a \int H^2$, because the theorem of Gauss-Bonnet states that $\int K$ is constant in these cases. Note that for closed surfaces this also holds for the discrete version of the integral Gaussian curvature:

$$\int_{\Omega_{\mathcal{T}}} K = \sum_{\boldsymbol{v}} \int_{S_{\boldsymbol{v}}} K = \sum_{\boldsymbol{v}} \left(2\pi - \sum_{i=1}^{n_{\boldsymbol{v}}} \alpha_{\boldsymbol{v}_i} \right) = |V|\,2\pi - |T|\,\pi = 2\pi\,\chi(\mathcal{T}),$$

where $\chi(\mathcal{T})$ is the Euler characteristic of $\mathcal{T}$. We therefore assume a constant integral Gaussian curvature in our setting.

Minimizing F_{TP} is therefore equivalent to the minimization of H in the L_2 norm, $\|H\|_2 = \left(\int H^2\right)^{1/2}$. Likewise, the minimization of the integral absolute mean curvature relates to the L_1 norm of H, $\|H\|_1 = \int |H|$. Besides these two energy functionals we have also used the L_1 norm of the principal curvatures, $\|\kappa\|_1 = \int |\kappa_1| + |\kappa_2|$, as an optimization criterion. Note that the minimum of $\|\kappa\|_2$ equals the minimum of $\|H\|_2$, since $\kappa_1^2 + \kappa_2^2 = 4H^2 - 2K$. Using (2.1), these three energy functionals are given by

$$F_1 = \|H\|_2^2 = \int_{\Omega_{\mathcal{T}}} |H|^2 = \sum_{\boldsymbol{v}\in V} \frac{1}{S_{\boldsymbol{v}}} |\bar{H}_{\boldsymbol{v}}|^2,$$

$$F_2 = \|H\|_1 = \int_{\Omega_{\mathcal{T}}} |H| = \sum_{\boldsymbol{v}\in V} |\bar{H}_{\boldsymbol{v}}|,$$

$$F_3 = \|\kappa\|_1 = \int_{\Omega_{\mathcal{T}}} |\kappa_1| + |\kappa_2| = \sum_{\boldsymbol{v}\in V} \begin{cases} 2\,|\bar{H}_{\boldsymbol{v}}|, & \text{if } \bar{K}_{\boldsymbol{v}} \geq 0, \\ 2\,\sqrt{|\bar{H}_{\boldsymbol{v}}|^2 - S_{\boldsymbol{v}}\bar{K}_{\boldsymbol{v}}}, & \text{otherwise.} \end{cases}$$

Choosing either of these three energy functionals as a cost function F and starting with an initial triangulation $\mathcal{T}$, we perform a local swapping algorithm that decreases the functional in each step. The key ingredients of this algorithm are the determination of a *swap value* s_j for each edge e_j and the use of a *priority queue* P. The swap value is the difference between the value of the functional before and after swapping the corresponding edge and indicates the reduction of the functional caused by this edge swap. Note that the swapping operation is not defined for boundary edges, and should be forbidden for edges that connect to a vertex of valency three, since it would result in two identical edges and two triangles glued together. We avoid swapping those edges by simply setting their swap value to $-\infty$. The priority queue P is a permutation of the set of integers

Figure 2.18: After swapping an edge (thick line) the swapping values of the edge itself and the edges in the butterfly-neighborhood (dashed lines) or the 2-neighbourhood (dotted lines) have to be updated.

$\{1, \ldots, |E|\}$, such that $s_{P(i)} \geq s_{P(j)}$ for all $1 \leq i < j \leq |E|$. The main advantage of using such a priority queue is the low complexity in building and updating it. Furthermore, $P(1)$ is always the index of the edge with the largest swap value and testing $s_{P(1)} > 0$ tells whether the functional can be further reduced by swapping one of the edges or not. When an edge swap is actually carried out, the swap value of the swapped edge and all neighbouring edges change and the priority queue has to be updated. The size of the neighbourhood in which swap values change depends on the functional chosen. For F_1 and F_3 we have to recompute the swap values of all edges in the *2-neighbourhood*, i.e. all edges for which the adjacent triangles share a common vertex with the adjacent triangles of the swapped edge (see Figure 2.18), while only the edges in the *butterfly-neighbourhood* need to be considered for the minimization of F_2.

As the number of all possible triangulations of the given data is finite and the functional is decreased by each swap, this algorithm is guaranteed to terminate in a finite number of steps. Unfortunately, it is generally impossible to determine whether the algorithm reaches a global minimum or not. For the L_1 norm of the Gaussian curvature, $\|K\|_1$, Alboul and van Damme could show that in case of convex data this optimization strategy always converges to the global minimum, which is the convex triangulation [3]. We have also tested our three functionals on convex data and observed that the convex triangulation was the global optimum for F_2 and F_3 in all the examples but we could not give a general prove yet. Nevertheless, this does not mean that the local swapping algorithm necessarily finds this global optimum because the functional can have local minima, too. Figure 2.19 shows an example of a convex data set for which the convex triangulation is the global minimum of F_1, F_2, and F_3. Still there exists a non-convex triangulation of the data corresponding to a local minimum of any of the functionals at which the local swapping algorithm might get trapped.

We have tested all three cost functions and observed that they behaved quite similarly within the scope of our investigations except for convex data, and it is hard to tell which one performs best. But we favour the use of F_2 for two reasons. Firstly, the computation of the absolute mean curvature is the simplest since it does not involve the calculation of the area around the vertices and secondly, the number of edges for which the swap values have to be updated after an edge swap is smallest for F_2 which speeds up the algorithm considerably.

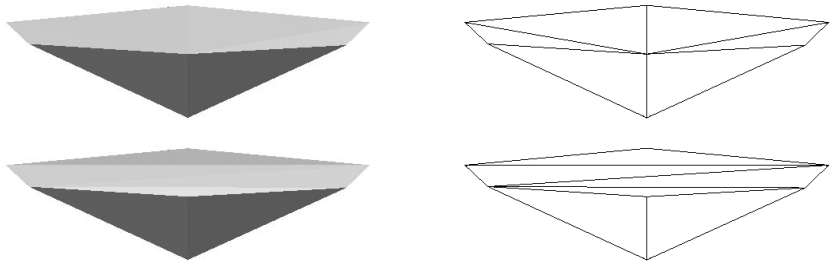


Figure 2.19: A convex data set with 7 vertices for which the non-convex triangulation in the first row corresponds to a local minimum of any of the functionals F_1, F_2, F_3. The second row shows the convex triangulation of the data.

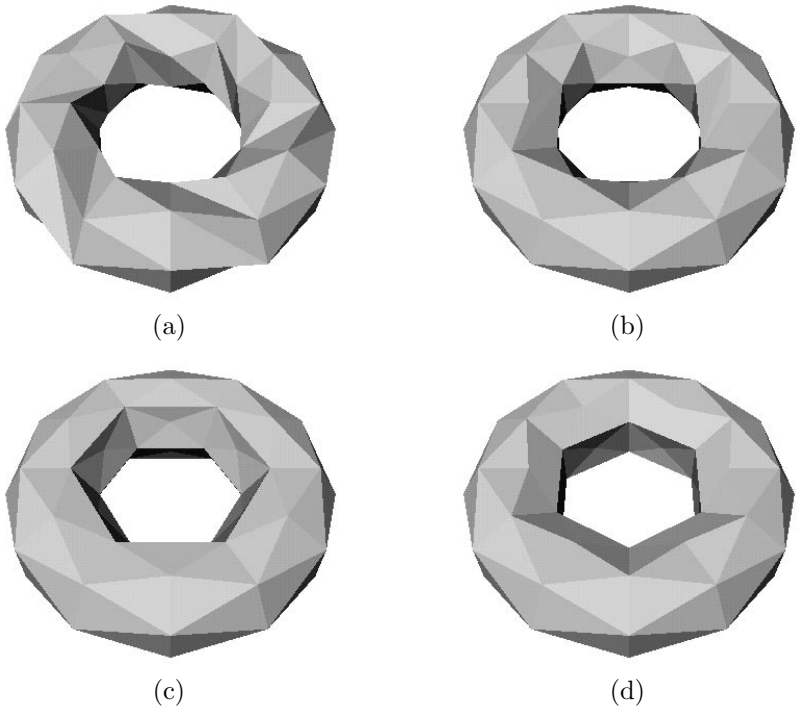


Figure 2.20: Optimizing a triangulation sampled from a torus (a) using different functionals: F_1 (b), F_2 (c), and F_3 (d).

Finally, we present some examples. Figure 2.20 shows the results using the different functionals on a simple data set. For the more complex data sets in Figures 2.12, 2.13, and 2.21–2.23, the results were so similar that we show the optimal triangulation with respect to our favourite functional F_2 only. Note how the optimized triangulations look much smoother and how the feature lines are enhanced in Figures 2.13, 2.22, and 2.23.

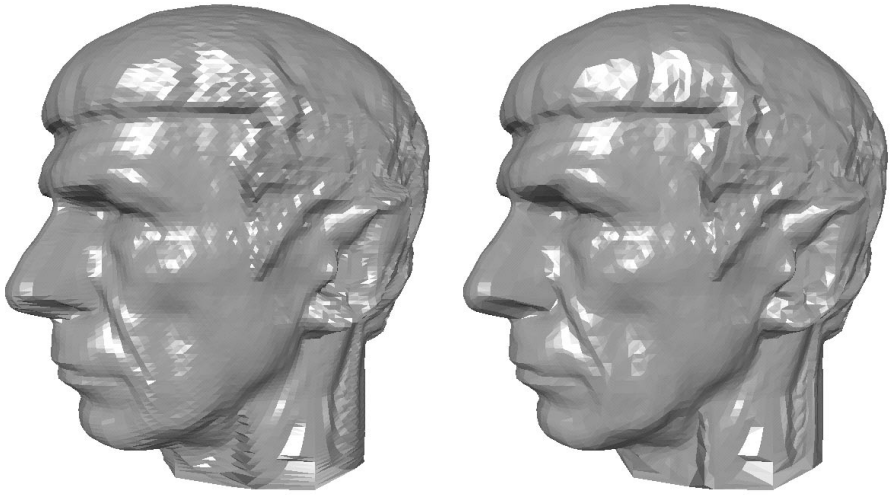


Figure 2.21: Optimizing a triangulation with 16 386 triangles.

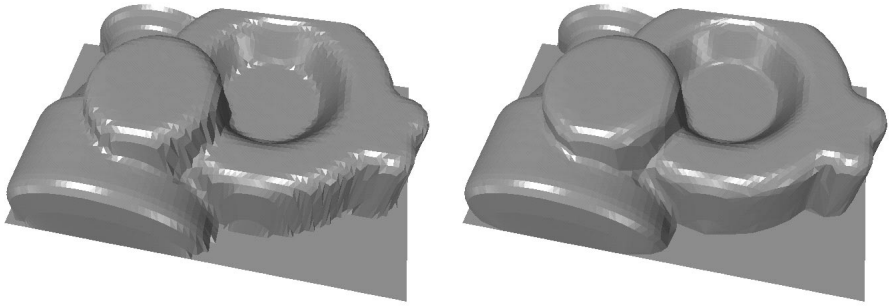


Figure 2.22: Optimizing a triangulation with 4 100 triangles.

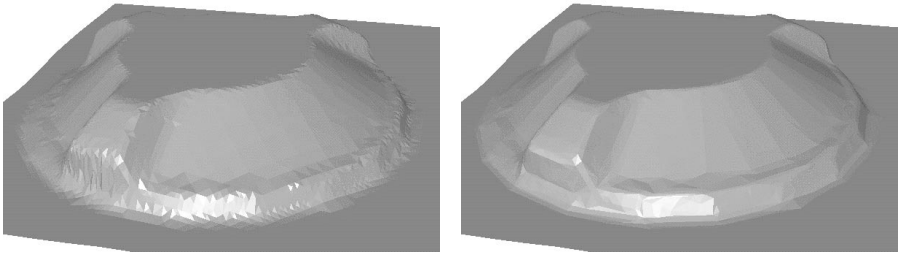


Figure 2.23: Optimizing a triangulation with 3 374 triangles.

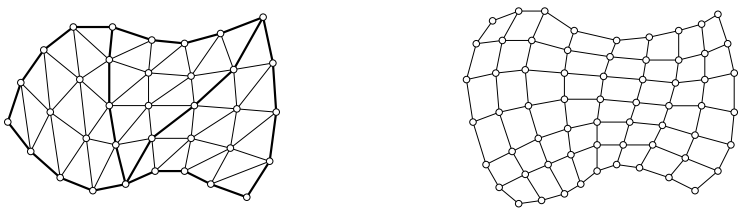
L'ordre est le plaisir de la raison,
mais le désordre est le délice de l'imagination.
Paul Claudel (1868–1955)

Chapter 3

Remeshing Triangulations

In this chapter we discuss how to approximate a given triangulation $\mathcal{T}$ by a new triangulation $\mathcal{T}'$ which has a certain connectivity structure. This process is commonly known as *remeshing* and motivated by the fact that the special structure of the new triangulation allows to apply very efficient algorithms for displaying, storing, transmitting, and editing. The special structure required by these algorithms is the *subdivision connectivity*, which is generated by iteratively refining a triangulation dyadically and is furthermore characterized by the property of almost all vertices having valency six (see Figure 3.1.a).

Several approaches exist to solve the remeshing problem of which the most important shall be listed in Section 3.1. The main idea behind most of these methods is similar to the one that we explain in detail in Section 3.2. We first find a parameterization $\phi : \Omega \to \Omega_{\mathcal{T}}$ of $\mathcal{T}$ over a parameter domain Ω which can easily be remeshed by a triangulation $\mathcal{S}'$ with vertices in Ω. Then ϕ is used to lift the vertices of $\mathcal{S}'$ to $\Omega_{\mathcal{T}}$ yielding the remesh $\mathcal{T}'$ (see Figure 3.4). In Section 3.3 we focus on remeshing a triangulation with another type of structured mesh, namely a *regular quadrilateral mesh* with quadrilateral facets and all vertices having valency four except for the boundary vertices (see Figure 3.1.b). We will use these kind of meshes for a very simple and efficient smooth surface reconstruction method with tensor product B-spline surfaces in Section 4.4.



(a) (b)

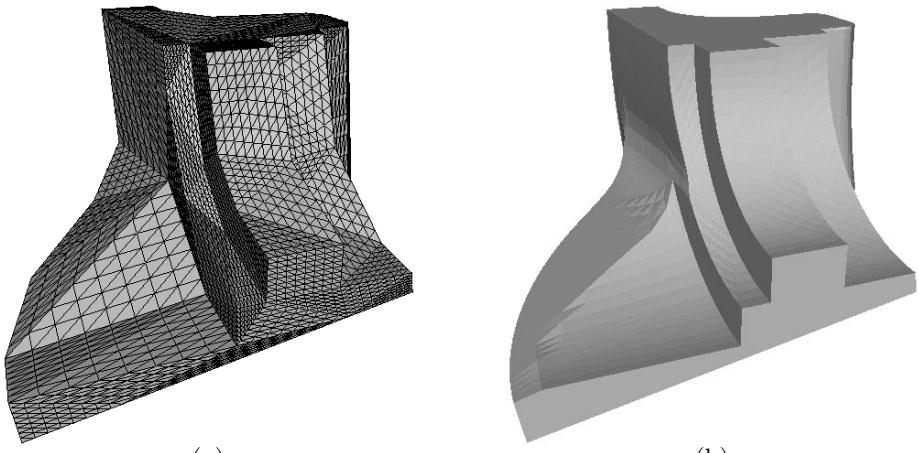
Figure 3.1: A triangulation with subdivision connectivity (a) and a regular quadrilateral mesh (b).

3.1 Related Work

The basic idea of the shrink wrapping approach by Kobbelt et al. [84] is to place a triangulation $\mathcal{T}'$ with subdivision connectivity around the given triangulation $\mathcal{T}$ and shrink it onto the surface $\Omega_{\mathcal{T}}$. This process simulates two forces: an attracting force, moving the vertices of $\mathcal{T}'$ in the direction of $\Omega_{\mathcal{T}}$, and a relaxing force, distributing the vertices uniformly over the surface. With this approach it is possible to remesh triangulations which are topologically equivalent to a sphere only.

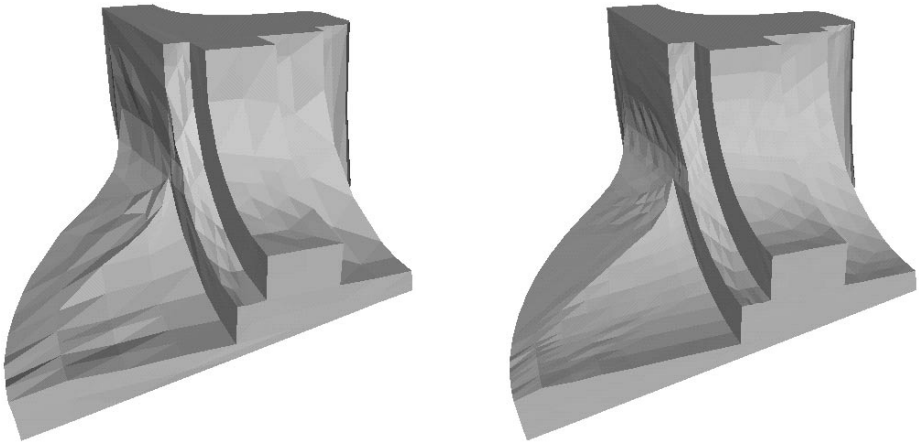
The problem of remeshing triangulations with arbitrary topology has been considered by Eck et al. in [35], Lee et al. in [90], and Floater et al. in [46]. All three methods start by constructing a simplified triangulation $\mathcal{T}^0$ which serves as a parameter domain, $\Omega = \Omega_{\mathcal{T}^0}$, and then finding a parameterization ϕ of $\mathcal{T}$ over Ω. In the next step they apply iterative dyadic refinement to the triangles in $\mathcal{T}^0$, yielding a sequence $\mathcal{T}^0, \mathcal{T}^1, \ldots, \mathcal{T}^j$ of triangulations with subdivision connectivity. Finally, the parameterization ϕ maps the vertices of $\mathcal{T}^j$ to the surface $\Omega_{\mathcal{T}}$ of the given triangulation so as to obtain the remesh $\mathcal{T}'$. Figure 3.2 shows an example, taken from [46]. In order to achieve global smoothness of $\mathcal{T}'$, the method in [90] applies a variant of Loop's subdivision scheme [93] to $\mathcal{T}^j$ prior to the mapping of the vertices.

The advantage of triangulations with subdivision connectivity is that the different refinement levels automatically provide a hierarchy which can be utilized by multiresolution algorithms such as level-of-detail rendering [19], progressive transmission [78, 87], multiresolution editing [142], and wavelet decomposition and reconstruction [117, 94, 46]. An example of the last application is shown in Figure 3.3. It was taken from [46] and we refer to that work for further details.



(a) (b)

Figure 3.2: Dyadic refinement $\mathcal{T}^3$ (a) of the triangulation $\mathcal{T}^0$ in Figure 1.40.b and remesh $\mathcal{T}'$ (b) of the triangulation $\mathcal{T}$ in Figure 1.40.a.



(a) (b)

Figure 3.3: Wavelet reconstruction of $\mathcal{T}'$ in Figure 3.2.b using 5% (a) and 20% (b) of the wavelet coefficients.

3.2 Subdivision Connectivity Triangulations

In this section we concentrate on the problem of remeshing a simple triangulation $\mathcal{T}$. The main idea of our remeshing strategy is to use a parameterization ϕ of $\mathcal{T}$ that enables us to shift the remeshing problem from three to two dimensions. Figure 3.4 sketches the whole concept. The inverse ψ of ϕ maps $\mathcal{T}$ to the planar triangulation $\mathcal{S}$ which is remeshed by a new planar triangulation $\mathcal{S}'$. Section 3.2.1 explains how to solve this planar remeshing problem. We finally find the remesh $\mathcal{T}'$ by lifting the vertices of $\mathcal{S}'$ with ϕ, in other words, we let $\mathcal{T}'$ be the set of triangles $[\phi(\boldsymbol{u}), \phi(\boldsymbol{v}), \phi(\boldsymbol{w})]$ for which $[\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}]$ is a triangle in $\mathcal{S}'$.

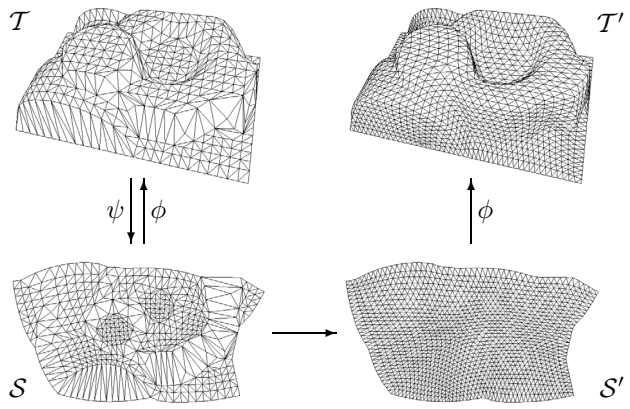


Figure 3.4: The general concept of remeshing a simple triangulation.

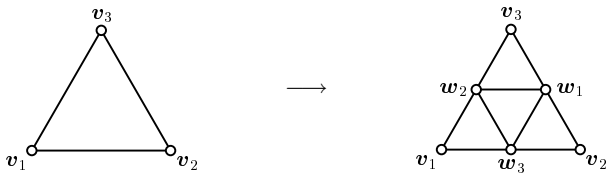


Figure 3.5: Dyadic refinement of a triangle.

As we have already mentioned we want the new triangulation $\mathcal{T}'$ to have a special kind of connectivity structure, namely subdivision connectivity. We call $\mathcal{T}'$ a *subdivision connectivity triangulation of level j* if it is the last element of a sequence of triangulations $\mathcal{T}^0, \mathcal{T}^1, \ldots, \mathcal{T}^j = \mathcal{T}'$ where each $\mathcal{T}^{i+1}$ emerges from $\mathcal{T}^i$ by dyadically refining each triangle $T = [\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3]$ in $\mathcal{T}^i$ into the four subtriangles

$$[\boldsymbol{v}_1, \boldsymbol{w}_2, \boldsymbol{w}_3], \quad [\boldsymbol{w}_1, \boldsymbol{v}_2, \boldsymbol{w}_3], \quad [\boldsymbol{w}_1, \boldsymbol{w}_2, \boldsymbol{v}_3], \quad [\boldsymbol{w}_1, \boldsymbol{w}_2, \boldsymbol{w}_3]$$

as shown in Figures 3.5 and 3.6. This refinement is also referred to as the one-to-four split. Note that all vertices of $\mathcal{T}^i$ for $i > 0$ have valency six except for the boundary vertices and those who correspond to the vertices of $\mathcal{T}^0$ with valency other than six. Apart from the connectivity structure, a good remesh should meet three other criteria.

1. The number of triangles in $\mathcal{T}^0$ shall be as small as possible for this results in a large number j of hierarchy levels in the triangulation $\mathcal{T}' = \mathcal{T}^j$ and assures a maximal utilization of the multiresolution techniques that can be applied to $\mathcal{T}'$.

2. The new triangulation $\mathcal{T}'$ shall be geometrically close to the given triangulation $\mathcal{T}$, in other words, $\Omega_{\mathcal{T}'} \approx \Omega_{\mathcal{T}}$.

3. All triangles in $\mathcal{T}'$ should be about equilateral and uniform in size.

These criteria can easily be fulfill in the plane which is our motivitation for shifting the remeshing problem from three to two dimensions with the help of the parameterization. However, a good planar remesh $\mathcal{S}'$ of $\mathcal{S}$ does not necessarily lead to a good remesh $\mathcal{T}'$ of $\mathcal{T}$ because the parameterization ϕ usually distorts the shape of the triangles. In Section 3.2.2 we discuss how to avoid this problem.

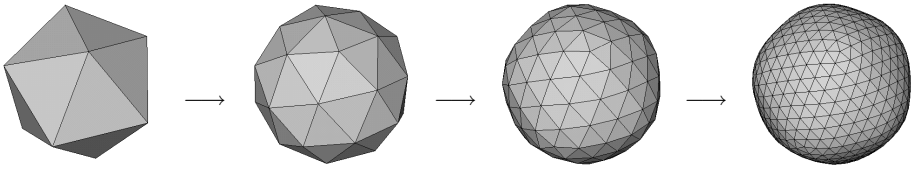


Figure 3.6: A sequence of subdivision connectivity triangulations.

