TECHNISCHE UNIVERSITÄT WIEN

VIENNA UNIVERSITY OF TECHNOLOGY

D I P L O M A T H E S I S

Planar quad meshes from relative principal curvature lines

Institute of
Discrete Mathematics and Geometry
Vienna University of Technology

supervised by o.Univ-Prof. Dr. Helmut Pottmann

Author:
Alexander Schiftner
Hustergasse 9/10, 1140 Wien

Date Signature

Abstract

This thesis proposes a technique for the approximation of surfaces by PQ meshes. These are meshes with planar and mostly quadrilateral faces. Relative differential geometry is used for the generation of conjugate curve networks. It is well known that a discrete choice of curves from these networks naturally leads to meshes with quadrilateral faces, which are in turn planarized using optimization algorithms. The possibility to choose a convex object, defining the relative differential geometry, gives rise to bounding the minimum intersecting angle of conjugate curves from below. This is a requirement for practical applications. Methods from convex geometry and Fourier analysis on the unit sphere are utilized to allow an interactive layout of the conjugate curve networks. This is followed by a discussion of the possibility to influence singularities in the conjugate curve networks, and consequently in the resulting PQ meshes. In a new approach, non-flat isotropic subdomains can be given an anisotropy, which is a replacement for the smoothing techniques introduced in recent papers on quad-dominant meshing. Finally, examples from architecture are used for demonstrating the capabilities of these techniques.

Kurzfassung

In dieser Diplomarbeit wird ein Verfahren zur Approximation von Flächen mit PQ Netzen vorgestellt. PQ Netze bestehen aus planaren und hauptsächlich viereckigen Flächenstücken. Relative Differentialgeometrie wird dazu benutzt um konjugierte Kurvennetze zu erzeugen, welche auf natürliche Weise zu Netzen mit viereckigen Flächenstücken führen. Die Flächenstücke werden danach mit Hilfe von Optimierungsmethoden planarisiert. Durch die Wahl einer entsprechenden konvexen Fläche, welche die relative Differentialgeometrie definiert, kann der minimale Schnittwinkel konjugierter Kurven nach unten beschränkt werden. Dies ist eine Forderung, die in praktischen Anwendungen auftaucht. Methoden der konvexen Geometrie, sowie der Fourieranalyse auf der Einheitssphäre, werden dazu verwendet um die Erzeugung von konjugierten Kurvennetzen interaktiv vorzunehmen. Darauf folgend wird beschrieben wie Singularitäten in den konjugierten Kurvennetzen, und dadurch auch in den resultierenden PQ Netzen, beeinflusst werden können. Darüber hinaus können isotrope Teilbereiche wie anisotrope behandelt werden. Dies führt zu einem Ersatz der Glättungstechniken, die in kürzlich erschienenen Veröffentlichungen zur Erzeugung von Viereckснetzen vorgestellt wurden. Schlussendlich werden die Möglichkeiten der untersuchten Methoden an Beispielen aus der Architektur demonstriert.

Acknowledgements

It is about five years ago, that I decided to quit working and take the chance of getting a full scholarship from the Austrian government to study mathematics. After nearly four years of regular work, this was not an easy decision to be taken. The idea of leaving great colleagues and an interesting job at Austrosoft behind, caused an uneasy feeling in my stomach. Nevertheless my decision had been taken, and it proved to be the right one. First of all I could work on part-time, which was beneficial for both sides and allowed me to keep in touch with some real-world problems. Thanks a lot to my colleagues and Austrosoft, who supported me through the last years! Besides working I became used to university life quite fast. I discovered the world of mathematics and soon made new friends. The past five years went by really fast, thanks to a lot of highlights on the way. These including a great time with my colleagues, friends and family, interesting lectures, BEST, courses at universities in Copenhagen, Paris und Munich and being a summer student at CERN.

About seven months ago, another important decision had to be taken: choosing a topic for my diploma thesis. I went on the look-out for different possibilities, and soon I had to choose from several topics that I was interested in. Again I took a decision that I do not regret. Differential geometry had sparked my interest already during the lectures one year before, and the outlook for a practical application provided an important motivation. Many thanks go to my supervisor Helmut Pottmann, who gave fruitful advice and provided me with a place to work at the institute, which definitely boosted my performance. Further thanks go to Yong-Liang Yang for passively teaching me by reading his source code, Yang Liu for patiently answering my questions about his software and Heinz Schmiedhofer for constructing some meshes for testing purposes. Now having finished the last but one step of my studies, I look back with satisfaction and forward to what may come.

Last but not least I want to thank my family for always supporting me. Unlimited thanks and kisses go to Marion, who has done without me in numerous nights and has always been there when I needed her most. I love you.

There is just one thing left to say:

"So long, and thanks for all the fish."
Douglas Adams

Contents

Preamble

The topic of this thesis was motivated by architecture. Architects commonly use freeform surfaces in their designs. These freeform surfaces are used for modelling roofs, walls or other components of buildings. Meshes arise in a natural way in the construction of such surfaces, because of constraints depending on the material used. As an example consider a roof built from glass, which can not be produced of arbitrary shape and size. Moreover a supporting structure carrying the roof is needed.

The use of quadrilateral meshes in the construction of freeform surfaces is motivated by different aspects. First of all stands aesthetics of a construction. Quadrilateral faces naturally define two distinguished directions, which can be used to emphasize the shape of a surface. Another aspect is the cost of a construction. Consider the supporting structure for a roof made of quadrilaterals respectively triangles. Less faces will meet in the vertices of the supporting structure when using quadrilaterals. In addition, if the quadrilaterals and triangles are of similar diameter, less quadrilaterals will be needed for covering the surface. Planarity of the faces arises as a further matter of cost.

There are various approaches for the construction of planar quad meshes. For example one can start from a polygonal boundary of a surface by successively cutting and glueing pieces of wooden plates. In contrast, the approach described in this thesis starts from a given freeform surface. Exploiting the geometry of this surface, one wants to construct an approximating planar quad mesh. The question addressed in this thesis is, how relative differential geometry can be applied to this approach.

1 Introduction

Given a surface, one wants to generate an approximating mesh consisting of planar quadrilateral[1] faces. Such meshes are commonly called PQ meshes. A mesh consisting of mainly quadrilateral faces could be generated by joining neighbouring faces of an isotropic triangular mesh, resulting in a mesh that does not reflect the geometry of the approximated surface. The planarity of the quadrilaterals leads to meshes that can be seen as discrete counterparts of so-called conjugate curve networks on a surface, this will be explained in section 4.1. PQ meshes thus reveal the underlying geometry and are anisotropic in general. In practice this will lead to the need of allowing not only quads but also some triangles or planar n-sided polygons (n-gons) in the resulting PQ mesh.

In [2] and [26] investigation has been done in quad-dominant meshing using principal curvature lines, which form a special conjugate curve network. The authors do not focus on planarity, but this can be addressed by the methods described in [23]. PQ meshes from principal curvature lines are also of interest from artistic and psychological viewpoints, see [14]. A disadvantage of using principal curvature lines for the generation of PQ meshes lies in the fact that they are uniquely determined on most surfaces. A lack of flexibility is the consequence. The generalization of differential geometry to relative differential geometry leads to relative principal curvature lines, which form conjugate curve networks as well[2]. A lot of flexibility is gained by the possibility to choose a specific relative differential geometry.

The thesis is structured as follows:

- Chapter 2 gives a short introduction to differential geometry, focussing on what will be needed throughout the thesis and providing a consistent notation. In the second section the concepts of relative differential geometry are motivated and summarized.
- Chapter 3 shortly describes how discrete surfaces can be represented by meshes. The surfaces which one starts from are usually given by triangular meshes and thus not differentiable, which leads to the nontrivial task of defining and computing curvature. This chapter also shortly addresses the method used for curvature computation within the accompanying software.
- Chapter 4 describes how to generate quad meshes approximating a surface using relative principal curvature lines. It addresses the difficulties of this step and explains how one can gain PQ meshes from quad meshes.

[1] "Quad" will be used as a short form of quadrilateral.

[2] This will be described in section 2.2. See [29] for another application of relative differential geometry.

- Chapter 5 shows two types of representations that were used for the relativesphere defining the relative differential geometry. Details regarding the calculation of curvature specific to the type of representation are given. Furthermore a method for prescribing curvature at certain points of the relative sphere is described, which is an important tool for influencing the resulting conjugate curve network.
- Chapter 6 describes important and critical details of the implementation in the accompanying software.
- Chapter 7 shows and discusses results for some freeform surfaces from architecture.
- Chapter 8 concludes and describes issues that were found to be of interest for further investigation.

2 Relative Differential Geometry

2.1 Basics from Differential Geometry

This section gives a brief summary of basic facts from differential geometry. The focus will be on two-dimensional embedded surfaces $\mathcal{S} \subset \mathbb{R}^3$. In the following a single parametric representation

$$\boldsymbol{x} : D \subset \mathbb{R}^2 \to V \subset \mathcal{S} \subset \mathbb{R}^3 : u \in D \mapsto \boldsymbol{x}(u) \in \mathcal{S} \tag{2.1}$$

of an open subset $V \subset \mathcal{S}$ will be used to define geometric objects or properties like tangential space, normal vector, curvature and so forth. These objects can be shown to be independent of a specific parameterisation by defining $\mathcal{S}$ in an appropriate way. In this chapter $\mathcal{S}$ is required to be a regular surface (for a definition of regular surface and further details see [12]). This basically means that $\mathcal{S}$ can *locally* be parametrized by diffeomorphisms as defined in 2.1.

2.1.1 Tangent Space

The partial derivatives of the parameterisation $\boldsymbol{x}$ shall be denoted by $\boldsymbol{x}_{,1}$ and $\boldsymbol{x}_{,2}$, their evaluation at a surface point[1] $p = \boldsymbol{x}(q), q \in D$ by $\boldsymbol{x}_{p,1}$ and $\boldsymbol{x}_{p,2}$. At a given point $p \in \mathcal{S}$ they span the tangent space $T_p(\mathcal{S})$ of $\mathcal{S}$ at p, which will be considered as a subspace of $\mathbb{R}^3$. Partial derivatives of higher order will be denoted analogously: $\boldsymbol{x}_{,ij} := \frac{\partial^2}{\partial_i \partial_j}\boldsymbol{x}$. $\boldsymbol{x}_{,1}$ and $\boldsymbol{x}_{,2}$ clearly depend on $\boldsymbol{x}$ and are called an associated basis to $\boldsymbol{x}$. Note that $T_p(\mathcal{S})$ does not depend on $\boldsymbol{x}$. The regularity of $\mathcal{S}$ ensures that the dimension of $T_p(\mathcal{S})$ is two for all $p \in \mathcal{S}$. The union of all tangent spaces[2] of $\mathcal{S}$ is called the *tangent bundle* $T\mathcal{S}$.

2.1.2 First Fundamental Form

Let $\mathbb{R}^3$ be equipped with the Euclidean scalar product, which will be denoted by $\langle ., . \rangle$. As the tangent spaces $T_p(\mathcal{S})$ are subspaces of $\mathbb{R}^3$, a scalar product can be defined in them by restricting $\langle ., . \rangle$ to $T_p(\mathcal{S}) \times T_p(\mathcal{S})$. This bilinear form is called first fundamental form of $\mathcal{S}$ in Euclidean space.

Consider the parameterisation $\boldsymbol{x}$. The functions $E := \langle \boldsymbol{x}_{,1}, \boldsymbol{x}_{,1} \rangle$, $F := \langle \boldsymbol{x}_{,1}, \boldsymbol{x}_{,2} \rangle$, $G := \langle \boldsymbol{x}_{,2}, \boldsymbol{x}_{,2} \rangle$ are called the associated coefficients of the first fundamental form. If vectors $v, w \in T_p(\mathcal{S})$ are given by their coordinate vectors $(v_1, v_2), (w_1, w_2)$ with respect to the associated basis, then $\langle v, w \rangle = v_1 w_1 E + (v_1 w_2 + v_2 w_1) F + v_2 w_2 G$.

[1] Throughout this chapter p and q will be used as defined here.

[2] This is not a strict definition, see e.g. [12] or [3].

2.1.3 Differentiability

A function $f : V \subset \mathcal{S} \to \mathbb{R}$ defined on an open subset V of $\mathcal{S}$ is called differentiable at $p \in V$, if $f \circ \boldsymbol{x} : D \subset \mathbb{R}^2 \to \mathbb{R}$ is differentiable at $\boldsymbol{x}^{-1}(p)$, f is called differentiable if it is differentiable at all $p \in V$. Differentiability of higher order is defined analogously. The notation $f \in C^k(V, \mathbb{R})$ will be used for a k-times differentiable f.

This definition can easily be extended to functions mapping from $\mathcal{S}$ to another regular surface $\mathcal{R} \subset \mathbb{R}^3$, which shall be parametrized by $\boldsymbol{y}$. A function $g : V \subset \mathcal{S} \to \mathcal{R}$, defined on an open subset V of $\mathcal{S}$, is called differentiable at $p \in V$, if $\boldsymbol{y}^{-1} \circ g \circ \boldsymbol{x} : D \subset \mathbb{R}^2 \to \mathbb{R}^2$ is differentiable at $\boldsymbol{x}^{-1}(p)$. Moreover g is called differentiable if it is differentiable at all $p \in V$, which will be denoted by $g \in C^k(\mathcal{S}, \mathcal{R})$. The regularity of $\mathcal{S}$ ensures that these notions of differentiability do not depend on the parameterisation $\boldsymbol{x}$.

Using the concept of tangent spaces, the differential of a mapping g between two regular surfaces can be defined as follows:
Given a vector $v \in T_p(\mathcal{S})$ there is a curve $c : (-\epsilon, \epsilon) \to \mathcal{S} : t \mapsto c(t) = \boldsymbol{x}(u_1(t), u_2(t))$ such that $c(0) = p$ and $c'(0) = v$. The mapping g gives a curve $g \circ c : (-\epsilon, \epsilon) \to \mathcal{R}$ with $(g \circ c)(0) = g(p) \in \mathcal{R}$ and $(g \circ c)'(0) = w \in T_{g(p)}(\mathcal{R})$. The vector w does not depend on the choice of c and thus a mapping $dg_p : T_p(\mathcal{S}) \to T_{g(p)}(\mathcal{R})$ can be defined, which is clearly linear.

If one considers the case $\mathcal{R} = \mathbb{R}$ the tangent space $T_{g(p)}(\mathbb{R})$ can be identified with $\mathbb{R}$ and thus the differential is a linear form on $T_p(\mathcal{S})$. The tangent space $T_p(\mathcal{S})$ equipped with $\langle ., . \rangle$ is a Hilbert space, which implies that there is a unique $w \in T_p(\mathcal{S})$ such that $\forall v \in T_p(\mathcal{S}) : dg_p(v) = \langle v, w \rangle$. This vector will be denoted by $\nabla^0_p g$ and is called the gradient of g with respect to $\mathcal{S}$. The coordinates t_i of this vector in the associated basis are given by

$$\frac{\partial}{\partial_i} g(\boldsymbol{x}) = dg_p(\boldsymbol{x}_{,i}) = \langle \nabla^0_p g, \boldsymbol{x}_{,i} \rangle = t_i \langle \boldsymbol{x}_{,i}, \boldsymbol{x}_{,i} \rangle. \tag{2.2}$$

If $g : \mathcal{S} \mapsto \mathbb{R}$ is given as the restriction of a $\tilde{g} : \mathbb{R}^3 \mapsto \mathbb{R}$ onto $\mathcal{S}$, then the following relation holds for the conventional gradient $\nabla \tilde{g}$ and $\nabla^0 g$ in a point $p \in \mathcal{S}$:

$$\nabla^0_p g = \pi_p(\nabla_p \tilde{g}). \tag{2.3}$$

Here π_p denotes the orthogonal projection onto $T_p(\mathcal{S})$.

2.1.4 Normal Vector and Orientation

Given a point $p \in \mathcal{S}$, the unit normal vector[3] at p is defined by

$$\boldsymbol{n}_p := \frac{\boldsymbol{x}_{p,1} \times \boldsymbol{x}_{p,2}}{||\boldsymbol{x}_{p,1} \times \boldsymbol{x}_{p,2}||} \tag{2.4}$$

Note that this definition depends on the parameterisation $\boldsymbol{x}$. Depending on the orientation of the associated basis $\boldsymbol{x}_{,1}, \boldsymbol{x}_{,2}$ the normal vector changes its sign. The definition

[3] In the following the phrase normal vector will always refer to unit normal vector.

(2.4) shows that the Gaussian map is locally differentiable (for a single parameterisation). If $\mathcal{S}$ can not be covered by a single parameterisation, it might be that the mapping $p \mapsto \boldsymbol{n}_p$ (also called the unit normal vector field) can not be extended differentiably to all of $\mathcal{S}$. An example for this case is a Möbius strip.

In contrast, a regular surface $\mathcal{S}$ is called orientable if it permits a globally differentiable unit normal vector field. An orientable surface with a chosen unit normal vector field is called *oriented*. This gives rise to the so-called *Gaussian* map:

Definition 2.1 (Gaussian Map)**.** The gaussian map $\boldsymbol{n}$ of an oriented regular surface $\mathcal{S}$ is defined by

$$\boldsymbol{n} : \mathcal{S} \to S^2 : p \mapsto \boldsymbol{n}_p, \tag{2.5}$$

where $S^2 \subset \mathbb{R}^3$ denotes the unit sphere.

It has to be kept in mind that this global definition still depends on the choice of an orientation. The orientability of $\mathcal{S}$ just guarantees that an orientation can be chosen, but there are still two possibilities to choose from.

2.1.5 The Second Fundamental Form and Normal Curvature

Further inspection of the differential of the Gaussian map will finally lead to the notion of curvature. The differential of the Gaussian map $d\boldsymbol{n}_p$ at $p \in \mathcal{S}$ maps from $T_p(\mathcal{S})$ to $T_{\boldsymbol{n}_p}(S^2)$. It is also referred to as *shape operator* or *Weingarten map*. As the unit normal vector to a $\boldsymbol{n} \in S^2$ equals $\boldsymbol{n}$, these two tangent spaces are parallel and can be identified. Thus $d\boldsymbol{n}_p$ can be considered as a mapping of $T_p\mathcal{S}$ onto itself:

$$d\boldsymbol{n}_p : T_p(\mathcal{S}) \to T_p(\mathcal{S}) \tag{2.6}$$

The most important property of $d\boldsymbol{n}_p$ is:

Proposition 2.2. *The differential $d\boldsymbol{n}_p : T_p(\mathcal{S}) \to T_p(\mathcal{S})$ of the Gaussian map is a self adjoint linear mapping.*

Proof. We have to show that $\forall v, w \in T_p(\mathcal{S}) : \langle d\boldsymbol{n}_p(v), w\rangle = \langle v, d\boldsymbol{n}_p(w)\rangle$. As $d\boldsymbol{n}_p$ is linear, it suffices to show that for a basis of $T_p(\mathcal{S})$, e.g. for $\boldsymbol{x}_{p,1}$ and $\boldsymbol{x}_{p,2}$.

We write $\boldsymbol{n}_{p,1}$ and $\boldsymbol{n}_{p,2}$ for the partial derivatives of $\boldsymbol{n}_p$ analogously to $\boldsymbol{x}_{p,1}$ and $\boldsymbol{x}_{p,2}$. Then it is easy to see that $d\boldsymbol{n}_p(\boldsymbol{x}_{p,1}) = \boldsymbol{n}_{p,1}$ and $d\boldsymbol{n}_p(\boldsymbol{x}_{p,2}) = \boldsymbol{n}_{p,2}$. Thus it suffices to show that $\langle \boldsymbol{n}_{p,1}, \boldsymbol{x}_{p,2}\rangle = \langle \boldsymbol{n}_{p,2}, \boldsymbol{x}_{p,1}\rangle$ holds.

This can be accomplished by differentiating the identities $\langle \boldsymbol{n}_p, \boldsymbol{x}_{p,1}\rangle$ and $\langle \boldsymbol{n}_p, \boldsymbol{x}_{p,2}\rangle$ in the second and first components respectively:

$$\frac{\partial}{\partial_2}\langle \boldsymbol{n}_p, \boldsymbol{x}_{p,1}\rangle = \langle \boldsymbol{n}_{p,2}, \boldsymbol{x}_{p,1}\rangle + \langle \boldsymbol{n}_p, \boldsymbol{x}_{p,12}\rangle = 0$$
$$\frac{\partial}{\partial_1}\langle \boldsymbol{n}_p, \boldsymbol{x}_{p,2}\rangle = \langle \boldsymbol{n}_{p,1}, \boldsymbol{x}_{p,2}\rangle + \langle \boldsymbol{n}_p, \boldsymbol{x}_{p,21}\rangle = 0$$

As $\boldsymbol{x}_{p,12} = \boldsymbol{x}_{p,21}$ one gets

$$\langle \boldsymbol{n}_{p,1}, \boldsymbol{x}_{p,2}\rangle = -\langle \boldsymbol{n}_p, \boldsymbol{x}_{p,12}\rangle = \langle \boldsymbol{n}_{p,2}, \boldsymbol{x}_{p,1}\rangle$$ □

From the linear mapping $d\boldsymbol{n}_p : T_p(\mathcal{S}) \to T_p(\mathcal{S})$ being self adjoint, one can directly deduce that $d\boldsymbol{n}_p$ has an orthonormal basis of two eigenvectors $\boldsymbol{e}_1, \boldsymbol{e}_2 \in T_p(\mathcal{S})$, corresponding to real eigenvalues $-k_1, -k_2$. We will now give a geometric meaning to these.

Remark. Without loss of generality we assume $k_1 \geq k_2$. The sense of reversing the sign of the eigenvalues will become clear in proposition 2.5.

Definition 2.3 (Second Fundamental Form)**.** The bilinear form

$$\Pi_p : T_p(\mathcal{S}) \times T_p(\mathcal{S}) \to \mathbb{R} : (v, w) \mapsto -\langle d\boldsymbol{n}_p(v), w\rangle \tag{2.7}$$

is called second fundamental form of $\mathcal{S}$ at p.

Consider the parameterisation $\boldsymbol{x}$. The functions $L := \Pi_p(\boldsymbol{x}_{,1}, \boldsymbol{x}_{,1})$, $M := \Pi_p(\boldsymbol{x}_{,1}, \boldsymbol{x}_{,2})$, $N := \Pi_p(\boldsymbol{x}_{,2}, \boldsymbol{x}_{,2})$ are called the associated coefficients of the second fundamental form. If vectors $v, w \in T_p(\mathcal{S})$ are given by their coordinate vectors $(v_1, v_2), (w_1, w_2)$ with respect to the associated basis, then

$$\Pi_p(v, w) = v_1 w_1 L + (v_1 w_2 + v_2 w_1) M + v_2 w_2 N. \tag{2.8}$$

Differentiating the identities $\langle \boldsymbol{n}_p, \boldsymbol{x}_{p,i}\rangle$ as in the proof of proposition 2.2, one can derive the following formula for the coefficients:

$$\begin{aligned}\Pi_p(\boldsymbol{x}_{p,i}, \boldsymbol{x}_{p,j}) &= -\langle d\boldsymbol{n}_p(\boldsymbol{x}_{p,i}), \boldsymbol{x}_{p,j}\rangle = -\langle \boldsymbol{n}_{p,i}, \boldsymbol{x}_{p,j}\rangle = \langle \boldsymbol{n}_p, \boldsymbol{x}_{p,ji}\rangle \\ &= \langle \frac{\boldsymbol{x}_{p,1} \times \boldsymbol{x}_{p,2}}{||\boldsymbol{x}_{p,1} \times \boldsymbol{x}_{p,2}||}, \boldsymbol{x}_{p,ji}\rangle = \frac{1}{||\boldsymbol{x}_{p,1} \times \boldsymbol{x}_{p,2}||} \det(\boldsymbol{x}_{p,1}, \boldsymbol{x}_{p,2}, \boldsymbol{x}_{p,ji}).\end{aligned} \tag{2.9}$$

Definition 2.4 (Normal Curvature)**.** Let c be a regular curve on $\mathcal{S}$ parameterized by its arc length and passing through $p \in \mathcal{S}$. The curvature of c in p shall be denoted by k, its tangent and normal vectors by $\boldsymbol{t}$ and $\boldsymbol{n}^c$ respectively[4]. It holds that $\boldsymbol{t}' = k\boldsymbol{n}^c$. The normal curvature of the curve c at p is defined by $k_n = k\langle \boldsymbol{n}^c_p, \boldsymbol{n}_p\rangle$. It can be interpreted as the curvature of the curve obtained by projecting c onto the normal plane to $\mathcal{S}$ along $\boldsymbol{t}_p$, which is spanned by $\boldsymbol{n}_p$ and $\boldsymbol{t}_p$.

Remark. The sign of k_n depends on the choice of orientation of $\mathcal{S}$.

Proposition 2.5. *Let c be a regular curve as above. For the normal curvature of c in p it holds that $k_n = \Pi_p(\boldsymbol{t}_p, \boldsymbol{t}_p)$.*

Proof. We use the identity $\langle \boldsymbol{n}_p, \boldsymbol{t}_p\rangle = 0$ to get

$$\langle \boldsymbol{n}'_p, \boldsymbol{t}_p\rangle + \langle \boldsymbol{n}_p, \boldsymbol{t}'_p\rangle = 0$$

Now a straightforward calculation shows

$$\begin{aligned}\Pi_p(\boldsymbol{t}_p, \boldsymbol{t}_p) &= -\langle d\boldsymbol{n}_p(\boldsymbol{t}_p), \boldsymbol{t}_p\rangle \\ &= -\langle \boldsymbol{n}'_p, \boldsymbol{t}_p\rangle \\ &= \langle \boldsymbol{n}_p, \boldsymbol{t}'_p\rangle \\ &= \langle \boldsymbol{n}_p, k\boldsymbol{n}^c_p\rangle = k_n \quad \square\end{aligned}$$

[4]Without loss of generality c shall be parameterized by its arc length, which implies $||\boldsymbol{t}|| = 1$.

This proposition implies that the normal curvature only depends on the tangent vector of a curve.

Corollary 2.6 (Meusnier)**.** *All curves on $\mathcal{S}$ passing through $p \in \mathcal{S}$ with the same tangent vector $\boldsymbol{t}_p \in T_p(\mathcal{S})$ share the same normal curvature.*

Given a $\boldsymbol{v} \in T_p(\mathcal{S})$ with $||\boldsymbol{v}|| = 1$ it can be uniquely be expressed as

$$\boldsymbol{v} = \langle \boldsymbol{v}, \boldsymbol{e}_1 \rangle \boldsymbol{e}_1 + \langle \boldsymbol{v}, \boldsymbol{e}_2 \rangle \boldsymbol{e}_2 = \cos\phi \boldsymbol{e}_1 + \sin\phi \boldsymbol{e}_2$$

for some $\phi \in [0, 2\pi)$. The normal curvature of curves on $\mathcal{S}$ passing through p with tangent vector $\boldsymbol{v}$ is

$$\begin{aligned} k_n &= \Pi_p(\boldsymbol{v}, \boldsymbol{v}) = -\langle d\boldsymbol{n}_p(\boldsymbol{v}), \boldsymbol{v} \rangle \\ &= \langle k_1 \cos\phi \boldsymbol{e}_1 + k_2 \sin\phi \boldsymbol{e}_2, \cos\phi \boldsymbol{e}_1 + \sin\phi \boldsymbol{e}_2 \rangle \\ &= k_1 \cos^2\phi + k_2 \sin^2\phi \end{aligned}$$

This shows that k_1 and k_2 are the extremal values of the normal curvature at $p \in \mathcal{S}$. These extremal values of normal curvature are attained by curves passing through p with tangent vectors $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$ respectively.

Definition 2.7 (Principal Curvatures)**.** k_1 and the k_2 are called principal curvatures, the corresponding directions given by $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$ are called principal directions.

The product of the principal curvatures is called Gaussian curvature $K := k_1 k_2$. It is obvious that it is equal to the determinant of the Weingarten map: $K = det(d\boldsymbol{n}_p)$. Points $p \in \mathcal{S}$ are classified regarding the sign of Gaussian curvature. For $K > 0, K = 0, K < 0$ they are called *elliptic, parabolic, hyperbolic* respectively. A point is called *flat* if $k_1 = k_2 = 0$.

Definition 2.8 (Curvature Tensor Field)**.** The mapping that assigns to each $p \in \mathcal{S}$ the linear mapping $-d\boldsymbol{n}_p$ on $T_p(\mathcal{S})$ is called *curvature tensor field.*

The curvature tensor field is a second order symmetric tensor field[5]. It determines two orthogonal eigenvector fields $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$ and two corresponding scalar fields k_1 and k_2. Without loss of generality we will consider the eigenvectors as normed. Notice that the eigenvectors are still only defined up to their sign, consequently the eigenvector fields do not possess a specific direction, as opposed to vector fields.

The degeneracies that can occur in the curvature tensor field, which are of great interest, will be defined in the following.

2.1.6 Umbilic Points

Consider a point on a plane or on a sphere. In these cases $k_1 = k_2$ and the differential of the Gaussian Map is just a multiple of the identity I: $d\boldsymbol{n}_p = k_1 I$. This means that the principal directions are not uniquely defined. These points will be of special interest later on. This gives rise to the following definition:

[5]For an introduction to tensor fields see [19].

Definition 2.9 (Umbilic point)**.** A point $p \in \mathcal{S}$ with $k_1 = k_2$ is called umbilic point.

A connected subset of $\mathcal{S}$ that contains umbilic points only, is called *umbilic area*. Examples for this are (connected subsets of) spheres or planes. For a further classification of umbilic points see section 4.2.1.

Remark 2.10. In points that are not umbilic, the different eigenvalues can be used to distinguish the two orthogonal eigenvector fields defined by the curvature tensor field. If $k_1 > k_2$, the eigenvector fields $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$ are called *major* and *minor eigenvector field* respectively.

2.1.7 Principal Curvature Lines

The principal directions on a surface can be used to trace networks of curves.

Definition 2.11 (Principal Curvature Line)**.** A regular connected curve c on $\mathcal{S}$ that is tangent to a principal direction of $\mathcal{S}$ in every $p \in c$ is called principal curvature line.

A necessary and sufficient condition for c to be a principal curvature line is:

Proposition 2.12. *Let $c(t), t \in (a,b) \subset \mathbb{R}$ be a regular curve. Then*

$$\forall t \in (a,b) : \boldsymbol{n}'_{c(t)} = \lambda(t) c'(t) \tag{2.10}$$

if and only if c is a principal curvature line. In this case $-\lambda(t)$ is the principal curvature in direction of $c'(t)$.

Proof. Let c be a principal curvature line, which means that the tangent vector $c'(t)$ is an eigenvector of $d\boldsymbol{n}_{c(t)}$. Thus $\boldsymbol{n}'_{c(t)} = d\boldsymbol{n}_{c(t)}(c'(t)) = -k(t)c'(t)$ where $k(t)$ is a principal curvature and (2.10) holds with $\lambda(t) = -k(t)$.

On the contrary if (2.10) holds then $c'(t)$ is an eigenvector of $d\boldsymbol{n}_{c(t)}$ for all $t \in (a,b)$, which is the defining property of a principal curvature line. □

In order to proof the existence and uniqueness of principal curvature lines starting from a point $p \in \mathcal{S}$, the classical Picard-Lindelöf theorem can not be applied directly, because the eigenvector fields are only defined up to their sign[6]. As follows from definition 2.9, there can not be two unique principal curvature lines passing through an umbilic p.

In non-umbilic areas it is possible to distinguish between the major and minor eigenvector fields. Therefore one can define a continous vector field from one of the eigenvector fields by assigning a consistent orientation to the eigenvectors. The theorem of Picard-Lindelöf can now be applied to these continous vector fields. In section 4.2.1 we will see that a consistent orientation is in general not possible in the vicinity of umbilic points. In practice the orientation of the eigenvector fields can be fixed on the fly between integration steps of a principal curvature line, as long as the line does not get too close to umbilic points.

[6]Remember that we have required $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$ to be of unit length without loss of generality.

In points of umbilic areas of $\mathcal{S}$ all directions are principal directions. This means that any regular connected curve c in such an area is a principal curvature line.

If one considers the normals of $\mathcal{S}$ along a curve on $\mathcal{S}$, the lines along these normals generate a ruled surface. The principal curvature lines can be characterized by the special property that these normal ruled surfaces along them are torsal and thus developable[7], see [30] and [34]. If $c(t), t \in (a,b) \subset \mathbb{R}$ denotes a principal curvature line of $\mathcal{S}$, the ruled surface can be parametrized by

$$(s,t) \in (a,b) \times \mathbb{R} : y(s,t) := c(t) + s\boldsymbol{n}_{c(t)}.$$

It is well known that the developability of a ruled surface can be characterized by the surface normals being constant along the rulings, see [34]. The normals in this case are given up to a constant by $\boldsymbol{n}_{c(t)} \times \big(c'(t) + sd\boldsymbol{n}_{c(t)}(c'(t))\big)$. Since the tangent vector $c'(t)$ is an eigenvector of $d\boldsymbol{n}_{c(t)}$, it follows directly that the normals along a ruling are constant.

Remark. Consider a regular curve $c \subset \mathcal{S}$ given by $c(t) := \boldsymbol{x}(u(t), v(t)), t \in (a,b) \subset \mathbb{R}$, where $(u(t), v(t))$ is a curve in the parameter domain D. The differential equation (2.10) leads to a differential equation for $u(t), v(t)$, see [34, page 96]. However this differential equation is not useful for numerical integration of the principal curvature lines, another approach will be described in 4.2.3.

2.1.8 Conjugate Directions and Curve Networks

Definition 2.13. Consider a point $p \in \mathcal{S}$ with attached tangent space $T_p(\mathcal{S})$. Two vectors $v, w \in T_p(\mathcal{S}) \setminus \{0\}$ are called conjugate if

$$\Pi_p(v,w) = -\langle d\boldsymbol{n}_p(v), w\rangle = 0. \tag{2.11}$$

Correspondingly, the directions given by v, w are called conjugate.

An example for conjugate directions are the principal directions. These are given by the orthonormal basis of eigenvectors $\boldsymbol{e}_1, \boldsymbol{e}_2$ of $d\boldsymbol{n}_p$ and thus

$$\langle d\boldsymbol{n}_p(\boldsymbol{e}_1), \boldsymbol{e}_2\rangle = -k_1\langle \boldsymbol{e}_1, \boldsymbol{e}_2\rangle = 0.$$

Given a $p \in \mathcal{S}$ and a $v \in T_p(\mathcal{S})$, there is at least one $w \in T_p(\mathcal{S})$ conjugate to v. All conjugate directions can be computed by solving (2.11) for w, which is a linear equation system for fixed v. If $p \in \mathcal{S}$ is an umbilic point, each pair of orthogonal vectors $v, w \in T_p(\mathcal{S}) \setminus \{0\}$ is conjugate. Self conjugate directions are called *asymptotic*. In hyperbolic points there are always two distinct asymptotic directions, in parabolic points there exists at least one.

A *conjugate curve network* consists of two one-parameter families of curves C_1, C_2 which cover $\mathcal{S}$. For all $p \in \mathcal{S}$ there are unique curves $c_1 \in C_1$ and $c_2 \in C_2$ passing through p, such that the tangents of c_1 and c_2 in p are conjugate. An example are the principal curvature lines.

[7] Developable essentially means locally isometric to the plane.

Remark. In section 4.2.1 it will become clear that the curvature tensor field can exhibit degenerate points which violate the requirement that there are unique curves passing through them.

The curves on the surface generated by fixing one of the parameters of $\boldsymbol{x}$ are called *parameter lines*. The following proposition will be needed later on:

Proposition 2.14. *The parameter lines of a parameterisation $\boldsymbol{x}$ of $\mathcal{S}$ are conjugate if and only if for the coefficients of the second fundamental form holds $M = 0$.*

Proof. The coordinates of tangent vectors belonging to the parameter lines clearly fulfill $v_1 = w_2 = 1$ and $v_2 = w_1 = 0$ for all $p \in \mathcal{S}$. Thus $\Pi_p(v, w) = M$ (see 2.8), which means that the tangents to the parameter lines are conjugate if and only if $M = 0$. □

2.2 From Differential Geometry to Relative Differential Geometry

In relative differential geometry the Gaussian Map is generalized. In particular the unit sphere S^2 will be replaced by a more general object, which will be called *relativesphere* in the following.

A compact convex nonempty subset $K \subset \mathbb{R}^3$ is called *convex body*. We will consider convex bodies with nonempty interior only. A convex body is called *strictly convex* if the interior of every line segment contains no boundary points of K. The boundary of a convex body will be called *convex surface*. In the following a convex surface $\mathcal{R}$ will always be assumed regular, such that the methods from differential geometry can be applied. In particular this leads to the equivalence of strict convexity and positive Gaussian curvature.

Let $\mathcal{R}$ be a strictly convex surface. As an example consider an ellipsoid. The Gaussian map of $\mathcal{R}$ shall be denoted by $\boldsymbol{\nu}$. The strict convexity of $\mathcal{R}$ ensures that $\boldsymbol{\nu} : \mathcal{R} \to S^2$ is a bijective mapping. Thus it is possible to define a *relative Gaussian map* of $\mathcal{S}$ with respect to $\mathcal{R}$:

Definition 2.15 (Relative Gaussian Map)**.** Let $\mathcal{R}$ be a strictly convex surface. The relative Gaussian map of an oriented regular surface $\mathcal{S}$ with respect to $\mathcal{R}$ is defined by:

$$\boldsymbol{n}^{\mathcal{R}} : \mathcal{S} \to \mathcal{R} : p \mapsto \boldsymbol{\nu}^{-1}(\boldsymbol{n}_p) \tag{2.12}$$

The differential of the relative Gaussian map, also referred to as *relative Weingarten map*, can easily be computed from the differentials of the Gaussian maps of $\mathcal{S}$ and $\mathcal{R}$:

$$\begin{aligned} d\boldsymbol{n}_p^{\mathcal{R}} &: T_p(\mathcal{S}) \to T_{\nu^{-1}(\boldsymbol{n}_p)}(\mathcal{R}) \\ d\boldsymbol{n}_p^{\mathcal{R}} &= \left(d\boldsymbol{\nu}_{\nu^{-1}(\boldsymbol{n}_p)}\right)^{-1} \circ d\boldsymbol{n}_p \\ &= d\boldsymbol{\nu}_{\boldsymbol{n}_p}^{-1} \circ d\boldsymbol{n}_p \end{aligned} \tag{2.13}$$

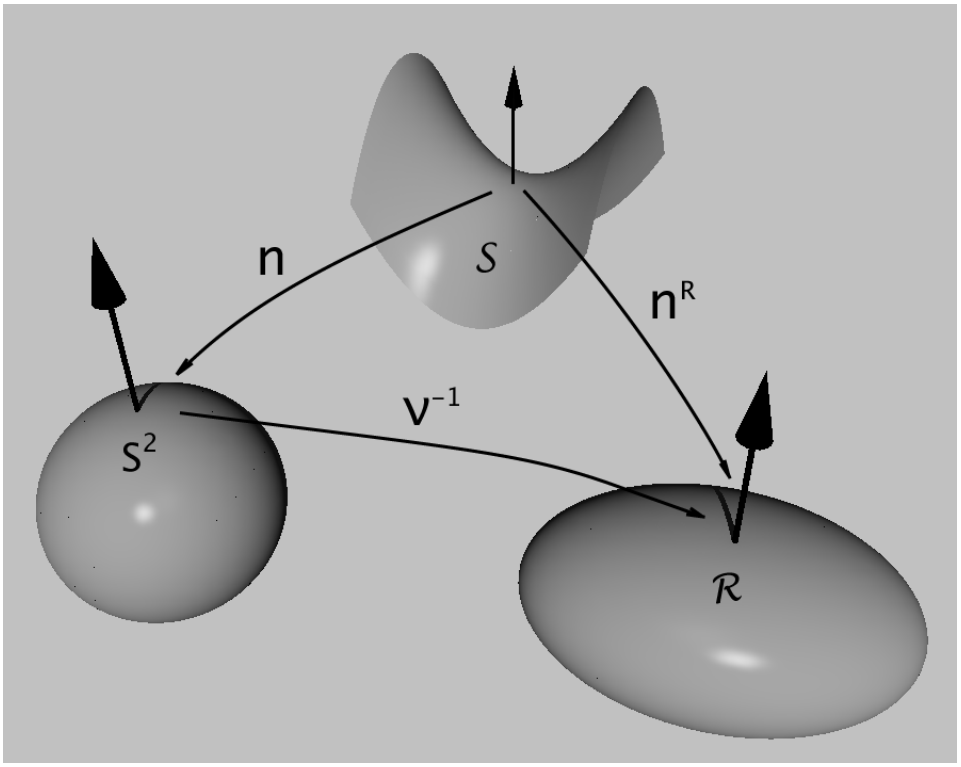


Figure 2.1: Composition of the relative Weingarten map

As the tangent spaces $T_p(\mathcal{S})$ and $T_{\nu^{-1}(\boldsymbol{n}_p)}(\mathcal{R})$ are parallel, one can identify them and view $d\boldsymbol{n}_p^{\mathcal{R}}$ as a mapping of $T_p(\mathcal{S})$ onto itself:

$$d\boldsymbol{n}_p^{\mathcal{R}} : T_p(\mathcal{S}) \to T_p(\mathcal{S})$$

In general this mapping will not be self adjoint any more. However, it still possesses a pair of real eigenvalues $-k_1^{\mathcal{R}}, -k_2^{\mathcal{R}} \in \mathbb{R}$ and corresponding real eigenvectors $\boldsymbol{e}_1^{\mathcal{R}}, \boldsymbol{e}_2^{\mathcal{R}} \in T_p(\mathcal{S})$ as will be shown in the following. For simplicity of notation we skip the superscript $^{\mathcal{R}}$ in $-k_i^{\mathcal{R}}, \boldsymbol{e}_i^{\mathcal{R}}$, if no confusion with the classical principal curvatures and principal directions is to be expected.

Theorem 2.16. *Let $-k_1, -k_2$ be the eigenvalues of $d\boldsymbol{n}_p^{\mathcal{R}}$ and $\boldsymbol{e}_1, \boldsymbol{e}_2 \in T_p(\mathcal{S})$ the corresponding eigenvectors. Using the strictly positive Gaussian curvature of $\mathcal{R}$ in $\nu^{-1}(\boldsymbol{n}_p)$, it follows that $k_1, k_2 \in \mathbb{R}$.*

Proof. First we show that $\boldsymbol{e}_1, \boldsymbol{e}_2$ are conjugate directions of $\mathcal{S}$ as well as $\mathcal{R}$. Using $k_i \boldsymbol{e}_i = d\boldsymbol{n}^{\mathcal{R}}(\boldsymbol{e}_i) = d\boldsymbol{\nu}^{-1} \circ d\boldsymbol{n}(\boldsymbol{e}_i), i \in \{1, 2\}$ and the self-adjointness of $d\boldsymbol{\nu}, d\boldsymbol{n}$ we get

$$\begin{aligned} \langle \boldsymbol{e}_2, d\boldsymbol{n}(\boldsymbol{e}_1) \rangle &= k_1 \langle \boldsymbol{e}_2, d\boldsymbol{\nu}(\boldsymbol{e}_1) \rangle = \\ \langle d\boldsymbol{n}(\boldsymbol{e}_2), \boldsymbol{e}_1 \rangle &= k_2 \langle d\boldsymbol{\nu}(\boldsymbol{e_2}), \boldsymbol{e}_1 \rangle = k_2 \langle \boldsymbol{e_2}, d\boldsymbol{\nu}(\boldsymbol{e}_1) \rangle \end{aligned}$$

and thus $(k_1 - k_2)\langle \boldsymbol{e}_2, d\boldsymbol{\nu}(\boldsymbol{e}_1) \rangle = 0$. If $k_1 \neq k_2$ it follows that

$$\langle \boldsymbol{e}_2, d\boldsymbol{\nu}(\boldsymbol{e}_1) \rangle = \langle \boldsymbol{e}_2, d\boldsymbol{n}(\boldsymbol{e}_1) \rangle = 0. \tag{2.14}$$

In case $k_1 = k_2$, $d\boldsymbol{n}_p^{\mathcal{R}}$ is a multiple of the identity and $\boldsymbol{e}_1, \boldsymbol{e}_2$ can be chosen as the eigenvectors of $d\boldsymbol{\nu}$, such that (2.14) holds too.
Now assume $k_1 \in \mathbb{C} \setminus \mathbb{R}$. As complex zeros of real polynomials appear in conjugated pairs it follows that $k_2 = \bar{k_1}$ and $\boldsymbol{e}_2 = \bar{\boldsymbol{e}_1}$. Let $c_1, c_2 \in \mathbb{C}$ denote the coefficients of $\boldsymbol{e}_1$ in the orthonormal basis of eigenvectors of $d\boldsymbol{\nu}$. The eigenvalues of $d\boldsymbol{\nu}$, which are both negative[8], shall be denoted by $l_1, l_2 \in \mathbb{R}$. Then

$$\langle \boldsymbol{e}_2, d\boldsymbol{\nu}(\boldsymbol{e}_1) \rangle = l_1 c_1 \bar{c_1} + l_2 c_2 \bar{c_2} = l_1 |c_1|^2 + l_2 |c_2|^2 < 0.$$

in contradiction to (2.14). □

The last proof shows why it is important to choose $\mathcal{R}$ as a convex surface with strictly positive Gaussian curvature. Consistent with definition 2.7 we now define:

Definition 2.17 (Relative Principal Curvatures)**.** $k_1^{\mathcal{R}}$ and $k_2^{\mathcal{R}}$ are called relative principal curvatures, the corresponding directions given by $\boldsymbol{e}_1^{\mathcal{R}}$ and $\boldsymbol{e}_2^{\mathcal{R}}$ are called relative principal directions.

Analogously the definitions of Gaussian curvature, umbilic points, curvature tensor field and principal curvature lines are carried over to relative differential geometry.

2.2.1 Relative umbilic points

Umbilic points play a significant role in the generation of networks of relative principal curvature lines.

A point $p \in \mathcal{S}$ is a relative umbilic point, if and only if $d\boldsymbol{n}_p^{\mathcal{R}}$ is a multiple of the identity. This happens if and only if the differential of the Gaussian map of $\mathcal{S}$ in p is a multiple of the Gaussian map of $\mathcal{R}$ in $\boldsymbol{\nu}^{-1}(\boldsymbol{n}_p)$, which means that the principal directions of $\mathcal{S}$ and $\mathcal{R}$ are the same and there is a constant $c \in \mathbb{R} \setminus \{0\}$ such that for the principal curvatures holds $ck_1 = \tilde{k_1}, ck_2 = \tilde{k_2}$, where $\tilde{k_1}, \tilde{k_2}$ denote the principal curvatures of $\mathcal{R}$. This means that $\mathcal{R}$ in $\boldsymbol{\nu}^{-1}(\boldsymbol{n}_p)$ multiplied by c approximates $\mathcal{S}$ in p up to second order[9]. The interesting topic of prescribing umbilic points by choosing an appropriate relativesphere will be addressed in section 5.3.

2.2.2 Relative principal curvature lines

Analogously to proposition 2.12, a necessary and sufficient condition for c to be a relative principal curvature line is:

Proposition 2.18. *Let $c(t), t \in (a, b) \subset \mathbb{R}$ be a regular curve. Then*

$$\forall t \in (a, b) : (\boldsymbol{n}_{c(t)}^{\mathcal{R}})' = \lambda(t) c'(t) \tag{2.15}$$

if and only if c is a relative principal curvature line. In this case $-\lambda(t)$ is the relative principal curvature in direction $c'(t)$.

[8] Assuming that the normal vector on $\mathcal{R}$ is directed outside.
[9] Neglecting a necessary translation.

Proof. See proof of proposition 2.12. □

The relative principal curvature lines possess a characteristic property similar to the one described in 2.1.7. In the case of relative principal curvature lines one considers the ruled surface generated by the relative normals given by the relative Gaussian map $\boldsymbol{n}^{\mathcal{R}}$. Let $c(t), t \in (a,b) \subset \mathbb{R}$ be a relative principal curvature line, then the ruled surface parametrized by

$$(s,t) \in (a,b) \times \mathbb{R} : y(s,t) := c(t) + s\boldsymbol{n}^{\mathcal{R}}_{c(t)}$$

is developable. As in section 2.1.7 it suffices to show that the normals of this surface are constant along the rulings. The normal in a point (s,t) of this surface is given by $\boldsymbol{n}^{\mathcal{R}}_{c(t)} \times \Big(c'(t) + sd\boldsymbol{n}^{\mathcal{R}}_{c(t)}(c'(t))\Big)$. As $c(t)$ is a relative principal curvature line, $c'(t)$ is an eigenvector of $d\boldsymbol{n}^{\mathcal{R}}_{c(t)}$ and thus the normal does not depend on s, which parameterizes the rulings.

Remark. If one considers a curve $(u(t), v(t))$ in D as parameterisation of a relative principal curvature line, a differential equation for this curve in D can be derived (cf. [29]). As this differential equation is not useful for numerical integration, another approach described in 4.2.3 will be used.

2.2.3 Influencing the Direction of Relative Principal Curvature Lines

In the proof of theorem 2.16 it has been shown that $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$ are conjugate directions of both $\mathcal{S}$ and $\mathcal{R}$. Consequently one can influence the direction of relative principal curvature lines by influencing the conjugate directions of $\mathcal{R}$. Without loss of generality consider a $q \in \mathcal{R}$ with $k_1/k_2 = a > 1$. The conjugate directions v, w in $T_q(\mathcal{R})$ can be parameterized by

$$\begin{aligned} v &= \cos\alpha\boldsymbol{e}_1 + \sqrt{a}\sin\alpha\boldsymbol{e}_2 \\ w &= -\sin\alpha\boldsymbol{e}_1 + \sqrt{a}\cos\alpha\boldsymbol{e}_2, \end{aligned}$$

which follows directly from the definition of conjugate directions in section 2.1.8. Clearly v and w represent points on the same ellipsoid in $T_q(\mathcal{R})$. Figure 2.2 shows how w can be geometrically constructed given v and vice versa. Apparently the distinguished direction for conjugate pairs is $\boldsymbol{e}_2$. This means that by choosing $\mathcal{R}$ such that $a \gg 1$ in large parts of $\mathcal{R}$, one can set a *preferred* direction for the relative principal curvature lines, see figure 7.6 for an example.