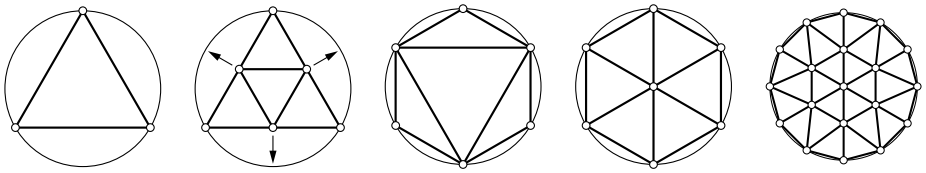


Figure 3.7: Using a base mesh with one (a–c) or six triangles (d–e) for remeshing a disc.

3.2.1 Planar Remeshing

In order to find an appropriate planar remesh $\mathcal{S}'$ of the planar triangulation $\mathcal{S}$ we have to address several problems. Firstly, we have to determine a suitable base mesh $\mathcal{S}^0$ with a small number of triangles which will then be subdivided iteratively to obtain a sequence $\mathcal{S}^0, \mathcal{S}^1, \ldots, \mathcal{S}^j$ of subdivision connectivity triangulations. Secondly, we need a strategy for placing the *boundary* vertices of the refinements $\mathcal{S}^i$ such that they lie on the boundary of $\mathcal{S}$ in order to fulfill the geometric criterion $\Omega_{\mathcal{S}^i} \approx \Omega_{\mathcal{S}}$. And thirdly, the *interior* vertices of each $\mathcal{S}^i$ should be located such that the triangles in $\mathcal{S}^i$ have approximately the same size and are as equilateral as possible.

Since we consider only simple triangulations it is feasible to use just a single triangle as a base mesh $\mathcal{S}^0$. But in most cases this will not lead to a satisfying remesh due to the distortion of the triangles in the subsequent refinements. Consider the situation in Figure 3.7 where the boundary vertices of the planar triangulation $\mathcal{S}$ lie on a circle so that $\Omega_{\mathcal{S}}$ is a polygonal approximation of a disc. Of course, we are able to distribute three vertices uniformly along $\partial\mathcal{S}$ to form a perfectly shaped single triangle base mesh $\mathcal{S}^0$ (a), but after performing a one-to-four split (b) and moving the boundary vertices to the boundary of $\mathcal{S}$ we obtain a refinement $\mathcal{S}^1$ with highly distorted triangles (c). It is therefore much better to use a base mesh $\mathcal{S}^0$ with six triangles arranged as a hexagon (d) in this case, yielding the refined triangulation $\mathcal{S}^1$ in (e).

If the boundary vertices of $\mathcal{S}$ do not lie on a circle but have been placed on a rectangle, found by projection of $\partial\mathcal{T}$, or determined by minimizing the MIPS energy, we propose another way of constructing a base mesh. At first we specify corner vertices among the boundary vertices of $\mathcal{S}$ by an angle criterion similarly to the way described in Section 1.2.5. In other words, we either define those k vertices as corner vertices at which the boundary polygon $\partial\mathcal{S}$ has the k smallest angles or all vertices with an angle below a certain threshold α. If these corner vertices are distributed very irregularly along $\partial\mathcal{S}$ we determine additional vertices so that the distance between all these vertices is about the same.

The set of vertices chosen so far defines the boundary $\partial\mathcal{S}^0$ of the base mesh. We then obtain the base mesh $\mathcal{S}^0$ by triangulating this boundary with Shewchuk's *triangle* program [125]. Based on a Delaunay refinement algorithm by Ruppert [113], this program inserts new interior vertices, called *Steiner*

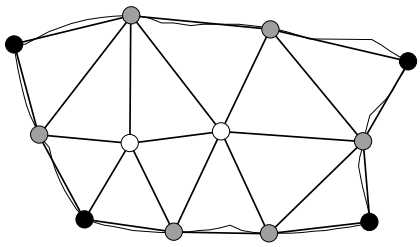


Figure 3.8: Constructing a base mesh for remeshing a triangulation with an arbitrary boundary.

points, to maximize the minimum angle of the triangles in the triangulation. An example is shown in Figure 3.8 where the base mesh construction for the most isometric parameterization (see Figure 1.32.a) of the triangulation in Figure 1.17 is illustrated. The four black vertices were identified as corner vertices, the grey vertices were inserted to adapt the distances between the boundary vertices, and the white interior vertices were added as Steiner points to improve the shape of the triangles.

We can now refine the base mesh $\mathcal{S}^0$ iteratively by applying one-to-four splits to the triangles. Topologically, $\mathcal{S}^{i+1}$ emerges from $\mathcal{S}^i$ by adding new vertices at the midpoint of the edges in $\mathcal{S}^i$ and connecting them as suggested by Figures 3.5 and 3.6. Geometrically, we place these new vertices as follows. New boundary vertices are moved to the boundary of $\mathcal{S}$, namely half way between the neighbouring boundary vertices along $\partial\mathcal{S}$ (see Figure 3.7.b). Finding the positions of the interior vertices $V_I = V_I(\mathcal{S}^{i+1})$ is then considered a linear parameterization problem as in Section 1.2.4. In other words, the vertices $\boldsymbol{v} \in V_I$ are determined by solving the linear system of equations

$$\boldsymbol{v} = \sum_{\boldsymbol{w} \in N_{\boldsymbol{v}}} \lambda_{\boldsymbol{v}\boldsymbol{w}} \boldsymbol{w}, \qquad \boldsymbol{v} \in V_I \tag{3.1}$$

with certain convex weights $\lambda_{\boldsymbol{v}\boldsymbol{w}}$ that sum to one. In order to obtain almost equilateral and uniformly sized triangles we use *uniform* weights

$$\lambda_{\boldsymbol{v}\boldsymbol{w}} = \frac{1}{|N_{\boldsymbol{v}}|}, \qquad \boldsymbol{v} \in V_I, \quad \boldsymbol{w} \in N_{\boldsymbol{v}}$$

so that every interior vertex is located at the barycentre of its neighbours. Figure 3.9 shows an example of planar remeshes computed with this strategy.

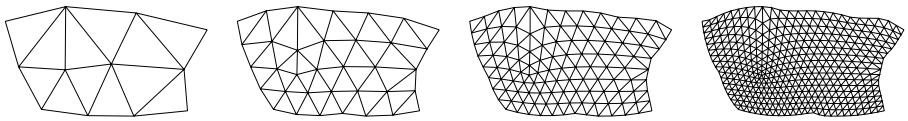


Figure 3.9: A refinement sequence of the base mesh in Figure 3.8.

3.2.2 Spatial Remeshing

After having remeshed the parameter triangulation $\mathcal{S} = \psi(\mathcal{T})$ with a subdivision connectivity triangulation $\mathcal{S}'$ we obtain a remesh $\mathcal{T}'$ of $\mathcal{T}$ by lifting the vertices of $\mathcal{S}'$ with the parameterization ϕ to the surface $\Omega_{\mathcal{T}}$. In this way $\mathcal{T}'$ inherits almost all properties of a good remesh from $\mathcal{S}'$. Firstly, the number of triangles in the spatial base mesh $\mathcal{T}^0$ is as small as the number of triangles in the planar base mesh $\mathcal{S}^0$, ensuring a large number of hierarchy levels in the remesh $\mathcal{T}'$. Secondly, $\mathcal{T}'$ is geometrically close to $\mathcal{T}$ because its vertices are located on the surface $\Omega_{\mathcal{T}}$. It is indeed not hard to verify

$$\lim_{i\to\infty} \Omega_{\mathcal{T}^i} = \Omega_{\mathcal{T}}.$$

But as the parameterization ϕ is not isometric in general, it inevitably modifies the size and the shape of the triangles so that $\mathcal{T}'$ does not inherit the third property of having triangles that are almost equilateral and uniformly sized from $\mathcal{S}'$. Even if we use most isometric parameterizations (see Section 1.3), which are designed to minimize the shape deformation of the triangles, we can still get non-uniformly sized triangles in $\mathcal{T}'$ since the MIPS energy is invariant to scalings. We therefore suggest the following modifications to the planar remeshing strategy in order to obtain good spatial remeshes.

We construct the base mesh $\mathcal{S}^0$ by first identifying corner vertices in the boundary polygon $\partial\mathcal{T}$ rather than $\partial\mathcal{S}$. Then we let the boundary $\partial\mathcal{S}^0$ consist of the images $\psi(\boldsymbol{v}) \in \partial\mathcal{S}$ of these corner vertices $\boldsymbol{v} \in \partial\mathcal{T}$ plus additional vertices if necessary so as to ensure the vertices in $\partial\mathcal{T}^0$ to be regularly distributed along $\partial\mathcal{T}$. After refining $\mathcal{S}^i$ to $\mathcal{S}^{i+1}$ we place the new boundary vertices of $\mathcal{S}^{i+1}$ such that the corresponding boundary vertices of $\mathcal{T}^{i+1}$ are halfway between the neighbouring boundary vertices along $\partial\mathcal{T}$. For determining the interior vertices $V_I = V_I(\mathcal{S}^{i+1})$ we solve again the linear system (3.1) but instead of taking uniform weights we use *area dependent* weights

$$\lambda_{\boldsymbol{vw}} = \frac{a_{\boldsymbol{w}}}{\sum_{\boldsymbol{u}\in N_{\boldsymbol{v}}} a_{\boldsymbol{u}}}, \qquad \boldsymbol{v} \in V_I, \quad \boldsymbol{w} \in N_{\boldsymbol{v}}, \tag{3.2}$$

where

$$a_{\boldsymbol{u}} = \rho_{\boldsymbol{u}} \sum_{[\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}]\in\mathcal{S}^{i+1}} \operatorname{area}\big([\phi(\boldsymbol{u}), \phi(\boldsymbol{v}), \phi(\boldsymbol{w})]\big)$$

is the area of the triangle fan around $\phi(\boldsymbol{u})$ in $\mathcal{T}^{i+1}$ with $\rho_{\boldsymbol{u}} = 1$ for interior vertices $\boldsymbol{u} \in V_I$ and

$$\rho_{\boldsymbol{u}} = \frac{2\pi}{\sum_{[\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}]\in\mathcal{S}^{i+1}} \sphericalangle\left(\phi(\boldsymbol{v}), \phi(\boldsymbol{u}), \phi(\boldsymbol{w})\right)}$$

for boundary vertices $\boldsymbol{u} \in V_B(\mathcal{S}^{i+1})$ thereby extrapolating the area of an open triangle fan to the area of a closed one. Since the area dependent weights $a_{\boldsymbol{u}}$

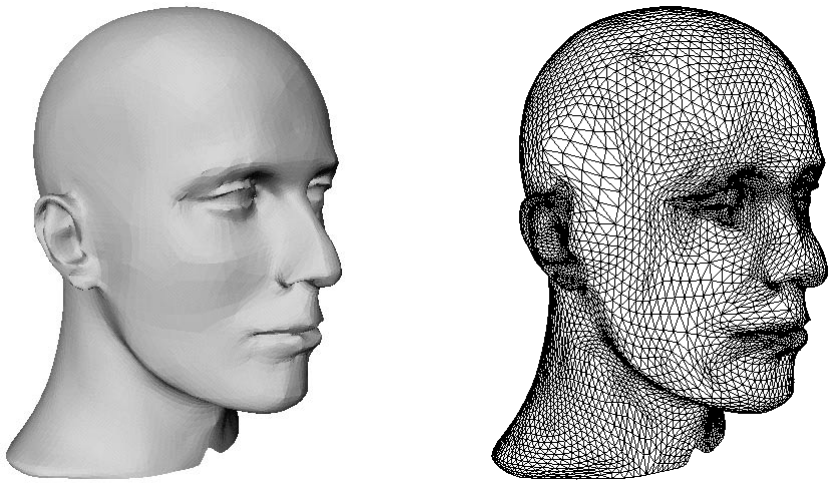
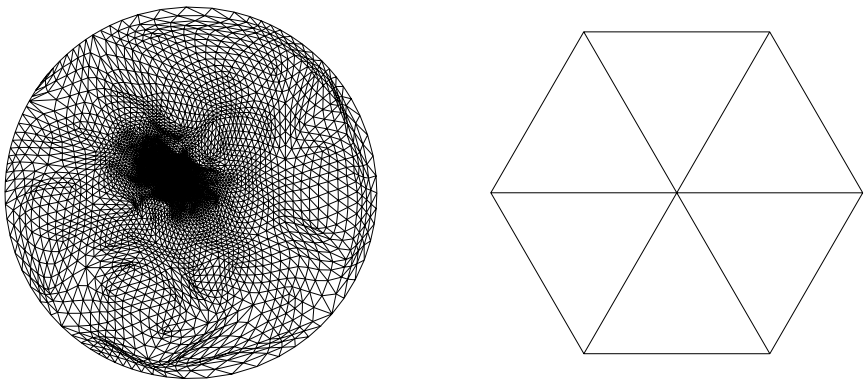


Figure 3.10: Triangulation of a head with 10 833 vertices and 21 680 triangles.

clearly depend on the position of the interior vertices V_I we alternately compute the weights $a_{\boldsymbol{u}}$ and solve (3.1) until numerical convergence, which usually takes a few iterations only. The idea behind these area dependent weights is that the interior vertices of $\mathcal{S}^{i+1}$ are pushed towards those neighbours whose corresponding vertices in $\mathcal{T}^{i+1}$ have the largest triangle fan areas and can be seen as an approach to compose the parameterization ϕ with a reparameterization ϕ' which compensates for the scaling defects in ϕ. As we will see in the following examples this strategy leads to uniformly sized triangles in $\mathcal{T}^{i+1}$ but may distort the triangles considerably at the same time.



(a) (b)

Figure 3.11: Shape preserving parameterization $\mathcal{S}$ (a) of the triangulation in Figure 3.10 and planar base mesh $\mathcal{S}^0$ (b).

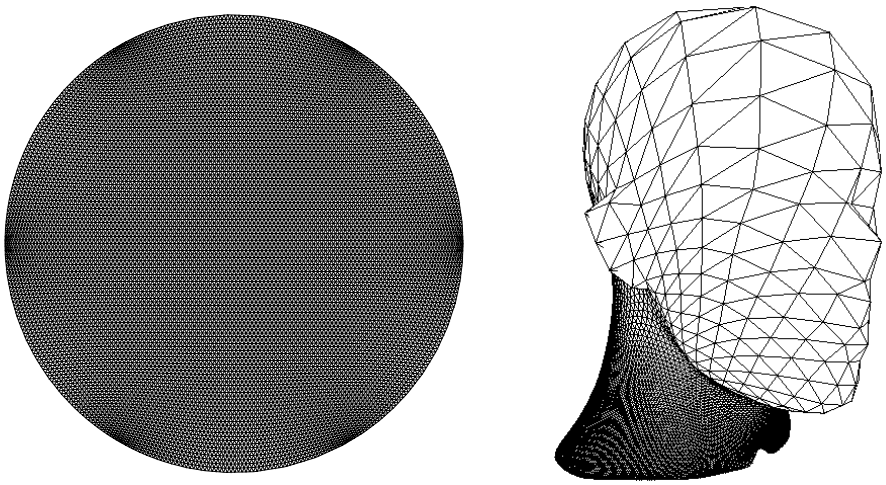


Figure 3.12: Planar remesh $\mathcal{S}^6$ (left) and spatial remesh $\mathcal{T}^6$ (right) of the triangulation in Figure 3.10 using uniform weights.

We parameterized the triangulation in Figure 3.10 by mapping the boundary to a circle and using shape preserving parameterization for the interior vertices. Figure 3.11 shows the result (a) as well as the triangulation that we chose as a base mesh. After 6 refinements, the planar remeshing method with uniform weights leads to the triangulations in Figure 3.12 while Figure 3.13 displays the result for area dependent weights. All remeshes have $6 \cdot 4^6 = 24\,576$ triangles.

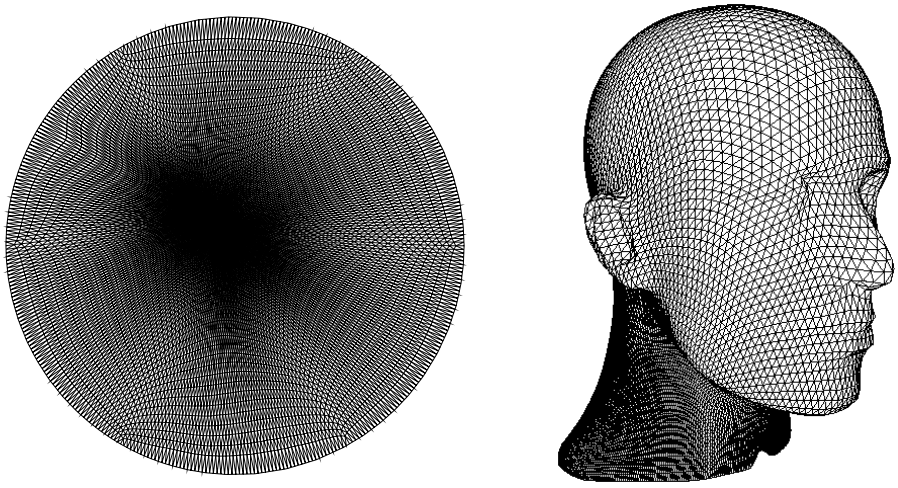


Figure 3.13: Planar remesh $\mathcal{S}^6$ (left) and spatial remesh $\mathcal{T}^6$ (right) of the triangulation in Figure 3.10 using area dependent weights.

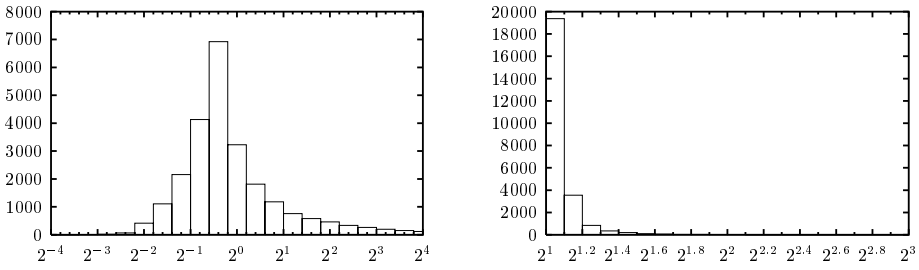


Figure 3.14: Histograms of relative triangle size (left) and shape deformation (right) for the spatial remesh in Figure 3.12.

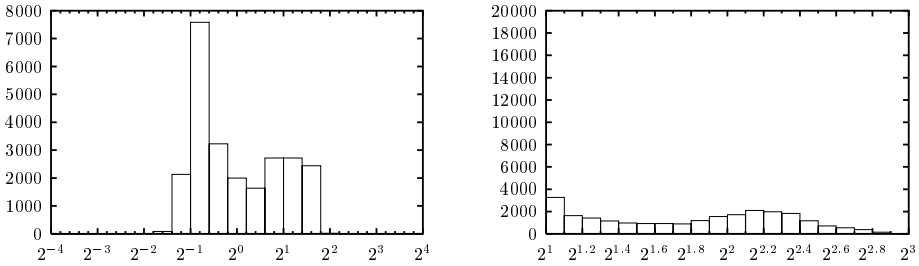


Figure 3.15: Histograms of relative triangle size (left) and shape deformation (right) for the spatial remesh in Figure 3.13.

At first sight we recognize that the triangles in the remesh with uniform weights vary largely in size while the bandwidth of triangle size in the other remesh is much smaller. The histograms in Figures 3.14 and 3.15 underpin this observation but also verify that the triangles are less equilateral in the remesh with area dependent weights. The equilaterality of a triangle T was measured by computing the shape deformation $E_2(\psi)$ (see Equation 1.21 on page 38) of the linear function ψ that maps an equilateral triangle to T. After all, getting almost equilateral triangles on the one hand and uniformly sized triangles on the other are two competing goals of the remeshing process and cannot be fulfilled simultaneously in general.

Let us now consider another example that illustrates how different parameterizations affect the remeshing process. Figure 3.17 shows three parameterizations of the triangulation in Figure 3.16 which we used for computing the triangulations in Figure 3.18. In this example we took area dependent weights and refined a base mesh with 10 triangles 5 times so that the remeshes consist of $10 \cdot 4^5 = 10\,240$ triangles. As we can see from the pictures and the histograms in Figure 3.19, the most isometric (a) and the shape preserving parameterization (b) give very similar results which are clearly better than the one generated with uniform parameterization (c). This confirms the fact that the spatial remesh inherits the properties of a good planar remesh if the parameterization that is used for lifting the vertices keeps the deformation of triangle shape and size low.

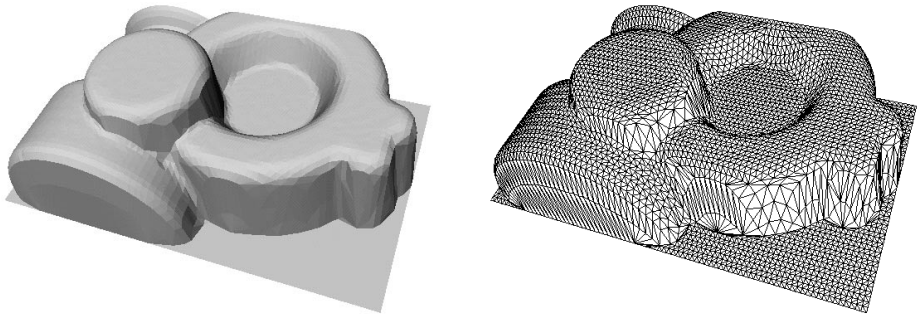
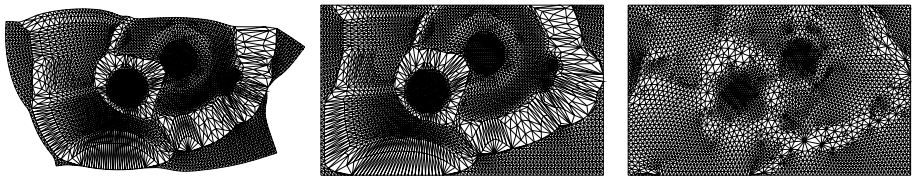
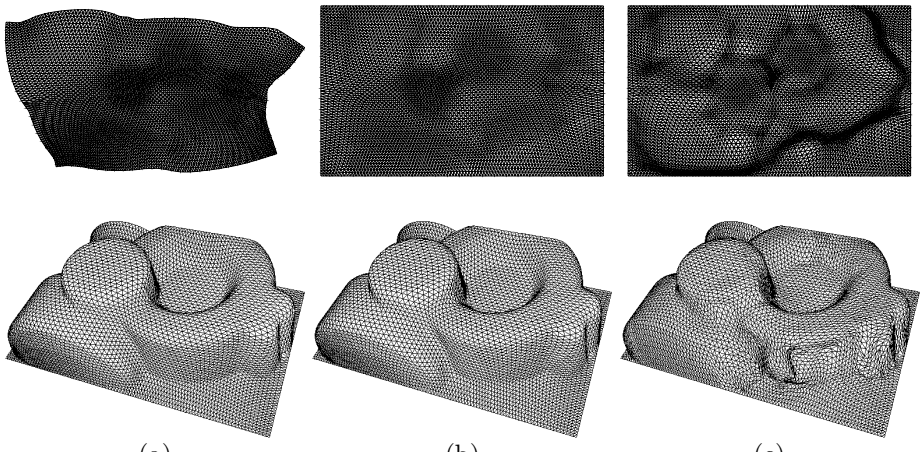


Figure 3.16: Triangulated technical data set with 4 100 vertices and 7 938 triangles.



(a) (b) (c)

Figure 3.17: Most isometric (a), shape preserving (b), and uniform parameterization (c) of the triangulation in Figure 3.16.



(a) (b) (c)

Figure 3.18: Planar (top) and spatial remeshes (bottom) of the triangulation in Figure 3.16 using the parameterizations in Figure 3.17.

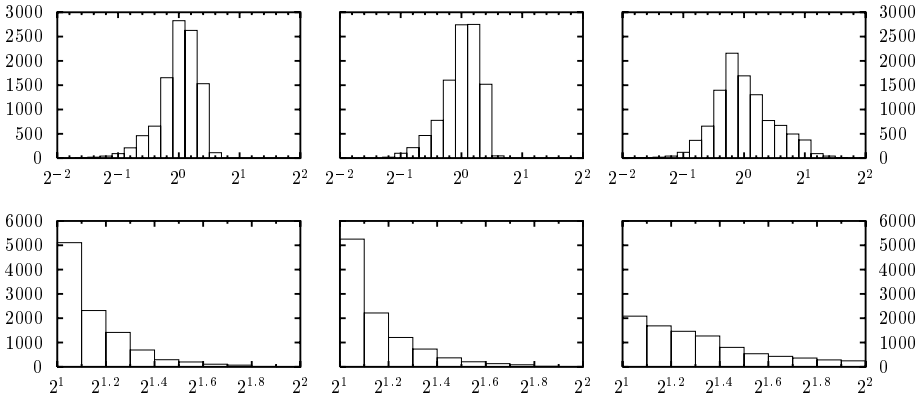


Figure 3.19: Histograms of relative triangle size (top) and shape deformation (bottom) for the spatial remeshes in Figure 3.18.

3.3 Regular Quadrilateral Meshes

The concept of remeshing a triangulation $\mathcal{T}$ with a subdivision connectivity triangulation $\mathcal{T}'$ can also be used for remeshing with other types of regular meshes, for example regular quadrilateral meshes. We call $\mathcal{Q}$ a *regular quadrilateral mesh of dimension* (m, n) if it consists of $(m+1) \cdot (n+1)$ vertices $V = \{\boldsymbol{v}_{r,s} : 0 \leq r \leq m,\ 0 \leq s \leq n\}$ which are arranged in $m+1$ columns and $n+1$ rows. Successive vertices in each row and column are connected by an edge so that the four vertices $\boldsymbol{v}_{r,s}$, $\boldsymbol{v}_{r+1,s}$, $\boldsymbol{v}_{r+1,s+1}$, $\boldsymbol{v}_{r,s+1}$ form a quadrilateral. Like for triangulations we distinguish between interior vertices $V_I = \{\boldsymbol{v}_{r,s} : 0 < r < m,\ 0 < s < n\}$ and boundary vertices $V_B = V \setminus V_I$ and call the four vertices $\boldsymbol{v}_{0,0}$, $\boldsymbol{v}_{m,0}$, $\boldsymbol{v}_{m,n}$, $\boldsymbol{v}_{0,n}$ the *corners* of $\mathcal{Q}$. Note that the corners have exactly two, all other boundary vertices three, and the interior vertices four neighbours. We further say that $\mathcal{Q}$ is of level j if j is the largest integer such that m and n are divisible by 2^j. Then $\mathcal{Q}$ is also the last element of a hierarchical sequence of regular quadrilateral meshes $\mathcal{Q}^0, \mathcal{Q}^1, \ldots, \mathcal{Q}^j = \mathcal{Q}$ where each $\mathcal{Q}^i$ is of level i and consists of the vertices $\{\boldsymbol{v}_{2^{j-i}r, 2^{j-i}s} : 0 \leq r \leq 2^{i-j}m,\ 0 \leq s \leq 2^{i-j}n\}$. Conversely, a regular quadrilateral mesh $\mathcal{Q}^{i+1}$ of level $i+1$ emerges from a regular quadrilateral mesh $\mathcal{Q}^i$ of level i by dyadically refining each quadrilateral $Q = (\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3, \boldsymbol{v}_4)$ in $\mathcal{Q}^i$ into the four subquadrilaterals

$$(\boldsymbol{v}_1, \boldsymbol{w}_1, \boldsymbol{w}_5, \boldsymbol{w}_4), \quad (\boldsymbol{v}_2, \boldsymbol{w}_2, \boldsymbol{w}_5, \boldsymbol{w}_1), \quad (\boldsymbol{v}_3, \boldsymbol{w}_3, \boldsymbol{w}_5, \boldsymbol{w}_2), \quad (\boldsymbol{v}_4, \boldsymbol{w}_4, \boldsymbol{w}_5, \boldsymbol{w}_3)$$

as shown in Figure 3.20. An example of a regular quadrilateral mesh of dimension $(8, 6)$ and level 1 is shown in Figure 3.1.b.

The objectives for remeshing with regular quadrilateral meshes are similar to the ones for remeshing with subdivision connectivity triangulations. We want a base mesh of small dimension and a remesh that is geometrically close to the

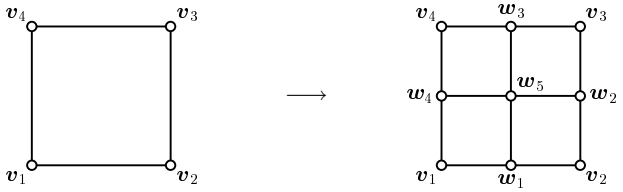


Figure 3.20: Dyadic refinement of a quadrilateral.

given triangulation with quadrilaterals that are about square in shape and uniform in size. Such remeshes can be created in the same way as discussed in the previous section. We start by finding four corner vertices $\boldsymbol{v}$ in the boundary polygon $\partial\mathcal{T}$ and let their images $\psi(\boldsymbol{v}) \in \partial\mathcal{S}$ be the corners of a planar base mesh $\mathcal{Q}^0$. Again, we may add vertices to the boundary $\partial\mathcal{Q}^0$ to ensure a regular distribution of boundary vertices and thus almost square and uniformly sized quadrilaterals in $\mathcal{Q}$. After refining $\mathcal{Q}^i$ to $\mathcal{Q}^{i+1}$ we place the new boundary vertices of $\mathcal{Q}^{i+1}$ halfway along the boundary between their neighbouring vertices. The interior vertices are then determined by solving the linear system (3.1) with either uniform weights $\lambda_{\boldsymbol{vw}} = \frac{1}{4}$ or area dependent weights as in (3.2). We also define $a_{\boldsymbol{u}}$ to be the area of the triangle fan formed by $\phi(\boldsymbol{u})$ and its neighbours $\phi(\boldsymbol{v})$, $\boldsymbol{v} \in N_{\boldsymbol{u}}$, multiplied by the extrapolation factor $\rho_{\boldsymbol{u}} = 4$ at the corners of $\mathcal{Q}$, $\rho_{\boldsymbol{u}} = 2$ for other boundary vertices, and $\rho_{\boldsymbol{u}} = 1$ for interior vertices. Finally, we define the spatial remesh $\mathcal{Q}'$ of $\mathcal{T}$ by lifting the vertices of the planar remesh $\mathcal{Q}$, in other words, we let $\mathcal{Q}'$ consist of the vertices $\phi(V(\mathcal{Q}))$.

Figure 3.21 shows a regular quadrilateral remesh of the triangulation in Figure 3.16 that was computed as explained above. In this example we took area dependent weights and refined a base mesh of dimension $(5, 3)$ four times yielding remeshes of dimension $(80, 48)$. Another example is illustrated in Figure 3.23 where we parameterized the triangulation in Figure 3.22 to a square using shape preserving parameterization and took the square itself as a base mesh of dimension $(1, 1)$. The remeshes are of dimension $(32, 32)$ and level 5.

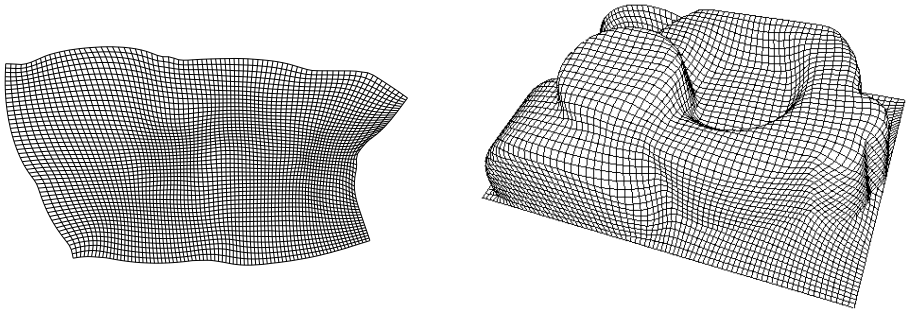


Figure 3.21: Planar (left) and spatial quadrilateral remesh (right) of the triangulation in Figure 3.16.

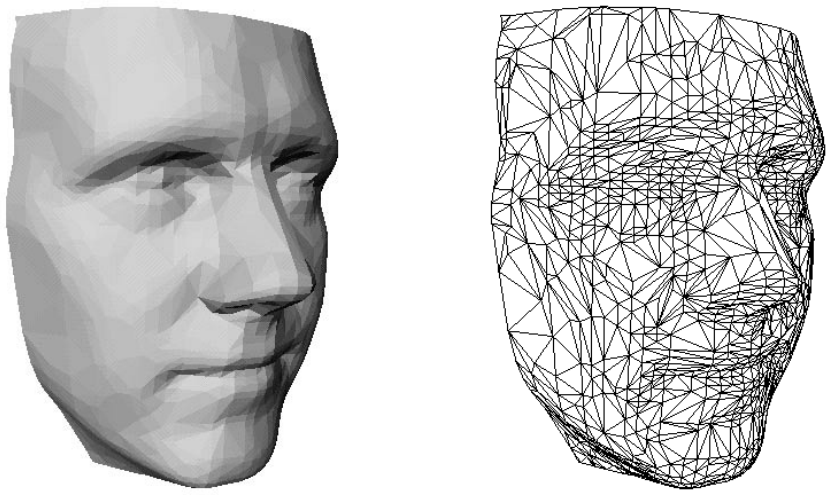


Figure 3.22: Triangulation of a face with 1 042 vertices and 1 999 triangles.

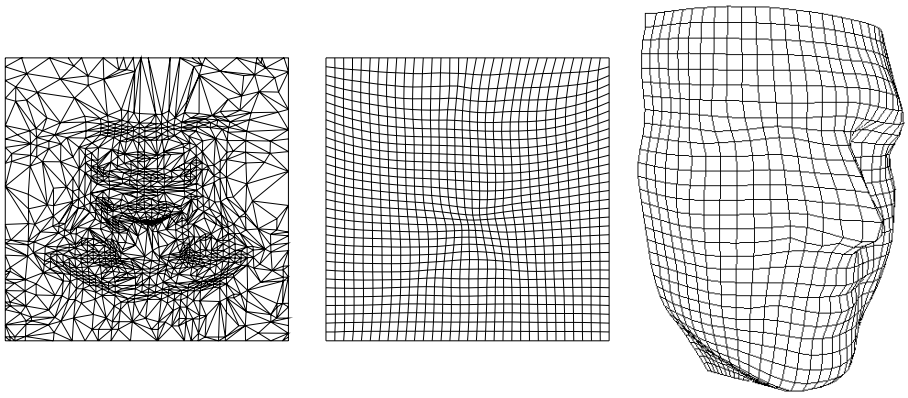


Figure 3.23: Shape preserving parameterization (left), planar (middle), and spatial remesh (right) of the triangulation in Figure 3.22.

Quam quidem propositionem in construendis
navibus non inutilem futuram esse cenceo.
Isaac Newton (1643–1727)

Chapter 4

Fitting Smooth Surfaces

The problem of reconstructing smooth surfaces from discrete scattered data arises in many fields of science and engineering and has been studied thoroughly for nearly 40 years. The data sources include measured values (meteorology, oceanography, optics, geology, laser range scanning, etc.) as well as experimental results (from physical, chemical or engineering experiments) and computational values (evaluation of mathematical functions, finite element solutions, or results of other numerical simulations). Due to the vast variety of data sources many different methods have been developed, each of them more or less suited to a specific problem.

After a brief survey of the related work in Section 4.1 we shall review several methods for solving the parametric surface fitting problem in the remaining sections of this chapter. Although many of the methods dicussed generalize to arbitrary linear spaces of functions we shall focus on tensor product B-splines, because the special structure of the B-spline basis functions leads to very efficient algorithms. After giving an introduction to this type of surfaces in Section 4.2 we discuss the variational approach to smooth surface fitting in Section 4.3 before presenting another more efficient technique in Section 4.4.

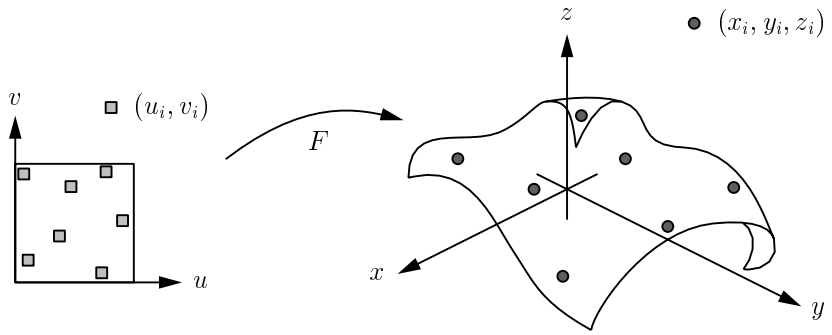


Figure 4.1: The parametric surface fitting problem.

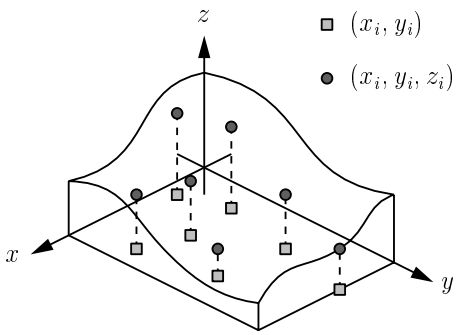


Figure 4.2: Scattered Data Interpolation.

4.1 Related Work

In the field of geology, meteorology, cartography, for example, the problem can typically be stated as follows: given data points $(x_i, y_i, z_i) \in \mathbb{R}^3$, find a scalar function $F : \Omega \to \mathbb{R}$, $\Omega \subset \mathbb{R}^2$ that approximates or interpolates the value z_i at (x_i, y_i), $F(x_i, y_i) \approx z_i$. This problem is generally known as *Scattered Data Interpolation* (see Figure 4.2) and there exist many solutions to that problem which include Shepard's method [124], radial basis functions [63] and finite element methods. Good surveys of these methods and further references can be found in [118, 51, 101].

In contrast to this *scalar problem* there is the *parametric problem*, where the task is to find a parameterized surface $F : \Omega \to \mathbb{R}^3$ that approximates or interpolates the data points. This is usually done by specifying additional parameter values $(u_i, v_i) \in \Omega$ and determining F with $F(u_i, v_i) \approx (x_i, y_i, z_i)$ (see Figure 4.1). While the theory of parametric surfaces was well understood in differential geometry, dating back to the time of Gauß [52, 30], their potential for the representation of surfaces in the field of engineering remained unknown for a long time. "The exploration of the use of parametric curves and surfaces can be viewed as the origin of *Computer Aided Geometric Design* (CAGD)" [39, p. xv].

The fundamental ideas in CAGD were developed in the French and US-American car industry in the 1960ies. While de Casteljau at Citroën and Bézier at Renault independently developed the theory of Bézier curves and surfaces, Coons at Ford and Gordon at General Motors worked on *transfinite interpolation methods*, which can be used to 'fill in' a network of curves. At the same time *mathematical splines*, introduced by Schoenberg in the 1940ies, were used by Ferguson at Boeing and Sabin at the British Aircraft Corporation.

Combining the ideas of Bézier curves and splines naturally leads to an efficient and numerically stable representation of B-splines and further on to non-uniform rational B-splines (NURBS), which is the de-facto standard of today's CAD systems. Detailed information about the theoretical and practical aspects of B-splines and NURBS can be found in [23, 128, 39, 122, 26].

4.2 Tensor Product B-Spline Surfaces

In the sections to follow we talk about B-spline surfaces with the tacit understanding that they are uniform and bicubic. A brief review explains how these surfaces are defined and why they are well suited for our purposes.

Given the *knot vector* $T = (t_0, \ldots, t_{m+k+1})$ with $t_i < t_{i+1}$, the *normalized B-spline functions* $N_i^k = N_i^{T,k}$ of degree k over T are defined by the following recursion:

$$N_i^0(u) = \begin{cases} 1, & \text{if } u \in [t_i, t_{i+1}[, \\ 0, & \text{else,} \end{cases}$$

$$N_i^j(u) = \frac{u - t_i}{t_{i+j} - t_i} N_i^{j-1}(u) + \frac{t_{i+j+1} - u}{t_{i+j+1} - t_{i+1}} N_{i+1}^{j-1}(u), \qquad j > 0.$$

The so defined functions N_i^k are polynomial of degree k on each interval $[t_i, t_{i+1}[$, form a basis[1] of all functions that are piecewise polynomial of degree k over $[t_k, t_{m+1}]$, and have a number of interesting properties:

1. *Continuity:* N_i^k is $\mathcal{C}^{k-1}$-continuous,
2. *Positivity:* $N_i^k(u) \geq 0$,
3. *Local support:* $N_i^k(u) = 0$, $u \notin [t_i, t_{i+k+1}[$,
4. *Partition of unity:* $\sum_{i=0}^{m} N_i^k(u) = 1$.

The B-spline functions are called *uniform* if the knot vector is equidistantly spaced, $t_{i+1} - t_i = h$. A d-dimensional *B-spline curve* $C : [t_k, t_{m+1}] \to \mathbb{R}^d$ over T is defined as

$$C(u) = \sum_{i=0}^{m} c_i N_i^k(u)$$

with *control points* or *de Boor points* $c_i \in \mathbb{R}^d$. The most important features of B-spline curves are

1. *Local control:* changing the control point c_i does not change the curve $C(u)$ for $u \notin [t_i, t_{i+k+1}[$,
2. *Local convex hull property:* for $u \in [t_i, t_{i+1}[$ the curve $C(u)$ is contained in the convex hull of the control points $c_{i-k}, \ldots, c_i$,
3. *Variation diminishing property:* the number of intersections of an affine hyperplane H with $C(u)$ is less or equal to the number of intersections of H with the control polygon $c_0, \ldots, c_m$,
4. *Affine invariance:* if the control polygon of a B-spline curve is transformed affinely the curve itself is transformed by the same affine transformation,

[1] hence the name B-spline as an abbreviation of basis spline

most of which derive directly from the properties of the basis functions. The proofs of these propositions and many more details about the theory of B-splines can be found in [119, 9, 23, 39, 74, 26].

A typical tensor product approach leads to the definition of *tensor product B-spline surfaces* $F : \Omega \to \mathbb{R}^3$, $\Omega = [t_k, t_{m+1}] \times [s_l, s_{n+1}]$ of degrees k and l over a knot grid $T \times S$:

$$F(u,v) = \sum_{i=0}^{m} \sum_{j=0}^{n} c_{ij} N_i^{T,k}(u) N_j^{S,l}(v)$$

with control points $c_{ij} \in \mathbb{R}^3$. The three-dimensional grid formed by these control points is called the *control mesh*. Analogously to the univariate case, the $(m+1)(n+1)$ basis functions $N_i^{T,k}(u)N_j^{S,l}(v)$ are a basis of all piecewise polynomial functions over Ω and similar properties as for curves can be stated for B-spline surfaces:

1. *Local control:* changing the control point c_{ij} does not change the surface $F(u,v)$ for $(u,v) \notin [t_i, t_{i+k+1}[\times[s_j, s_{j+l+1}[$,
2. *Local convex hull property:* for $(u,v) \in [t_i, t_{i+1}[\times[s_j, s_{j+1}[$ the surface $F(u,v)$ is contained in the convex hull of the control points c_{pq} with $i-k \le p \le i$ and $j-l \le q \le j$,
3. *Affine invariance:* if the control mesh of a B-spline surface is transformed affinely the surface itself is transformed by the same affine transformation.

The reason for using bicubic B-spline surfaces ($k = l = 3$) is that they are sufficiently often (i.e. twice) differentiable on the one hand but relatively cheap to compute on the other. Usually, B-spline curves and surface are evaluated with the *de Boor algorithm* but for uniform knot vectors a more efficient method can be applied. For a uniform knot vector $T = (t_0, t_0 + h, t_0 + 2h, \dots)$ the B-spline functions differ by translation only, $N_{i+j}^{T,k}(u) = N_i^{T,k}(u - hj)$ and evaluation of any basis function can be reduced to evaluating the B-spline function N_0^k defined on the integer knot vector, $N_j^{T,k}(u) = N_0^k(\frac{u-t_0}{h} - j)$. For uniform, cubic B-splines it is therefore advisable to use the explicit piecewise polynomial representation of N_0^3 over $(0,1,2,3,4)$,

$$N_0^3(u) = \begin{cases} \frac{1}{6}u^3, & u \in [0,1[, \\ -\frac{1}{2}u^3 + 2u^2 - 2u + \frac{2}{3}, & u \in [1,2], \\ N_0^3(4-u), & u \in]2,4[, \\ 0, & \text{else}, \end{cases}$$

and *Horner's method* to evaluate curves and surfaces. Note that first and second order derivatives of a uniform, bicubic tensor product B-spline surface can be computed efficiently in the same way.

Figure 4.3: The configuration to the left is suitable for interpolation while approximation should be used for the noisy data points to the right.

4.3 The Variational Approach

The process of reconstructing smooth surfaces from discrete data generally offers two possibilities: interpolation and approximation. Since the solution of the interpolation problem coincides with the data points one might assume it to be the more accurate reconstruction. But if the samples carry some noise (due to measurement errors, for example) an approximating solution is more adequate. Therefore, the choice of the appropriate method depends on the specific structure of the problem (see Figure 4.3).

In both cases we have to specify a parameterization that assigns a parameter point p_i to each data point P_i so as to be able to formulate the interpolation conditions or the approximation error. In Chapters 1 and 2 we already discussed several methods for finding such a parameterization. After briefly mentioning how to optimize the parameter points in Section 4.3.1 we study the interpolation and the least squares approximation problem in Section 4.3.2 and show how to extend the concepts in order to find smooth reconstruction surfaces. A summary of different smoothness functionals is given in Section 4.3.3. We further discuss how the approximation problem can be optimized in three ways. Firstly, the use of hierarchical surfaces is reviewed in Section 4.3.4. Secondly, we show how to solve the Chebychev approximation problem in Section 4.3.5 and thirdly, an efficient iterative linear method for reducing the maximum approximation error is presented in Section 4.3.6.

4.3.1 Parameter Correction

Once an approximating reconstruction surface $F : \Omega \to \mathbb{R}^3$ is found, we can improve the parametrization, thereby exploiting the additional information given by the approximating surface itself [73, 115]. The standard method as introduced by Hoschek in [73] is derived from the observation that the error vector $Q = P - F(p)$ will not be orthogonal with respect to the approximating surface in general. Hence it does not represent the minimal distance between the data point P and the corresponding point $F(p)$ on the approximating surface and the parameter value p should be corrected (see Figure 4.4).

This is done by linearly approximating the surface at $F(p)$ and projecting the data point P orthogonally onto this tangent plane. The correction term $\Delta p = (\Delta u, \Delta v)$ can be determined by solving $L = F(p) + \Delta u\, F_u + \Delta v\, F_v$

Figure 4.4: Distance between a data point and the corresponding point on the approximating surface before (left) and after (right) parameter correction.

(see Figure 4.5), which can be rewritten as

$$\Delta p = \begin{pmatrix} F_u^2 & F_u F_v \\ F_u F_v & F_v^2 \end{pmatrix}^{-1} \begin{pmatrix} \langle\, Q \,|\, F_u \,\rangle \\ \langle\, Q \,|\, F_v \,\rangle \end{pmatrix}.$$

The approximation process will then be repeated with the improved parameter values $\tilde{p} = p + \Delta p$. The underlying idea of this method is to split the non-linear approximation problem, in which both the surface parameters and the parameter values are unknowns, into two steps:

1. The *approximation step*, which finds the optimal surface parameters for given parameter values.

2. The *parameter correction step*, which determines the optimal parameter values while the surface parameters are fixed.

Solving these problems alternately leads to good results after a few iterations only. Examples are shown in Figure 4.11 and Table 4.3 in Section 4.3.3 and Figure 4.20 and Table 4.7 in Section 4.3.6.