The above representation of conjugate directions can also be used to calculate the minimum angle that can occur between $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$. Denote the angle between between v and w by $\beta(\alpha)$. It holds that

$$\cos\beta(\alpha) = \frac{\langle v, w\rangle}{||v||||w||}.$$

As cos is a monotonous function on $[0,\pi]$, the extremal values of $\beta(\alpha)$ can be calculated using the extremal values of $\cos\beta(\alpha)$. Differentiating the right hand side in α shows that

these are attended for $\cos\alpha = \pm\sin\alpha = 1/\sqrt{2}$ and are equal to $\pm\frac{a-1}{a+1}$. Consequently, if $a \leq \tilde{a}$ for all of $\mathcal{R}$, the minimum angle $\tilde{\beta}$ that can occur between $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$ is given by

$$\tilde{\beta} = \cos^{-1}\left(\frac{\tilde{a}-1}{\tilde{a}+1}\right).$$

Remark 2.19. The above considerations show, that setting a preferred direction leads to a small minimum angle between conjugate directions. This is unwanted in many applications. Consequently a compromise has to be taken.

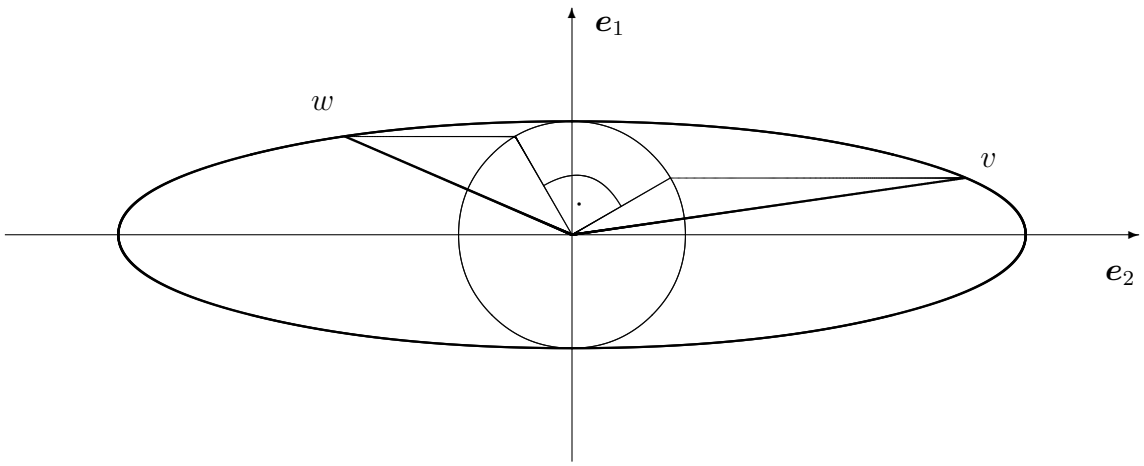


Figure 2.2: Geometrical construction of conjugate directions in the elliptic case

3 Surface Representation by Meshes

In differential geometry surfaces are given as differentiable manifolds. Usually the degree of differentiability of the manifolds is not explicitly mentioned. It is just assumed to be high enough to justify all the operations from differential calculus that need to be done for the definitions of geometrical objects.

In contrast to surfaces represented by differentiable manifolds are surfaces represented by meshes. Loosely speaking, these are given as the union of polyhedrons, consequently they are discrete by nature and in general not differentiable. The definitions of tangent space, normal vector, curvature, etc. can not be carried over directly. However, similar objects can be defined using the discrete surface representation. If a mesh stems from the discretisation of a differentiable manifold one can study the behavior of these objects in the limit when the mesh is refined. This field of mathematics is called discrete differential geometry, see [34] and [10].

Nowadays many surfaces that are considered in different applications are represented by meshes, since it is easy to construct meshes from other representations. Meshes can easily be exchanged between different kinds of software, they are convenient to work with and allow for a fast rendering using modern computer graphics libraries and hardware. Therefore, in practical applications the ability to export and import meshes is a must-have. In fact, the software which implements the methods described in this thesis implements what is called a remeshing process. A new coarse mesh is generated from a surface given by a mesh on a much finer level.

3.1 Overview of Meshes

A formal definition of a mesh can be given by a graph (V, E), $V := \{v_1, \ldots, v_n\}$ denoting the set of vertices and E the set of edges[1] (v_i, v_j). The vertex degree, also called *valence* in the context of meshes, is the number of edges incident to a vertex. A path between the vertices a and b is a sequence of vertices and edges $(v_1, e_1, v_2, e_2, \ldots, e_{n-1}, v_n)$ such that $v_1 = a, v_n = b$ and the vertices as well as the edges are pairwise different. If $a = b$ the path is called cycle. The graph (V, E) is called *connected* if for any two vertices there is a path connecting them. Faces are cycles which can not be shortened, the set of all faces shall be denoted by F. A graph is *embedded* in $\mathbb{R}^n$ if a $p \in \mathbb{R}^n$ is assigned to each vertex. A *planar graph* is a graph that can be embedded in $\mathbb{R}^2$ such that its edges do not intersect. Planar graphs can be drawn using straight lines only, called *straight*

[1] We do not consider graphs with edges connecting vertices to themselves, i.e. edges (v_i, v_i).

line planar graphs. A *triangulation* is a straight line planar graph all of whose faces are triangles. Finally a *mesh* is a straight line graph embedded in $\mathbb{R}^3$.

In the following we will not distinguish strictly between a vertex and its coordinate vector. The meaning of the symbol v should be clear from the context.
The edges of a mesh can be classified as follows:

Boundary edge adjacent to exactly one face

Regular edge adjacent to exactly two faces

Singular edge adjacent to more than two faces

Meshes without boundary/singular edges are called *closed*/*manifold* meshes respectively. In the following we will consider connected manifold meshes only. Meshes representing $\mathcal{S}$ shall always be triangular.

There are numerous ways to represent a mesh within data structures on a computer, see [5] for an overview. When choosing between these representations, one has to balance between performance and storage requirements. A possibility, usually used as an exchange format for meshes, is to store a table of vertices along with their coordinates and a table of faces pointing to the vertex table.

Vertex table			
Vertex	Coordinates		
v_1	x_1	y_1	z_1
v_2	x_2	y_2	z_2
v_3	x_3	y_3	z_3
v_4	x_4	y_4	z_4

Face table			
Face	Vertices		
f_1	v_1	v_2	v_3
f_2	v_1	v_4	v_2

Table 3.1: Vertex and Face Tables

This representation requires little memory, but it is not useful for quickly answering questions like

Question 1 Which faces are adjacent to vertex a?

Question 2 What are the vertices directly connected[2] to vertex a?

Question 3 Which faces are adjacent to the edge (a, b)?

In order to answer these questions it is necessary to do full table scans of the face table.

A commonly used structure that resolves these problems is the *doubly-connected edge list*. It is also called *half-edge structure*, because each regular edge is considered consisting of two half-edges, individually assigned to the adjacent faces. The data structure consists of vertex, face and half-edge records.

[2]The set of these vertices is called the *1-ring* of a.

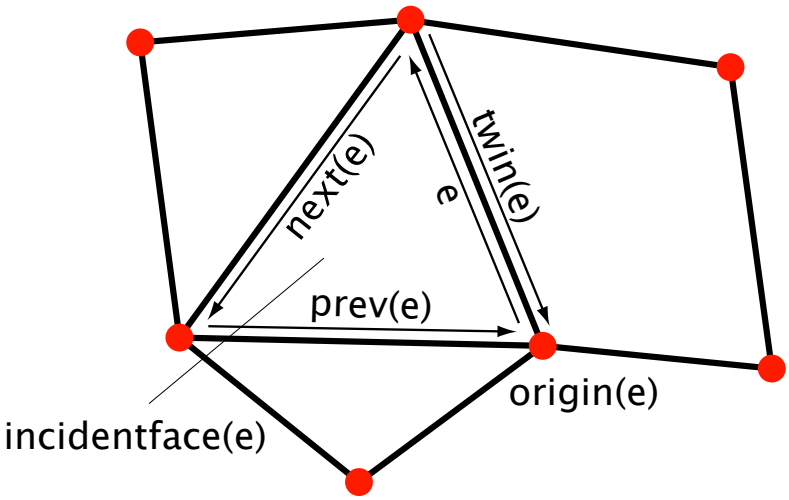


Figure 3.1: Illustration of a half-edge record

Vertex v

- Coordinates
- Pointer to one half-edge originating from the vertex (halfedge(v))

Face f

- Pointer to one half-edge on its boundary (halfedge(f))

Half-edge e

- Pointer to the vertex it originates from (origin(e))
- Pointer to the face it bounds (incidentface(e))
- Pointers to next/previous edge on boundary of incidentface (next(e)/prev(e))
- Pointer to its twin half-edge (twin(e))

Figure 3.1 gives an example for a half-edge record. At the expense of a higher memory requirement, the above questions can now be answered much easier:

Answer 1 incidentface(halfedge(a)), incidentface(next(twin(halfedge(a)))), incidentface(next(twin(next(twin(halfedge(a)))))),...

Answer 2 origin(next(halfedge(a))), origin(next(next(twin(halfedge(a))))), origin(next(next(twin(next(twin(halfedge(a))))))), ...

Answer 3 incidentface(halfedge(a)), incidentface(halfedge(b))

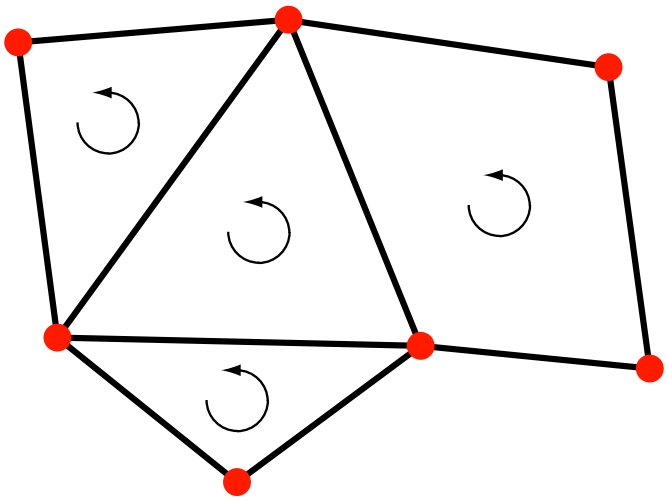


Figure 3.2: An orientable mesh

Remark. A half-edge structure can represent manifold meshes only. The examples above show meshes consisting of triangular faces only, nevertheless the representations presented are not limited to this kind of meshes. In the first representation the face table just needs to be enhanced to the maximum amount of vertices adjacent to each face. The half-edge representation the data structure directly supports the representation of polyhedral faces.

The orientation of faces in the face table / half-edge representations is given by the order of vertices / half-edges respectively. A mesh is called *orientable* if the faces can be oriented consistently, such that each regular edge is oriented in both directions, see figure 3.2.

Consider a manifold orientable triangular mesh. For a face f with vertex coordinates (v_1, v_2, v_3) the *face normal* n_f is defined by

$$\tilde{n}_f := (v_2 - v_1) \times (v_3 - v_1) \tag{3.1}$$

$$n_f := \frac{\tilde{n}_f}{||\tilde{n}_f||} \tag{3.2}$$

Let $N^e(v)$ denote the set of edges adjacent to vertex v. The vertex normal can then be defined by

$$n_v := \frac{\sum_{f \in N^e(v)} n_f}{||\sum_{f \in N^e(v)} n_f||}. \tag{3.3}$$

If the vertices of a given face are not collinear (which shall be assumed for all faces of the meshes considered), they span a plane which is the tangent plane in all points inside the face. The definition of vertex normal leads to a natural definition of the tangent space in a vertex v. It is defined as the plane orthogonal to n_v through v.

3.2 Defining and Computing the Curvature Tensor Field

The classical concepts from differential geometry are based on differentiation, as the name suggests. Consequently differentiability of the objects studied is a prerequisite. The stringent requirement that surfaces $\mathcal{S}$ be regular can be relaxed, e.g. for the definition of the Weingarten map it is sufficient that $\mathcal{S}$ is C^2. Nevertheless it is not possible to directly carry over these concepts to polyhedral surfaces like triangle meshes, which are not smooth.

In recent years numerous ways of overcoming this have been investigated. One way is to generate a smooth representation from a mesh. For example this can be done by subdivision and smoothing of the mesh, or using local approximations of the mesh by smooth surfaces. Other ways of generalizing the curvature tensor to triangle meshes stem from discrete differential geometry, for an example see [6], which has been used in [2] for tracing principal curvature lines.

For our purposes *integral invariants* obtained by integration over local neighbourhoods were used, an approach that has been proposed in [31], [38]. These neighbourhoods are defined by the intersection of balls or spheres of radius r with the surface $\mathcal{S}$ or the domain bounded by it. Principal component analysis of these point sets leads to a definition of principal curvatures at the scale r. In [31] asymptotic analysis for $r \to 0$ is used to show that this definition is consistent with the classical curvature theory.

In practice this technique is used to compute the curvature tensor in the vertices of the mesh. A continous curvature tensor field can then be defined by linearly interpolating between the curvature tensors in the vertices adjacent to a face.

Remark 3.1. Consider a face f of $\mathcal{S}$, remember that we require f to be a triangle. If one wants to introduce local coordinates on f, it is convenient to work with *barycentric coordinates*. We denote the coordinate vectors of the vertices adjacent to f by v_1, v_2 and v_3. Any point p on the face can be written as an affine combination $p = \lambda_1 v_1 + \lambda_2 v_2 + \lambda_3 v_3$ subject to the constraints $\lambda_1 + \lambda_2 + \lambda_3 = 1$ and $0 \leq \lambda_i \leq 1, 1 \leq i \leq 3$. $(\lambda_1, \lambda_2, \lambda_3)$ are called the barycentric or area coordinates. These coordinates are extremely useful for linear interpolation of data a_1, a_2, a_3 defined in the vertices of f. This interpolation evaluated at a point p with barycentric coordinates $(\lambda_1, \lambda_2, \lambda_3)$ is given by $\lambda_1 a_1 + \lambda_2 a_2 + \lambda_3 a_3$. Note that by adding a face-index to the barycentric coordinates $(f, \lambda_1, \lambda_2, \lambda_3)$, a global coordinate system for the mesh can be defined.

4 Planar Quad Meshes

First of all a few words on the existence of PQ meshes. Consider two polylines given by vertices $v_i, i \in \{1, \dots, m\}$ and $v_j, j \in \{1, \dots, n\}$. Translate one of the polylines along the other, which produces a mesh with vertices $v_{i,j} = v_i + v_j$. Let the edges between neighbouring vertices $v_{i,j}$ and $v_{i',j'}$ be denoted by $v_{i,j}v_{i',j'}$. Clearly the translation of the polyline ensures that opposing edges on the faces are parallel, which means the faces are planar. An example can be seen in figure 4.1 left.

An infinite number of PQ meshes can be constructed by using two polylines for prescribing the boundary vertices $v_{i,1}, i \in \{1, \dots, m\}$ and $v_{1,j}, j \in \{2, \dots, n\}$. PQ meshes can be constructed by choosing vertices in a sequence ordered by the sum of the indices $k = i+j, k > 2$ such that $v_{i,j}$ lies in the plane spanned by the vertices $v_{i-1,j-1}, v_{i-1,j}, v_{i,j-1}$. Figure 4.1 right gives an example where the faces are numbered in order of creation. Clearly this construction is a generalization of the first one.

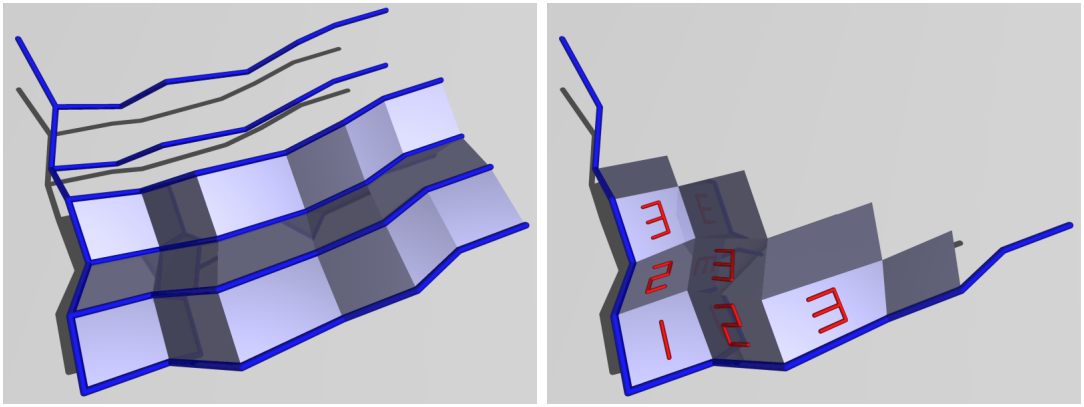


Figure 4.1: Examples for the construction of PQ meshes

In the following we are interested in the construction of a PQ mesh approximating a given surface.

4.1 Analogy to Conjugate Curve Networks

In 2.1.8 the notion of conjugate curve networks was defined. In the following it should become clear, that PQ meshes can be understood as a discrete analogon to such curve networks.

Consider a curve $c(t), t \in (a, b)$ on $\mathcal{S}$. The family of tangent planes along this curve

can be parameterized by

$$\left(T_{c(t)}(\mathcal{S})\right)_{t\in(a,b)} = \left(\{v \in \mathbb{R}^3 | \langle \boldsymbol{n}_{c(t)}, v\rangle = \langle \boldsymbol{n}_{c(t)}, c(t)\rangle\right)_{t\in(a,b)}.$$

The right hand side $\langle \boldsymbol{n}_{c(t)}, c(t)\rangle$ is the distance of the tangent plane to the origin. These tangent planes envelop a surface $\mathcal{T}$. Surfaces enveloped by a family of planes are developable ruled surfaces also called *torsal*, see [34], [30]. $\mathcal{T}$ is determined by the equations[1]

$$\begin{aligned} \langle \boldsymbol{n}_{c(t)}, p\rangle &= \langle \boldsymbol{n}_{c(t)}, c(t)\rangle \\ \langle d\boldsymbol{n}(c'(t)), p\rangle &= \langle \boldsymbol{n}_{c(t)}, c(t)\rangle' \end{aligned} \tag{4.1}$$

Consider a solution $p(t)$ to these equations for some fixed parameter t. The vector $r(t) := \boldsymbol{n}_{c(t)} \times d\boldsymbol{n}(c'(t))$ solves the homogenous equations and thus all points of the line $p(t)+sr(t), s \in \mathbb{R}$ are solutions to 4.1. Consequently these lines are the rulings of $\mathcal{T}$. The direction of the rulings can also be calculated using the shape operator. Let $w \in T_{c(t)}(\mathcal{S})$ be the conjugate direction to the tangent vector $c'(t)$, which means $\langle d\boldsymbol{n}(c'(t)), w\rangle = 0$ by definition. Therefore w is a solution of the homogenous equations too, and thus a multiple of r. In other words:

A ruling of the tangent surface generated by a curve c on $\mathcal{S}$ and passing through a $p \in c$, is conjugate to the tangent vector of c in p. The two families C_1 and C_2 of conjugate curves are thus dual to each other in the sense of tangent surfaces and their rulings. The rulings of tangent surfaces generated by curves from C_1 are tangent to curves from C_2 and vice versa.

Now we will focus on PQ meshes. First we study a single row of planar quadrilateral faces, a so-called PQ strip. The rows of vertices shall be denoted by $v_{1,j}, v_{2,j}, j \in \{1,\dots,n\}$, a line through vertices v_1 and v_2 by v_1v_2, see figure 4.2 left. A PQ strip is a discrete model for a developable surface, it can be mapped isometrically to the plane by unfolding it along the lines $v_{1,j}v_{2,j}$. These lines may be parallel, in which case the PQ strip is cylindrical. They can meet in a single point a, thus defining a conical surface with apex a. In the general case two consecutive lines $v_{1,j}v_{2,j}, v_{1,j+1}v_{2,j+1}$ cross each other in a point r_j. These points define a polyline $r_1,\dots,r_{n-1}$. The PQ strip is a patch on the tangent surface of this polyline, which is a discrete counterpart of the fact that a developable surface is in general part of the tangent surface of a space curve (cf. [30] and [34]). The rulings of the discrete tangent surface are represented by $v_{1,j}v_{2,j}$.

These considerations lead to a discrete version of the duality of conjugate curves for PQ meshes. The column of faces between two consecutive polylines along $v_{i,j}, j \in \{1,\dots,n\}$ and $v_{i+1,j}, j \in \{1,\dots,n\}$ create a PQ strip, see figure 4.2 right. As we have seen above, this PQ strip represents a patch of a discrete developable surface,

[1] More generally consider a family of planes given by the equation $F(t,p) = 0$. This equation defines a surface $\mathcal{Q}$ in $\mathbb{R}^4$. The envelope can be defined as the boundary of this surface projected to $\mathbb{R}^3$, also called the *contour*. The normal vector in a point of $\mathcal{Q}$, whose projection belongs to the contour, must be orthogonal to the t-axis. These are precisely the points with $\partial F/\partial t = 0$. This is represented by the second equation in 4.1.

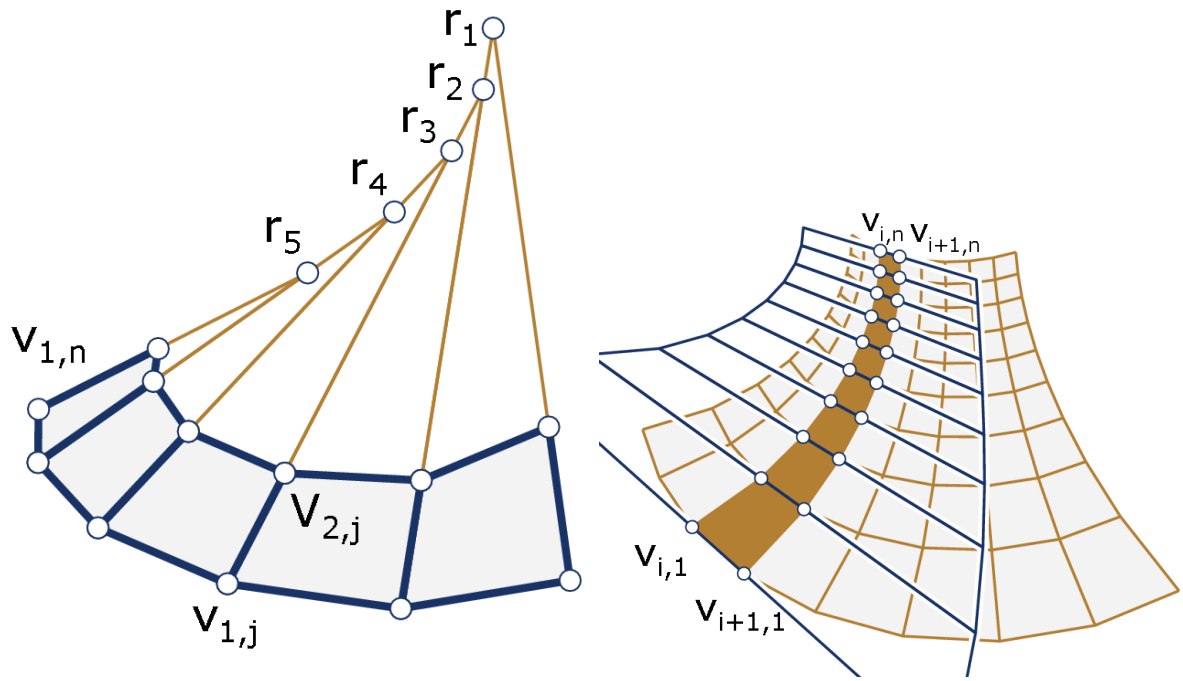


Figure 4.2: PQ strip on tangent surface and PQ strip on PQ mesh

which is tangent to the PQ mesh between the two polylines. The edges along rows $v_{i,j}v_{i+1,j}, j \in \{1,\dots,n\}$ are rulings of this developable surface. The same property holds if one exchanges rows with columns, thus considering PQ strips along polylines $v_{i,j}, i \in \{1,\dots,m\}$ and $v_{i,j+1}, i \in \{1,\dots,m\}$. The rulings of these developable surfaces are represented by $v_{i,j}v_{i,j+1}, i \in \{1,\dots,m\}$. Thus the rows and columns of polylines are a discrete conjugate network of polylines.

The authors of [23] point out the following conclusion: If a PQ mesh gets refined by a subdivision process, which preserves the PQ property and produces a curve network in the limit, then this curve network must be a conjugate network on a surface.