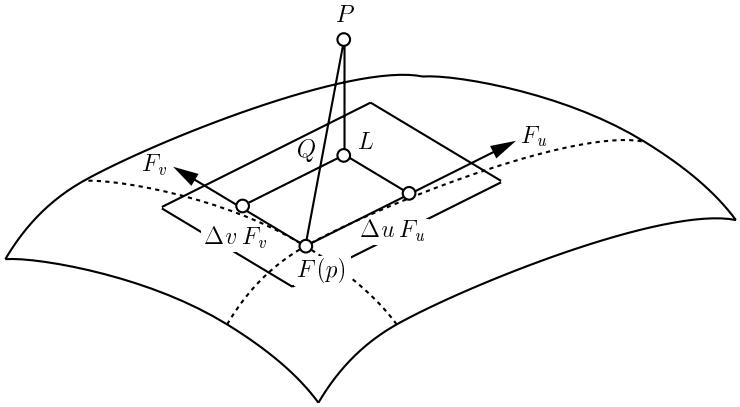


Figure 4.5: Parameter correction for surfaces.

4.3.2 Interpolation and Least Squares Approximation

The interpolation problem can be stated as follows: given n data points $P_i = (x_i, y_i, z_i) \in \mathbb{R}^3$ with corresponding parameter values $p_i = (u_i, v_i) \in \Omega$ and some class S of parameterized surfaces $F : \Omega \to \mathbb{R}^3$, find $F \in S$ such that $F(p_i) = (P_i)$ for all i. The linear space S may consist of bivariate polynomials, tensor product B-splines or NURBS, piecewise Bézier patches, or triangular splines. Suppose S to be spanned by k basis functions B_j, $S = \text{span}\{B_j\}_j$, then the interpolation function $F = \sum_j c_j B_j$ satisfies $F(p_i) = \sum_j c_j B_j(p_i) = P_i$ for all i, hence the unknowns $c_1, \ldots, c_k$ can be found by solving the linear system

$$B\boldsymbol{c} = \boldsymbol{P}$$

with

$$B = \begin{pmatrix} B_1(p_1) & \cdots & B_k(p_1) \\ \vdots & \ddots & \vdots \\ B_1(p_n) & \cdots & B_k(p_n) \end{pmatrix}, \quad \boldsymbol{c} = \begin{pmatrix} c_1 \\ \vdots \\ c_k \end{pmatrix}, \quad \boldsymbol{P} = \begin{pmatrix} P_1 \\ \vdots \\ P_n \end{pmatrix}. \tag{4.1}$$

Note that this problem will not be solvable in general if $k < n$ and that the existence of a unique solution requires $k = n$.

In case of curve interpolation with polynomials, B will be a *Vandermonde-Matrix*, that is regular for mutually different parameter values p_i. An interpolating B-spline curve can be found if and only if the *Schoenberg-Whitney* conditions are fulfilled [23, 39, 26]. These results can be extended to surface interpolation with polynomials or tensor product B-splines if the parameter values p_i are distributed on a grid. The problem then decouples into a sequence of curve interpolation problems [49, 95] (see Section 4.4).

However, if the parameter values are not gridded, the problem becomes more complicated. For example, a biquadratic polynomial ($k = 6$) that interpolates $n = 6$ given data points cannot be found if the corresponding 6 parameter values lie on an algebraic curve of degree 2 (circles, ellipses, parabolas and hyperbolas), although $k = n$. Another problem is that the interpolating surfaces might have unwanted oscillations.

To overcome these difficulties, a variational approach can be used for solving the interpolation problem. The idea is to start with a class of surfaces having more degrees of freedom than necessary to fulfill the interpolation conditions and to use the remaining degrees of freedom to smooth the surface [55, 56]. This is achieved by minimizing a functional that measures the smoothness (see Section 4.3.3).

If $\mathcal{J} : S \to \mathbb{R}$ is such a smoothness functional, the task now is to find $F \in S$ such that

1. $F(p_i) = (P_i)$ for all i and
2. $\mathcal{J}(F) \leq \mathcal{J}(G)$ for any $G \in S$ satisfying 1.

Many reasonable measures of smoothness can be expressed by a quadratic and positive semidefinite functional, $\mathcal{J}(F) = \langle F \,|\, F \rangle_{\mathcal{J}}$ for a suitable positive

semidefinite, symmetric, bilinear form $\langle \cdot \,|\, \cdot \rangle_{\mathcal{J}}$. By introducing *Lagrangian multipliers* λ_i, the smooth interpolation function $F = \sum_j c_j B_j$ can be found by solving the linear system

$$\begin{pmatrix} A & B^t \\ B & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{c} \\ \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} \boldsymbol{0} \\ \boldsymbol{P} \end{pmatrix} \tag{4.2}$$

with

$$A = \begin{pmatrix} \langle B_1 \,|\, B_1 \rangle_{\mathcal{J}} & \cdots & \langle B_1 \,|\, B_k \rangle_{\mathcal{J}} \\ \vdots & \ddots & \vdots \\ \langle B_k \,|\, B_1 \rangle_{\mathcal{J}} & \cdots & \langle B_k \,|\, B_k \rangle_{\mathcal{J}} \end{pmatrix}, \quad \boldsymbol{\lambda} = \begin{pmatrix} \lambda_1 \\ \vdots \\ \lambda_n \end{pmatrix},$$

and $B, \boldsymbol{c}, \boldsymbol{P}$ as defined in (4.1). An efficient way of solving (4.2) was presented in [112].

However, interpolation is not always the best method for surface fitting. If the data points are not exact it is advisable to give up interpolation and better look for an approximating surface. The most common approach to this problem is the classical method of *least squares*, whereby the approximation error in the (squared) ℓ_2 norm

$$\mathcal{E}_2(F) = \sum_{i=1}^{n} |F(p_i) - P_i|^2 = \|B\boldsymbol{c} - \boldsymbol{P}\|_2^2 \tag{4.3}$$

is minimized. With the definitions from above the ℓ_2 error can be rewritten as

$$\mathcal{E}_2(F) = (B\boldsymbol{c} - \boldsymbol{P})^t (B\boldsymbol{c} - \boldsymbol{P}) = \boldsymbol{c}^t B^t B \boldsymbol{c} - 2\boldsymbol{c}^t B^t \boldsymbol{P} + \boldsymbol{P}^t \boldsymbol{P}$$

and the minimum can be found by solving the *normal equation*

$$B^t B \boldsymbol{c} = B^t \boldsymbol{P}. \tag{4.4}$$

Note that the matrix $M = B^t B$ is symmetric, positive semidefinite. In case of approximation with B-spline curves or surfaces, M is also sparse and has a band or multi-band structure. While the *Schoenberg-Whitney* condition is sufficient for the solvability of the normal equation in the univariate case as it guarantees the linear independence of B's column vectors, such a simple condition does not exist in the bivariate setting unless the given data is gridded so that the uniqueness of the solution cannot be guaranteed for scattered data approximation.

But even if the normal equation has a unique solution, the same problem as with interpolating surfaces may occur: the surface tends to oscillate (see Figure 4.7.f). In such cases it is better to minimize a combination of the error functional $\mathcal{E}_2$ and a smoothness functional $\mathcal{J}$,

$$\mathcal{K}_\mu(F) = \mu \mathcal{J}(F) + \mathcal{E}_2(F), \tag{4.5}$$

with a *smoothing parameter* $\mu \geq 0$ that controls the tradeoff between smoothness and approximation quality of the surface. Using the definitions from above the

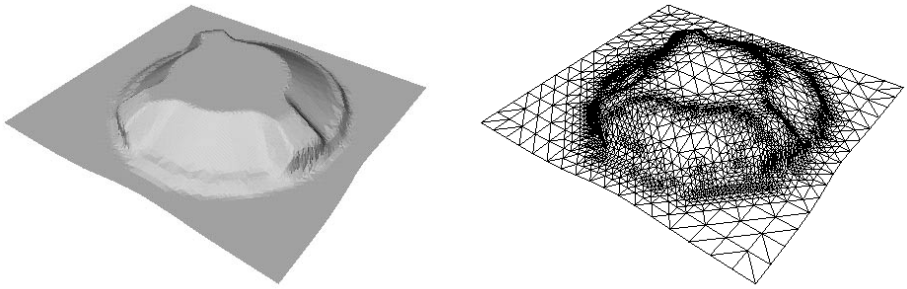


Figure 4.6: Triangulated data set of a tank cap with 3 374 data points.

minimum of this combined functional $\mathcal{K}_\mu$ can be found by solving

$$(\mu A + B^t B)\boldsymbol{c} = B^t \boldsymbol{P}. \tag{4.6}$$

Assuming $\mathcal{J}$ to be a symmetric positive definite smoothness functional, (4.6) always has a unique solution, say F_μ, for which some interesting properties hold [136, 67].

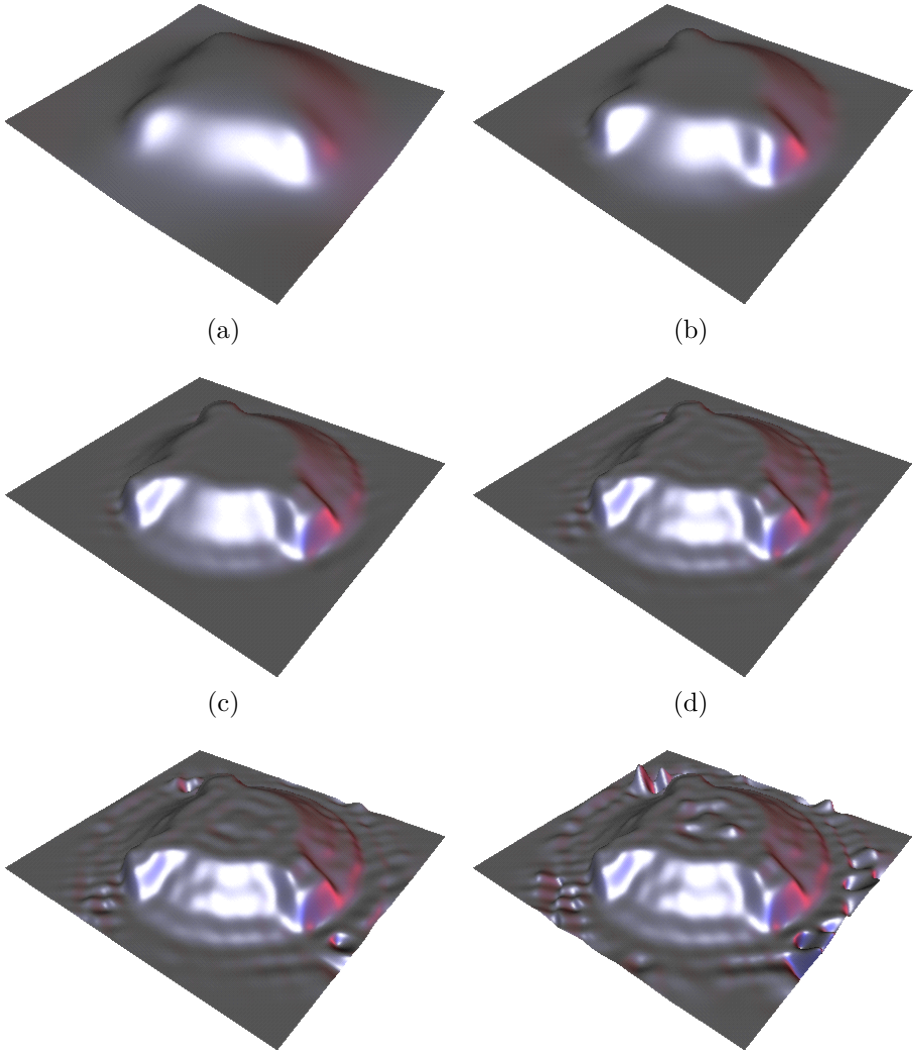
1. The function $\mu \to \mathcal{E}_2(F_\mu)$ is monotonically increasing,
2. The function $\mu \to \mathcal{J}(F_\mu)$ is monotonically decreasing,
3. $F_0 = \lim_{\mu \to 0} F_\mu$ exists, and
4. F_0 minimizes $\mathcal{E}_2$ and is among all minima of $\mathcal{E}_2$ the one that additionally minimizes $\mathcal{J}$.

The first and second property support the statements on the effect of the smoothing parameter μ: the smaller μ is chosen, the smaller the approximation error is, while the smoothness of the surface increases with larger μ. An immediate consequence of the last property is that if there exist interpolating surfaces in S, then F_0 is the interpolant with optimal smoothness. Thus, it will also be the solution to (4.2).

Such penalized least squares methods have been studied thoroughly by von Golitschek and Schumaker in [136]. The authors also describe a method for automatically adjusting the smoothing parameter μ but their choice is very costly to compute. Floater [43] suggests to set

$$\mu^* = \frac{\|B^t B\|}{\|A\|}$$

for some matrix norm $\|\cdot\|$, such that $\mu^* \mathcal{J}$ and $\mathcal{E}_2$ contribute equally to $\mathcal{K}_{\mu^*}$. For the *Frobenius Norm* $\|A\|_F = \sqrt{\sum_{i,j} a_{ij}^2}$ this is an inexpensive and good choice. We propose to take this μ^* as an initial smoothing parameter and iteratively decrease it until a user-specified error tolerance is met.



(a) (b) (c) (d) (e) (f)

Figure 4.7: Surface reconstruction of the data set in Figure 4.6 for different smoothing parameters.

	μ	error in %		mean curvature		
		max	average	min	max	average
(a)	10^{-2}	1.703330	0.536520	−0.022074	0.009693	0.005886
(b)	10^{-3}	0.888876	0.252356	−0.040364	0.042386	0.010124
(c)	10^{-4}	0.755660	0.139964	−0.073339	0.103838	0.014599
(d)	10^{-5}	0.728642	0.119589	−0.092256	0.155166	0.017927
(e)	10^{-6}	0.727063	0.116882	−0.168611	0.263485	0.019536
(f)	10^{-7}	0.687708	0.114353	−0.484673	0.377598	0.020681

Table 4.1: Comparison of the different surfaces in Figure 4.7. The maximum error, $\max_{i=1,\dots,n} |F(p_i) - P_i|$, and the average ℓ_2 error $\sqrt{\frac{1}{n}\mathcal{E}_2(F)}$ are given in proportion to the length of the diagonal of the axis aligned bounding box that encompasses the data set. The curvature was evaluated on a regular $1\,000 \times 1\,000$ grid; the average corresponds to the arithmetic mean of the values.

Figure 4.7 and Table 4.1 show how the smoothing parameter affects the approximating surface. We reconstructed the data set in Figure 4.6 using MIPS (see Section 1.3) for parameterizing the data points, the simplified thin plate energy (4.8) as smoothness functional $\mathcal{J}$ and a 30×30 tensor product B-spline as reconstruction surface. In this example we have $\|B^t B\|_F = 67.7976$ and $\|A\|_F = 73\,005.19$, hence $\mu^* \approx 10^{-3}$ which relates to the surface shown in Figure 4.7.b. Table 4.1 proves the theoretical assertion that decreasing μ reduces the average approximation error. But Figure 4.7 also shows the unwanted oscillations to occur in insufficiently dense sampled areas the closer we get to the pure least squares solution without any smoothing.

In order to avoid these oscillations while further improving the approximation quality more sophisticated methods are needed. One solution is the use of non-uniform smoothing proposed by Dietz in [27] where the constant smoothing parameter μ is replaced by a function $\mu : \Omega \to \mathbb{R}$. This enables to enforce more smoothing in those areas where the surface tends to oscillate and can improve the results significantly, especially for irregularly distributed data. However, the additional costs of this technique are immense and two other approaches are discussed in Sections 4.3.4 and 4.3.6.

Figure 4.8 and Table 4.2 illustrate the effect of choosing different parameterizations for the reconstruction process. The data set in Figure 1.17.a was reconstructed with 23×15 tensor product B-splines using the simplified thin plate energy and $\frac{1}{10}\mu^*$ as smoothing parameter. This example clearly shows that the surface quality depends on the distortion introduced by the parameterization and that shape preserving or most isometric parameterization should be used in order to obtain good results. Note that the surface reconstructed with MIPS was trimmed with the boundary of the parameterization and is therefore controlled by 321 control points only as opposed to the $23 \cdot 15 = 345$ coefficients that describe the other results.

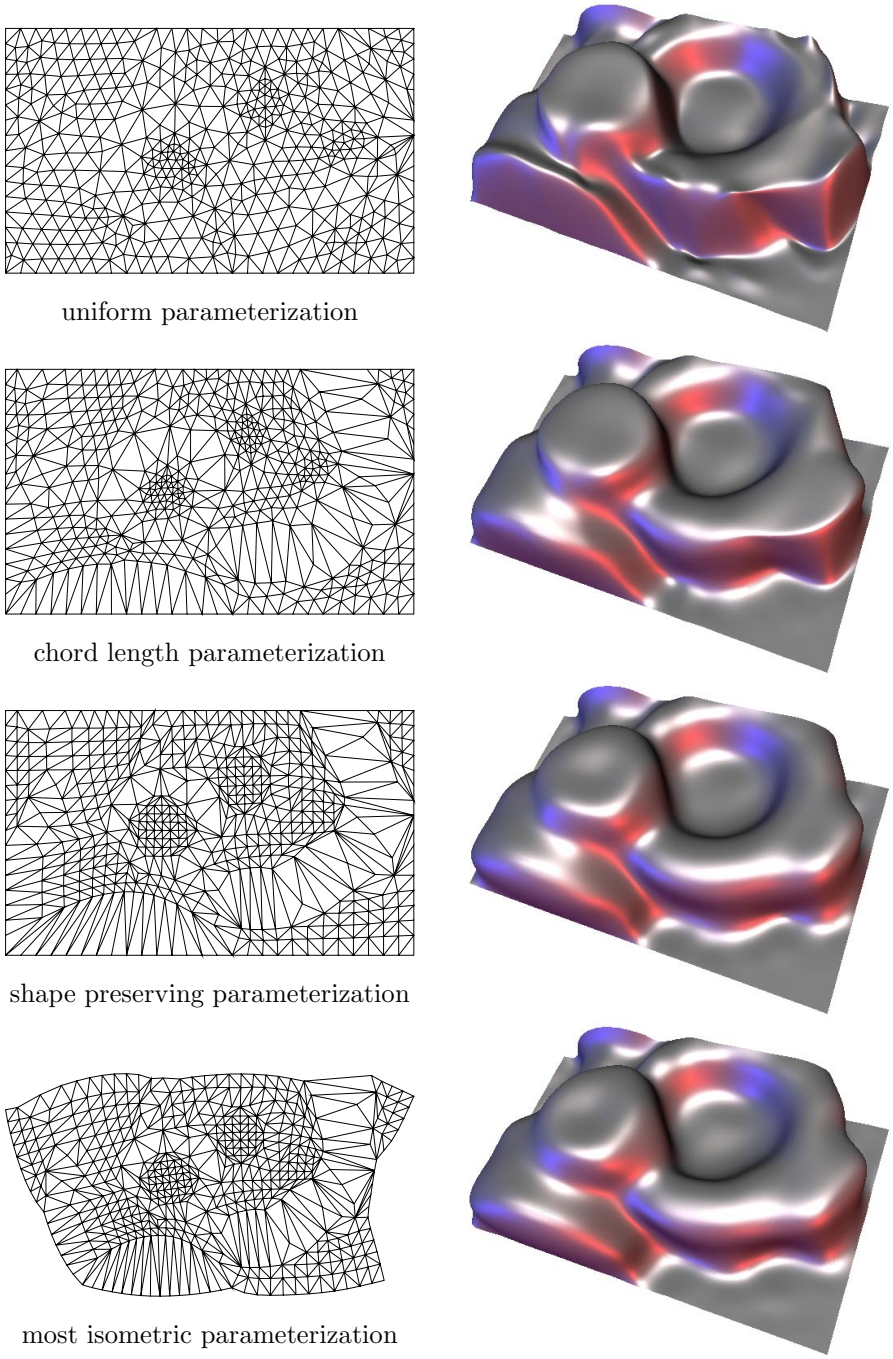


Figure 4.8: Surface reconstruction of the data set in Figure 1.17.a on page 27 using different parameterizations.

	error in %		mean curvature		
	max	average	min	max	average
(a)	2.001594	0.437398	−2.447136	1.815809	0.090254
(b)	1.103259	0.262610	−0.682774	0.489944	0.064370
(c)	1.082219	0.285898	−0.245731	0.418965	0.062680
(d)	1.090881	0.281849	−0.264913	0.438860	0.062929

Table 4.2: Comparison of the different surfaces in Figure 4.8.

4.3.3 Smoothness Functionals

The construction of 'smooth' or 'visually pleasing' surfaces is of vital importance in many areas of geometric modelling, especially in industrial design and styling [114]. While the human eye can easily rate the quality of a surface, the translation of this rating strategy to mathematical formulas is a crucial step. A lot of work has been dedicated to this problem and lots of different approaches how to measure the quality of a surface have been proposed by different authors.

In principal there are two approaches: one can either construct a functional by physical analogy (e.g. minimizing the energy of thin plates) or by geometric reasoning (e.g. minimizing area, curvature or the variation of curvature). But since every reasonable physical quantity has to be a generic property of the surface, it does not depend on the special parameterization that is used to describe the surface. This is, by definition, a geometric property and therefore every physical quantity is a geometric one.

When selecting a smoothness functional we have to consider two aspects.

1. Does the functional yield surfaces with a pleasant shape?
2. Can the functional be minimized in a reasonable amount of time?

It turns out that the functionals involving surface curvature yield surfaces of high quality, but the computation of the solutions is very expensive. The *thin plate energy functional*

$$\int_\Omega a(\kappa_1^2 + \kappa_2^2) + 2(1-b)\kappa_1\kappa_2 \, d\omega$$

that describes the energy of a thin plate [22] is of that type. Here, κ_1 and κ_2 are the principal curvatures of the surface and a and b are constants describing properties of the material of the thin plate (resistance to bending and sheering). In [61] a special case ($a = b = 1$) of this functional,

$$\int_\Omega \kappa_1^2 + \kappa_2^2 \, d\omega, \tag{4.7}$$

is used to determine smooth tensor product B-spline surfaces. To overcome the difficulty that this functional is highly nonlinear and thus the numerical solution

is very time consuming, the integrand of (4.7) is evaluated only at the corners of the rectangular control grid. In [99] the functional

$$\int_\Omega \left(\frac{\partial \kappa_1}{\partial \epsilon_1}\right)^2 + \left(\frac{\partial \kappa_2}{\partial \epsilon_2}\right)^2 d\omega$$

with ϵ_1 and ϵ_2 denoting the directions of principal curvature is considered. Although this MVC functional that minimizes the variation of the curvature has proven to yield surfaces of perfect shape, the complexity and hence the numerical treatment is even worse than for the thin plate energy.

In contrast to these highly nonlinear functionals there are simpler, quadratic functionals that can be minimized by solving a linear system and are thus suitable for interactive use. The most commonly used functional is the *simplified* version of the thin plate energy (4.7):

$$\int_\Omega F_{uu}^2 + 2F_{uv}^2 + F_{vv}^2 \, du\, dv, \tag{4.8}$$

where F_{uu}, F_{uv} and F_{vv} denote the second order partial derivatives of the surface F. As this functional is quadratic and much easier to minimize than the exact version, it is widely used [62, 103, 40, 59, 139]. However, it is a good approximation to (4.7) only in the functional case, whereas the surfaces obtained in the parametric case may fail to have a pleasant shape. The more general quadratic, second order functional

$$\int_\Omega \alpha_{11}F_u^2 + \alpha_{12}F_uF_v + \alpha_{22}F_v^2 + \beta_{11}F_{uu}^2 + \beta_{12}F_{uv}^2 + \beta_{22}F_{vv}^2 \, du\, dv$$

has been used in [18], where the coefficients α_{ij}, β_{ij} are chosen by physical reasoning, and in [138], where the choice is based on geometric properties.

In contrast to the functionals based on the surface curvature, these simplified versions depend on the parameterization of the surface and different parameterizations of the same surface lead to different values of the functional. This statement also holds for energy functionals based on the third order derivatives, like

$$\int_\Omega F_{uuu}^2 + F_{vvv}^2 \, du\, dv,$$

as proposed in [13] and

$$\int_\Omega (F_{uuu} + F_{uvv})^2 + (F_{uuv} + F_{vvv})^2 \, du\, dv,$$

introduced in [56].

A compromise between the simplicity of the quadratic functionals and the quality of the exact ones are data dependent functionals as introduced in [55, 56, 59]. The basic idea is the following: since the simple quadratic functionals approximate the exact energies well if the shape of the surface is nearly planar and thus similar to the shape of the parameter space, one can take the

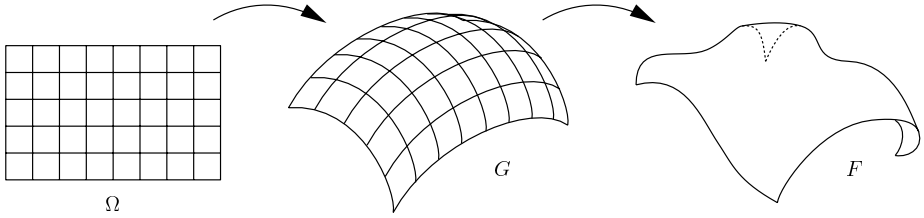


Figure 4.9: Parameter space Ω, reference surface G and surface F.

inverse approach and adapt the parameter space in such a way that its shape is close to that of the surface (see Figure 4.9). The concepts needed to do so are well-known in differential geometry [29, 80]. Given a *reference surface* $G : \Omega \to \mathbb{R}^3$ and considering *gradient* grad_G, *divergence* div_G and *Laplacian* Δ_G with respect to this reference surface, we can introduce the data dependent energy functionals

$$\int_\Omega \text{grad}_G(F)^2 \, d\omega_G$$

$$\int_\Omega \Delta_G(F)^2 \, d\omega_G \tag{4.9}$$

$$\int_\Omega \text{grad}_G(\text{div}_G(\text{grad}_G(F)))^2 \, d\omega_G$$

with $d\omega_G$ denoting the surface element $\|G_u \times G_v\| \, du \, dv$. These functionals are still quadratic and can thus be minimized by solving a linear system. If the reference surface G is close to F, the data dependent functionals will be good approximations to the exact energies. Note that the functionals in (4.9) can be proven not to depend on the specific parameterization of the reference surface G [57]. Still their computation is very expensive compared to the simple quadratic smoothness functionals which makes them unsuitable for interactive use.

An example of smooth surface reconstruction with different smoothness functionals is shown in Figure 4.11 where the data set of a ship hull (see Figure 4.10) was reconstructed with a 7×7 tensor product B-spline surface. The colour in the pictures relates to the Gaussian curvature K of the surfaces:

$$\text{blue} \iff K < 0, \qquad \text{green} \iff K = 0, \qquad \text{red} \iff K > 0.$$

This data set was part of a benchmark experiment organized by the Human Capital & Mobility Project FAIRSHAPE (ERB-CHRXCT-940522) and presented at the International Benchmarking Workshop on "Creating Fair and Shape-Preserving Curves and Surfaces" [6]. The reconstructed surface was asked to be a bicubic B-spline surface with positive Gaussian curvature.

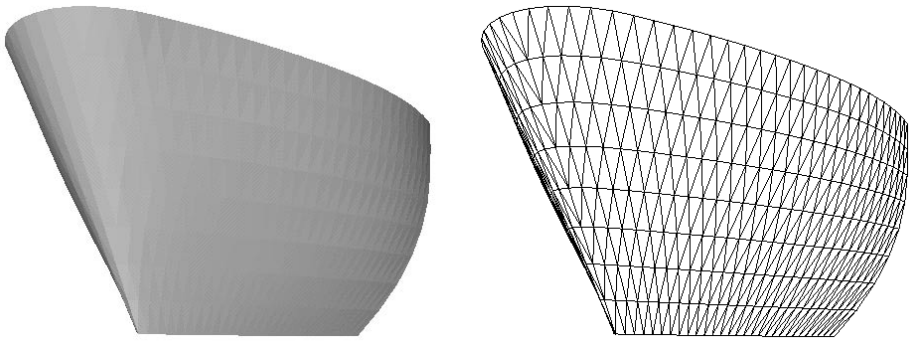


Figure 4.10: Triangulated data set of a ship hull with 531 data points.

Using the simple thin plate energy (4.8) and a smoothing factor $\mu = 10^{-4}$ gives the results to the left. From the pictures and the values in Table 4.3 we can see that an iterative parameter correction as discussed in Section 4.3.1 improves the surface quality but does not lead to surfaces with positive Gaussian curvature. In order to get rid of the change of curvature at the bow we have to apply a third order smoothness functional that measures the curvature variation. In Figure 4.11.d–f we used the smoothness functional

$$\int_\Omega F_{uu}^2 + 2F_{uv}^2 + F_{vv}^2 \, du\, dv + \int_\Omega \alpha F_{uuu}^2 + \beta F_{uuv}^2 + \gamma F_{uvv}^2 + \delta F_{vvv}^2 \, du\, dv$$

with $\alpha = 20$, $\beta = \gamma = 5$, and $\delta = 500$, thus smoothing particularly with regard to the curvature variation in vertical direction. After 8 parameter correction steps we obtain a surface with positive Gaussian curvature whose approximation quality can be improved by further iterations.

	iterations	Gaussian curvature			error in %	
		min	max	average	max	average
(a)	1	−0.091740	0.003311	0.002784	2.928813	0.507866
(b)	8	−0.006904	0.003057	0.001097	2.563619	0.436098
(c)	15	−0.003731	0.008313	0.000985	2.679641	0.439336
(d)	1	−0.005372	0.004111	0.002341	4.481793	1.072675
(e)	8	0.0	0.003747	0.001558	2.968360	0.509084
(f)	15	0.0	0.002587	0.001289	2.565349	0.474674

Table 4.3: Comparison of the different surfaces in Figure 4.11.

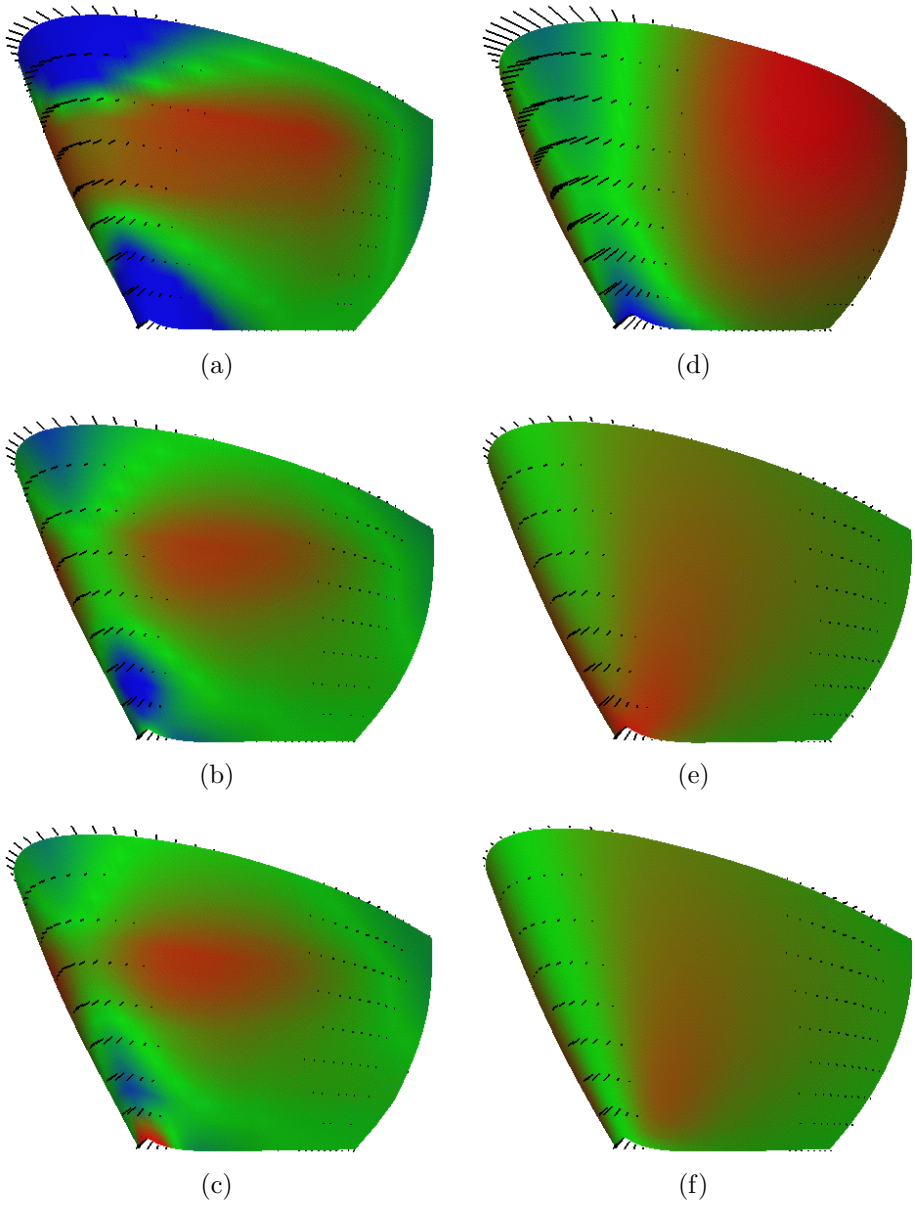


Figure 4.11: Surface reconstruction of the data set in Figure 4.10 using a quadratic smoothness functional of order 2 (left) and 3 (right) after 1 (top), 8 (middle) and 15 (bottom) parameter correction steps (see Section 4.3.1). The black dashes illustrate the distance between the data points and the surface.

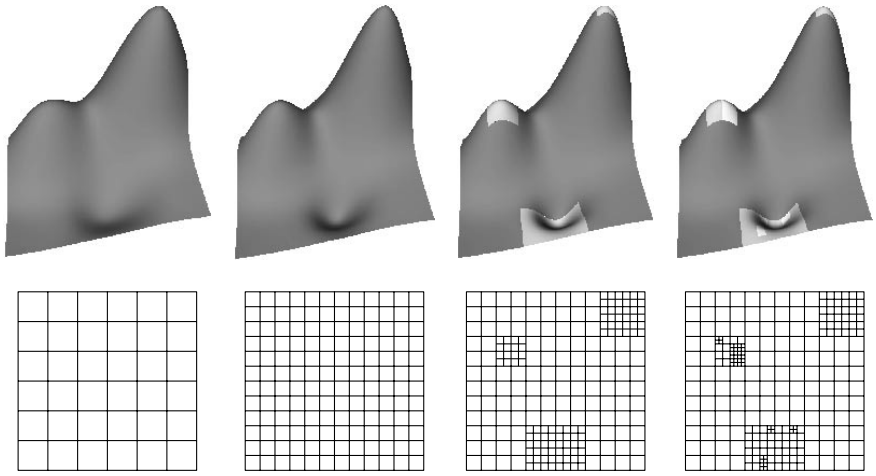


Figure 4.12: Approximating surface with one global and two local refinement steps.

4.3.4 Hierachical Surface Fitting

A major problem that arises by using the surface fitting methods discussed in Section 4.3.2 is the size of the linear systems that have to be solved. One way to overcome this drawback is the use of hierarchical surface models like the *Hierarchical B-Splines* introduced by Forsey and Bartels in [48]. Although these surfaces were originally designed for the purpose of modelling, the ideas have successfully been transferred to the surface fitting problem [49, 58, 91].

The basic idea is to subdivide the *global* approximation problem adaptively into several *local* problems where only a comparatively small part of the data has to be taken into account. These local problems will lead to linear systems of small size which can be solved efficiently. The first step of this method is to start with a class of surfaces S_0 defined on a coarse control lattice, thus having only a small number of control points, and determine the surface $F_0 \in S_0$ that solves the problem

$$\min_{F \in S_0} \mathcal{K}_\mu(F),$$

where $\mathcal{K}_\mu$ is the combined smoothness and approximation error functional (4.5) from Section 4.3.2.

If the quality of F_0 suits our needs, i.e. the approximation error is within a specified tolerance and the surface is sufficiently smooth, nothing has to be done. Otherwise we will either have to adapt the smoothing parameter μ or increase the number of degrees of freedom. Halving the grid size of the control lattice yields a new class of surfaces $S_1 \supset S_0$ whose dimension is roughly four times the dimension of S_0. However, this *global refinement* step increases the size of the linear system that has to be solved by factor 16. The strategy of *local refinement* offers a better alternative: only at those regions where the

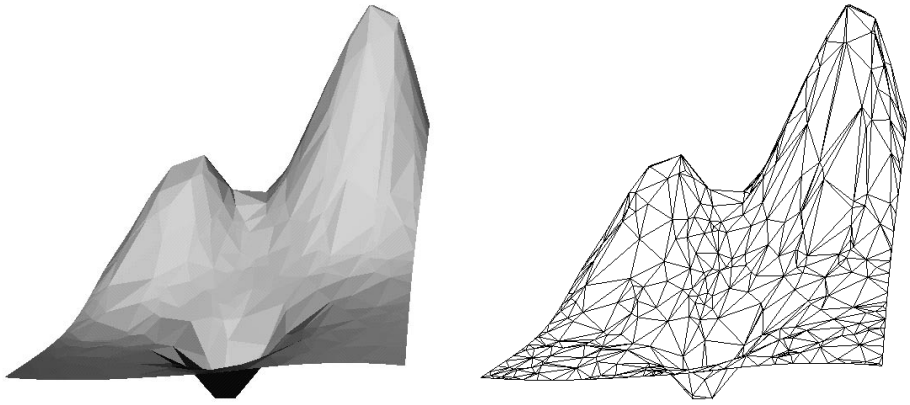


Figure 4.13: Triangulated data set of Franke's test function with 500 data points.

approximation error exceeds the tolerance, local overlay patches T_k with finer control lattices are added to the surface (see Figure 4.12). Now, instead of solving the approximation problem in the globally refined space S_1 we only look for the optimal surface contained in the affine subspace $\tilde{S}_1 = F_0 + T \subset S_1$ with $T = \sum_k T_k$.

If the overlay patches are chosen such that they are orthogonal, i.e. having disjunct support, this problem decomposes into several local problems. Indeed, if F_1 solves the optimization problem

$$\min_{F \in \tilde{S}_1} \mathcal{K}_\mu(F) = \min_{G \in T} \mathcal{K}_\mu(F_0 + G),$$

then F_1 can be written as $F_1 = F_0 + \sum_k G_k$ with the overlay patches $G_k \in T_k$ determined by solving the local problems

$$\min_{G \in T_k} \mathcal{K}_\mu(F_0 + G).$$

These local problems can be solved efficiently due to their small size. Obviously, this strategy can be applied recursively, as shown in Figure 4.12.

An example that illustrates the advantages of hierarchical surface fitting is shown in Figure 4.14 and Table 4.4. The data set to be reconstructed (see Figure 4.13) was derived by triangulating 500 randomly sampled data points of Franke's test function

$$\begin{aligned} F(x,y) = {} & \frac{3}{4}\, e^{-\frac{(9x-2)^2+(9y-2)^2}{4}} + \frac{3}{4}\, e^{-\frac{(9x+1)^2}{49}-\frac{9y+1}{10}} + \\ & \frac{1}{2}\, e^{-\frac{(9x-7)^2+(9y-3)^2}{4}} - \frac{1}{5}\, e^{-(9x-4)^2-(9y-7)^2} \end{aligned} \tag{4.10}$$

and the maximum error threshold was set to $\varepsilon = 0.2\,\%$. The optimal approximating surface F_0 within the class S_0 of all B-spline surfaces defined on a

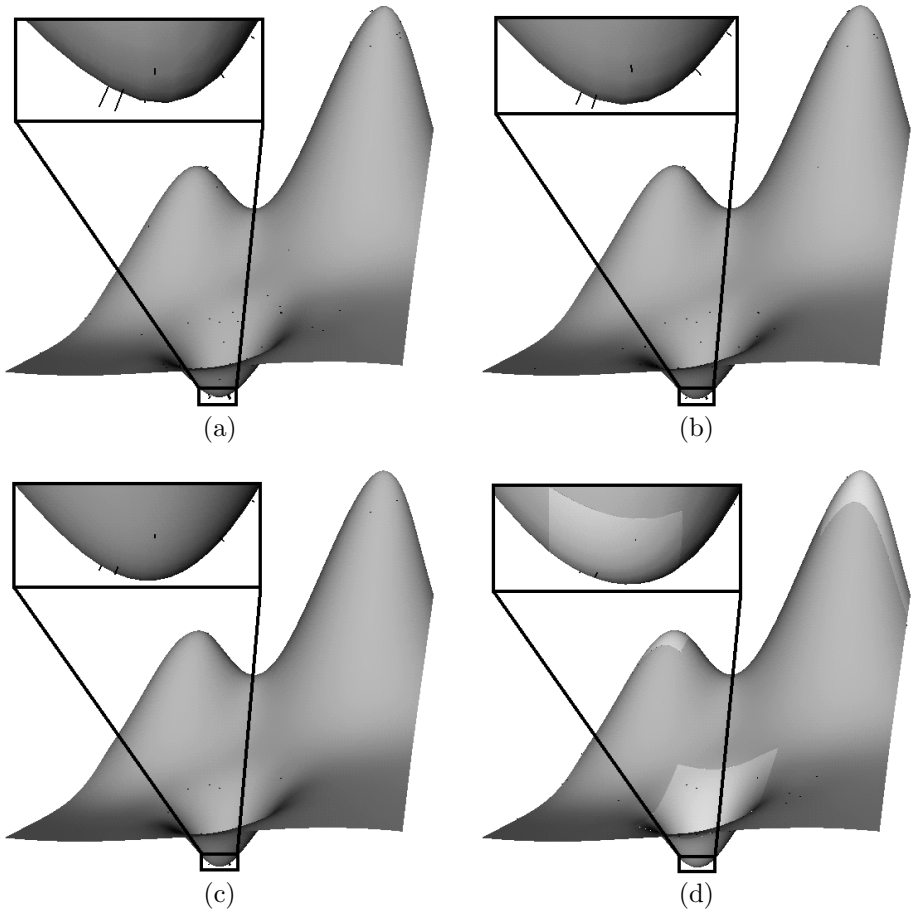


Figure 4.14: Surface reconstruction of the data set in Figure 4.13 using regular (a–c) and hierarchical (d) B-spline surfaces.

	number of control points		time in sec.	mean curvature			error in %	
				min	max	average	max	average
(a)	13×13	169	3.5	-28.884	23.166	1.9853	0.8873	0.0553
(b)	20×20	400	7.1	-28.212	27.665	1.9935	0.3877	0.0340
(c)	43×43	1 849	29.3	-28.745	28.740	2.0211	0.2036	0.0181
(d)	13×13 9×9 9×9 7×5 7×5	401	9.3	-27.322	23.294	1.9429	0.1399	0.0254

Table 4.4: Comparison of the different surfaces in Figure 4.14.

uniform 13×13 control grid is shown in Figure 4.14.a. This surface fails to fulfill the error criterion and we observe the largest errors to occur in the highly curved regions of the three 'bumps' (see enlarged detail).

Refining the control lattice locally by adding two overlay patches with 9×9 and one with 7×5 control points at those regions where the approximation error exceeds ε and refining the overlay patch at the depression again by adding another 7×5 patch gives the surface $\tilde{F}_2 \in \tilde{S}_2$ in Figure 4.14.d which clearly meets the error requirement. This surface has three hierarchy levels and 401 control points altogether.

If we take the same number of basis functions defined on a uniform control lattice, i.e. a surface with 20×20 control points, we obtain the surface in Figure 4.14.b whose maximum error is thrice as large. The superiority of the hierarchical surface is due to the fact that hierarchical approximation concentrates the degrees of freedom at those regions where they are mainly needed. Thereby the surface gains more flexibility to approximate the data points in these areas.

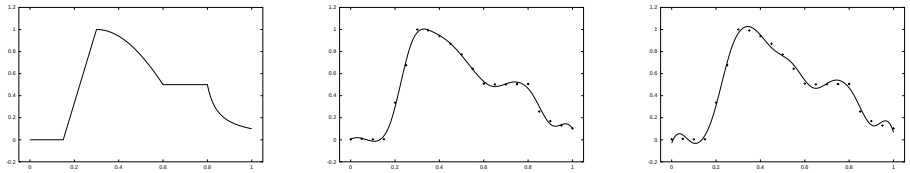
Refining the initial control grid twice globally rather than locally results in the surface $F_2 \in S_2$ shown in Figure 4.14.c. Although the globally refined function space contains the locally refined one, $S_2 \supset \tilde{S}_2$, and therefore $\mathcal{K}_\mu(F_2) < \mathcal{K}_\mu(\tilde{F}_2)$, the hierarchical solution $\tilde{F}_2$ is better than F_2 which even misses the error criterion marginally. This may seem strange at first glance but please remember that $\mathcal{K}_\mu$ does not measure the maximum but the least squares error $\mathcal{E}_2$ and Table 4.4 confirms $\mathcal{E}_2(F_2) < \mathcal{E}_2(\tilde{F}_2)$ indeed.

Let us recapitulate the three main advantages of using hierarchical surfaces for solving the approximation problem. Firstly, the computation is speeded up significantly (compare the forth column in Table 4.4) since only small local problems with few unknowns need to be solved. Secondly, the number of control points that define the approximating surface and thus the memory requirements are reduced, and thirdly, the localization of the degrees of freedoms to 'critical' regions helps to minimize the maximum instead of the least squares error. The next two sections will discuss how the maximum error can be minimized directly.

4.3.5 Chebychev Approximation

Although the concept of smooth least squares approximation as discussed in the previous sections is often appropriate we already saw in the examples that it is not the best choice if a user-specified error tolerance needs to be complied with. The disadvantage of using the (squared) ℓ_2 norm $\mathcal{E}_2$ from (4.3) is that it allows the error to become rather big at a few data points as long as the average error is small. In order to fulfil an error criterion we should better minimize the error measured in the ℓ_∞ or *Chebychev norm*

$$\mathcal{E}_\infty(F) = \max_{i=1,\dots,n} |F(p_i) - P_i| = \|B\boldsymbol{c} - \boldsymbol{P}\|_\infty$$



(a) (b) (c)

Figure 4.15: Reconstructing the function in (a) by approximating 21 uniformly sampled data points best in the ℓ_2 (b) and the ℓ_∞ norm (c).

directly. Since $\mathcal{E}_\infty$ is no longer a quadratic functional, its minimum cannot be found by solving a linear system and the method of *linear programming* has to be applied instead [8, 38, 76]. Although the uniqueness of the solution and the theoretical convergence of the linear programming algorithm can only be guaranteed if the basis functions are a *Chebychev* or *Haar system*, like univariate polynomials for example, the algorithm usually works as well for univariate B-splines which are a *weak Chebychev system* only and even for B-spline surfaces.

Another theorem that is valid for Chebychev systems only but applies nicely to B-spline curves and surfaces in practice is

Theorem 4.1 *(Alternating Extremal Points Theorem) If S is a k-dimensional Chebychev system $S \in \mathcal{C}[a,b]$ and $F^* \in S$ is the best Chebychev approximation to $n > k$ data points $(p_i, P_i) \in \mathbb{R}^2$, and $I^* = \{i : |F^*(p_i) - P_i| = \mathcal{E}_\infty(F^*)\}$ is the index set of all* extremal points *where F^* attains the maximum error, then there exist at least $k+1$ points $a \le p_{i_0} < \cdots < p_{i_k} \le b$ with $i_j \in I^*$ such that the sign of the error at p_{i_j} alternates, i.e.*

$$\sigma(-1)^j (F^*(p_{i_j}) - P_{i_j}) = \mathcal{E}_\infty(F^*), \qquad j = 0, \ldots, k,$$

for some $\sigma \in \{-1, 1\}$.

Of course there is no equivalence to the alternating property in the bivariate setting, but still there usually exist at least $k+1$ extremal points although counter-examples can be constructed.

The example in Figure 4.15 illustrates the effect of Chebychev approximation compared to least squares approximation. We used univariate polynomials of degree 13 (thus $k = 14$) to reconstruct the function shown in (a) by fitting it to 21 uniformly sampled data points. The maximum error of the Chebychev solution is significantly smaller than the one of the least squares solution (see Table 4.6, page 116) but the function oscillates due to the 15 alternating extremal points.