On the contrary, if one starts from a conjugate curve network on a surface, then the faces of a quad mesh produced from a discrete choice of conjugate curves will be close to planarity. This has been shown already in [34], and will be outlined in the following.

Consider a quad face on $\mathcal{S}$ defined by the points $\boldsymbol{x}(u_1,u_2), \boldsymbol{x}(u_1+\epsilon,u_2), \boldsymbol{x}(u_1,u_2+\delta), \boldsymbol{x}(u_1+\epsilon,u_2+\delta)$ on parameter lines of $\boldsymbol{x}$, as sketched in figure 4.3. The sides $a_\epsilon, b_\delta, d_{\epsilon,\delta}$ span a tetrahedron, whose volume V is a measure for the planarity of the quad. V can be calculated using the well-known formula $V = Ah/3$, where A denotes the base area and h the height. The base area of the triangle spanned by a_ϵ, b_ϵ is of order[2] $\epsilon\delta$, as well as the height of $\boldsymbol{x}(u_1+\epsilon,u_2+\delta)$ above the plane spanned by a_ϵ, b_ϵ. Consequently one

[2] $\mathcal{O}((\epsilon,\delta)^k)$ will be used as short notation for $\mathcal{O}(\sum_{i=0}^{k}\epsilon^i\delta^{k-i})$. For a definition of the Landau symbol $\mathcal{O}$ see a book on numerics, e.g. [27].

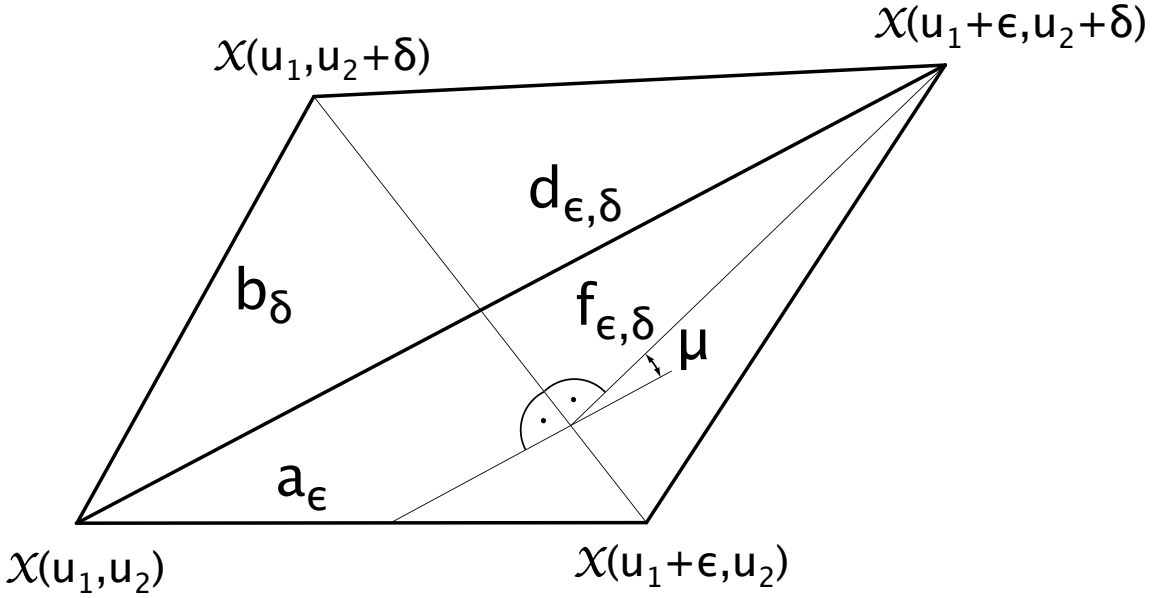


Figure 4.3: Quad face on $\mathcal{S}$ defined by points on parameter lines

can expect the volume of the tetraeder to be of order $\epsilon^2\delta^2$.

Another way of measuring the planarity is the angle μ, see figure 4.3. The height h can be calculated by means of $h = \sin\mu||f_{\epsilon,\delta}|| \approx \mu||f_{\epsilon,\delta}||$. As $||f_{\epsilon,\delta}|| = \mathcal{O}((\epsilon,\delta))$, which means that it depends linearly on δ and ϵ, it follows that $\mu = \mathcal{O}((\epsilon,\delta))$ too.
We will use the following second order Taylor expansions for estimation of V:

$$\begin{aligned} a_\epsilon :=& \boldsymbol{x}(u_1+\epsilon, u_2) - \boldsymbol{x}(u_1,u_2) = \epsilon\boldsymbol{x}_{,1} + \epsilon^2\boldsymbol{x}_{,11} + \mathcal{O}(\epsilon^3) \\ b_\delta :=& \boldsymbol{x}(u_1, u_2+\delta) - \boldsymbol{x}(u_1,u_2) = \delta\boldsymbol{x}_{,2} + \delta^2\boldsymbol{x}_{,22} + \mathcal{O}(\delta^3) \\ d_{\epsilon,\delta} :=& \boldsymbol{x}(u_1+\epsilon, u_2+\delta) - \boldsymbol{x}(u_1,u_2) \\ =& \epsilon\boldsymbol{x}_{,1} + \delta\boldsymbol{x}_{,2} + \epsilon^2\boldsymbol{x}_{,11} + 2\epsilon\delta\boldsymbol{x}_{,12} + \delta^2\boldsymbol{x}_{,22} + \mathcal{O}((\epsilon,\delta)^3) \end{aligned} \tag{4.2}$$

Calculating $\det(a_\epsilon, b_\delta, d_{\epsilon,\delta}) = \langle a_\epsilon \times b_\delta, d_{\epsilon,\delta}\rangle = 2Ah = 6V$ we get

$$a_\epsilon \times b_\delta = \epsilon\delta(\boldsymbol{x}_{,1}\times\boldsymbol{x}_{,2}) + \epsilon^2\delta(\boldsymbol{x}_{,11}\times\boldsymbol{x}_{,2}) + \epsilon\delta^2(\boldsymbol{x}_{,1}\times\boldsymbol{x}_{,22}) + \mathcal{O}((\epsilon,\delta)^4)$$

$$\begin{aligned} 6V &= \langle a_\epsilon \times b_\delta, d_{\epsilon,\delta}\rangle \\ &= \epsilon^3\delta\det(\boldsymbol{x}_{,1},\boldsymbol{x}_{,2},\boldsymbol{x}_{,11}) + 2\epsilon^2\delta^2\det(\boldsymbol{x}_{,1},\boldsymbol{x}_{,2},\boldsymbol{x}_{,12}) + \epsilon\delta^3\det(\boldsymbol{x}_{,1},\boldsymbol{x}_{,2},\boldsymbol{x}_{,22}) \\ &\quad + \epsilon^3\delta\det(\boldsymbol{x}_{,11},\boldsymbol{x}_{,2},\boldsymbol{x}_{,1}) + \epsilon\delta^3\det(\boldsymbol{x}_{,1},\boldsymbol{x}_{,22},\boldsymbol{x}_{,2}) + \mathcal{O}((\epsilon,\delta)^5) \\ &= 2\epsilon^2\delta^2\det(\boldsymbol{x}_{,1},\boldsymbol{x}_{,2},\boldsymbol{x}_{,12}) + \mathcal{O}((\epsilon,\delta)^5) \\ &= 2\epsilon^2\delta^2||\boldsymbol{x}_{,1}\times\boldsymbol{x}_{,2}||M + \mathcal{O}((\epsilon,\delta)^5) \end{aligned} \tag{4.3}$$

where we have used 2.9. As a result the volume V is of order $\mathcal{O}((\epsilon, \delta)^5)$ if and only if the parameter lines are conjugate, see proposition 2.14. In this case the angle μ is of order $\mathcal{O}((\epsilon, \delta)^2)$.

4.2 Quad Meshes from Relative Principal Curvature Lines

We have seen that PQ meshes are a discrete analogon of conjugate curve networks. Therefore it is reasonable to use a conjugate curve network for the generation of an approximating PQ mesh of a given surface $\mathcal{S}$. Roughly speaking, the core idea is to

1. Pick some subset D_1 of curves from family C_1 as well as a subset D_2 of curves from family C_2 such that they cover $\mathcal{S}$ in a *reasonable* way[3].
2. Generate mesh vertices at intersection points of curves in D_1 with curves in D_2.
3. Connect these mesh vertices using straight edges that are direct neighbours along curves in D_1 or D_2.
4. Generate a quad-dominant mesh from this set of vertices and edges.
5. Perturb the mesh such that all quadrilateral faces become planar.

Most of these points have pitfalls and will be analyzed in more detail in the following. Principal curvature lines can be traced uniquely through points of a surface that are not umbilic and they form a special conjugate curve network. The use of principal curvature lines for quad-dominant (re-)meshing has been studied e.g. in [2] and [26]. Relative differential geometry leads to relative principal curvature lines, which form conjugate curve networks as well[4]. Consequently one can influence the relative curvature flow and thus the mesh generation by choosing different relativespheres. In the following the phrase *curvature line* will refer to relative curvature line as well, if no distinction between the two is done within the context.

Remark. The usage of relative differential geometry is just one way to generate a conjugate curve network consisting of curve families C_1, C_2. Another possibility is to prescribe one of the curve families and determine the second family by integrating the eigenvector field determined by the conjugate directions of the prescribed family.

4.2.1 Classification of Umbilic Points

Before discussing point 1 in more detail, it is important to understand singularities that can occur in the curvature tensor field. The notions of umbilic point and umbilic area where defined in 2.1.6. In these points the surface $\mathcal{S}$ is isotropic, which means

[3] See section 2.1.8 for the definition of C_1 and C_2.
[4] This was shown in 2.16.

the principal curvature directions are not uniquely defined. As a result the principal curvature lines belonging to either one of the curve families D_1 or D_2 are not uniquely defined and will meet each other in these points.

A brief introduction to the singularities[5] in second order symmetric tensor fields is given in [9]. Elaborate discussions can be found in [28], [8] and [35]. In the following a brief recapitulation will be given.

First of all we will concentrate on symmetric tensor fields like the curvature tensor field[6]. The Weingarten map $d\boldsymbol{n}_p$ is defined on the tangent space $T_p(\mathcal{S})$, for a further treatment it is convenient to *flatten* the surface $\mathcal{S}$ and the curvature tensor field via a conformal[7] mapping to $\mathbb{R}^2$. This can be done at least locally, see [12, Chapter 4]. A recent approach to this is described in [33]. The corresponding parameterisation shall be denoted by $\boldsymbol{x}$. In the plane we choose the Cartesian coordinate system and consider the coordinate matrix $T(x, y) \in \mathbb{R}^{2\times 2}, (x, y) \in \mathbb{R}^2$ of the flattened curvature tensor field $-d\boldsymbol{n}_{\boldsymbol{x}(x,y)}$ in the associated basis. For an umbilic point $p = \boldsymbol{x}(\bar{x}, \bar{y})$ the coordinate matrix must be a multiple of the identity matrix:

$$T(\bar{x}, \bar{y}) = \begin{pmatrix} k_1 & 0 \\ 0 & k_1 \end{pmatrix}$$

Consequently umbilic points fulfill the conditions

$$T_{11}(\bar{x}, \bar{y}) - T_{22}(\bar{x}, \bar{y}) = T_{12}(\bar{x}, \bar{y}) = 0. \tag{4.4}$$

Another possibility is to define the deviator[8] part of our tensor field

$$D := T - \frac{tr(T)}{2} I_2 = \begin{pmatrix} \alpha & \beta \\ \beta & -\alpha \end{pmatrix} \tag{4.5}$$

and check for the condition $\alpha = \beta = 0$. These conditions can be used to locate umbilic points.

In the following we want to inspect the curvature flow in close neighbourhood of an umbilic point. Therefore we will consider the Taylor expansion of D around the umbilic point, neglecting higher order terms. For convenience, the origin of the coordinate system shall be translated to the umbilic point.

$$D(x, y) \approx \nabla D(x, y)^T = \begin{pmatrix} ax + by & cx + dy \\ cx + dy & -ax - by \end{pmatrix} \tag{4.6}$$

Regarding the curvature flow around the umbilic point, one is interested in curvature lines that run into the umbilic point, these are called *separatrices*. Accordingly we are looking for points in the neighbourhood with the curvature flow direction pointing to

[5] Also called degenerate points in [9] and [8], umbilic points is used in the context of differential geometry.

[6] Remember that the relative curvature tensor field is not symmetric.

[7] Conformal meaning angle-preserving.

[8] A trace free tensor is called deviator.

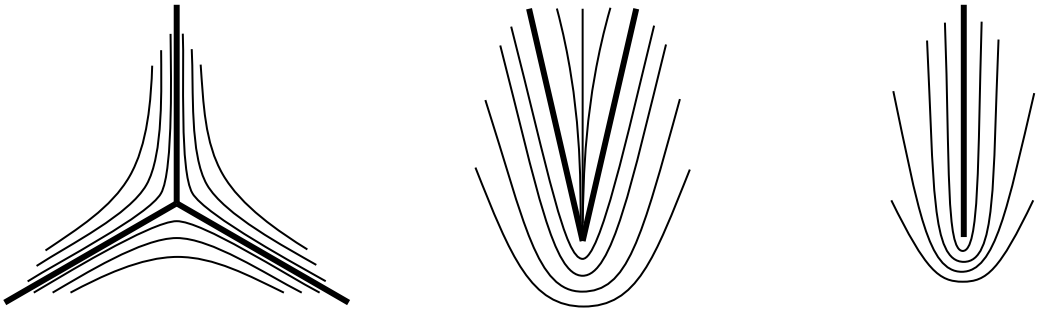


Figure 4.4: Umbilic points: trisector and wedge types

the umbilic point. This leads to the condition $\det(D(x,y)^T,(x,y)^T) = 0$. We choose polar coordinates $(x,y) = r(\cos\theta, \sin\theta)$, use the approximation 4.6 and get

$$r^2 \det(\nabla D(\cos\theta,\sin\theta)^T,(\cos\theta,\sin\theta)^T) = 0, \tag{4.7}$$

thus in the linear case there is no dependency on r. Further we get

$$\begin{aligned}\det(D(x,y)^T,(x,y)^T) &= \\ &= (c\cos\theta + d\sin\theta)(\sin^2\theta - \cos^2\theta) + 2(a\cos\theta + b\sin\theta)\sin\theta\cos\theta \\ &= 0 \qquad (4.8)\\ & \qquad (4.9)\end{aligned}$$

Multiplying by $\cos^{-3}(\theta)$ and substituting $\tan\theta = u$ one arrives at the cubic polynomial

$$du^3 + (c+2b)u^2 + (2a-d)u - c = 0 \tag{4.10}$$

Each real root $\bar{u}$ of this equation corresponds to an angle between a separatrix and the x-axis via $\bar{\theta} = \tan^{-1}\bar{u}$. As there are exactly one or three real roots, there are no more than three separatrices running into an umbilic point. If equation 4.10 has three real roots there are two possibilities, shown in figure 4.4 left. The one with three distinct separatrices is called *trisector type*, the middle one *wedge type*. The case of only one real root is shown in figure 4.4 right. Such an umbilic point is classified as *wedge type* too, it is a special case of the middle one such that the two separatrices degenerate into one.

Remark 4.1. Notice that the angles $\bar{\theta}$ are defined via $\tan^{-1}$, which means they are only defined modulo π so that we actually get six solutions. This is due to the two eigenvector fields of the curvature tensor field. As we focus on one of the eigenvector fields only, this can not be seen in figure 4.4.

The step to equation 4.10 can not be done for $\cos(\theta) = 0$. In this case one has to multiply by $\sin^{-3}(\theta)$, substitute $\cot\theta = v$ and gets

$$cv^3 + (d-2a)v^2 - (c+2b)v - d = 0. \tag{4.11}$$

The roots of this polynomial are equivalent to the roots of 4.10 in case $c \neq 0$ and $d \neq 0$ via the substitution $u = 1/v$. As an example assume $d = 0$, in which case 4.10 is a quadratic polynomial. In this case the solution $v = 0$ of 4.11 corresponding to $\theta = \pi/2 \mod \pi$ has to be considered too.
Figure 4.4 illustrates that the eigenvector fields can not be oriented consistently in the vicinity of umbilic points in general. Moreover the separatrices divide the neighbourhood of umbilic points into sectors with different behavior of the curvature lines. In the sectors of the trisector type, the integral lines sweep past the umbilic point, while in the wedge type another sector occurs where integral lines end in the umbilic point.

Up to now we have been focussing on the linear approximation of $D(x, y)$. It is pointed out in [35, Section 3.4.2] that this is a good approximation if ∇D is a regular matrix. Otherwise there may be additional solutions to 4.7. The regularity of ∇D may be expressed in the coefficients a, b, c, d:

$$\delta := \begin{vmatrix} a & b \\ c & d \end{vmatrix} \neq 0 \tag{4.12}$$

The sign of this determinant can be used to distinguish between the trisector and wedge types as pointed out in [9]. For the case $\delta = 0$ I refer to [28], [8] and [35].

The classification for the flattened relative curvature tensor field $-d\boldsymbol{n}^{\mathcal{R}}_{\boldsymbol{x}(x,y)}$ can be done in a very similar way, taking care of the fact that this tensor field is not symmetric. Let $T^{\mathcal{R}}(x, y)$ be its coordinate matrix. Condition 4.4 has to be replaced by

$$T_{11}(\bar{x}, \bar{y}) - T_{22}(\bar{x}, \bar{y}) = T_{12}(\bar{x}, \bar{y}) = T_{21}(\bar{x}, \bar{y}) = 0 \tag{4.13}$$

to check for umbilic points, and the deviator is not symmetric anymore.

$$D^{\mathcal{R}} := T^{\mathcal{R}} - \frac{tr(T^{\mathcal{R}})}{2} I_2 = \begin{pmatrix} \alpha & \gamma \\ \beta & -\alpha \end{pmatrix} \tag{4.14}$$

Now one has to check for $\alpha = \beta = \gamma = 0$ in order to locate umbilic points. Consequently the linearization becomes assymetric:

$$D^{\mathcal{R}}(x, y) \approx \nabla D^{\mathcal{R}} \cdot (x, y)^T = \begin{pmatrix} ax + by & ex + fy \\ cx + dy & -ax - by \end{pmatrix} \tag{4.15}$$

Analogous steps as in 4.7 and 4.8 lead to the polynomials

$$fu^3 + 8e + 2b)u^2 + (2a - d)u - c = 0 \tag{4.16}$$

$$cv^3 + (d - 2a)v^2 - (e + 2b)v - f = 0 \tag{4.17}$$

which are equivalent to 4.10 and 4.11 respectively for $c = e$ and $d = f$, as could be expected.

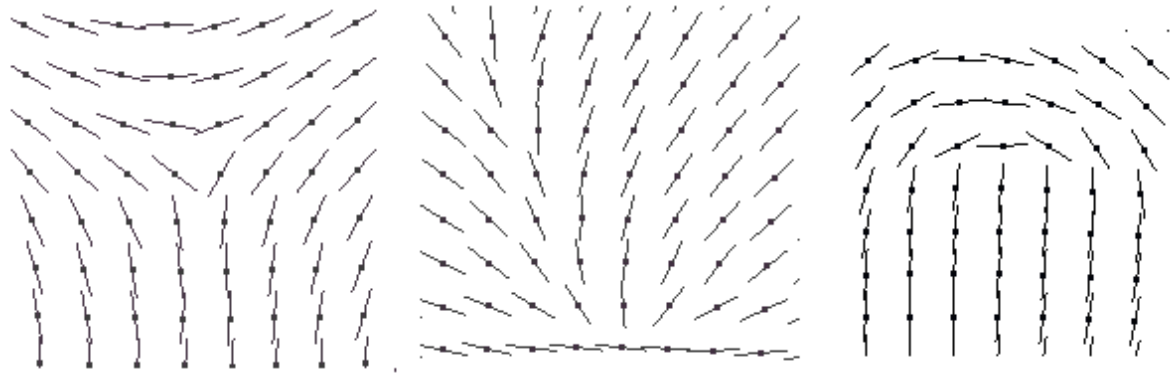


Figure 4.5: Practical examples of umbilic points: trisector and wedge types

Remark. In the practical case, the surface $\mathcal{S}$ is defined via a triangular mesh and the curvature tensor is available only in the vertices, see section 3.2. Linear interpolation is used to define the curvature tensor on the faces, such that checking condition 4.4 or 4.13 results in 2×2 or 3×2 linear equation systems respectively. For a description of the implementation see section 6.4.

Figure 4.5 shows some practical examples of an eigenvector field near umbilic points of different type.

4.2.2 Smoothing of the Curvature Tensor Field

Smoothing of the curvature tensor field is reasonable because of the following:

1. Given a discete surface by a mesh, the curvature tensor field may be available only in discrete points of the surface (usually in the vertices). As a consequence of the discreetness, the curvature tensor field is prone to jitter.

2. In umbilic areas the eigenvector fields are not uniquely defined. One wants to propagate the directions of the eigenvectors from neighbouring non-umbilic areas into the umbilic areas by smoothing the eigenvector fields.

Several techniques have been proposed for the smoothing of discretely defined curvature tensor fields. In [26] a technique involving a discrete Laplacian operator was proposed that addresses both points above. A global approach is described in [33]. In the following short introductions to both methods are given. Note that these methods were developed for the smoothing of the classical curvature tensor field, and are not directly applicable to the relative curvature tensor field. This will be discussed in the following.

Laplacian smoothing

This technique originates from the discretisation of the heat equation. Let u be a scalar field defined in the vertices of a mesh. We consider the discretisation of the equation

$$u_t = (1 - \lambda)\Delta u \tag{4.18}$$

where $0 < \lambda < 1$ may depend on the spatial coordinates x and y. Now we discretise the time derivative by a forward difference quotient $(u^{k+1} - u^k)/dt$ and the Laplacian according to [11] and obtain

$$\begin{aligned} u_v^{k+1} &= u_v^t + dt(1-\lambda)\Delta u \\ &= u_v^t + dt(1-\lambda) \sum_{q \in Nb(v)} w_{vq}(u_q^k - u_v^k) \\ &= \lambda u_v^t + dt(1-\lambda) \sum_{q \in Nb(v)} w_{vq} u_v^k. \end{aligned} \tag{4.19}$$

Here u_v^k denotes u at time step k at vertex v, $Nb(v)$ the neighbouring vertices to v and w_{vq} weights assigned to the edges. These weights shall fulfill $\sum_{q \in Nb(v)} w_{vq} = 1$. It is well known that for this type of explicit Euler integration scheme $dt(1-\lambda) < 1$ must hold for stability. One may choose $dt = 1$, as $0 < \lambda < 1$.

In [26] Marinov and Kobbelt suggest to plug the flattened curvature tensor matrices T instead of u into 4.19 and use *confidence* coefficients for λ and the w_{vq}, as described in the following.

The parameter λ in 4.18 controls the amount of smoothing, this can also be seen in 4.19. The smaller λ, the more influence the neighbouring vertices will have and thus more smoothing will occur. This is what one would like to have near and in umbilic areas. Consequently one could use conditions 4.4 and 4.13, which occur exactly at umbilic points, as a basis for λ, e.g. $\lambda = ||D||$ or $\lambda = |k_1/k_2 - 1|$.

Another possibility can be seen in figure 4.5. Consider the direction of the eigenvectors in vertices attached to a single face. For faces near umbilic points or in umbilic areas there will be bigger differences in these directions. Therefore in [26] it is suggested to use this direction coherence as a measure for the confidence coefficients. This can be done for each face of the triangulated surface by projecting the directions of one eigenvector field onto the plane defined by the face and computing the cosine of the angles $\alpha_0, \alpha_1, \alpha_2$ between each pair of such flattened directions. The confidence coefficient for a face f can then be defined by

$$\lambda(f) = \min_{j=0,1,2} |\cos \alpha_j| \tag{4.20}$$

Note that a possible inconsistency in the direction of the eigenvectors does not matter in this case because of $|\cos \alpha_j|$. For the vertices the confidence coefficients can now be defined by averaging the confidence coefficients of the adjacent faces. Marinov and Kobbelt in [26] suggest to use the following setup:

$$T_v^{k+1} = \frac{\lambda(v)}{2} T_v^t + \left(1 - \frac{\lambda(v)}{2}\right) \frac{\sum_{q \in Nb(v)} \lambda(q) T_q^k}{\sum_{q \in Nb(v)} \lambda(q)} \tag{4.21}$$

The damping factor of 1/2 is chosen in order to allow for more smoothing in non-umbilic areas. With each iteration the directions of neighbouring anisotropic areas are propagated into umbilic regions, as can be seen in figure 4.6.