This effect also appears for bivariate approximation as shown in Figure 4.16, where we randomly sampled Franke's test function (4.10) at 400 data points and computed the best-fitting bivariate B-spline function with respect to the ℓ_2 and the ℓ_∞ norm. Table 4.5 confirms the Chebyshev solution to have a smaller maximum approximation error than the least squares solution but the surface

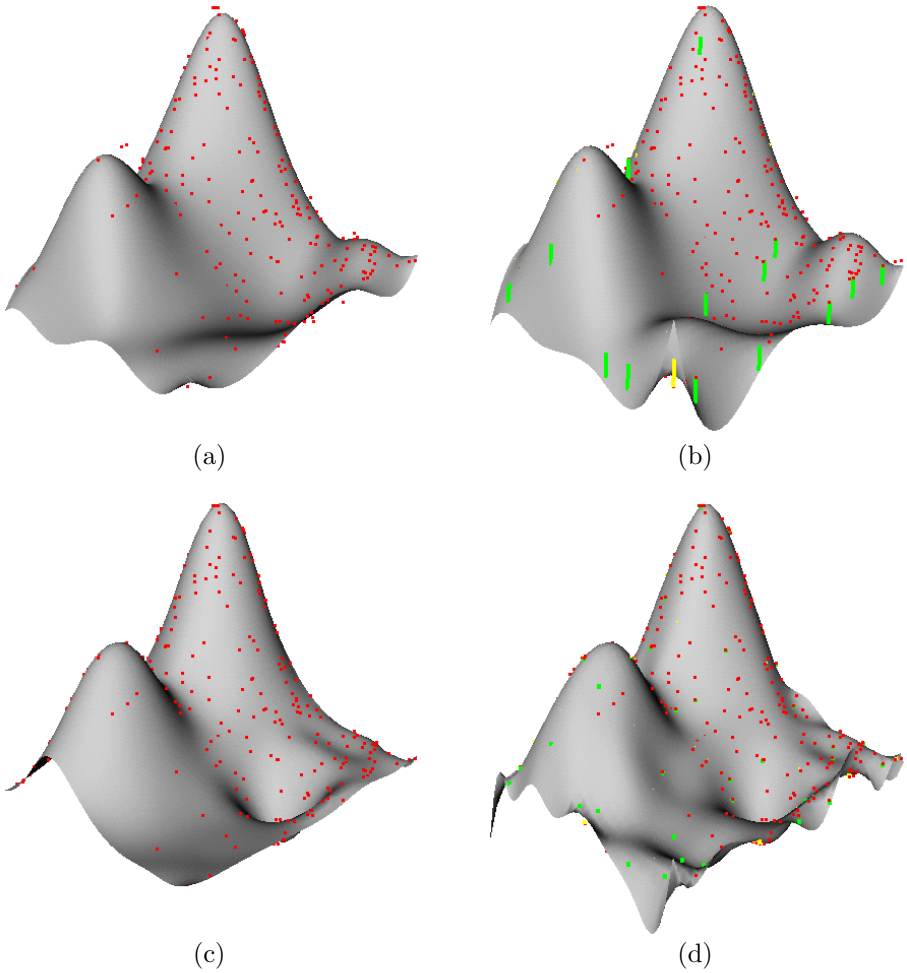
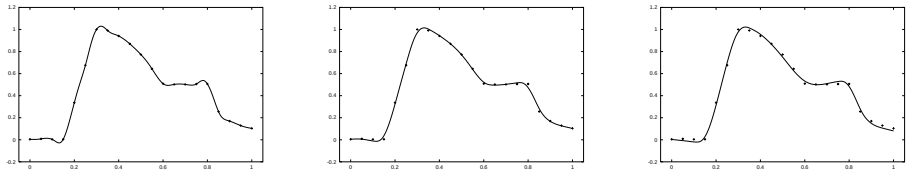


Figure 4.16: Least squares (left) and Chebychev (right) approximation with 6×6 (top) and 10×10 B-spline functions to 400 data points (red dots). The yellow and green dashes symbolize the positive and negative maximal errors to the data at extremal points.

	k	$\mathcal{E}_\infty(F)$
(a)	36	0.126812
(b)	36	0.067878
(c)	100	0.020306
(d)	100	0.013613

Table 4.5: Comparison of the different surfaces in Figure 4.16.



(a) (b) (c)

Figure 4.17: Smooth reconstruction of the function in Figure 4.15.a by interpolation (a), least squares (b), and Chebychev approximation (c).

wiggles between its $k+1$ extremal points, especially at the boundary, in almost the same manner as in the univariate example.

To avoid these oscillations we take the same approach as for least squares approximation by combining the approximation error $\mathcal{E}_\infty$ with a smoothness functional $\mathcal{J}$ and minimizing

$$\mathcal{K}_{\mu,\infty}(F) = \mu\mathcal{J}(F) + \mathcal{E}_\infty(F).$$

Using the definitions from Section 4.3.2 we can rewrite this minimization problem as a *quadratic programming* problem [41]. That is, the quadratic objective function

$$\mu \boldsymbol{c}^t A \boldsymbol{c} + \epsilon$$

with $k+1$ variables $\boldsymbol{c}$ and ϵ is minimized subject to the $2n$ linear inequality constraints

$$\|B\boldsymbol{c} - \boldsymbol{P}\|_\infty \leq \epsilon \qquad \Longleftrightarrow \qquad -\epsilon \leq F(p_i) - P_i \leq \epsilon, \qquad i = 1, \ldots, n.$$

Quadratic programming problems are well-known in literature and non-commercial software code for solving them is available [133].

An example for smooth Chebychev approximation is given in Figure 4.17 where we reconstructed the function in Figure 4.15.a by univariate polynomials of degree 30. While the smooth interpolation (compare Equation 4.2 in Section 4.3.2) overshoots at the 'kinks', the two smooth approximations nicely reconstruct the function. The maximum error of the Chebychev solution is one third smaller than the one of the least squares solution (see Table 4.6, page 116).

4.3.6 Iteratively Reweighted Least Squares

While Chebychev approximation as discussed in the previous section is best suited for fitting a surface to given data points within specified error bounds, the main problem of this approach is that linear and quadratic programming problems are very costly in terms of computation time in comparison to the linear problems that have to be solved for least squares approximation. However, the following theorem from [100] hints at a way out of this dilemma.

Theorem 4.2 *(Motzkin, Walsh) Let S be a Chebychev system $S \in \mathcal{C}[a,b]$ and $F^* \in S$ be the Chebychev approximation to n data points (p_i, P_i). Then there exist weights $\boldsymbol{w} = (w_1, \ldots, w_n)$ such that F^* also minimizes the* weighted least squares norm

$$\mathcal{E}_{\boldsymbol{w}}(F) = \sum_{i=1}^{n} w_i |F(p_i) - P_i|^2.$$

The minimum of this quadratic functional can be found by solving the linear system

$$B^t W B \boldsymbol{c} = B^t W \boldsymbol{P},$$

where $W = \mathrm{diag}(\boldsymbol{w})$ is the diagonal matrix containing the weights $w_1, \ldots, w_n$. It was Lawson [88] who utilized these facts to create an algorithm for finding the Chebychev approximation in an iterative procedure which was later extended by Rice and Usow [110] to find best ℓ_p approximations $(2 < p < \infty)$.

Lawson's algorithm:

1. Let $w_1^{(0)} = \cdots = w_n^{(0)} = 1$ and $r = 0$.
2. Find the minimum $F^{(r)}$ of $\mathcal{E}_{\boldsymbol{w}^{(r)}}$ by solving $B^t W^{(r)} B \boldsymbol{c}^{(r)} = B^t W^{(r)} \boldsymbol{P}$.
3. Update the weights,

$$w_i^{(r+1)} = w_i^{(r)} |F^{(r)}(p_i) - P_i|, \qquad i = 1, \ldots, n,$$

   increment r and continue with 2.

Lawson proved that the sequence $\left\{F^{(r)}\right\}$ converges to the Chebychev approximation,

$$\lim_{r \to \infty} F^{(r)} = F^*,$$

and that the sequence $\left\{\mathcal{E}_{\boldsymbol{w}^{(r)}}(F^{(r)})\right\}$ increases monotonically with

$$\lim_{r \to \infty} \mathcal{E}_{\boldsymbol{w}^{(r)}}(F^{(r)}) = \mathcal{E}_\infty(F^*).$$

Furthermore, if I^* is the index set of F^*'s extremal points (compare Theorem 4.1) then

$$\lim_{r \to \infty} w_i^{(r)} = 0 \qquad \Longleftrightarrow \qquad i \notin I^*,$$

and if $w_i^{(r)} \neq 0$, $i \in I^*$ and $w_i^{(r)} = 0$, $i \notin I^*$ for some r, then the algorithm converges after 2 more steps, otherwise the convergence is linear [21]. This

motivates to speed up the iterative process by making the weights tend to zero faster, for example by updating them with

$$w_i^{(r+1)} = w_i^{(r)}|F^{(r)}(p_i) - P_i|^2 \qquad \text{or} \qquad w_i^{(r+1)} = \left(w_i^{(r)}\right)^2|F^{(r)}(p_i) - P_i|,$$

and setting $w_i^{(r)} = 0$ if $w_i^{(r)} < \tau$ for some small $\tau > 0$. But it was already observed by Lawson that these modifications sometimes lead to divergence.

The strategy of updating the weights in Lawson's algorithm can be understood by looking at the ℓ_p approximation problem. In fact, if we minimize the ℓ_p norm (to the power of p)

$$\mathcal{E}_p(F) = \sum_{i=1}^{n} |F(p_i) - P_i|^p = \|B\boldsymbol{c} - \boldsymbol{P}\|_p^p$$

for $2 \le p < \infty$ via Newton iteration, then we have to solve

$$B^t W'^{(r)} B\boldsymbol{c}^{(r)} = B^t W'^{(r)} \boldsymbol{P}$$

in the r-th Newton step, exactly as in Lawson's algorithm, but with weights $w_i'^{(r)} = |F^{(r-1)}(p_i) - P_i|^{p-2}$ instead. Further details on ℓ_p approximation for $1 \le p \le \infty$ can be found in [137].

Lawson's algorithm can therefore be seen as a kind of Newton iteration for solving the ℓ_p problem with $p \to \infty$. It starts with $p^{(0)} = 2$ and iteratively applies one modified Newton step for the $\ell_{p^{(r)}}$ problem with $p^{(r+1)} = p^{(r)} + 1$ while accumulating all previous errors

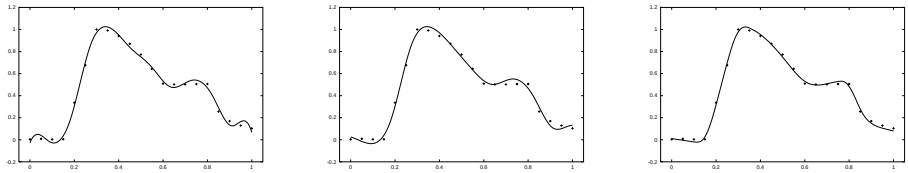
$$w_i^{(r+1)} = \prod_{j=0}^{r} |F^{(j)}(p_i) - P_i|$$

in order to get the weights rather than taking the most recent error to the power of $p^{(r+1)} - 2 = r + 1$,

$$w_i'^{(r+1)} = \prod_{j=0}^{r} |F^{(r)}(p_i) - P_i|.$$

Besides these theoretical aspects, the *iteratively reweighted least squares* (IRLS) method is of particular interest because it usually reduces the maximum error significantly after just a few iterations. Furthermore, it can easily be combined with a smoothness functional and as an iterative process it matches nicely with the concept of parameter correction (see Section 4.3.1) while the extra costs for updating the weights are negligible.

Figure 4.18 shows some examples of function reconstruction after four IRLS steps. As in Figures 4.15 and 4.17 we used univariate polynomials of degree 13 (a,b) and 30 (c) respectively and added a smoothness term to the error functional for the reconstructions in (b) and (c). Table 4.6 summarizes the results of the different reconstruction methods. Note that for $k = 14$ the maximum error of



(a) (b) (c)

Figure 4.18: Different IRLS reconstructions of the function in Figure 4.15.a.

	method	smoothing	k	$\mathcal{E}_\infty(F)$
Figure 4.15.b	ℓ_2	no	14	0.046318
Figure 4.18.b	IRLS$^{(4)}$	yes	14	0.041512
Figure 4.18.a	IRLS$^{(4)}$	no	14	0.035286
Figure 4.15.c	ℓ_∞	no	14	0.035173
Figure 4.17.b	ℓ_2	yes	31	0.033317
Figure 4.18.c	IRLS$^{(4)}$	yes	31	0.023781
Figure 4.17.c	ℓ_∞	yes	31	0.022973
Figure 4.17.a	interpolation	yes	31	0.0

Table 4.6: Comparison of the different reconstructions of the function in Figure 4.15.a.

the IRLS solution with smoothing is less than the one of the ℓ_2 solution without smoothing which clearly indicates the ability of IRLS to reduce the maximum error after just a few iterations.

The behaviour of the maximum error during the IRLS process is shown in Figure 4.19. We took the same 400 data points sampled from Franke's test function as in Figure 4.16 and bivariate B-spline functions over a uniform 6×6 knot grid for reconstructing them with IRLS approximation without smoothing.

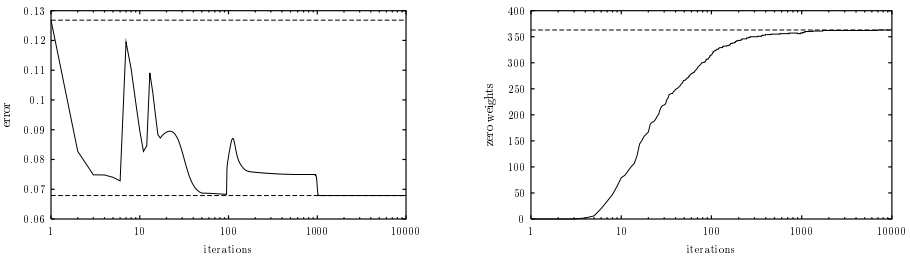
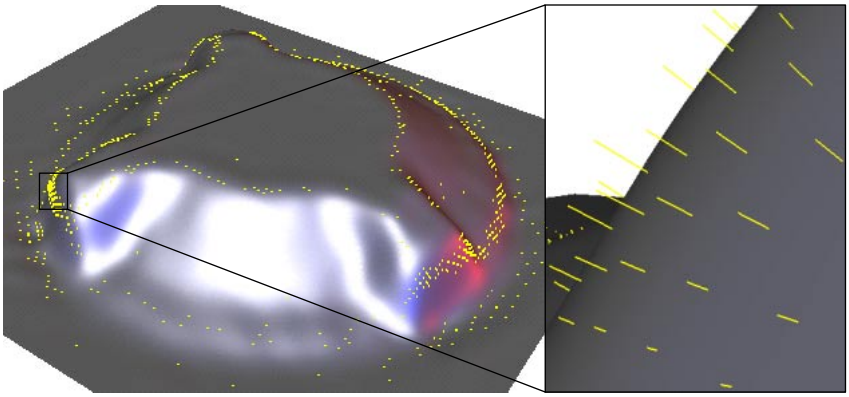
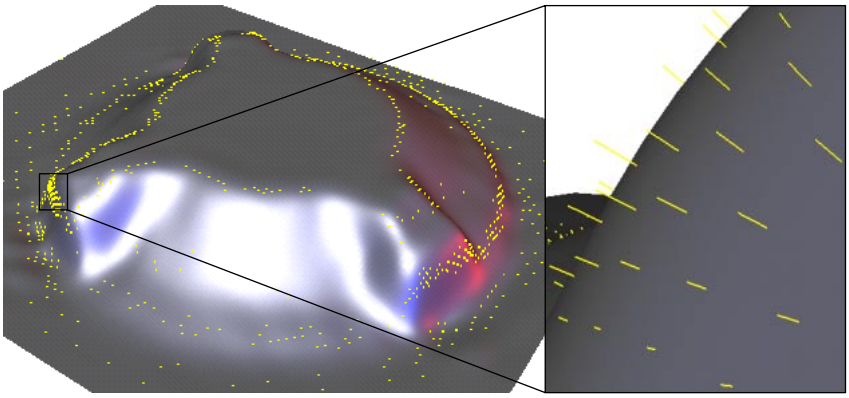


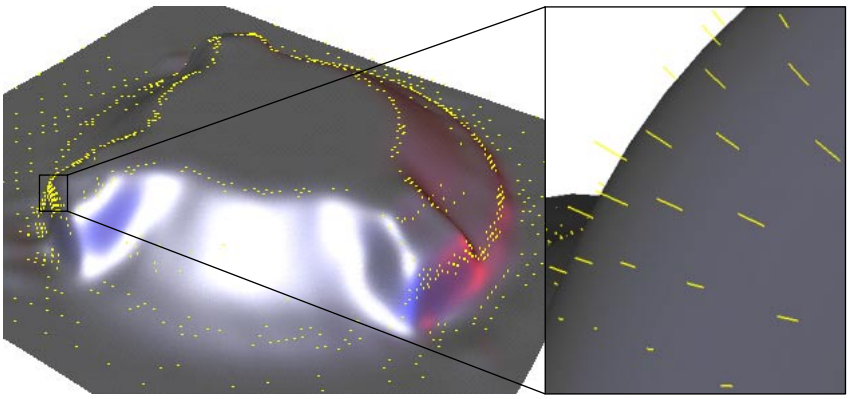
Figure 4.19: Convergence of the maximum error and the number of zero weights for IRLS approximation with 6×6 B-spline functions to 400 data points (compare Figure 4.16).



(a)



(b)



(c)

Figure 4.20: Surface reconstruction of the data set in Figure 4.11 with IRLS approximation. The yellow dashes illustrate the distance between the data points and the surface.

	iterations	IRLS	error in %		mean curvature		
			max	average	min	max	average
(a)	0		0.755660	0.139964	−0.073339	0.103838	0.014599
(b)	1	yes	0.515890	0.131807	−0.079671	0.102377	0.014519
(c)	2	yes	0.396737	0.130225	−0.085258	0.103718	0.014748
	1	no	0.716089	0.132498	−0.070113	0.093952	0.014670
	2	no	0.662751	0.126273	−0.070991	0.085567	0.014794

Table 4.7: Comparison of the different surfaces in Figure 4.20.

Again, we can see that the maximum error is reduced significantly after a few iterations ($k \leq 5$) only and although it increases temporarily for $7 \leq k \leq 22$ and $95 \leq k \leq 112$ it is always smaller than the maximum error of the ℓ_2 solution (top dashed line). We also plotted the number $\#\{i : w_i^{(r)} = 0\}$ of zero weights per iteration so as to show that it increases almost monotonically (sometimes it decreases by one or two) and that the maximum error converges to the maximum error of the ℓ_∞ solution (bottom dashed line) as soon as the maximum number of zero weights ($n - k - 1 = 363$) is reached.

In Figure 4.20 we finally present an example demonstrating that the concept of IRLS approximation can also be successfully applied to the parametric surface fitting problem. As in Figure 4.7 we reconstructed the data set in Figure 4.6 using MIPS for parameterizing the data points, the simplified thin plate energy as smoothness functional with smoothing factor $\mu = 10^{-4}$, and 30×30 B-spline surfaces. In (a) we can see the smooth ℓ_2 reconstruction, the same as in Figure 4.7.c. If we now apply Lawson's reweighting strategy and iterate the smooth approximation process we obtain the result in (b) and another IRLS step gives the surface shown in (c). The close up view to the right clearly demonstrates how the approximation error decreases with each iteration step while the smoothness of the surface does not vary (see Table 4.7). In fact, the maximum error is reduced by almost 50 %. Note that we also optimized the parameter values after each approximation step. Table 4.7 also shows the results for iterating the smooth approximation process with parameter correction only. This also helps to reduce the approximation error but the effect is less significant than with concurrent reweighting.

4.4 Indirect Approximation

So far we have discussed smooth surface reconstruction from discrete *scattered data* and have discovered several problems thereby—and solutions as well. From a theoretical point of view, the hardest problem is that adequate interpolation conditions like the *Schoenberg-Whitney* conditions for curves are yet unknown for tensor product B-splines. Therefore, the solution of the interpolation problem can not be guaranteed to exist in general and the solution of the approximation

problem may not be unique. As we saw in Section 4.3 we can regularize interpolation and approximation of scattered data by adding smoothness functionals but this also increases the computation costs significantly.

We believe that the main reason for these problems is that scattered data points are *unstructured* while tensor product surfaces have a natural rectangular structure and that scattered data and tensor product surfaces are simply not compatible for this reason. This assumption is confirmed by two observations. If we organize the scattered data points in a triangulation and use bivariate spline surfaces that are piecewise continuous over this triangulation instead of tensor product splines, then the data and the surface have the same structure and there exist several interpolation methods which give pleasing results. For a good survey on recent developments in this field we refer the reader to [102]. If on the other hand the data is organized in a rectangular grid and thus has the same structure as a tensor product surface, then interpolation and approximation become much simpler and decouple into a series of curve fitting problems.

Let us consider a general tensor product surface

$$F(u,v) = \sum_{i=0}^{m} \sum_{j=0}^{n} c_{ij} A_i(u) B_j(v)$$

with basis functions $\{A_i\}$ and $\{B_j\}$ and gridded parameter points $p_{rs} = (u_r, v_s)$, $r = 0, \ldots, m$, $s = 0, \ldots, n$ with associated data points P_{rs}. Then the interpolation conditions $F(p_{rs}) = P_{rs}$ can be written as the matrix equation

$$\underbrace{\begin{pmatrix} A_0(u_0) & \cdots & A_m(u_0) \\ \vdots & \ddots & \vdots \\ A_0(u_m) & \cdots & A_m(u_m) \end{pmatrix}}_{A} \underbrace{\begin{pmatrix} c_{00} & \cdots & c_{0n} \\ \vdots & \ddots & \vdots \\ c_{m0} & \cdots & c_{mn} \end{pmatrix}}_{C} \underbrace{\begin{pmatrix} B_0(v_0) & \cdots & B_0(v_n) \\ \vdots & \ddots & \vdots \\ B_n(v_0) & \cdots & B_n(v_n) \end{pmatrix}}_{B^t} = \underbrace{\begin{pmatrix} P_{00} & \cdots & P_{0n} \\ \vdots & \ddots & \vdots \\ P_{m0} & \cdots & P_{mn} \end{pmatrix}}_{P}$$

and the unknown control points $\{c_{ij}\}$ of the interpolating surface can be found by first solving $AD = P$ with a $(m+1) \times (n+1)$ matrix D, in other words,

$$A\boldsymbol{d}_j = \boldsymbol{P}_j, \qquad j = 0, \ldots, n, \tag{4.11}$$

where $\boldsymbol{d}_j$ and $\boldsymbol{P}_j$ are the j-th column vectors of D and P respectively, and then solving $D = CB^t$ which is equivalent to $BC^t = D^t$ or

$$B\bar{\boldsymbol{c}}_i = \bar{\boldsymbol{d}}_i, \qquad i = 0, \ldots, m, \tag{4.12}$$

where $\bar{\boldsymbol{c}}_i$ and $\bar{\boldsymbol{d}}_i$ are the i-th row vectors of C and D respectively. Note that (4.11) and (4.12) are curve interpolation problems of rather small size relative to the surface interpolation problem and that only two matrix factorizations are required, one for A and one for B.

We therefore propose to use the regular quadrilateral remeshing algorithm from Section 3.3 to replace a set of given scattered data points P_i with a set

of gridded data points P_{rs}. Then we interpolate the gridded data, thereby approximating the original data values *indirectly* rather than approximating them directly as in the previous sections. If we consider interpolation with uniform, bicubic tensor product B-spline surfaces we can make this interpolation process very efficient by interpolating the data points at the knots of the grid formed by the knot vectors in u- and v-direction since this leads to tridiagonal matrices A and B. In detail, we proceed as follows.