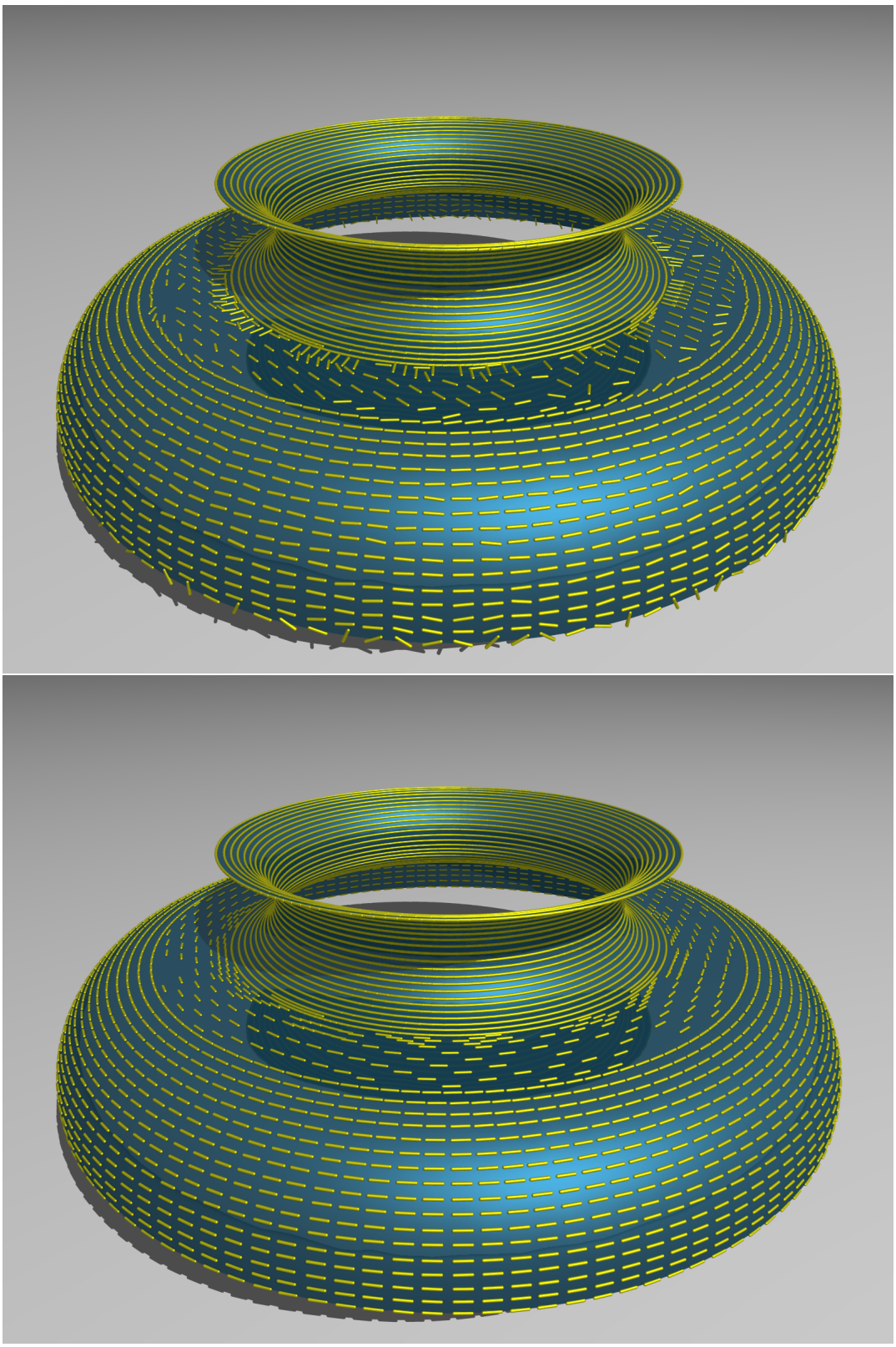


Figure 4.6: Laplacian smoothing of eigenvector field. Note that the eigenvector directions have been ordered according to the absolute value of the corresponding curvatures, as opposed to remark 2.10. The plots show the eigenvector directions corresponding to the curvature with less absolute value.

Global smoothing

A global approach based on the minimization of an energy functional is presented in [33]. The degrees of freedom in this energy functional are the angles α by which the eigenvector fields are rotated in the vertices. Roughly speaking the energy functional consists of two terms. A *fitting term* that controls the deviation of the eigenvector fields from their original directions, and a *smoothing term* that aims at minimizing the variation of the directions in the faces. In order to define this terms one uses reference directions in each of the vertices. This can be achieved by choosing an arbitrary edge adjacent to a vertex and projecting it to the tangent plane in this vertex. Using these reference directions one computes the angles α_v^0 between one of the curvature directions and the reference direction for the vertices v. These angles are then used as references for measuring the deviation and variation as mentioned above. In [33] the energy functional is defined in the form

$$E(\boldsymbol{\alpha}) = (1-\rho)\underbrace{\sum_{v\in V} |\frac{k_1}{k_2}| g(\alpha_v - \alpha_v^0)^2}_{\text{fitting term}} + \rho \underbrace{\sum_{f\in F} R_f(\boldsymbol{\alpha})^2}_{\text{smoothing term}} \tag{4.22}$$

where $\boldsymbol{\alpha}$ denotes a vector consisting of all α_v, g is used to take care of the periodicity of α_v and R_f measures the variation of the eigenvector field on the face f and depends on the angles α_v corresponding to vertices adjacent to f only. The parameter ρ is used to control the amount of smoothing. This method gives very good results for the generation of quad meshes because of its global nature, see [33]. It evenly distributes umbilic points over the surface, a property not exhibited by the Laplacian smoothing. These advantages come at the cost of higher computational effort, because the energy functional possesses number of vertices parameters. The method is not directly applicable to the smoothing of the relative curvature tensor field, as will become clear in the following.

Implications of the Smoothing to the Generation of PQ Meshes

In practice the relative curvature tensor field is computed from the curvature tensor fields of $\mathcal{S}$ and $\mathcal{R}$. If one wants to smooth the relative curvature tensor field, one has to take care about the conjugacy of the relative eigenvector fields with respect to the curvature tensor field of $\mathcal{S}$. Remember that the conjugacy of the relative eigenvector fields is important because we want to generate a quad mesh with nearly planar faces, which we can expect using conjugate curvature lines only. This could be incorporated into the global smoothing method by smoothing one of the relative eigenvector fields and afterwards calculating the second eigenvector field using equation 2.11. Moreover, the two points at the beginning of this section, that motivated the smoothing, can also be addressed differently:

1. The method for computation of the curvature tensor field described in section 3.2 smooths the curvature tensor field by nature.

2. Umbilic areas that are not flat can be transformed to relative non-umbilic areas by choosing an appropriate relativesphere. This is illustrated in figure 4.7, which

shows a cylinder topped by a half of a sphere. The short lines indicate the direction of the minor eigenvector field. Clearly the half sphere part of the surface is umbilic. The relativesphere used in the example was generated by slightly stretching the unitsphere in the upward direction using a factor of 1/2, resulting in an ellipsoid. Consequently the umbilic area becomes relative non-umbilic and a single umbilic point is left at the top. Another example, were an appropriate choice of a relativesphere led to the removal of an umbilic region, will be given in example 5.13, see also figure 5.1.

4.2.3 Integration of Relative Curvature Lines

In this section we will focus on the integration of integral curves of the eigenvector fields[9] given by the relative principal directions. Assume that starting points for the relative curvature lines are given, the question how to choose these will be addressed in the next section. Different techniques for integration of (eigen)vector fields have been described in [13], [26] and [2], requiring no, a local and a global surface parameterisation respectively. The following listing gives a short comparison.

No parameterisation, [13] This is the easiest of the three methods. Integral lines are traced directly on the given triangulated surface. Straight line segments are used on faces, which are parallel to the eigenvector direction averaged over the respective face. The intersection points of these line segments with the faces' edges define the vertices of the integral lines. They are used as starting points for the continuation of the integration on the neighbouring faces. This can be interpreted as an explicit Euler scheme with step lengths chosen such that one reaches an edge. Implementation of this method is straightforward, but results depend strongly on the original mesh and can be expected to be inaccurate.

Local parameterisation, [26] In this method faces are mapped isometrically to the plane. Line integration is done via an explicit Euler scheme again, but the step length is chosen according to the norm of the deviator, see 4.5 and 4.14. Consequently the step length will become smaller in the vicinity of umbilic points, where the eigenvector field is prone to fast turns. Using this concept, crossing of one or several edges within one integration step might occur, in which case the corresponding faces are unfolded isometrically to the plane. The confidence coefficients as defined in 4.20 are used to judge between regions where the directions of the eigenvector field are reliable or not. Accordingly the authors propose to switch between two types of line tracing. In anisotropic regions, where the directions given by the eigenvector field are clearly defined, they use those to trace curvature lines. When the confidence coefficients fall below a given threshold, they trace the lines in the last reliable direction. On leaving an isotropic area (detected by the confidence coefficients rising above the threshold) they start to trace curvature

[9] In this section *eigenvector field* refers to either the major or the minor eigenvector field.

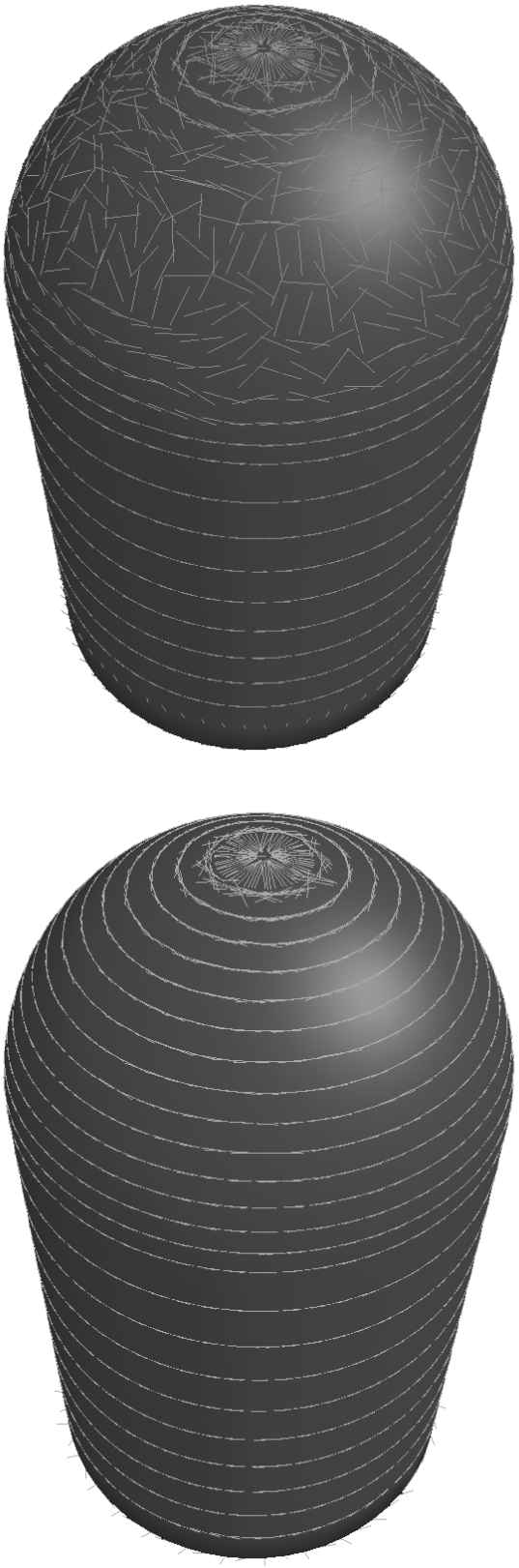


Figure 4.7: Smoothing of eigenvector fields by using an appropriate relativesphere

lines again, but the use of minimum or maximum eigenvector field may change according to which direction fits better the last direction traced in the isotropic region.

Global parameterisation, [2] If a global parameterisation is available, it is possible to trace the principal curvature lines directly in parameter space, omitting the need to take care of the discrete surface representation. Therefore sophisticated higher order integration schemes can be used, the authors of [2] use a fourth order Runge-Kutta integration scheme with adaptive step length.

The first method was implemented in the accompanying software. Given a starting point, the relative curvature lines are traced in both directions defined by the eigenvector field, until one of the following **stopping criteria** is fulfilled:

1. A boundary of the given surface is reached.
2. The relative curvature line comes too close to an existing one or itself. Closeness in this context depends on the local curvature of the surface, this will be addressed in the next section.
3. The relative curvature line reaches a face containing an umbilic point.
4. $|k_1/k_2 - 1|$ falls below a predefined threshold.

If a curvature line does not reach a predefined length, it will be removed again and another starting point will be chosen. This is done because few long curvature lines produce less hanging nodes in the resulting mesh than a lot of short curvature lines, see section 4.2.5.

Point 2 indicates that a lot of proximity queries need to be done during the integration, namely in each integration step. For this reason the performance of the integration routine will depend strongly on the response time of these queries.

A classical tool for the subdivision of 3D data is an octree. Roughly speaking a tree is a data structure consisting of connected nodes, which are accessed starting at a root node. Each node is classified either as leaf or internal node. An internal nodes has at least one child node, the unique internal node connected to a child is also referred to as its parent. For a formal definition see [7]. An octree is a tree such that all internal nodes have exactly 8 child nodes, figure 4.8 shows an example.

The cube corresponding to the root node is chosen such that it contains the given surface. Next this root cube is subdivided into 8 subcubes and the spatial objects one wants to structurize are assigned to the cubes they lie in. This process is iterated until all objects have been assigned to leaf cubes of a given predefined size. Using this data structure, the amount of objects to be processed in proximity queries can be restricted significantly. Assume that given a point $p \in \mathcal{S}$ one wants to find all objects within a geodesic disc of radius r on $\mathcal{S}$, denote this set by U. A superset $V \supset U$ can be found by choosing all objects that lie in leaf cubes which intersect the ball with radius r around p, these is a direct conclusion from the fact that the distance on $\mathcal{S}$ is bounded below

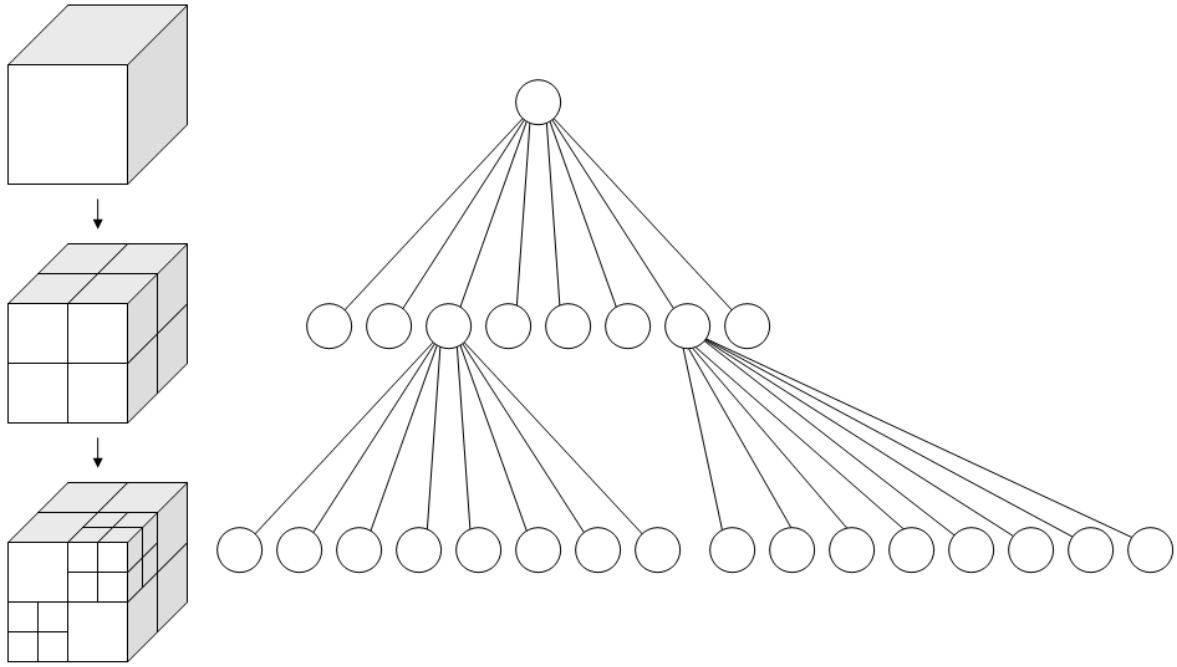


Figure 4.8: An example octree

by the Euclidean distance in $\mathcal{R}^3$. The leaf cubes fulfilling this property are found easily by traversing the octree structure. V can now be further limited to all objects with Euclidean distance less or equal r to p, which gives a good upper approximation to U at least in directions starting from p with small normal curvature. This has been implemented in the accompanying software, using the meshes vertices' as objects for doing proximity queries to.

Remark. Using this octree data structure we get a list of vertices, but we want to check for the proximity to segments of the curvature lines. Since the integration method used produces line segments on the faces, one can approximate the distance to a line segment by taking the distance to the faces' closest vertex plus the distance from this vertex to the line segment.

Another possibility would be to build an octree structure of the line segments directly. A drawback of this approach is, that the octree would have to be updated after each integration step in order to include the new line segments.

In [26] the authors propose a method that does not use an octree or similar global search structure. Instead they keep a list of faces with at least one vertex within distance r to p. A list of such vertices can be computed using Dijkstra's algorithm (cf. [7]). Afterwards the corresponding list of faces is looked through for line segments.

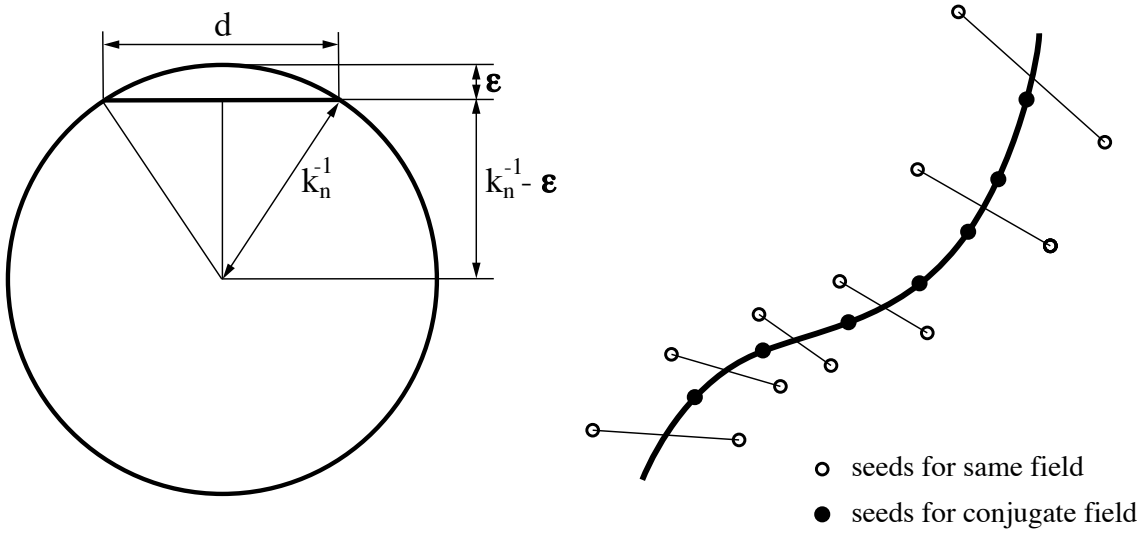


Figure 4.9: Osculating circle and linear approximation, seed point selection

4.2.4 Seed Point Selection

So far the selection of starting points has not been discussed. This is maybe the most crucial point for the remeshing process. The relative curvature lines should cover $\mathcal{S}$ evenly and still follow its anisotropy, which seems to be an antagonism at first glance. In practice one maintains a list of possible starting points, these points are referred to as seed points. Within this list the candidates get prioritized according to different criteria. After each generation of a curvature line the list gets updated: candidates that are too close to existing curvature lines are removed, new seed points might be added. Then the seed point with the highest priority is chosen as the next starting point. Usually seed points are also added to the list during the integration of a curvature line. Each seed point in the list might be marked belonging to a specific one of the two eigenvector fields. The creation of curvature lines goes on until no more seed points are left in the queue.

First we will have a look at the local density that the integral lines of the two eigenvector fields should have. Apparently this density will depend on the local curvature. Consider an integral line through a point $p \in \mathcal{S}$ and its orthogonal projection onto the normal plane to $\mathcal{S}$ along the integral line[10]. The curvature of this projection is the normal curvature k_n of the integral line in p. As we have seen in 2.5 it can be calculated by means of the tangent vector and the second fundamental form. The osculating circle with radius $1/k_n$ and midpoint $p + 1/k_n \boldsymbol{n}_p$ approximates the projection of the integral line up to second order. We will use this approximation of the integral line to derive an expression for the local density. According to figure 4.9 left, if we want the maximum error between the osculating circle and the linear approximation to be less than a

[10]Cf. definition 2.4.

predefined parameter $\epsilon > 0$, a simple calculation shows that for the distance $d(k_n)$ the following must hold:

$$\left(\frac{1}{|k_n|}\right)^2 = \left(\frac{1}{|k_n|} - \epsilon\right)^2 + \left(\frac{d(k_n)}{2}\right)^2$$
$$\Leftrightarrow d(k_n) = 2\sqrt{\epsilon\left(\frac{2}{|k_n|} - \epsilon\right)} \tag{4.23}$$

As a result, an approximation of the optimum distance of integral lines for one eigenvector field is given by 4.23 evaluated at the normal curvature in the direction of the other eigenvector field.

Remark. When integrating a curvature line, 4.23 is used as a stopping criterion. Typically one stops the integration if the distance to the closest curvature line falls below the optimum distance times a user-defined parameter. It makes sense to relax this criterion in the vicinity of umbilic points, as the curvature lines will meet in these points anyway. Thus one gets a finer discretisation near these points in order to better resolve the singularity.

In practice 4.23 can not be used directly, because of its singularity for $k_n = 0$. Therefore one can replace it by

$$d(k_n) = 2\sqrt{\epsilon\left(\frac{2}{c + |k_n|} - \epsilon\right)} \tag{4.24}$$

where the choice of $c > 0$ determines a maximum distance of curvature lines in flat regions of $\mathcal{S}$.

From the ideas above, a first possibility of seed point generation can be deduced. If one integrates a curvature line for the eigenvector field $\boldsymbol{e}_1^{\mathcal{R}}$, seed points for both of the eigenvector fields can be generated along the way.

Type 1 Seed points for the conjugate eigenvector field $\boldsymbol{e}_2^{\mathcal{R}}$ can be generated directly along the curvature line in distances $\mathrm{d}(k_n(\boldsymbol{e}_1^{\mathcal{R}}))$.

Type 2 Seed points for the same eigenvector field can be calculated from $p \pm d(k_n(\boldsymbol{e}_2^{\mathcal{R}}))\boldsymbol{e}_2^{\mathcal{R}}$, where p denotes a point on the curvature line.

Note that the distance along the curvature line for seed points of the second type does not matter. Figure 4.9 right illustrates the two types of seeding.

Type 1 generation of seed points was originally introduced in [20]. The same method is used in [2], the two eigenvector fields are treated separately. The authors of [26] use type 2 seeding, they integrate curvature lines alternating between the eigenvector fields.

Depending on the surface, the type of seeding and the prioritisation in the queue of seed points, there might be some parts of the surface that are left uncovered. This can

be resolved by randomly picking points on the surface and checking the optimum versus the actual distance of curvature lines, or by scanning the surface systematically. In the accompanying software the user also has the possibility to manually choose starting points for curvature line integration.

Both types of seeding need some initial seeds to be chosen. In [2] umbilic points are chosen as initial seeds. Depending on wedge or trisector type, one or three lines are started respectively, see figure 4.4. In [26] arguments against this option are presented, because the directions near umbilic points are by nature unreliable. Thus the authors propose to choose the starting seeds in anisotropic areas, where the eigenvector directions are clearly defined.

The prioritisation in the queue of seed points can be done by comparing the local optimum distance to the actual distance of the seed point to neighbouring lines. Furthermore hierarchy levels can be introduced. If a curvature line belongs to level l, the seed points that are generated while integrating this line will be marked belonging to level $l+1$. This makes sense if one uses the umbilic points as initial seeds, because lines starting from these usually give a good topological division of the surface. By using the hierarchy level as primary sorting criterion, the coverage with curvature lines will grow uniformly around the initial lines.

4.2.5 Construction of the Output Mesh

The previous steps have covered the surface with a network of curvature lines, which partition the surface into quadrilateral patches in non-umbilic regions. In the next step a mesh has to be constructed from these curvature lines.

Vertex creation

First of all a list of intersection points of curvature lines is generated. The curvature lines are stored as straight line segments per face, consequently the intersection points can easily be calculated. One only needs to consider faces that contain line segments of curvature lines from different eigenvector fields.

Additional points added to the list are the umbilic points, as well as intersection points of curvature lines with the boundary.

Edge creation

The list of edges is generated by walking along the curvature lines and connecting subsequent intersection points that have been calculated in the previous step. At last the intersection points on the boundary are connected by tracing the boundary on the original mesh.

As the remeshing is done without parameterisation, care has to be taken about the orientation. At this step of the process a vertex list and an edge list are available. A half-edge list is created in the next step, containing one record for boundary edges and two records for the remaining edges. In order to keep the orientation consistent with

the original surface, the normal vector of the underlying face is attached to the newly generated vertices in the vertex creation phase. It can now be used to define tangent spaces in these vertices and consistently orient them. Consider the projection of the edges adjacent to a vertex onto the tangent space. An arbitrary reference edge is chosen from these projections. The edges can now be sorted according to their angle to the reference edge. The next/prev pointers of an incoming half-edge record can thus be set pointing to the next outgoing half-edge record in counter-clockwise direction.