Given the $(m+1)\cdot(n+1)$ data points P_{rs} of a quadrilateral remesh, we define the knot vectors $T = (-3, -2, \ldots, m+2, m+3)$ and $S = (-3, -2, \ldots, n+2, n+3)$ and consider the space of bicubic tensor product B-spline surfaces $F : \Omega \to \mathbb{R}^3$, $\Omega = [0, m] \times [0, n]$ over the knot grid $T \times S$:

$$F(u, v) = \sum_{i=0}^{m+2} \sum_{j=0}^{n+2} c_{ij} N_i(u) N_j(v).$$

Note that in order to have full support of the basis functions over the parameter domain Ω we need $(m + 3) \cdot (n + 3)$ basis functions as opposed to the $(m + 1) \cdot (n + 1)$ interpolation conditions $F(r, s) = P_{rs}$, $r = 0, \ldots, m$, $s = 0, \ldots, n$. We therefore have to add further conditions so as to obtain a uniquely solvable problem. A typical choice are the *natural end conditions* which demand the second derivative of the surface to be zero along the boundary. This results in the $2(n + 1) + 2(m + 1) + 4$ additional conditions

$$\begin{aligned} F_{uu}(0, s) = F_{uu}(m, s) &= 0, \qquad s = 0, \ldots, n, \\ F_{vv}(r, 0) = F_{uu}(r, n) &= 0, \qquad r = 0, \ldots, m, \end{aligned}$$

and

$$F_{uuvv}(0, 0) = F_{uuvv}(m, 0) = F_{uuvv}(m, n) = F_{uuvv}(0, n) = 0,$$

so that all conditions sum up to the number of basis functions and unknown coefficients c_{ij} respectively.

If we write the additional natural end conditions in terms of coefficients and add them to the matrix notation of the interpolation conditions, we obtain the system $ACB^t = P$ with

$$A = \begin{pmatrix} 1 & -2 & 1 & & & \\ \frac{1}{6} & \frac{4}{6} & \frac{1}{6} & & & \\ & \ddots & \ddots & \ddots & & \\ & & \frac{1}{6} & \frac{4}{6} & \frac{1}{6} & \\ & & & 1 & -2 & 1 \end{pmatrix}, C = \begin{pmatrix} c_{00} & \cdots & c_{0,n+2} \\ \vdots & \ddots & \vdots \\ c_{m+2,0} & \cdots & c_{m+2,n+2} \end{pmatrix}, P = \begin{pmatrix} 0 & 0 & \cdots & 0 & 0 \\ 0 & P_{00} & \cdots & P_{0n} & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & P_{m0} & \cdots & P_{mn} & 0 \\ 0 & 0 & \cdots & 0 & 0 \end{pmatrix},$$

and B with the same layout as A but of size $(n + 3) \times (n + 3)$ rather than $(m + 3) \times (m + 3)$. The first and last row of A and B relate to the natural end conditions and the other rows reflect the interpolation conditions. Note that due to the compact support of the basis functions and because we chose knots

as parameter points these matrices are tridiagonal except for one entry in the first and one in the last row. However, multiplying A and B by

$$M = \begin{pmatrix} -\frac{1}{6} & 1 & & & \\ & 1 & & & \\ & & \ddots & & \\ & & & 1 & \\ & & & 1 & -\frac{1}{6} \end{pmatrix},$$

from left gives tridiagonal matrices $A' = MA$ and $B' = MB$. We finally consider the $(n+3)+(m+3)$ curve interpolation problems

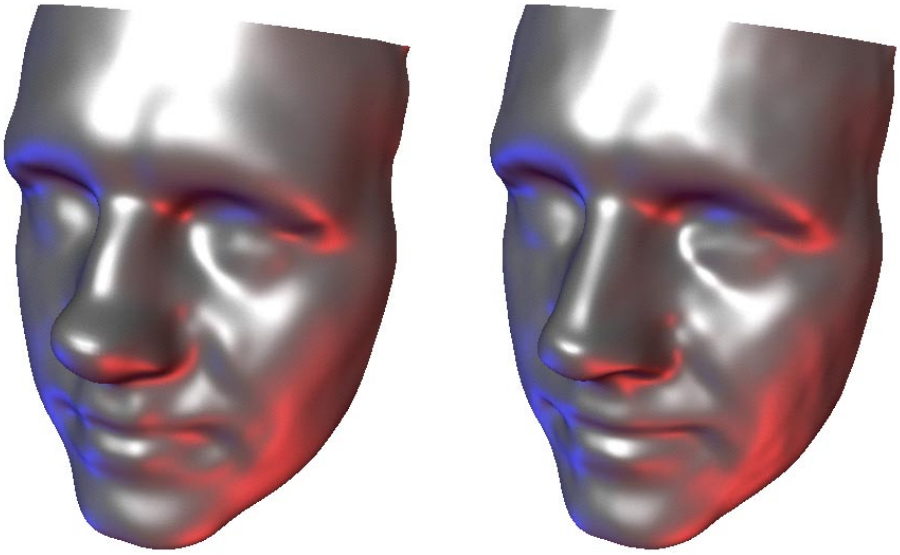
$$A'\boldsymbol{d}_j = M\boldsymbol{P}_j \qquad \text{and} \qquad B'\bar{\boldsymbol{c}}_i = M\bar{\boldsymbol{d}}_i,$$

each of which can be solved in linear time since A' and B' are tridiagonal. Assuming $m = n$ we conclude that this indirect approximation approach amounts to $\mathcal{O}(m^2)$ computations in contrast to the $\mathcal{O}(m^6)$ operations needed for solving the scattered data approximation problem which makes it extremely fast.

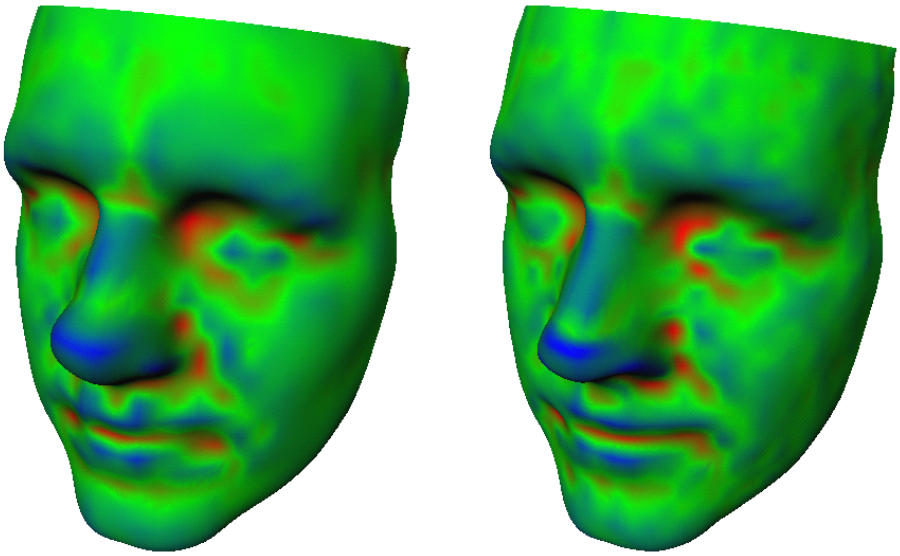
Another advantage of this approximation method is that it does not require to introduce smoothness functionals. We observed in our examples that whenever the remesh is of a nice shape the interpolating surface somehow inherits this smoothness, probably due to the energy minmizing properties of B-splines, a result that needs further investigation. So instead of applying rather complicated smoothness functionals to the approximating surface we can consider the discrete and therefore much simpler problem of smoothing the regular quadrilateral remesh. Furthermore, this method can easily be combined with the hierarchical spline surfaces from Section 4.3.4. We can start by interpolating the vertices of a coarse quadrilateral remesh and further refine the remesh only at those regions where the interpolation surface does not approximate the given data points sufficiently well. The refined regions can then be interpolated by overlay patches which are added to the coarse interpolation surface as in Section 4.3.4.

The drawback of this method is that it seems impossible to establish a priori error bounds of the approximation error to the data points altough it poses no problem to actually compute this error after having determined the surface. It is also difficult to specify the approximation order of this method, except for the obvious proposition that the error tends to zero as the refinement level of the quadrilateral remesh increases. However, since solving the interpolation problem is very fast, it might be possible to establish an iterative process in which the data points that are interpolated are changed in dependency of the approximation error at the data points nearby so that the error decreases after solving the interpolation problem again.

Let us close this section with two examples of the proposed indirect approximation method. Figures 4.21–4.23 show the interpolating spline surface of the quadrilateral remesh in Figure 3.23 with $35 \cdot 35 = 1\,225$ control points and compare it to a smooth least squares approximation of the data points with the same number of control points. As we can see from the mean curvature plots in

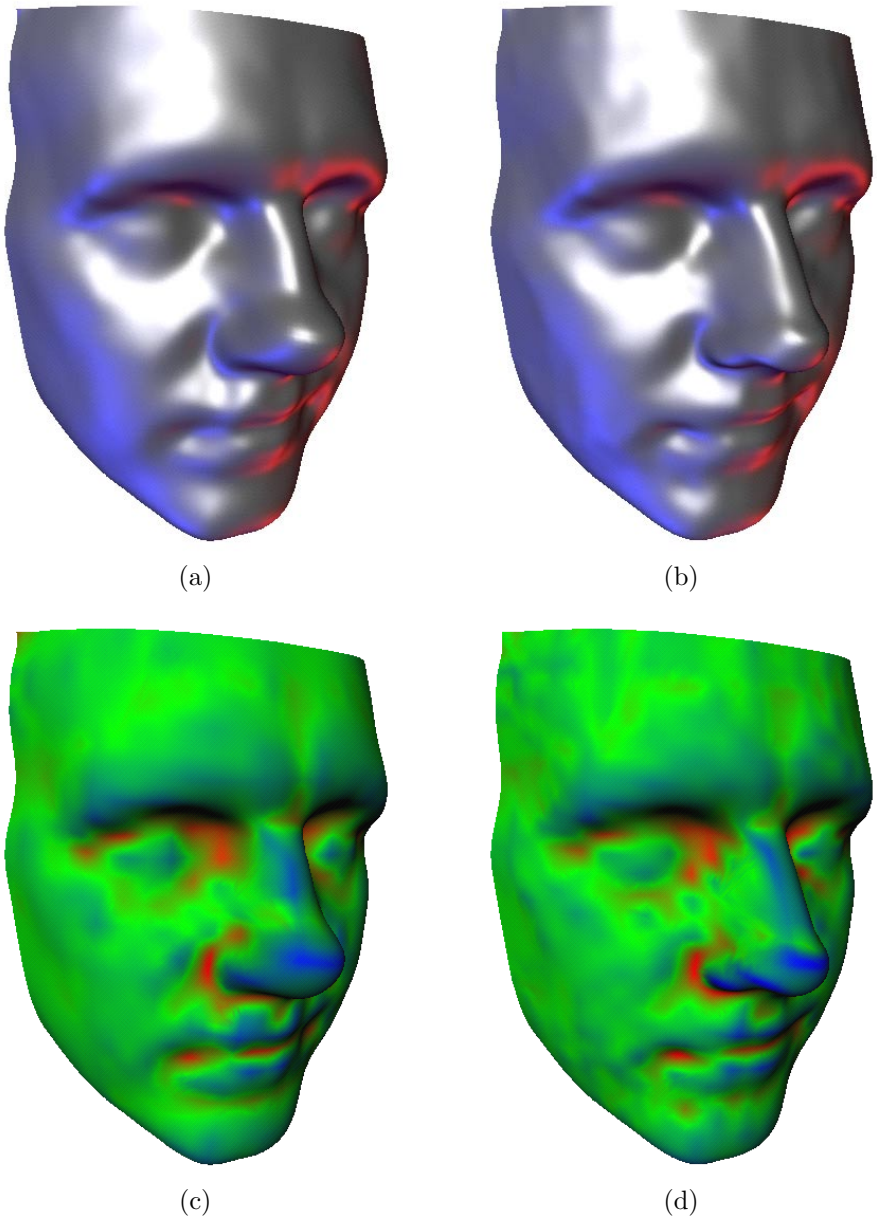


(a) (b)



(c) (d)

Figure 4.21: Smooth least squares (left) and indirect (right) approximation of the data set in Figure 3.22 on page 89.



(a) (b)

(c) (d)

Figure 4.22: Smooth least squares (left) and indirect (right) approximation of the data set in Figure 3.22 on page 89.

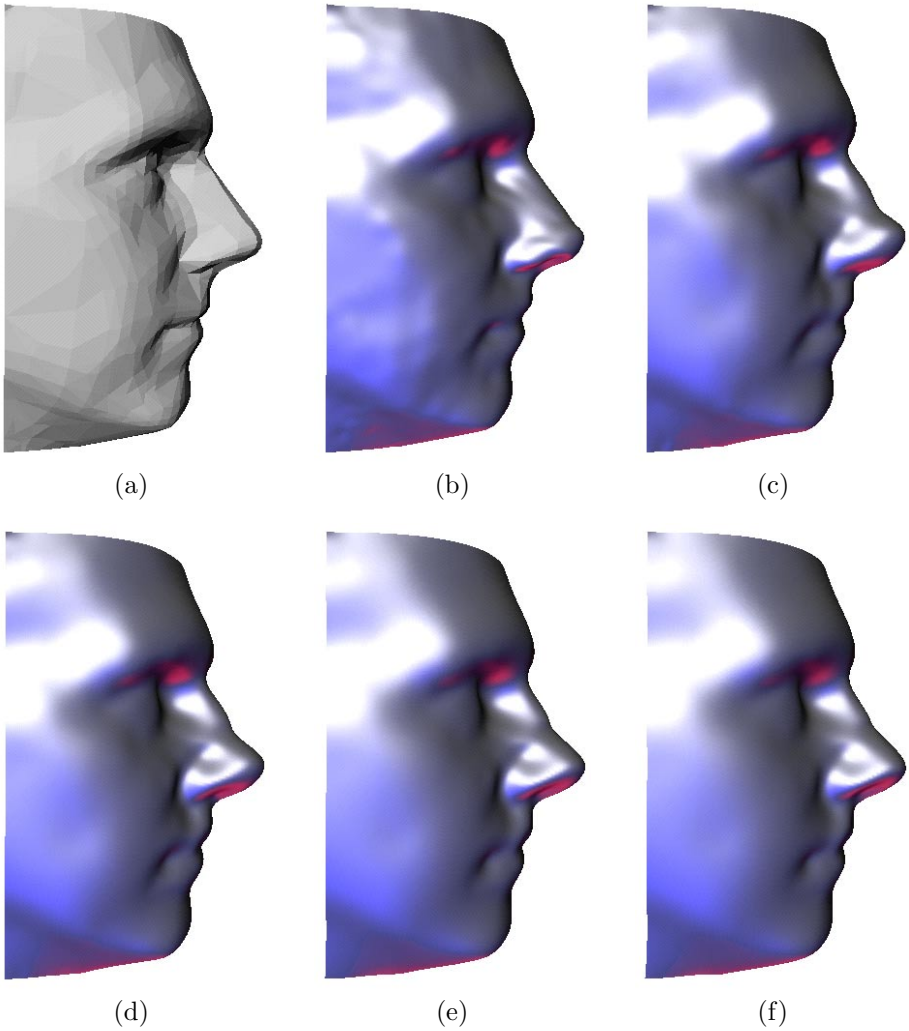


Figure 4.23: Different reconstructions of the data set in Figure 3.22 on page 89.

	method	error in %		mean curvature ($\cdot 10^{-4}$)			time in sec.
		max	average	min	max	average	
(b)	indirect	0.652034	0.103547	−1.951587	1.770751	0.300773	6
(c)	ℓ_2	0.807107	0.102735	−1.478671	1.578625	0.291950	37
(d)	IRLS$^{(1)}$	0.499620	0.096863	−2.445003	1.829205	0.297460	75
(e)	IRLS$^{(2)}$	0.379449	0.121866	−3.212197	1.707961	0.285877	112
(f)	IRLS$^{(3)}$	0.357732	0.139740	−3.857718	1.641943	0.279558	150

Table 4.8: Comparison of the surfaces in Figure 4.23.

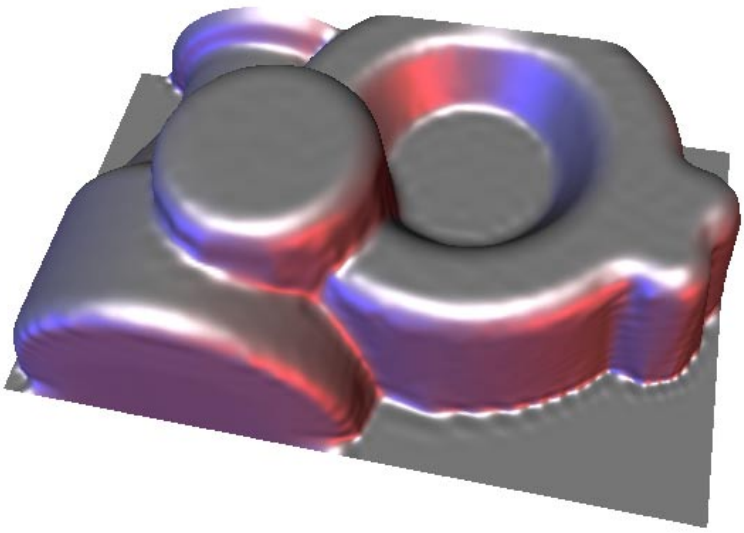


Figure 4.24: Indirect approximation of the data set in Figure 3.16 on page 86.

Figures 4.21.c, 4.21.d, 4.22.c, and 4.22.d, and the profiles in 4.23.b, and 4.23.c, the surface obtained by indirect approximation is almost as smooth as the one computed with the variational approach. Furthermore, Table 4.8 states that it clearly outranges the classical solution in terms of approximation error. However, this table also confirms that we can diminish the error of the smooth approximation significantly by applying the IRLS method from Section 4.3.6 and Figure 4.23 shows that this also improves the shape of the approximation surface. Taking a closer look at the profile of the nose we observe that the indirect approximation (b) nicely matches the shape of the original nose (a) whereas the smooth least squares solution (c) features a snub nose. But as we iterate the smooth approximation process with reweighted least squares, the nose is straightened out (d–f). Still, the result of the indirect approximation method is remarkable if we keep in mind that the quadrilateral remesh and the interpolating surface can be computed in a few seconds while solving the smooth approximation problem takes up to several minutes on an SGI Octane with a 400 MHz R12000 processor.

Another example is shown in Figure 4.24 where we interpolated the quadrilateral remesh in Figure 3.21 yielding a spline surface with $83 \cdot 51 = 4\,233$ control points. This example illustrates the rigidity of the tensor product structure which leads to oscillations along crest lines in the given data that do not align with the iso-parameter lines of the surface. In order to avoid this effect, such feature lines need to be detected in advance. Then we can either dissect the data set into several patches which are reconstructed individually or align the edges of the remesh and thus the iso-parameter lines of the interpolating surface to the features.

Das Chaos will anerkannt, will gelebt sein,
ehe es sich in eine neue Ordnung bringen läßt.
Hermann Hesse (1877–1962)

Conclusion

The subject of this dissertation has been a thorough investigation into the parameterization of triangulations in principle and practice. But this work seems to be incomplete without a summary of the results and an outlook on possible future investigations in this field.

In the first part of this thesis several methods for parameterizing triangulations were reviewed and their advantages and disadvantages were discussed from a theoretical point of view. The most efficient methods can be expressed in a common framework where every parameter point is an affine combination of its neighbours and found by solving a sparse linear system. They only differ in the way how the weights of the affine combinations are chosen and the results of the applications in the second part of this thesis suggest that the quality of the parameterization is the higher the more knowledge about the triangulation enters the computation of these weights.

While uniform weights, for example, only depend on the number of neighbouring vertices, chord length weights also take the distance to these neighbours into account which improves the parameterization considerably. Both choices are derived from an edge-based spring model and do not utilize any information about the triangles of the triangulation. On the one hand this is advantageous, because the methods thus also apply to the parameterization of point clouds with connectivity structure as discussed in Chapter 2, but on the other hand, the resulting parameterizations are rather poor for triangulations, probably because they lack the ability to reproduce planar triangulations.

In contrast, the computation of harmonic and shape preserving weights requires the neighbouring vertices to be ordered in a triangulation since both choices depend on triangle information. It is due to this extra knowledge that the corresponding parameterizations have the reproduction property and give much better results in the applications than the spring model parameterizations. In practice, harmonic and shape preserving parameterizations are almost identical but the latter are to be preferred as they also guarantee injectivity.

The only disadvantage of linear methods is that they need the boundary of the triangulation to be parameterized in advance and it is not always clear how this is done best. The new concept of most isometric parameterizations overcomes this drawback and can be used to determine the parameter points of all vertices in the same manner. But although the computation of MIPS

can be speeded up considerably by a hierarchical approach it remains a comparatively slow method. As the results obtained by using shape preserving and most isometric parameterization are very similar it is advisable to use MIPS only if no reasonable parameterization of the boundary can be found otherwise.

It might be worth noticing that harmonic, shape preserving, and most isometric parameterization all aim at preserving angles at the expense of areas. And while this concept of conformality in the form of the Mercator projection safely navigates sailors over the oceans, in the context of parameterizations it steers the geometric modeller soundly through the seas of applications which were discussed in the second part of this thesis.

In Chapter 2 a method for triangulating point clouds with patch topology was reviewed and extended to spherical topology. Both strategies are based on the fact that the triangulation problem is rather easy to solve in two dimensions and that the three-dimensional problem is reduced to that simple case by means of a parameterization of the point cloud. But the triangulations obtained that way tend to look crinkly and need to be postprocessed, for example with the newly developed optimization method in Section 2.3 which can generally be used to optimize triangulations without modifying the vertex positions.

The same concept of reducing the problem's dimensionality was used in Chapter 3 to remesh a given triangulation with a subdivision connectivity triangulation. Again, remeshing in two dimensions is quite straightforward and sufficient to solve the general problem provided that a suitable parameterization of the triangulation is given. Chapter 3 also treated remeshing with another kind of regular surfaces, namely quadrilateral meshes which leads to a new indirect smooth surface approximation scheme.

This method was discussed and compared to the classical variational approach to smooth surface fitting in Chapter 4 and found to be competitive in terms of the results but substantially more efficient. In the variational approach the approximating surface is usually determined by minimizing a weighted combination of some quadratic smoothness functional and a norm quantifying the approximation error. Normally the least squares norm is used because it leads to sparse linear systems, while the Chebychev norm which should rather be reduced in order to guarantee error tolerances requires to solve a costly quadratic programming problem. A convenient compromise between both choices is the concept of IRLS which was successfully adapted to the context of smooth surface fitting. This method repeatedly solves sparse linear systems and reduces the maximum approximation error significantly after a few iterations only.

While discussing parameterizations of triangulations and their applications, this thesis mainly focussed on the case of disc-like topology, in other words, simple triangulations and surface patches with one boundary and no holes, because more complex topology can always be handled by segmenting and reducing it to that case. In Section 1.4 such a segmentation strategy was developed for triangulations and adapted to point clouds in Section 2.2.3. But it remains to improve that method and to further research on the segmentation problem.

I accept chaos,
I'm not sure whether it accepts me.
Bob Dylan (∗ 1941)

Future Work

As the investigations in the applications part indicate that the results are the better the less distorted or more conformal the parameterization is, it seems reasonable to consider this observation as a segmentation criterion. A triangulation could, for example, be split into subtriangulations such that the shape deformation energy of the single parameterizations is below a certain threshold. Obviously the subtriangulations can be parameterized without distortion if they consist of one or two triangles only so that it is important to find the proper balance between number of segments and shape deformation.

Another important issue of segmentation are feature or crest lines along which a data set should be cut. One way of modifying triangulations so as to enhance such structures was presented in Section 2.3. However, a robust strategy to reliably detect feature lines is not available yet. Especially in the context of smooth surface fitting, segmentation of the given data leads to the further question of how to join the single surface patches that approximate the individual subsets of data to give an overall reconstruction surface. While a non-differentiable joint along a crest line might be realized by trimming two neighbouring surface patches along the common intersection curve, blending with a higher order of continuity remains an open and unsolved problem.

Bibliography

[1] D. Adams. *The Hitchhiker's Guide to the Galaxy*. Pan, 1979.

[2] L. Alboul and R. van Damme. Polyhedral metrics in surface reconstruction. In G. Mullineux, editor, *The Mathematics of Surfaces VI*, pages 171–200, Oxford, 1996. Clarendon Press.