Face creation

In this step the half-edge list is searched for loops, which will define the faces. First of all the half-edges are added to a list of *free* half-edges. Then the free half-edges are traversed using the next-pointers starting from an arbitrary half-edge. On reaching the starting half-edge again, a new face is created and the corresponding half-edges are removed from the list of free half-edges. This process is continued until no more free half-edges remain. It might happen that no more loops can be found in the half-edge list while it still is not empty. This might happen if the original mesh is coarse and the projection onto the tangent space changes the orientation (see previous step). This can resolved by removing the wrongly oriented edges from the half-edge list at the cost of loosing details in the resulting mesh.

Remark. The steps described above produce a mesh that consists mainly of quad faces. Note that it is not clear whether this mesh is topologically equivalent to the original one. It is very likely that holes are created near umbilic points, because the integration of curvature lines towards these points may stop before reaching them. This must be resolved by connecting the vertices at umbilic points to nearby mesh vertices. Usually this can not be done using quadrilateral faces only.

Subdomains of the original surface that are umbilic should be addressed in earlier steps. Non-flat umbilic subdomains can be dealt with easily by choosing a different relativesphere. For flat subdomains the approach described in [26] (cf. section 4.2.3) gives very good results.

Remark. The faces created in the last step will possibly include n-gons for $n > 4$. Moreover quadrilaterals and these n-gons might be concave. One can avoid this by decomposing them into convex quadrilaterals and triangles, thereby creating a mesh consisting of mostly quadrilaterals and some triangles (called *quad-dominant mesh* in the following).

4.3 Planarization of Quad Meshes

Section 4.1 suggests that the quadrilaterals generated will be close to planar. The topic of this section is how to transform the quad-dominant mesh into a planar quad-dominant mesh. The method presented here was initially mentioned in [23] and is called *PQ perturbation*. In the following we will consider a mesh (V, E) consisting of quad faces only,

which is no restriction because the constraint of planarity addresses these faces only.

The main idea is to minimally perturb the vertices of the quad mesh in order to planarize the faces. The resulting mesh shall still be close to the original surface. A possibility to measure this is to use the squared distances from vertices v to corresponding closest points $p_v \in \mathcal{S}$. Summing over the vertices this gives the so-called closeness term

$$f_{close} := \sum_{v \in V} ||v - p_v||^2. \tag{4.25}$$

In addition, the smoothness of the mesh can be controlled by adding a so-called fairness term that measures the bending energies in the rows and columns defined by the quad mesh. Let v_+^1, v_-^1 and v_+^2, v_-^2 denote the neighbouring vertices to v in the two eigenvector directions respectively. The bending energy is defined by means of a second order central difference quotient:

$$f_{fair} := \sum_{v \in V} \left(||v_+^1 - 2v + v_-^1||^2 + ||v_+^2 - 2v + v_-^2||^2 \right). \tag{4.26}$$

The perturbation can then be stated as a constrained optimization problem, where the constraints express the planarity of the faces. A possible choice for the constraints is to consider the angles enclosed by consecutive edges of the quad faces. For a face f we denote the angles corresponding to vertices $v_1, \ldots, v_4$ by $\alpha_1^f, \ldots, \alpha_4^f$. They shall be measured in the interval $[0, \pi]$, see figure 4.10 for examples. Consider the unit vectors $e_{ij} := \frac{v_i - v_j}{||v_i - v_j||}, i \neq j$. The angles can be computed by

$$\begin{aligned} \alpha_1^f &= \cos^{-1}(\langle e_{21}, e_{41} \rangle) \\ \alpha_2^f &= \cos^{-1}(\langle e_{32}, e_{12} \rangle) \\ \alpha_3^f &= \cos^{-1}(\langle e_{43}, e_{23} \rangle) \\ \alpha_4^f &= \cos^{-1}(\langle e_{14}, e_{34} \rangle) \end{aligned} \tag{4.27}$$

The following proposition holds for the sum of these angles:

Proposition 4.2. *Given a quad face f consider the quadrilateral polygon defined by its edges. The sum of angles*

$$a_f := \alpha_1^f + \alpha_2^f + \alpha_3^f + \alpha_4^f \tag{4.28}$$

is equal to 2π if and only if f is planar and convex.

Proof. If f is planar and convex then obviously $a_f = 2\pi$.

On the contrary we show that f not planar or f not convex implies $a_f < 2\pi$. In order to do this consider the following cases:

f is planar but concave: The sum of inner angles equals 2π, but one of the inner angles, say β, exceeds π (see figure 4.10). It will be measured by $\alpha_3^f = 2\pi - \beta$ and thus $a_f < 2\pi$.

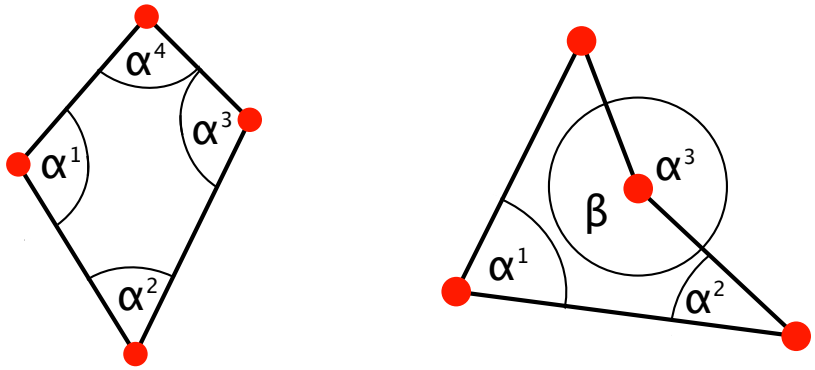


Figure 4.10: Definition of $\alpha_1^f, \ldots, \alpha_4^f$

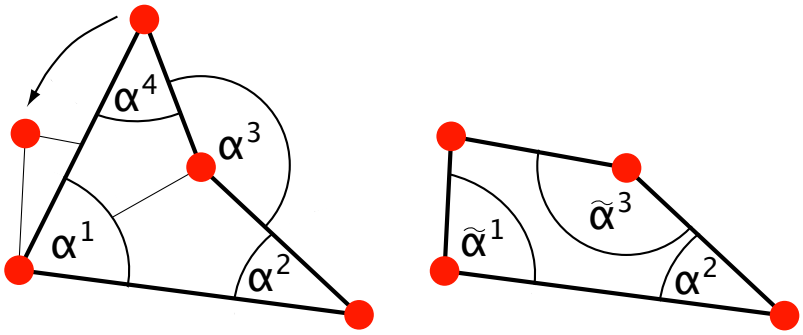


Figure 4.11: Unfolding of a non-planar face

f is not planar: Choose two vertices of the polygon that are not connected by an edge, without loss of generality these shall be α_1^f and α_3^f. Rotate vertex v_4 around the axis defined by v_1 and v_3 such that f becomes planar. Doing this, the angles α_2^f and α_4^f stay the same, but α_1^f, α_3^f change to $\tilde{\alpha}_1^f$ and $\tilde{\alpha}_3^f$ respectively, see figure 4.11. According to the first case it holds that $\tilde{\alpha}_1^f + \alpha_2^f + \tilde{\alpha}_3^f + \alpha_4^f \leq 2\pi$, depending on the convexity of f. Now it suffices to show $\alpha_1^f < \tilde{\alpha}_1^f$ and $\alpha_3^f < \tilde{\alpha}_3^f$.

It holds that $\cos(\tilde{\alpha}_3^f) = \langle \tilde{e}_{43}, e_{23} \rangle$. Choose a unit vector n orthogonal to $\tilde{e}_{43}$ and e_{23} and $0 < \gamma < \pi$ such that $e_{43} = \cos\gamma \tilde{e}_{43} + \sin\gamma n$. From this follows

$$\cos\alpha_3^f = \langle e_{43}, e_{23} \rangle = \cos\gamma \langle \tilde{e}_{43}, e_{23} \rangle = \cos\gamma \cos\tilde{\alpha}_3^f < \cos\tilde{\alpha}_3^f$$

and thus $\alpha_3^f < \tilde{\alpha}_3^f$. The inequality for α_1^f can be shown analogously. □

We define constraints $c_f := a_f - 2\pi$. The constrained optimization problem can thus be stated using the Lagrangian multiplier formulation

$$f_{pq}(x, \lambda) := w_{close} f_{close}(x) + w_{fair} f_{fair}(x) + \sum_{f \in F} \lambda_f c_f(x). \tag{4.29}$$

where x denotes a vector of all vertex coordinates, $\lambda = (\lambda_f)_{f \in F}$ is the vector of Lagrangian multipliers and w_{close}, w_{fair} are user-defined constants that control the weighting of approximation error and smoothness. An unconstrained minimizer $(\tilde{x}, \tilde{\lambda})$ of 4.29 minimizes $w_{close} f_{close} + w_{fair} f_{fair}$ subject to the constraints $(c_f)_{f \in F} = 0$. The authors of [23] use Sequential Quadratic Programming (SQP) as described in [25] for the computation of a minimizer. In the following an overview of SQP applied to this minimization problem is given, for a more detailed description see [23] and [25].
For better readability we define $f(x) := w_{close} f_{close}(x) + w_{fair} f_{fair}(x)$, $c(x) := -(c_f(x))_{f \in F}$ and rewrite 4.29 in vector notation $f_{pq}(x, \lambda) = f(x) - \lambda^T c(x)$. Let $\nabla_{x,\lambda} f_{pq}$ denote the gradient of f_{pq} with respect to (x, λ), $\nabla f(x)$ shall represent the gradient of $f(x)$ and $J(x)$ the Jacobian of the constraints $c(x)$. The following condition is necessary for a minimizer $(\tilde{x}, \tilde{\lambda})$ of f_{pq}:

$$\nabla_{x,\lambda} f_{pq}(\tilde{x}, \tilde{\lambda}) = \begin{pmatrix} \nabla f(\tilde{x}) - J^T(\tilde{x})\tilde{\lambda} \\ -c(\tilde{x}) \end{pmatrix} = 0 \tag{4.30}$$

This nonlinear equation can be solved iteratively using Newton's method of linearizing around a given pair (x_i, λ_i):

$$\begin{aligned} &\nabla_{(x,\lambda)} f_{pq}(x_i + h_i, \lambda_i + \eta_i) \approx \\ &\approx \begin{pmatrix} \nabla f(x_i) - J^T(x_i)\lambda_i \\ -c(x_i) \end{pmatrix} + \begin{pmatrix} [\nabla^2 f(x_i) + \sum_{f \in F} \nabla^2 c_f(x_i)\lambda_{if}] & -J^T(x_i) \\ -J^T(x_i) & 0 \end{pmatrix} \begin{pmatrix} h_i \\ \eta_i \end{pmatrix} \end{aligned} \tag{4.31}$$

Here $\nabla^2 f$ and $\nabla^2 c_f$ denote the Hessians of f and c_f with respect to x. For shorter notation we define $H(x, \lambda) := \nabla^2 f(x) + \sum_{f \in F} \nabla^2 c_f(x)\lambda_f$. Plugging the linearization into 4.30 we get a linear equation system for the update steps (h_i, η_i):

$$\begin{pmatrix} H(x_i, \lambda_i) & -J^T(x_i) \\ -J^T(x_i) & 0 \end{pmatrix} \begin{pmatrix} h_i \\ \eta_i \end{pmatrix} = \begin{pmatrix} -\nabla f(x_i) + J^T(x_i)\lambda_i \\ c(x_i) \end{pmatrix} \tag{4.32}$$

Substituting $\lambda_{i+1} = \lambda_i + \eta_i$ one gets

$$\begin{pmatrix} H & -J^T(x_i) \\ -J^T(x_i) & 0 \end{pmatrix} \begin{pmatrix} h_i \\ \lambda_{i+1} \end{pmatrix} = \begin{pmatrix} -\nabla f(x_i) \\ c(x_i) \end{pmatrix} \tag{4.33}$$

The actual update step is determined using a line-search strategy, which roughly means that a factor $0 < d \leq 1$ is determined such that $f_{pq}(x_i + dh_i) < f_{pq}(x_i)$. Several update steps will be necessary in order to reach a predefined stopping criterion. Usually one uses $\max_{f \in F} |c_f|$ for that purpose.

Remark. Note that 4.30 is not a sufficient condition, in fact it is also necessary for local maxima. However these will not occur in practice.

It is necessary to repeatedly compute the Hessians $\nabla^2 c_f, \nabla^2 f_{fair}, \nabla^2 f_{close}$. Expressions for the first two Hessians can be derived from 4.27 and 4.26 respectively, but the definition

of f_{close} involves the computation of closest points on $\mathcal{S}$ and thus the Hessian can not be given in a closed form. The authors of [23] therefore suggest the following approximation for $||v - p_v||^2$. As the difference $v - p_v$ is orthogonal to the tangent plane $T_{p_v}(\mathcal{S})$, the normal vector n_{p_v} in the point p_v on $\mathcal{S}$ is a multiple of $v - p_v$, consequently $||v - p_v||^2 = \langle v - p_v, n_{p_v}\rangle^2$. The right hand side of this equation is used as an approximation by fixing p_v and n_{p_v}.
The matrix in 4.33 is highly sparse, which follows directly from 4.29 because all terms depend on neighbouring vertices only. Thus efficient solvers can be used.

Remark 4.3. In practice one might want to fix certain vertices, e.g. the boundary vertices. This is simple done by optimizing for the remaining vertex coordinates only.

Penalty method

The optimization problem can also be stated without the use of constraints for expressing the planarity of the quads. In this case 4.29 is replaced by

$$f_{pq}^{penalty}(x) := w_{close} f_{close}(x) + w_{fair} f_{fair}(x) + f_{angle} \tag{4.34}$$

where

$$f_{angle} := \sum_{f \in F} |c_f|. \tag{4.35}$$

The minimizer of 4.34 can effectively be computed using the Gauss-Newton method with Levenberg-Marquardt regularization, see [24].

Example 4.4. Figure 4.12 shows a section of a quad mesh before and after planarization. The lower part shows two different results, which were obtained by the penalty method using the stopping criterion $\max_{f \in F} |c_f| < 5 \cdot 10^{-4}$ and $w_{close} = w_{fair} = 1$. The right result shows the case of fixed boundary vertices. Fixing all boundary vertices imposes a strict constraint on faces touching the boundary, depending on the number of vertices of the face that lie on the boundary. Therefore it makes sense to subdivide faces with more than two vertices on the boundary, if all of the boundary vertices should stay fixed. Otherwise the planarization might result in a mesh that is less fair, as can be seen in figure 4.12.

Remark. In section 4.2.5 it has become clear that the construction of the output mesh will possibly include n-gons with $n > 4$. For aesthetical purposes one might want to refrain from decomposing them into quads and triangles. The planarization of such faces is done by planarizing sufficiently many overlapping sub-quads, see also [23].

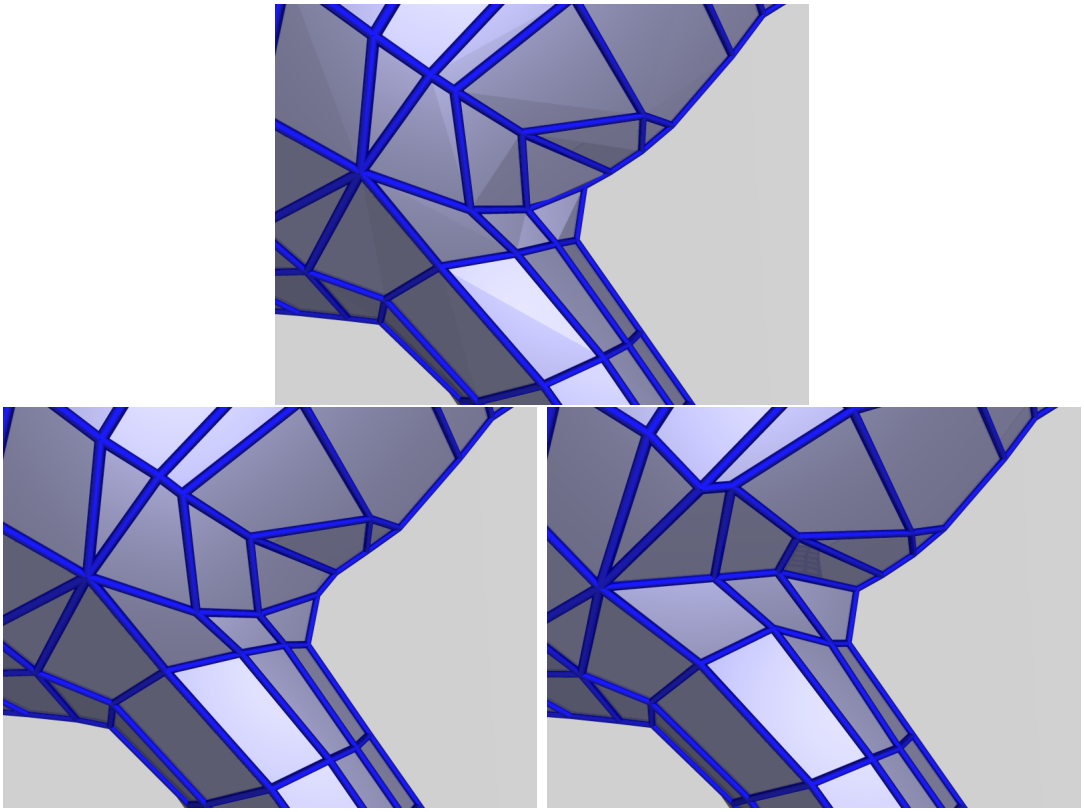


Figure 4.12: Quad mesh before and after planarization - detail

5 The Relative Sphere

5.1 Representation by a Triangular Mesh

A straightforward way to represent a relativesphere $\mathcal{R}$ is by means of a triangular closed orientable mesh. The requirement that $\mathcal{R}$ be strictly convex can be replaced by requiring that the surface normals $n_v, v \in V$ are unique. The curvature tensor field of $\mathcal{R}$ can be computed using the methods described in section 3.2. Computation of the relative curvature tensor field in a $p \in \mathcal{S}$ involves finding a vertex $v \in V$ that best matches $\boldsymbol{n}_p$, for a description of how to do this efficiently see section 6.1. In the second step the mapping $d\boldsymbol{\nu}_v$ has to be reconstructed from the curvature tensor field of $\mathcal{R}$. Finally it is applied to $d\boldsymbol{n}_p$ and the new eigenvectors and -values are computed, see also equation 2.13 and section 6.1.

Remark. The advantage of this representation of $\mathcal{R}$ is its simpleness, it can be implemented easily using the methods also needed for computing the curvature tensor field of $\mathcal{S}$. Its disadvantages are:

- The computational cost of computing the curvature tensor field as described in section 3.2 is high. In an implementation one would like to change the relativesphere interactively and see the effects on the relative curvature tensor fields within a short response time. This is not possible, because a recomputation of the curvature tensor field would be necessary in most cases. An exception to this are rotations of the relativesphere, as the curvature tensor field is invariant under these transformations.

- The curvature tensor field of $\mathcal{R}$ is available in the vertices only, thus the matching of normal vectors and calculation of the relative curvature tensor field is prone to inaccuracy and jitter. An optimum mesh representing $\mathcal{R}$ should be isotropic regarding the vertex normals n_v. A possibility how to obtain such a mesh will be explained in the following section.

5.2 Representation by a Support Function

The representation of a convex surface by its support function is a classical tool in convex geometry, see [4]. Support functions will proof to have some very useful properties for the representation of a relativesphere.

A convex surface $\mathcal{R} \subset \mathbb{R}^3$ can be described by the distance of its tangent planes to the origin of the coordinate system. Given an $n \in S^2$ and a $d \in \mathbb{R}$ the hyperplane $H(n, d) :=$

$\{v \in \mathbb{R}^3 | \langle v, n\rangle = d\}$ separates $\mathbb{R}^3$ into the half spaces $H_+(n,d) := \{v \in \mathbb{R}^3 | \langle v, n\rangle > d\}$ and $H_-(n,d) := \{v \in \mathbb{R}^3 | \langle v, n\rangle < d\}$. For a given convex surface its support function can now be defined by

$$h : S^2 \to \mathbb{R} : n \mapsto \inf\{d \in \mathbb{R} | H_+(n,d) \cap \mathcal{R} = \emptyset\} \tag{5.1}$$

Thus $H(n, h(n))$ intersects with $\mathcal{R}$ in at least one $p \in \mathcal{R}$, and the set $H(n, h(n)) \cap \mathcal{R}$ is called the *support set*. As $||n|| = 1$ it follows that $\text{dist}(0, H(n,d)) = d$. p is called a *support point* and $H(n, h(n))$ is called the *support plane* in p. It is clear that $H(n, h(n))$ is a tangent plane to $\mathcal{R}$.

In the following it is assumed that $0 \in \mathbb{R}^3$ is an interior point of the convex body enveloped by $\mathcal{R}$. In this case the support function can also be defined by

$$\tilde{h}(n) = \sup\{< n, p > | \ p \in \mathcal{R}\}.$$

This gives a natural extension of h from S^2 to $\mathcal{R}^3$. The mapping $\tilde{h} : \mathbb{R}^3 \mapsto \mathcal{R}$ is called *extended support function*. The following relationship holds for a support function and its extension:

$$\tilde{h}(n) = ||n|| h\left(\frac{n}{||n||}\right). \tag{5.2}$$

It follows directly from the definition that $\tilde{h}$ is a convex function:

$$\begin{aligned}\tilde{h}(\lambda x + (1-\lambda)y) &= \sup\{\langle \lambda x + (1-\lambda)y, p\rangle | p \in \mathcal{R}\} \\ &\leq \lambda \sup\{\langle x, p\rangle | p \in \mathcal{R}\} + (1-\lambda)\sup\{\langle y, p\rangle | p \in \mathcal{R}\} \\ &= \lambda\tilde{h}(x) + (1-\lambda)\tilde{h}(y)\end{aligned}$$

Furthermore $\tilde{h}$ is positively homogenous of degree 1, which means

$$\forall \lambda \geq 0 : \forall x \in \mathbb{R}^3 : \tilde{h}(\lambda x) = \lambda \tilde{h}(x). \tag{5.3}$$

If one starts from a convex, positively homogeneous[1] $\tilde{h}$, there is exactly one convex surface whose extended support function is $\tilde{h}$, for a proof see [4, section 17]. The convexity and homogeneity imply that $\tilde{h}$ is Lipschitz-continous and thus differentiable nearly everywhere.

Remark 5.1. The concept of *Minkowski addition* has a direct analogon regarding support functions. The Minkowski sum of two convex surfaces $\mathcal{R}^1, \mathcal{R}^2$ is defined as

$$\mathcal{R}^1 + \mathcal{R}^2 := \left\{a + b \ | \ a \in \mathcal{R}^1, b \in \mathcal{R}^2\right\}$$

Obviously $\mathcal{R}^1 + \mathcal{R}^2$ is a convex surface again. Similarly one defines

$$\lambda\mathcal{R} := \left\{\lambda a \ | \ a \in \mathcal{R}\right\}.$$

Let $h_{\mathcal{R}^1}, h_{\mathcal{R}^2}$ denote the corresponding support functions. It is easy to show (cf. [4]) that for $\lambda_1, \lambda_2 \geq 0$

$$h_{\lambda_1\mathcal{R}^1 + \lambda_2\mathcal{R}^2}(n) = \lambda_1 h_{\mathcal{R}^1}(n) + \lambda_2 h_{\mathcal{R}^2}(n)$$

is the support function of $\lambda_1\mathcal{R}^1 + \lambda_2\mathcal{R}^2$.

[1] In the following *positively homogeneous* will always refer to positive homogeneity of degree 1.