[3] L. Alboul and R. van Damme. Polyhedral metrics in surface reconstruction: tight triangulations. In T. Goodman and R. Martin, editors, *The Mathematics of Surfaces VII*, pages 309–336, Oxford, 1997. Clarendon Press.

[4] M.-E. Algorri and F. Schmitt. Surface reconstruction from unstructured 3D data. *Computer Graphics Forum*, 15:47–60, 1996.

[5] N. Amenta, M. Bern, and M. Kamvysselis. A new Voronoi-based surface reconstruction algorithm. In *ACM Computer Graphics (SIGGRAPH '98 Proceedings)*, pages 415–421, 1998.

[6] I. Applegarth, P. D. Kaklis, and S. Wahl, editors. *Benchmark Tests on the Generation of Fair Shapes subject to Constraints*. B. G. Teubner, Stuttgart, Leipzig, 2000.

[7] C. L. Bajaj, F. Bernardini, and G. Xu. Automatic reconstruction of surfaces and scalar fields from 3D scans. In *ACM Computer Graphics (SIGGRAPH '95 Proceedings)*, pages 109–118, 1995.

[8] I. Barrodale and A. Young. A note on numerical procedures for approximation by spline functions. *The Computer Journal*, 9:318–320, 1966.

[9] R. H. Bartels, J. C. Beatty, and B. A. Barsky. *An introduction to splines for use in computer graphics and geometric modeling*. Kaufmann, Los Altos, 1987.

[10] G. Baszenski and L. L. Schumaker. Use of simulated annealing to construct triangular facet surfaces. In P.-J. Laurent, A. LeMéhauté, and L. L. Schumaker, editors, *Curves and Surfaces*, pages 27–32, New York, 1991. Academic Press.

[11] C. Bennis, J.-M. Vézien, and G. Iglésias. Piecewise surface flattening for non-distorted texture mapping. In *ACM Computer Graphics (SIGGRAPH '91 Proceedings)*, pages 237–246, 1991.

[12] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, and G. Taubin. The ball-pivoting algorithm for surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, pages 349–359, 1999.

[13] M. I. Bloor, M. J. Wilson, and H. Hagen. The smoothing properties of variational schemes for surface design. *Computer Aided Geometric Design*, 12:381–394, 1995.

[14] J.-D. Boissonat. Geometric structures for three-dimensional shape representation. *ACM Transactions on Graphics*, 3:266–286, 1984.

[15] W. L. Briggs, V. E. Henson, and S. F. McCormick. *A Multigrid Tutorial*. SIAM Books, 2nd edition, 2000.

[16] S. Campagna, L. Kobbelt, and H.-P. Seidel. Efficient generation of hierarchical triangle meshes. In Robert Cripps, editor, *The Mathematics of Surfaces VIII*, pages 105–123. Information Geometers Ltd., 1998.

[17] E. Catmull and J. Clark. Recursively generated B-Spline surfaces on arbitrary topological meshes. *Computer-Aided Design*, 10(6):350–355, 1978.

[18] G. Celniker and D. Gossard. Deformable curve and surface finite-element for free-form shape design. In *ACM Computer Graphics (SIGGRAPH '91 Proceedings)*, pages 257–266, 1991.

[19] A. Certain, J. Popović, T. DeRose, T. Duchamp, D. Salesin, and W. Stuetzle. Interactive multiresolution surface viewing. In *ACM Computer Graphics (SIGGRAPH '96 Proceedings)*, pages 91–98, 1996.

[20] J. Chen and Y. Han. Shortest paths on a polyhedron. *International Journal of Computational Geometry & Applications*, 6:127–144, 1996.

[21] A. K. Cline. Rate of convergence of Lawson's algorithm. *Mathematics of Computation*, 26:167–176, 1972.

[22] R. Courant and D. Hilbert. *Methods of Mathematical Physics*, volume 1. Wiley, New York, 1953.

[23] C. de Boor. *A practical guide to splines*. Springer, New York, 1987.

[24] B. N. Delaunay. Sur la sphère cide. *Izvestia Akademia Nauk SSSR*, VII:793–800, 1934.

[25] M. Desbrun, M. Meyer, P. Schröder, and A. H. Barr. Implicit fairing of irregular meshes using diffusion and curvature flow. In *ACM Computer Graphics (SIGGRAPH '99 Proceedings)*, pages 317–324, 1999.

[26] P. Dierckx. *Curve and surface fitting with splines.* Oxford Clarendon Press, 1995.

[27] U. Dietz. Fair surface reconstruction from point clouds. In M. Dæhlen, T. Lyche, and L. L. Schumaker, editors, *Mathematical Methods for Curves and Surfaces II*, pages 79–86. Vanderbilt University Press, 1998.

[28] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numer. Math.*, 1:269–271, 1959.

[29] M. P. do Carmo. *Differential Geometry of curves and surfaces.* Prentice-Hall Inc., 1976.

[30] P. Dombrowski. *150 Years after Gauss' "Disquisitiones Generales Circa Superficies Curvas".* Paris Societé Mathématique de France, 1979.

[31] D. Doo and M. Sabin. Behaviour of recursive division surfaces near extraordinary points. *Computer-Aided Design*, 10(6):356–360, 1978.

[32] N. Dyn, J. Gregory, and D. Levin. A butterfly subdivision scheme for surface. *ACM Transactions on Graphics*, 9:160–169, 1990.

[33] N. Dyn, K. Hormann, S.-J. Kim, and D. Levin. Optimizing 3D triangulations using discrete curvature analysis. In T. Lyche and L. L. Schumaker, editors, *Mathematical Methods for Curves and Surfaces: Oslo 2000*, pages 135–146. Vanderbilt University Press, 2001.

[34] N. Dyn, D. Levin, and S. Rippa. Data dependent triangulations for piecewise linear interpolation. *IMA J. Numer. Anal.*, 10:137–154, 1990.

[35] M. Eck, T. DeRose, T. Duchamp, H. Hoppe, M. Lounsbery, and W. Stuetzle. Multiresolution analysis of arbitrary meshes. In *ACM Computer Graphics (SIGGRAPH '95 Proceedings)*, pages 173–182, 1995.

[36] H. Edelsbrunner and E. P. Mücke. Three-Dimensional alpha shapes. *ACM Transactions on Graphics*, 13:43–72, 1994.

[37] J. Eells and J. H. Sampson. Harmonic mappings of Riemannian manifolds. *American Journal of Mathematics*, 86:109–160, 1964.

[38] R. E. Esch and W. L. Eastman. Computational methods for best spline function approximation. *Journal of Approximation Theory*, 2:85–96, 1969.

[39] G. Farin. *Curves and surfaces for computer aided geometric design.* Acadamic Press, Boston, 1993.

[40] G. E. Fasshauer and L. L. Schumaker. Minimal energy surfaces using parametric splines. *Computer Aided Geometric Design*, 13:45–79, 1996.

[41] R. Fletcher. *Practical Methods of Optimization.* Wiley-Interscience, Chichester, 1990.

[42] M. S. Floater. Parameterization and smooth approximation of surface triangulations. *Computer Aided Geometric Design*, 14:231–250, 1997.

[43] M. S. Floater. How to approximate scattered data by least squares. Technical Report STF42 A98013, SINTEF, Oslo, 1998.

[44] M. S. Floater. One-to-one piecewise linear mappings over triangulations. Preprint, 2001.

[45] M. S. Floater and K. Hormann. Parameterization of triangulations and unorganized points. In M. S. Floater, A. Iske, and E. Quak, editors, *Principles of Multiresolution in Geometric Modelling*, Summer School Lecture Notes, Munich, August 22–30, 2001. Springer.

[46] M. S. Floater, K. Hormann, and M. Reimers. Parameterization of manifold triangulations. In C. K. Chui, L. L. Schumaker, and J. Stoeckler, editors, *Approximation Theory X*. Vanderbilt University Press, 2001. To appear.

[47] M. S. Floater and M. Reimers. Meshless parameterization and surface reconstruction. *Computer Aided Geometric Design*, 18:77–92, 2001.

[48] D. R. Forsey and R. H. Bartels. Hierarchical B-spline refinement. In *ACM Computer Graphics (SIGGRAPH '88 Proceedings)*, pages 205–212, 1988.

[49] D. R. Forsey and R. H. Bartels. Surface fitting with hierarchical splines. *ACM Transactions on Graphics*, 14:134–161, 1995.

[50] S. Fortune. A sweepline algorithm for Voronoi diagrams. *Algorithmica*, 2:153–174, 1987.

[51] R. Franke and G. M. Nielson. Scattered data interpolation and applications: A tutorial and survey. In H. Hagen and D. Roller, editors, *Geometric Modeling*, pages 131–161. Springer, Berlin, 1991.

[52] C. F. Gauß. Disquisitiones generales circa superficies curvas. *Comm. Soc. Gött.*, 6:99–146, 1827.

[53] G. H. Golub and C. F. van Loan. *Matrix Computations*. The John Hopkins University Press, Baltimore, Maryland, 1989.

[54] M. Gopi, S. Krishnan, and C. T. Silva. Surface reconstruction based on lower dimensional localized Delauney triangulation. In *Computer Graphics Forum (EUROGRAPHICS '00 Proceedings)*, volume 19, pages 119–130, 2000.

[55] G. Greiner. Surface construction based on variational principles. In P. J. Laurent, A. LeMéhauté, and L. L. Schumaker, editors, *Wavelets, Images, and Surface Fitting*, pages 277–286. AK Peters, Wellesley, 1994.

[56] G. Greiner. Variational design and fairing of spline surfaces. In *Computer Graphics Forum (EUROGRAPHICS '94 Proceedings)*, volume 13, pages 143–154, 1994.

[57] G. Greiner. Data dependent fairness functionals. Preprint, 1996.

[58] G. Greiner and K. Hormann. Interpolating and approximating scattered 3D data with hierarchical tensor product B-splines. In A. Méhauté, C. Rabut, and L. L. Schumaker, editors, *Surface Fitting and Multiresolution Methods*, pages 163–172. Vanderbilt University Press, 1997.

[59] G. Greiner, J. Loos, and W. Wesselink. Data dependent thin plate energy and its use in interactive surface modeling. In *Computer Graphics Forum (EUROGRAPHICS '96 Proceedings)*, volume 15, pages 175–185, 1996.

[60] L. J. Guibas, D. E. Knuth, and M. Sharir. Randomized incremental construction of Delaunay and Voronoi diagrams. *Algorithmica*, 7:381–413, 1992.

[61] H. Hagen, G.-P. Bonneau, and S. Hahmann. Variational design and surface interrogation. *Computer Graphics Forum (EUROGRAPHICS '93 Proceedings)*, 13:447–459, 1993.

[62] M. Halstead, M. Kass, and T. DeRose. Efficient, fair interpolation using Catmull-Clark surfaces. In *ACM Computer Graphics (SIGGRAPH '93 Proceedings)*, pages 35–44, 1993.

[63] R. L. Hardy. Multiquadric equation of topography and other irregular surfaces. *J. Geophysical Res.*, 76:1905–1915, 1971.

[64] P. S. Heckbert and M. Garland. Survey of polygonal surface simplification algorithms. Technical report, Carnegie Mellon University, 1997.

[65] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. In *ACM Computer Graphics (SIGGRAPH '92 Proceedings)*, pages 71–78, 1992.

[66] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Mesh optimization. In *ACM Computer Graphics (SIGGRAPH '93 Proceedings)*, pages 19–26, 1993.

[67] K. Hormann. Glatte Approximation mit hierarchischen Splineflächen. Diplomarbeit, Mathematisches Institut, Universität Erlangen-Nürnberg, 1997.

[68] K. Hormann. Fitting free form surfaces. In B. Girod, G. Greiner, and H. Niemann, editors, *Principles of 3D Image Analysis and Synthesis*, pages 192–202. Kluwer Academic Publishers, Boston, 2000.

[69] K. Hormann and G. Greiner. MIPS: an efficient global parametrization method. In P.-J. Laurent, P. Sablonnière, and L. L. Schumaker, editors, *Curve and Surface Design: Saint-Malo 1999*, pages 153–162. Vanderbilt University Press, 2000.

[70] K. Hormann and G. Greiner. Quadrilateral remeshing. In B. Girod, G. Greiner, H. Niemann, and H.-P. Seidel, editors, *Vision, Modeling and Visualization 2000*, pages 153–162. infix, 2000.

[71] K. Hormann, G. Greiner, and S. Campagna. Hierarchical parametrization of triangulated surfaces. In B. Girod, H. Niemann, and H.-P. Seidel, editors, *Vision, Modeling and Visualization '99*, pages 219–226. infix, 1999.

[72] K. Hormann, U. Labsik, and G. Greiner. Remeshing triangulated surfaces with optimal parametrizations. *Computer-Aided Design*, 33(11):779–788, 2001.

[73] J. Hoschek. Intrinsic parametrization for approximation. *Computer Aided Geometric Design*, 5:27–31, 1988.

[74] J. Hoschek and D. Lasser. *Fundamentals of Computer Aided Geometric Design*. AK Peters, Wellesley MA, 1993.

[75] F. Isselhard, G. Brunnett, and T. Schreiber. Polyhedral reconstruction of 3D objects by tetrahedral removement. Technical Report No. 288/97, University of Kaiserslautern, Germany, 1997.

[76] B. Jüttler. Computational methods for discrete parametric ℓ_1 and ℓ_∞ curve fitting. *International Journal of Shape Modeling*, 4:21–34, 1998.

[77] B. Kaneva and J. O'Rourke. An implementation of Chen & Han's shortest paths algorithm. In *Proceedings of the 12th Canadian Conference on Computational Geometry*, 2000. To appear.

[78] A. Khodakovsky, W. Sweldens, and P. Schröder. Progressive geometry compression. In *ACM Computer Graphics (SIGGRAPH '00 Proceedings)*, pages 271–278, 2000.

[79] R. Kimmel and J. A. Sethian. Computing geodesic paths on manifolds. In *Proceedings of National Academy of Sciences*, volume 95, pages 8431–8435, 1998.

[80] W. Klingenberg. *A Course in Differential Geometry*. Springer, Berlin, Heidelberg, 1978.

[81] L. Kobbelt. Discrete fairing. In T. Goodman and R. Martin, editors, *The Mathematics of Surfaces VII*, pages 101–131, Oxford, 1997. Clarendon Press.

[82] L. Kobbelt. $\sqrt{3}$-subdivision. In *ACM Computer Graphics (SIGGRAPH '00 Proceedings)*, pages 102–112, 2000.

[83] L. Kobbelt, S. Campagna, and H.-P. Seidel. A general framework for mesh decimation. In *Proceedings of Graphics Interface '98*, pages 43–50, 1998.

[84] L. Kobbelt, J. Vorsatz, U. Labsik, and H.-P. Seidel. A shrink wrapping approach to remeshing polygonal surfaces. In *Computer Graphics Forum (EUROGRAPHICS '99 Proceedings)*, volume 18, pages 119–130, 1999.

[85] U. Labsik and G. Greiner. Interpolatory $\sqrt{3}$-subdivision. In *Computer Graphics Forum (EUROGRAPHICS '00 Proceedings)*, volume 19, pages 131–138, 2000.

[86] U. Labsik, K. Hormann, and G. Greiner. Using most isometric parametrizations for remeshing polygonal surfaces. In *Geometric Modelling and Processing 2000 Proceedings*, pages 220–228, 2000.

[87] U. Labsik, L. Kobbelt, R. Schneider, and H.-P. Seidel. Progressive transmission of subdivision surfaces. *Computational Geometry*, 15:25–39, 2000.

[88] C. L. Lawson. *Contributions to the Theory of Linear Least Maximum Approximation.* PhD thesis, University of California, Los Angeles, CA, 1961.

[89] C. L. Lawson. Software for C^1 surface interpolation. In J. R. Rice, editor, *Mathematical Software III*, pages 161–194. Academic Press, New York, 1977.

[90] A. W. F. Lee, W. Sweldens, P. Schröder, L. Cowsar, and D. Dobkin. MAPS: Multiresolution adaptive parameterization of surfaces. In *ACM Computer Graphics (SIGGRAPH '98 Proceedings)*, pages 95–104, 1998.

[91] S. Lee, G. Wolberg, and S. Y. Shin. Scattered data interpolation with multilevel B-splines. *IEEE Transactions on Visualization and Computer Graphics*, 3:1–17, 1997.

[92] B. Lévy and J.-L. Mallet. Non-distorted texture mapping for sheared triangulated meshes. In *ACM Computer Graphics (SIGGRAPH '98 Proceedings)*, pages 343–352, 1998.

[93] C. Loop. Smooth subdivision surfaces based on triangles. Master's thesis, University of Utah, Seattle, 1987.

[94] M. Lounsbery, T. DeRose, and J. Warren. Multiresolution analysis for surfaces of arbitrary topological type. *ACM Transactions on Graphics*, 16:34–73, 1997.

[95] W. Ma and J. P. Kruth. Parameterization of randomly measured points for least squares fitting of B-spline curves and surfaces. *Computer-Aided Design*, 27:663–675, 1995.

[96] J. Maillot, H. Yahia, and A. Verroust. Interactive texture mapping. In *ACM Computer Graphics (SIGGRAPH '93 Proceedings)*, pages 27–34, 1993.

[97] R. Mencl and H. Müller. Graph-based surface reconstruction using structures in scattered point sets. In *Proceedings of CGI '98*, pages 298–311, 1998.

[98] R. Mencl and H. Müller. Interpolation and approximation of surfaces from three-dimensional scattered data points. *State of the Art Report (STAR) for EUROGRAPHICS '98*, 1998.

[99] H. P. Moreton and C. H. Séquin. Functional optimization for fair surface design. In *ACM Computer Graphics (SIGGRAPH '92 Proceedings)*, pages 167–176, 1992.

[100] T. S. Motzkin and J. L. Walsh. Polynomials of best approximation on a real finite point set *i*. *TAMS*, 91:231–245, 1959.

[101] G. M. Nielson. Scattered data modeling. *Computer Graphics and Applications*, pages 60–70, 1993.

[102] G. Nürnberger and F. Zeilfelder. Developments in bivariate spline interpolation. *J. Comput. Appl. Math.*, 121:125–152, 2000.

[103] R. Pfeifle. *Approximation und Interpolation mit quadratischen Dreiecks-B-Splines*. PhD thesis, IFI 9, University Erlangen-Nürnberg, 1995.

[104] U. Pinkall and K. Polthier. Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics*, 2(1):15–36, 1993.

[105] E. Polak. *Computational Methods in Optimization*. Academic Press, New York, 1971.

[106] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction*. Springer, 1985.

[107] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C*. Cambridge University Press, 2nd edition, 1992.

[108] M. Python. *The Meaning of Life*. Mandarin Paperbacks, 1991.

[109] E. Quak and L. L. Schumaker. Cubic spline interpolation using data dependent triangulations. *Computer Aided Geometric Design*, 7:293–301, 1990.

[110] J. R. Rice and K. H. Usow. The Lawson algorithm and extensions. *Math. Comp.*, 22:118–127, 1968.

[111] B. Riemann. *Grundlagen für eine allgemeine Theorie der Functionen einer veränderlichen complexen Größe*. PhD thesis, Universität Göttingen, 1851.

[112] A. Riepl. Interpolation gestreuter Daten mit krümmungsminimierenden Flächen. Diplomarbeit, IFI 9, University Erlangen-Nürnberg, 1995.

[113] J. Ruppert. A Delaunay refinement algorithm for quality 2-dimensional mesh generation. *J. Algorithms*, 18:548–585, 1995.

[114] N. S. Sapidis. *Designing fair curves and surfaces.* SIAM, Philadelphia, 1994.

[115] B. Sarkar and C.-H. Menq. Parameter optimization in approximating curves and surfaces to measurement data. *Computer Aided Geometric Design*, 8:267–290, 1991.

[116] T. Schreiber and G. Brunnett. Approximating 3D objects from measured points. In *Proceedings of 30th ISATA*, Florence, Italy, 1997.

[117] P. Schröder and W. Sweldens. Spherical wavelets: Efficiently representing functions on the sphere. In *ACM Computer Graphics (SIGGRAPH '95 Proceedings)*, pages 161–172, 1995.

[118] L. L. Schumaker. Fitting surfaces to scattered data. In G. G. Lorentz, C. K. Chui, and L. L. Schumaker, editors, *Approximation Theory II*, pages 203–268. Academic Press, Boston, 1976.

[119] L. L. Schumaker. *Spline Functions: Basic Theory.* John Wiley & Sons, New York, 1981.

[120] L. L. Schumaker. Triangulation methods. In C. K. Chui, L. L. Schumaker, and F. Utreras, editors, *Topics in Multivariate Approximation*, pages 219–232. Academic Press, New York, 1987.

[121] L. L. Schumaker. Computing optimal triangulations using simulated annealing. *Computer Aided Geometric Design*, 10:329–345, 1993.

[122] H.-P. Seidel. Geometrische Grundlagen des Computer Aided Geometric Design. In *Geometrie und ihre Anwendungen*, pages 201–246. Hanser, 1994.

[123] A. Sheffer and E. de Sturler. Surface parameterization for meshing by triangulation flattening. In *Proceedings of the International Meshing Round Table Conference 2000*, 2000.

[124] D. Shepard. A two dimensional interpolation function for irregularly spaced data. In *Proc. 23rd Nat. Conf. ACM*, pages 517–524, 1968.

[125] J. R. Shewchuk. Triangle: Engineering a 2D quality mesh generator and Delaunay triangulator. In M. C. Lin and D. Manocha, editors, *Applied Computational Geometry: Towards Geometric Engineering*, volume 1148 of *Lecture Notes in Computer Science*, pages 203–222. Springer, 1996.

[126] R. Sibson. Locally equiangular triangulations. *The Computer Journal*, 2:243–245, 1973.

[127] J. Stoer and R. Bulirsch. *Introduction to Numerical Analysis*. Springer, New York, 1980.

[128] W. Straßer and H.-P. Seidel. *Theory and Practice of Geometric Modeling*. Springer, 1989.

[129] G. Taubin. A signal processing approach to fair surface design. In *ACM Computer Graphics (SIGGRAPH '95 Proceedings)*, pages 351–358, 1995.

[130] W. T. Tutte. How to draw a graph. *Proc. London Math. Soc.*, 13:743–768, 1963.

[131] R. van Damme and L. Alboul. Tight triangulations. In L.L. Schumaker M. Daehlen, T. Lyche, editor, *Mathematical Methods for Curves and Surfaces*, pages 517–526. Vanderbilt University Press, 1995.

[132] H. A. van der Vorst. Bi-CGSTAB: A fast and smoothly converging variant of bi-CG for the solution of nonsymmetric linear systems. *SIAM Journal of Scientific Statistical Computing*, 13:631–644, 1992.

[133] R. Vanderbei. LOQO. Available at `http://orfe.princeton.edu/~loqo`.

[134] R. S. Varga. *Matrix Iterative Analysis*. Prentice-Hall, Englewood Cliffs, NJ, 1962.

[135] R. C. Veltkamp. *Closed Object Boundaries from Scattered Points*, volume 885 of *Lecture Notes in Computer Science*. Springer, Berlin, Heidelberg, New York, 1994.

[136] M. von Golitschek and L. L. Schumaker. Data fitting by penalized least squares. In J. C. Mason, editor, *Algorithms for approximation II*, pages 210–227, Shrivenham, 1988.

[137] G. A. Watson. Choice of norms for data fitting and function approximation. In *Acta Numerica*, volume 8, pages 337–377. Cambridge University Press, 1999.

[138] W. Welch and A. Witkin. Variational surface modeling. In *ACM Computer Graphics (SIGGRAPH '92 Proceedings)*, pages 157–166, 1992.

[139] W. Wesselink. *Variational Modeling of Curves and Surfaces*. PhD thesis, University of Technology, Eindhoven, 1996.

[140] D. M. Young. *Iterative Solution of Large Linear Systems*. Academic Press, New York, 1971.

[141] D. Zorin, P. Schröder, and W. Sweldens. Interpolating subdivision for meshes with arbitrary topology. In *ACM Computer Graphics (SIGGRAPH '96 Proceedings)*, pages 189–192, 1996.

[142] D. Zorin, P. Schröder, and W. Sweldens. Interactive multiresolution mesh editing. In *ACM Computer Graphics (SIGGRAPH '97 Proceedings)*, pages 259–268, 1997.