From here on we will concentrate on strictly convex surfaces $\mathcal{R}$, for more general results see [4] and [15]. The strict convexity implies that the support sets of all support planes consist of single points. For $p \in \mathcal{R}$ the support function can be computed by $\langle p, \boldsymbol{n}_p \rangle$. It can easily be shown that $\mathcal{R}$ can be fully reconstructed given h. Let $h : S^2 \to \mathbb{R}$ be the support function of $\mathcal{R}$. The strict convexity implies that $\mathcal{R}$ can be parameterized by its normal vectors, say $p = p(n) \in \mathcal{R}$. Thus one gets the following formula for computing the support function:

$$h(n) = \langle p(n), n \rangle. \tag{5.4}$$

The envelope $\boldsymbol{y}_h$ of a support function h is defined by

$$\boldsymbol{y}_h : S^2 \to \mathbb{R}^3 : n \mapsto h(n)n + \nabla_n^0 h. \tag{5.5}$$

Using 5.4, one can prove that the envelope gives exactly the parameterisation $p(n)$:

Proposition 5.2. *Let h be the support function of a strictly convex surface $\mathcal{R}$. The envelope $\boldsymbol{y}_h$ of h is a parameterisation of $\mathcal{R}$ such that $\boldsymbol{n}_{\boldsymbol{y}_h(n)} = n$.*

Proof. Notice that the two summands in (5.5) are orthogonal, because $\nabla_n^0 h \in T_n(S^2)$. Consider a $p \in \mathcal{R}$, let n be the normal vector of $\mathcal{R}$ at p, and decompose it into orthogonal components:

$$p = h(n)n + [p - h(n)n] = h(n)n + [p - \langle p, n \rangle n] = h(n)n + \pi_n(p) \tag{5.6}$$

where π_n denotes the orthogonal projection onto $T_n(S^2)$ (Notice that $T_n(S^2)$ and $T_p(\mathcal{R})$ are parallel). We will now calculate $\nabla_n^0 h$ using $h(n) = \langle p(n), n \rangle$ and a local parameterisation $n(x_1, x_2)$ of S^2:

$$\frac{\partial h}{\partial x_i}(n) = \langle dp_n(n_{,i}), n \rangle + \langle p(n), n_{,i} \rangle = \langle p(n), n_{,i} \rangle$$

From this it follows immediately that the differential of h at n is given by

$$dh_n : T_n(S^2) \to \mathbb{R} : v \mapsto \langle p(n), v \rangle = \langle \pi_n(p(n)), v \rangle. \tag{5.7}$$

Thus $\nabla_n^0 h = \pi_n(p)$ and a comparison of (5.5) and (5.6) completes the proof. □

The property that $\boldsymbol{y}_h$ parameterizes $\mathcal{R}$ by its normal vectors leads to

Corollary 5.3. *The Gaussian map and the Weingarten map of $\mathcal{R}$ are given by $\boldsymbol{y}_h^{-1} : \mathcal{R} \to S^2$ and $d\boldsymbol{y}_h^{-1}$ respectively.*

Remark 5.4. The envelope $\boldsymbol{y}_h$ can also be expressed in terms of the extended support function $\tilde{h}$. Differentiating the identity in 5.3 to λ one gets the so-called Euler relation $\langle \nabla \tilde{h}, x \rangle = \tilde{h}(x)$. Using this and equation 2.3 gives

$$\nabla_n^0 h = \pi_n(\nabla_n \tilde{h}) = \nabla_n \tilde{h} - \langle \nabla_n \tilde{h}, n \rangle n = \nabla_n \tilde{h} - h(n)n. \tag{5.8}$$

Consequently the envelope can also be defined by

$$\boldsymbol{y}_{\tilde{h}} : S^2 \to \mathbb{R}^3 : n \mapsto \nabla_n \tilde{h}. \tag{5.9}$$

Example 5.5. Consider the function $\tilde{h}(x) = \sqrt{a_1^2x_1^2 + a_2^2x_2^2 + a_3^2x_3^2}$ with $a_i > 0$. Obviously this function is convex and positively homogeneous and therefore it is the support function of a strictly convex surface. According to remark 5.4 the envelope of this support function has the coordinates $y_i = a_i^2x_i/\sqrt{a_1^2x_1^2 + a_2^2x_2^2 + a_3^2x_3^2}$. From this follows $\sum_{i=1}^3 y_i^2/a_i^2 = 1$, which is the defining equation of an ellipsoid. More generally an analogous calculation shows that $\tilde{h}(x) = \sqrt[l/(l-1)]{|a_1x_1|^{l/(l-1)} + |a_2x_2|^{l/(l-1)} + |a_3x_3|^{l/(l-1)}}$ is the support function of the superellipsoid $\sum_{i=1}^3 |y_i/a_i|^l = 1$.

The support function of the sphere with radius r and center 0 is given by the constant function r.

It is a direct consequence of 5.4, that the support function of a point $p \in \mathbb{R}^3$ is given by $\langle p, n\rangle$. Together with remark 5.1 this implies that the support function of $\mathcal{R}$ translated by p is given by $h_{\mathcal{R}} + \langle p, n\rangle$.

In the following we would like to choose a relativesphere such that it fulfills certain conditions. This can be done by parameterizing h. However care has to be taken that the resulting h is still a support function of a strictly convex surface. If we define h on S^2, this involves checking for the convexity of $\tilde{h}$, which is not a straightforward task. In [15] the author proves a proposition on the perturbation of support functions. The following corollary is a direct consequence of it, which will be referred to in the following.

Corollary 5.6. *If h is a twice differentiable function on S^{d-1} such that for all $p \in S^{d-1}$ and all $i, j \in \{1, \dots, d\}$*

$$2d^3|D_0^{ij}h(p)| + \sqrt{2d^3}|D_0^ih(p)| + |h(p)| < r_o$$

then $r_0 + h$ is the support function of a convex surface in $\mathbb{R}^d$.
$D_0^ih(p)$ and $D_0^{ij}h(p)$ are defined as

$$D_0^ih(p) := \left(\frac{\partial h(x/|x|)}{\partial x_i}\right)_{x=p}, \quad D_0^{ij}h(p) := \left(\frac{\partial^2 h(x/|x|)}{\partial x_i \partial x_j}\right)_{x=p}$$

Proof. See [15, Proposition 2.2.4] □

Up to now, no parameterisation of S^2 was considered, which we will need for the evaluation of the envelope and its differential. We choose the following parameterisation:

$$n : [0, \pi] \times [0, 2\pi) \to S^2 \subset \mathbb{R}^3 : (\theta, \phi) \mapsto (\sin\theta\sin\phi, \sin\theta\cos\phi, \cos\theta) \tag{5.10}$$

The associated basis to this parameterisation is given by

$$\begin{aligned} n_\theta &= (\cos\theta\sin\phi, \cos\theta\cos\phi, -\sin\theta) \quad , \quad ||n_\theta|| = 1 \\ n_\phi &= (\sin\theta\cos\phi, -\sin\theta\sin\phi, 0) \quad , \quad ||n_\phi|| = \sin\theta \end{aligned} \tag{5.11}$$

In the following we derive expressions for $\boldsymbol{y}_h$ and $d\boldsymbol{y}_h$ in this parameterisation. First of all we compute the coefficients of $\nabla_n^0 h$ in the basis n_θ, n_ϕ. Using the parameterisation

(5.10) we can write $h = h(\theta, \phi)$. According to (2.2) the coordinate vector of $\nabla_n^0 h$ reads $(h_\theta/||n_\theta||^2, h_\phi/||n_\phi||^2) = (h_\theta, h_\phi/\sin^2\theta)$. Thus we get

$$\boldsymbol{y}_h(\theta,\phi) = h(\theta,\phi)n(\theta,\phi) + h_\theta(\theta,\phi)n_\theta(\theta,\phi) + \frac{h_\phi(\theta,\phi)}{\sin^2\theta}n_\phi(\theta,\phi) \tag{5.12}$$

for the envelope, see also [18]. This expression, and the fact that the tangent spaces of S^2 and $\mathcal{R}$ at given values (θ, ϕ) are parallel, gives an easy way of calculating the coefficients of $d\boldsymbol{y}_h$ in the basis (n_θ, n_ϕ) by simply computing $\frac{\partial}{\partial\theta}\boldsymbol{y}_h$ and $\frac{\partial}{\partial\phi}\boldsymbol{y}_h$ and projecting these onto the tangent space. One gets

$$\begin{aligned} d\boldsymbol{y}_h(n_\theta)\Big|_{(\theta,\phi)} &= (h + h_{\theta\theta})n_\theta + (-h_\phi\frac{\cos\theta}{\sin^3\theta} + \frac{h_{\theta\phi}}{\sin^2\theta})n_\phi \\ d\boldsymbol{y}_h(n_\phi)\Big|_{(\theta,\phi)} &= (-\frac{h_\phi\cos\theta}{\sin\theta} + h_{\theta\phi})n_\theta + (h + \frac{h_\theta\cos\theta}{\sin\theta} + \frac{h_{\phi\phi}}{\sin^2\theta})n_\phi \end{aligned} \tag{5.13}$$

using the identities

$$\begin{array}{ll} \langle n, n_\theta\rangle = 0 & \langle n, n_\phi\rangle = 0 \\ \langle n_{\theta\theta}, n_\theta\rangle = 0 & \langle n_{\theta\theta}, n_\phi\rangle = 0 \\ \langle n_{\theta\phi}, n_\theta\rangle = 0 & \langle n_{\theta\phi}, n_\phi\rangle = \sin\theta\cos\theta \\ \langle n_{\phi\phi}, n_\theta\rangle = -\sin\theta\cos\theta & \langle n_{\phi\phi}, n_\phi\rangle = 0 \end{array}$$

which follow directly from 5.11.

Remark. The formulas 5.12 and 5.13 suggest singularities at the poles for $\theta \in \{0, \pi\}$. These are no singularities but cancel with $||n_\phi||$ and inner derivatives of $h(\theta, \phi)$. However, numerical problems can arise, because 5.12 and 5.13 are evaluated in the software as shown. Therefore an alternative coordinate system was used for evaluation at the poles, see section 6.3.

Remark 5.7. Assume that $h(\theta, \phi)$ and its derivatives up to second order can be computed by given expressions. Formulas 5.12 and 5.13 show that for a given normal vector n it is easily possible to evaluate the corresponding point on $\mathcal{R}$ as well as the inverse Weingarten map, see also corollary 5.3. This is exactly what is needed for the computation of the relative curvature tensor in a point $p \in \mathcal{S}$ with $\boldsymbol{n}_p = n$. The cost[2] needed for the curvature tensor update in p is bounded by a constant, depending on the cost of the evaluation of h and its derivatives. Note that it is not necessary to evaluate the trigonometric functions. Given $\boldsymbol{n}_p = (x, y, z)$ they can easily be computed using 5.10:

$$\cos\theta = z, \sin\theta = \sqrt{1 - z^2}, \sin\phi = x/\sin\theta, \cos\phi = y/\sin\theta \tag{5.14}$$

Again, special care has to be taken near the poles where $\sin\theta \approx 0$, see section 6.3.

[2]Cost meaning amount of arithmetic operations.

Remark. The support function representation can also be used to generate a mesh of a convex surface $\mathcal{R}$ whose vertex normals are evenly spread[3]. Given such a mesh on S^2, simply map it to $\mathcal{R}$ using $\boldsymbol{y}_h$.

We have seen above, that the cost of the curvature tensor update strongly depends on the cost of the evaluation of h and its derivatives up to second order. A representation that allows approximation of a big class of support functions up to arbitrary accuracy, and still can be evaluated cheaply, will be described in the following section.

5.2.1 Representation of Support Functions by Spherical Harmonics

Consider Laplace's equation in $\mathbb{R}^3$. The spherical harmonics are the angular portion of an orthogonal set of solutions, if Laplace's equation is represented in spherical coordinates (cf. [37]). They are defined on S^2 and can thus be represented using the parameterisation 5.10.

Definition 5.8. The real spherical harmonics for degree $l \geq 0$ are defined as

$$Y_l^m(\theta,\phi) := \begin{cases} C_l^m P_l^m(\cos\theta)\cos(m\phi) & 0 \leq m \leq l \\ C_l^{|m|} P_l^{|m|}(\cos\theta)\sin(m\phi) & -l \leq m < 0 \end{cases} \tag{5.15}$$

where $(\theta,\phi) \in [0,\pi] \times [0,2\pi)$.

Here P_l^m and C_l^m denote the so called associated Legendre functions[4] and normalization constants respectively, which are defined by

$$P_l^m(x) := \frac{1}{2^l l!}(1-x^2)^{m/2}\frac{\partial^{l+m}}{\partial x^{l+m}}[(x^2-1)^l] \tag{5.16}$$

$$C_l^m := \begin{cases} \sqrt{\frac{(2l+1)(l-m)!}{4\pi(l+m)!}} & m = 0 \\ \\ \sqrt{\frac{(2l+1)(l-m)!}{2\pi(l+m)!}} & m > 0 \end{cases} \tag{5.17}$$

l and m are called degree and order of Y_l^m respectively.

Consider the Hilbert-space $\mathcal{H}(S^2,\mathbb{R})$ of real-valued square-integrable functions defined on the unit sphere, with the canonical inner product

$$< f, g >:= \int_{S^2} fg \; dF.$$

It can be shown that the family $(Y_l^m | -l \leq m \leq l)_{l\in\mathbb{N}}$ is a complete orthonormal set of $\mathcal{H}(S^2,\mathbb{R})$, see [15, Thm. 3.2.10]. Thus any square integrable support function $h \in \mathcal{H}(S^2,\mathbb{R})$ can be expressed as its expansion into spherical harmonics:

[3] These meshes are ideal for the representation of a relative sphere.
[4] Cf. [36].

$$h_l^m := \; < h, Y_l^m > \; = \; \int_{\theta=0}^{\pi} \int_{\phi=-\pi}^{\pi} h(\theta,\phi) Y_l^m(\theta,\phi) \sin(\theta) \; d\phi \; d\theta \tag{5.18}$$

$$h(\theta,\phi) \; = \; \sum_{l=0}^{\infty} \sum_{m=-l}^{m=l} h_l^m Y_l^m(\theta,\phi) \tag{5.19}$$

This implies a natural way of approximating h by limiting the expansion to a finite degree $l_{max} \in \mathbb{N}$:

$$h(\theta,\phi) \; \approx \; h_{l_{max}}(\theta,\phi) := \sum_{l=0}^{l_{max}} \sum_{m=-l}^{m=l} h_l^m Y_l^m(\theta,\phi) \tag{5.20}$$

Remark 5.9. If h is the support function of a convex surface $\mathcal{R}$ then it is not clear whether $h_{l_{max}}$ is the support function of a convex surface too. However it can be proven (see [15, Thm. 5.1.4]) that their is a $l_0 \in \mathbb{N}$, which depends on $\mathcal{R}$, such that this is the case for all $l_{max} > l_0$. The proof of this theorem is based on corollary 5.6.

Remark 5.10. A simple calculation shows that the number of spherical harmonics with degree $\leq l_{max}$ is given by

$$k(l_{max}) := l_{max}(l_{max} + 2) + 1.$$

Example 5.11. The real spherical harmonics of positive order up to degree 2 are given by:

Y_l^m			
$l \backslash m$	0	1	2
0	$\sqrt{\frac{1}{4\pi}}$		
1	$\sqrt{\frac{3}{4\pi}} \cos\theta$	$\sqrt{\frac{3}{2\pi}} \sin\theta \cos\phi$	
2	$\sqrt{\frac{5}{8\pi}}(3\cos^2\theta - 1)$	$\sqrt{\frac{15}{4\pi}} \cos\theta \sin\theta \cos\phi$	$\sqrt{\frac{15}{16\pi}} \sin^2\theta \cos 2\phi$

For $Y_l^{-m}, m > 0$, replace $\cos m\phi$ by $\sin m\phi$ in Y_l^m. This shows that the Y_1^m are multiples of the coordinates of the parameterisation 5.10. As a result the contribution of the spherical harmonics of degree 1 to h is of the form $\langle p, n \rangle$ with $p = (\sqrt{\frac{3}{2\pi}} h_1^{-1}, \sqrt{\frac{3}{2\pi}} h_1^1, \sqrt{\frac{3}{4\pi}} h_1^0)$. According to example 5.5 this corresponds to a translation, which is not of interest for the representation of the relativesphere. Consequently the coefficients of degree 1 will be assumed being equal to 0 in the following.

5.2.2 Evaluation of Spherical Harmonics

If one wants to numerically evaluate the envelope $\boldsymbol{y}_h$ of a support function defined via a finite spherical harmonic expansion, it is necessary to evaluate value and first order

derivatives of the spherical harmonics. Computation of the curvature tensor enforces us to calculate the second order derivatives of the spherical harmonics too.

Considering 5.15, the core problem is to compute the associated Legendre functions $P_l^m(\cos\theta)$ and its derivatives

$$\begin{aligned} \frac{\partial}{\partial\theta}P_l^m(\cos\theta) &= -P_l^{m'}(\cos\theta)\sin\theta \qquad \text{and} && (5.21)\\ \frac{\partial^2}{\partial\theta^2}P_l^m(\cos\theta) &= P_l^{m''}(\cos\theta)\sin^2\theta - P_l^{m'}(\cos\theta)\cos\theta. && (5.22) \end{aligned}$$

The P_l^m fulfil several recurrence relations, an extensive listing can be found in [1]. These are commonly used for evaluation, see [32]. However, in common literature, recurrence relations are given for the associated Legendre functions and their derivatives only. The following recurrence relation in the degree l holds for the derivatives of the P_l^m as defined above.

$$\begin{aligned} \frac{\partial}{\partial x}P_l^m(x) &= \frac{lxP_l^m(x) - (l+m)P_{l-1}^m(x)}{x^2-1} && (5.23)\\ \frac{\partial}{\partial x}P_m^m(x) &= \frac{\partial}{\partial x}\left(\frac{1}{2^m m!}(1-x^2)^{m/2}\frac{\partial^{2m}}{\partial x^{2m}}[(x^2-1)^m]\right)\\ &= \frac{\partial}{\partial x}\left((1-x^2)^{m/2}(2m-1)!!\right)\\ &= -mx(1-x^2)^{m/2-1}(2m-1)!! && (5.24) \end{aligned}$$

From this recurrence relation, one can derive another one for the second derivatives by differentiation.

$$\begin{aligned} (1-x^2)\frac{\partial^2}{\partial x^2}P_l^m(x) &= -lP_l^m(x) + (2-l)x\frac{\partial}{\partial x}P_l^m(x) + (l+m)\frac{\partial}{\partial x}P_{l-1}^m(x) && (5.25)\\ (1-x^2)\frac{\partial^2}{\partial x^2}P_m^m(x) &= m(2m-1)!!(1-x^2)^{m/2-1}\left[(m-1)x^2-1\right] && (5.26) \end{aligned}$$

The term $(1-x^2)$ is kept on the left hand side, because it will be equal to $\sin^2\theta$ if evaluated at $\cos\theta$ as needed in equation 5.22. Using equations 5.21 to 5.26, the derivatives up to second order can easily be evaluated:

$$\frac{\partial}{\partial\theta}Y_l^m(\theta,\phi) = \begin{cases} -C_l^m P_l^{m'}(\cos\theta)\sin\theta\cos(m\phi) & 0 \le m \le l \\ -C_l^{|m|} P_l^{|m|'}(\cos\theta)\sin\theta\sin(m\phi) & -l \le m < 0 \end{cases} \tag{5.27}$$

$$\frac{\partial}{\partial\phi}Y_l^m(\theta,\phi) = \begin{cases} -mC_l^m P_l^m(\cos\theta)\sin(m\phi) & 0 \le m \le l \\ mC_l^{|m|} P_l^{|m|}(\cos\theta)\cos(m\phi) & -l \le m < 0 \end{cases} \tag{5.28}$$

$$\frac{\partial^2}{\partial\theta^2}Y_l^m(\theta,\phi) = \begin{cases} C_l^m \left(P_l^{m''}(\cos\theta)\sin^2\theta - P_l^{m'}(\cos\theta)\cos\theta\right)\cos(m\phi) & 0 \le m \le l \\ C_l^{|m|} \left(P_l^{|m|''}(\cos\theta)\sin^2\theta - P_l^{|m|'}(\cos\theta)\cos\theta\right)\sin(m\phi) & -l \le m < 0 \end{cases} \tag{5.29}$$

$$\frac{\partial^2}{\partial\theta\partial\phi}Y_l^m(\theta,\phi) = \begin{cases} mC_l^m P_l^{m'}(\cos\theta)\sin\theta\sin(m\phi) & 0 \le m \le l \\ -mC_l^{|m|} P_l^{|m|'}(\cos\theta)\sin\theta\cos(m\phi) & -l \le m < 0 \end{cases} \tag{5.30}$$

$$\frac{\partial^2}{\partial\phi^2}Y_l^m(\theta,\phi) = \begin{cases} -m^2 C_l^m P_l^m(\cos\theta)\cos(m\phi) & 0 \le m \le l \\ -m^2 C_l^{|m|} P_l^{|m|}(\cos\theta)\sin(m\phi) & -l \le m < 0 \end{cases} \tag{5.31}$$

5.3 Prescribing Curvature

In chapter 4 we have seen that umbilic points and areas are critical for the generation of a PQ mesh. Consequently one would like to have some sort of control of umbilic points and remove umbilic areas. In section 4.2.2 and figure 4.7 an example has been given, showing how an appropriate choice of the relativesphere can lead to the transformation of an umbilic area into a single relative umbilic point. In the following we will consider the placement of umbilic points. Umbilic points are clearly recognizable when viewing curvature lines or the resulting quad mesh, which means it is desirable to have a possibility to choose their location. An arbitrary choice is not possible, as will become clear in the following.
The defining property of a relative umbilic point is, that there exists a $\lambda \in \mathbb{R}$ such that

$$d\boldsymbol{\nu}_{\boldsymbol{n}_p}^{-1} \circ d\boldsymbol{n}_p = \lambda I. \tag{5.32}$$

In flat points $p \in \mathcal{S}$ this equation is fulfilled for any choice of $\mathcal{R}$ with $\lambda = 0$, because $d\boldsymbol{n}_p = 0$. Note that the relativesphere $\mathcal{R}$ is elliptic everywhere by definition, which means the eigenvalues of $d\boldsymbol{\nu}^{-1}$ are of the same sign. Consequently 5.32 can be fulfilled for elliptic or flat points $p \in \mathcal{S}$ only.

If one wants to choose a single elliptic $p \in \mathcal{S}$ as relative umbilic point, $\mathcal{R}$ has to be chosen such that 5.32 holds. This can be achieved by an ellipsoid, consider example 5.5. The principal curvatures k_1, k_2 in the point $q : y_1 = a_1, y_2 = y_3 = 0$ are equal to the curvature of the intersection curves of $\mathcal{R}$ and the two symmetry planes passing through q. It is well known that $k_1 = a_1/a_2^2, k_2 = a_1/a_3^2$. The principal curvatures of $\mathcal{R}$ in q can thus be chosen equal to the principal curvatures of $\mathcal{S}$ in q. Afterwards two rotations are applied to the ellipsoid. The first rotation is chosen such that the normal vectors $\boldsymbol{n}_p$

and $\boldsymbol{n}_q^{\mathcal{R}}$ coincide. In a second step the ellipsoid is rotated around the axis defined by $\boldsymbol{n}_p$ in order to align the corresponding principal directions.

Consider a family $P := (p_1, \ldots, p_n)$ of points in $\mathcal{S}$ that one would like to be relative umbilic points. Note that it is not clear at all whether a strictly convex relativesphere exists such that 5.32 holds for all $p \in P$. In the following a possible approach to this problem is outlined, using the representation by spherical harmonics.

In the rest of this section $h(\theta, \phi)$ denotes a support function defined according to 5.20 up to degree l_{max}. For each $p \in P$ equation 5.32 can be expressed in terms of a matrix equation. We know from corollary 5.3 that

$$d\boldsymbol{\nu}_{\boldsymbol{n}_p}^{-1} = d\boldsymbol{y}_h\big|_{\boldsymbol{n}_p}.$$

Consequently its coordinate matrix A_p in the basis (n_θ, n_ϕ) can be computed using equation 5.13. As h is given by a linear combination of spherical harmonics Y_l^m, A_p can be written as a linear combination of matrices $A_p^{l,m}$ as well: $A_p = \sum_{l,m} h_l^m A_p^{l,m}$ where $A_p^{l,m}$ denotes the coordinate matrix of $d\boldsymbol{y}_{Y_l^m}$. Furthermore we define B_p to be the coordinate matrix of $d\boldsymbol{n}_p$ in the same basis. Using this notation one arrives at the following formulation:

$$\text{Find coefficients } h_l^m \text{ such that } \forall p \in P : \sum_{l,m} h_l^m A_p^{l,m} = B_p \tag{5.33}$$

Note that this is a linear equation system, thus the λ_p could be dropped. In order to write this linear equation system in matrix-vector notation, one introduces a lexicographical ordering of the coefficients h_l^m:

$$\boldsymbol{h} := (h_0^0, h_2^{-2}, h_2^{-1}, h_2^0, h_2^1, h_2^2, h_3^{-3}, h_3^{-2}, \ldots, h_{l_{max}}^{l_{max}})^T \in \mathbb{R}^{\tilde{k}(l_{max})}$$

where $\tilde{k}(l_{max}) := l_{max}(l_{max}+2) - 2$. Using this ordering we define

$$\boldsymbol{A} := \begin{pmatrix} (A_{p_1}^{0,0})_{(1,1)} & (A_{p_1}^{2,-2})_{(1,1)} & (A_{p_1}^{2,-1})_{(1,1)} & \cdots \\ (A_{p_1}^{0,0})_{(1,2)} & (A_{p_1}^{2,-2})_{(1,2)} & (A_{p_1}^{2,-1})_{(1,2)} & \cdots \\ (A_{p_1}^{0,0})_{(2,1)} & (A_{p_1}^{2,-2})_{(2,1)} & (A_{p_1}^{2,-1})_{(2,1)} & \cdots \\ (A_{p_1}^{0,0})_{(2,2)} & (A_{p_1}^{2,-2})_{(2,2)} & (A_{p_1}^{2,-1})_{(2,2)} & \cdots \\ (A_{p_2}^{0,0})_{(1,1)} & (A_{p_2}^{2,-2})_{(1,1)} & (A_{p_2}^{2,-1})_{(1,1)} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix} \in \mathbb{R}^{4n \times \tilde{k}(l_{max})}$$

$$\boldsymbol{b} := \big((B_{p_1})_{(1,1)}, (B_{p_1})_{(1,2)}, (B_{p_1})_{(2,1)}, (B_{p_1})_{(2,2)}, (B_{p_2})_{(1,1)}, \ldots\big)^T \in \mathbb{R}^{4n}$$

which allows to replace 5.33 by $\boldsymbol{A}\boldsymbol{h} = \boldsymbol{b}$. As was mentioned before, the existence of a solution depends on the choice of P. If a solution exists, it might not correspond to the support function of a strictly convex surface. In the following it will be assumed that a solution $\boldsymbol{h}$ exists.

Remark 5.12. Further investigation should be done in clarifying the existence of a convex surface that fulfills 5.32 for all $p \in P$, which seems to be a non-trivial task.

As no result on that could be obtained, the following rudimentary approach was chosen. It is easy to see that for each $p \in P$ condition 5.32 fixes three degrees of freedom. Therefore l_{max} should be chosen such that $\tilde{k}(l_{max}) \geq 3n$. Note that l_{max} in general can not be chosen such that $\tilde{k}(l_{max}) = 3n$. Consequently, depending on the choice of l_{max} the kernel of the matrix $\boldsymbol{A}$ will change. This additional degrees of freedom have proven useful in practice, because they might allow to choose $\boldsymbol{h}$ such that the resulting surface is strictly convex. In order to compute a solution $\boldsymbol{h}$, the singular value decomposition of $\boldsymbol{A}$ can be used. If a solution exists, the set of all solutions is given by $\boldsymbol{h} + ker(\boldsymbol{A})$, where $ker(\boldsymbol{A})$ denotes the kernel. A basis of $ker(\boldsymbol{A})$ can be computed using again its singular value decomposition. Corollary 5.6 suggests that a solution should be chosen from $\boldsymbol{h} + ker(\boldsymbol{A})$ such that $h_0^0 \gg h_l^m, l > 0$.

Remark. The approach described above should be considered as a method of trial and error, as one does not know in advance whether a strictly convex relativesphere exists that fulfills the desired criteria.

Example 5.13. Figure 5.1 shows an example where the location of two umbilic points was successfully prescribed using the method described above. The maximum degree l_{max} was set to three in this example, resulting in $\tilde{k}(l_{max}) = 13$. Consequently 7 degrees of freedom where left, which allowed to choose a strictly convex relativesphere. The upper plot shows classical principal curvature lines and umbilic points, whereas the lower plot displays the resulting relative principal curvature lines. The prescribed umbilic points are marked by green points. This example shows that the prescription of umbilic points has a global influence on the relative curvature tensor field. In this case it resulted in the removal of the central umbilic area, that can be seen in the upper plot.

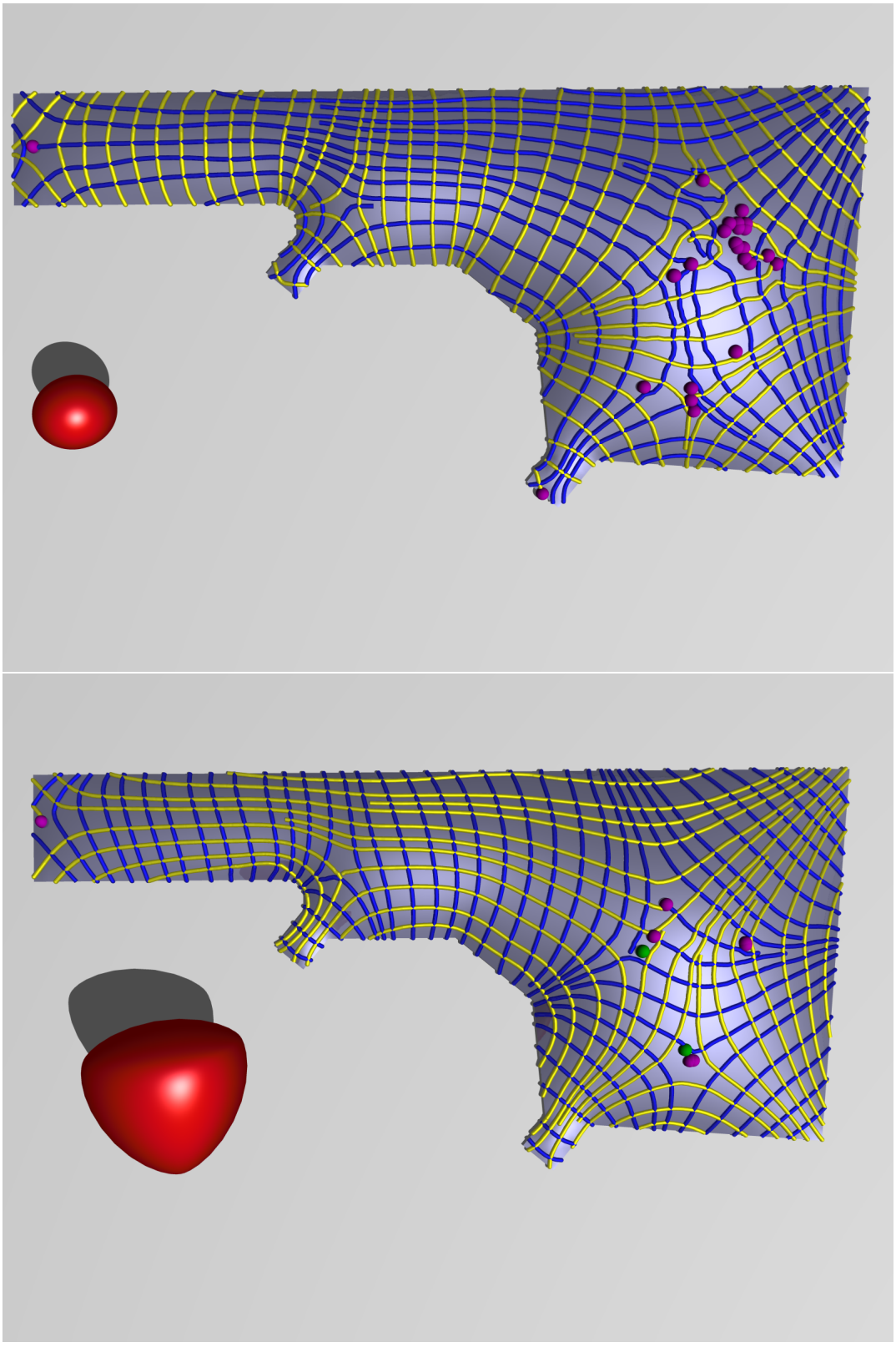


Figure 5.1: Example for the prescription of umbilic points

6 Implementation

6.1 Computation of Relative Curvature Tensor Field

According to 2.13, the relative curvature tensor field of $\mathcal{S}$ with respect to $\mathcal{R}$ can be computed using the curvature tensor fields of $\mathcal{S}$ and $\mathcal{R}$. In practice $\mathcal{S}$ is given by a triangulation. Its curvature tensor field is computed using the method described in section 3.2 in discrete points of $\mathcal{S}$, usually its vertices. The computation of the curvature tensor field of $\mathcal{R}$ depends on the type of representation of $\mathcal{R}$.

In the accompanying software the curvature tensor field is represented as a data structure attached to the vertices. This consists of normed coordinate vectors $b_1, b_2 \in \mathbb{R}^3$ corresponding to the eigenvectors $\boldsymbol{e}_1, \boldsymbol{e}_2$ and the principal curvatures k_1, k_2. The following procedure needs to be done for the computation of the relative curvature tensor in a vertex v:

1. Find a $q \in \mathcal{R}$ with matching normal vector $\boldsymbol{\nu}_q$ to n_v.

2. Set up the coordinate matrix $\boldsymbol{B}$ of $d\boldsymbol{n}_v$ with respect to the basis $\boldsymbol{e}_1, \boldsymbol{e}_2$, which is clearly given by
$$\boldsymbol{B} = \begin{pmatrix} -k_1 & 0 \\ 0 & -k_2 \end{pmatrix}.$$

3. Set up the coordinate matrix $\boldsymbol{C}$ of $(d\boldsymbol{\nu}_q)^{-1}$ with respect to the same basis. Let $c_1, c_2 \in \mathbb{R}^3$ denote the normed coordinate vectors corresponding to the eigenvectors of $d\boldsymbol{\nu}_q$ and l_1, l_2 the matching principal curvatures. Define the transformation matrix $\boldsymbol{D} = (d_{ij}) \in \mathbb{R}^{2\times 2}$ by $d_{ij} = b_i^T c_j$. Then $\boldsymbol{C}$ can be calculated by
$$\boldsymbol{C} = \boldsymbol{D} \begin{pmatrix} -l_1^{-1} & 0 \\ 0 & -l_2^{-1} \end{pmatrix} \boldsymbol{D}^{-1} \tag{6.1}$$

4. Compute the eigenvectors and -values of the composition $\boldsymbol{CB}$. The negative eigenvalues are the relative principal curvatures $k_1^{\mathcal{R}}, k_2^{\mathcal{R}}$, whereas the eigenvectors are coordinate vectors of $\boldsymbol{e}_1^{\mathcal{R}}, \boldsymbol{e}_2^{\mathcal{R}}$ in the basis $\boldsymbol{e}_1, \boldsymbol{e}_2$.

Interactivity

In practice one would like to parameterize the relativesphere interactively and immediately see the results in a visualization of the relative curvature tensor field. Consequently the computational effort for the steps described above needs to be kept as low as possible, because these steps have to be done for all vertices of the given mesh representing

$\mathcal{S}$. Therefore the curvature tensor field of $\mathcal{S}$ is precomputed. The following sections describe the specialties of the different representations of $\mathcal{R}$ regarding the interactivity.

Relativesphere represented by a Triangular Mesh

In this case the curvature tensor field of $\mathcal{R}$ has to be precomputed too. This gives a strong limitation on the interactive parameterisation of $\mathcal{R}$, because the curvature tensor field should stay invariant under the transformations involved, which is true only for affine transformations consisting of a rotation and a translation. Translations of the relativesphere will not change its Gaussian map, thus rotations of $\mathcal{R}$ are left in this case for the parameterisation.

The computational effort for steps 2 to 4 is not significant, it involves a fixed number of operations and thus is of order $\mathcal{O}(|V|)$. More care has to be taken regarding step 1. Let the set of vertices of $\mathcal{R}$ be denoted by $V^{\mathcal{R}}$. For a given $v \in V$ one wants to find a vertex $w \in V^{\mathcal{R}}$ such that

$$\langle n_w, n_v \rangle = \max_{w' \in V^{\mathcal{R}}} \langle n_{w'}, n_v \rangle \tag{6.2}$$

Note that this vertex is not necessarily a unique maximizer, although the existence of two maximizers is very unlikely in practice. A naive approach for finding w would be to compute this inner product for all $w \in V^{\mathcal{R}}$, resulting in an overall number of operations of order $\mathcal{O}(|V||V^{\mathcal{R}}|)$. The following algorithms drastically reduce this effort by making use of the connectivity information of the mesh representing $\mathcal{R}$. The notation $N^v(v)$ will be used to denote the 1-ring of v.

Algorithm 6.1 (Matching of a single vertex)**.** *For $v \in V$ determine $w \in V^{\mathcal{R}}$ as in 6.2 by :*

1. *Choose a starting vertex $w_0 \in V^{\mathcal{R}}$, set $ipmax := \langle n_{w_0}, n_v \rangle$, $vList := \{\}$ and $i := 0$*
2. *Loop over*
   a) *Find u such that*
   $$\langle n_u, n_v \rangle = \max_{u' \in N^v(w_i) \setminus vList} \langle n_{u'}, n_v \rangle$$
   b) *If $ipmax \geq \langle n_u, n_v \rangle$ then break*
   c) *else set $ipmax := \langle n_u, n_v \rangle$, $w_{i+1} := u$, $vList := vList \cup N^v(w_i)$*
   d) *Set $i := i + 1$*
3. *Set $w := w_i$*

This algorithm locally makes the best possible choice, as $\mathcal{R}$ is convex it will stop at a maximizer of 6.2 regardless of the starting vertex w_0. Note that the amount of vertices to be visited depends strongly on the choice of w_0. Up to now we have made use of the connectivity of $\mathcal{R}$ only. This gives rise to an algorithm which does the matching for all $v \in V$.

Algorithm 6.2 (Matching of all vertices)**.** *For all $v \in V$ determine $w_v \in V^{\mathcal{R}}$ as in 6.2 by :*

1. *Choose a starting vertex $v \in V$*
2. *Compute w_v according to algorithm 6.1*
3. *Set $vQueue := (N^v(v) \times \{w_v\})$ and $vList := \{v\} \cup N^v(v)$*
4. *Loop over*
   a) *If vQueue is empty then break*
   b) *Pop first element (v, w_0) from vQueue*
   c) *Compute w_v according to algorithm 6.1 using starting vertex w_0*
   d) *Set $vQueue := (vQueue, (N^v(v) \setminus vList) \times \{w_v\})$*
   e) *Set $vList := vList \cup N^v(v))$*

Remark. The main idea behind this algorithm is to provide good starting vertices for algorithm 6.1. If the matching vertex $w_v \in V^{\mathcal{R}}$ has been computed for a vertex $v \in V$, it is reasonable to choose w_v as starting vertex for the neighbouring vertices to v, which is represented in the ordered pairs $(v, w_0) \in vQueue$. This algorithm visits all vertices in rings[1] around the starting vertex, provided that the mesh is connected.

Relativesphere represented by a Support Function given by Spherical Harmonics

In this case step 1 is trivial because according to proposition 5.2 the envelope of the support function is a parameterization by normal vectors. Step 3 is done according to remark 5.7. The corresponding coordinate matrix $\boldsymbol{C}$ can thus be calculated by:

$$\begin{aligned}
\boldsymbol{Y} &:= \begin{pmatrix} (h + h_{\theta\theta}) & (-h_\phi \frac{\cos\theta}{\sin^3\theta} + \frac{h_{\theta\phi}}{\sin^2\theta}) \\ (-\frac{h_\phi \cos\theta}{\sin\theta} + h_{\theta\phi}) & (h + \frac{h_\theta \cos\theta}{\sin\theta} + \frac{h_{\phi\phi}}{\sin^2\theta}) \end{pmatrix} \\
\boldsymbol{D} &:= \begin{pmatrix} b_1^T n_\theta & b_1^T n_\phi \\ b_2^T n_\theta & b_2^T n_\phi \end{pmatrix} \\
\boldsymbol{C} &:= \boldsymbol{D}\boldsymbol{Y}\boldsymbol{D}^{-1}
\end{aligned}$$

It can be expected that the cost of the evaluation of $\boldsymbol{Y}$ depends quadratically on the maximum degree l_{max} of the spherical harmonics used in the representation of the support function, see remark 5.10. Consequently this representation can be used for a fully interactive parameterisation of the relativesphere, as long as l_{max} is kept low enough to not disturb the interactivity.

[1] In the sense of connectivity by edges.

Computation Times

The following table 6.1 shows computation times needed for updating the relative curvature tensor field of a mesh with 4845 vertices and 9408 faces for both types of representations of $\mathcal{R}$. The experiments were done on a workstation using a 1.8GHz processor. The figures for the representation by meshes clearly show, that the matching of normal vectors scales better than linearly with the number of vertices of $\mathcal{R}$. In remark 5.10 we have seen that the number of spherical harmonics grows quadratically depending on l_{max}. Consequently a quadratic dependency of the computation time on l_{max} could be expected. In fact it scales better than quadratic, which can be explained by the use of recurrence relations for the evaluation of the spherical harmonics. Table 6.1 might suggest that the mesh representation performs better. This has to be put into perspective with the need of precomputation of the curvature tensor field, which is necessary for the relativesphere in this case.

Mesh representation	
Number of Vertices of $\mathcal{R}$	Time [ms]
2039	291
4487	389
16004	651

Support function representation	
Maximum Degree l_{max}	Time [ms]
5	274
10	464
20	1109
30	2123
40	3610
50	5377

Table 6.1: Computation times needed for updating the relative curvature tensor field

6.2 Approximation of Support Functions using Spherical Harmonics

As was mentioned in section 5.2.1, the representation by spherical harmonics can be used to approximate a given support function h up to a predefined degree l_{max}. Remember that there is $l_0 \in \mathbb{N}$ depending on h such that the approximation of degree $l_{max} > l_0$ is again the support function of a convex surface, see remark 5.9. This is a theoretical result and there is no known way to compute l_0. Still, the following method of trial and error can be used to choose a suitable l_{max}. Step 4 in the update procedure described in section 6.1 can be used as indicator whether l_{max} was chosen high enough. If the envelope of the approximating support function is not convex, it is not guaranteed that the relative Weingarten map has real eigenvalues. Therefore, by encountering a $v \in V$ such that the eigenvalues are not real, one knows that l_{max} should be increased. Figure 6.1 shows examples for the approximation of a support function with $l_{max} \leq l_0$ and $l_{max} > l_0$.

In order to calculate the coefficients h_l^m, one needs to compute the integrals in 5.18. These integrals can be approximated and efficiently computed by a FFT on S^2 (cf. [21]).

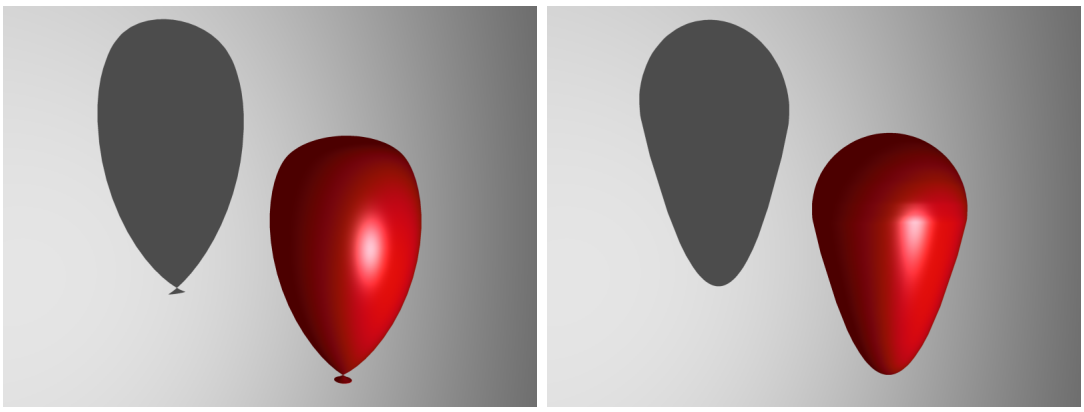


Figure 6.1: Example for approximation of support function with too small l_{max}

In order to do this, $h(\theta, \phi)$ is sampled on an equiangular grid of size $2l_{max} \times 2l_{max}$: $\theta_j = \pi(2j+1)/4l_{max}$, $\phi_k = 2\pi k/2l_{max}$, $j, k = 0, 1, \ldots, 2B-1$. The collection of sums of the form

$$\hat{h}(l,m) = \frac{\sqrt{2\pi}}{2l_{max}} \sum_{j=0}^{2l_{max}-1} \omega_j P_l^m(\cos\theta_j) \sum_{k=0}^{2l_{max}-1} e^{-im\phi_k} h(\theta_j, \phi_k)$$

for $|m| \leq l \leq l_{max}$ is called *Discrete Spherical Harmonic Transform*, where the constants ω_j are quadrature weights. Note that this transform uses a complex set of spherical harmonics $\tilde{Y}_l^m$ instead of definition 5.15, because it allows for a more efficient computation by FFTs.

$$\tilde{Y}_l^m := \begin{cases} \frac{1}{\sqrt{2}}(Y_l^m + iY_l^{-m}) & m \geq 0 \\ \frac{1}{\sqrt{2}}(Y_l^{-m} - iY_l^m) & m < 0 \end{cases}$$

The real coefficients h_l^m can thus be calculated from the coefficients $\hat{h}_l^m$ by

$$h_l^m = \begin{cases} \frac{1}{\sqrt{2}}(\hat{h}_l^m + \hat{h}_l^{-m}) & m \geq 0 \\ \frac{1}{\sqrt{2}i}(h_l^{-m} - \hat{h}_l^m) & m < 0 \end{cases}$$

In the accompanying software the freely available implementation S2kit (cf. [22]) of discrete spherical harmonic transforms was used.

6.3 Avoiding Numerical Problems when Evaluating Spherical Harmonics

As was mentioned before, equations 5.12 and 5.13 can not be evaluated at the poles $\theta \in \{0, \pi\}$. Moreover numerical problems can be expected and were encountered for θ close to 0 or π. A possible solution to this problem is to change the parameterisation $n(\theta, \phi)$ (see 5.10) of the unit sphere. This can be done by applying a bijective mapping

$f : S^2 \to S^2$ to n. We choose an orthogonal transformation represented by a matrix $R \in \mathbb{R}^{3\times3}, R^TR = I$, such that the poles are not mapped onto themselves, e.g. by choosing

$$R = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{pmatrix}. \tag{6.3}$$

Denote the poles of the new parameterisation by $P := \{Rn(0,0), Rn(\pi,0)\}$. Given a $p \in S^2 \setminus P$ its parameters $(\bar{\theta}, \bar{\phi})$ in the new parameterisation Rn can be calculated by applying n^{-1} to R^Tp.

The representation of h by spherical harmonics needs to be transformed to the new coordinate system too, this changed representation of h shall be denoted by $\bar{h}$. Given the coefficients h_l^m we need to calculate the *rotated* coefficients $\bar{h}_l^m$ such that for all (θ, ϕ) holds $h(\theta,\phi) = \bar{h}(n^{-1}(R^Tn(\theta,\phi)))$. It can be shown that the rotated coefficients $\bar{h}_l^m$ depend on the coefficients h_l^m of same degree l only, see [16] and [17]. In these papers a recurrence relation for the transformation matrices R_l, corresponding to the coefficients of degree l, is derived. In the accompanying software the rotation matrices R_l are precomputed up to a configurable degree at startup and kept in memory. This allows for fast computation of the rotated coefficients $\bar{h}_l^m$ by simple matrix vector multiplications, whenever the coefficients h_l^m are changed.

6.4 Umbilic Points

This section describes how relative umbilic points can be determined, making use of the linearity of the curvature tensor field on the faces. Let $v_i, 1 \le i \le 3$ denote the vertices adjacent to face f. Computation of umbilic points is done per face $f \in F$ using barycentric coordinates[2].

1. Compute the flattened curvature tensor matrices for adjacent vertices.
2. Compute the asymmetric deviator matrices $D_i^{\mathcal{R}}$.
3. Compute solutions $(\lambda_1, \lambda_2, \lambda_3)$ of $\sum_{i=1}^3 \lambda_i D_i^{\mathcal{R}} = 0$.

According to 4.14 the deviator matrices $D_i^{\mathcal{R}}$ of the relative curvature tensor field in the vertices v_i are given by

$$D_i^{\mathcal{R}} = \begin{pmatrix} \alpha_i & \gamma_i \\ \beta_i & -\alpha_i \end{pmatrix}.$$

Using $\sum_{i=1}^3 \lambda_i = 1$, point 3 leads to the linear equation system

$$\begin{pmatrix} \alpha_1 - \alpha_3 & \alpha_2 - \alpha_3 \\ \beta_1 - \beta_3 & \beta_2 - \beta_3 \\ \gamma_1 - \gamma_3 & \gamma_2 - \gamma_3 \end{pmatrix} \begin{pmatrix} \lambda_1 \\ \lambda_2 \end{pmatrix} = \begin{pmatrix} \alpha_3 \\ \beta_3 \\ \gamma_3 \end{pmatrix}.$$

[2]For the definition of barycentric coordinates see remark 3.1.

Note that we are only interested in solutions such that $0 \leq \lambda_i \leq 1$. As this linear equation system can be solved in a fixed number of steps for every vertex, the computation of umbilic points can be done in $\mathcal{O}(|V|)$, which allows to do it interactively.

Remark 6.3. In the symmetric case, when determining classical umbilic points, it holds that $\beta_i = \gamma_i$. Thus the above linear equation system reduces to a 2×2 linear equation system.

Remark 6.4. Due to the linear interpolation of the relative curvature tensor field between vertices, it is not guaranteed that all umbilic points are detected. Consequently, in the accompanying software a possibility was built in to manually add umbilic points, because they are used for the integration of curvature lines, see section 4.2.3. Examples for the failing detection of umbilic points can be seen in the following chapter.

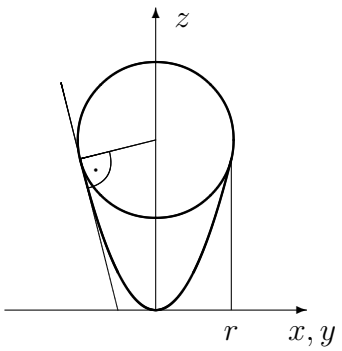
7 Results

This chapter illustrates some results that were obtained using the accompanying software. Most of the plots show the particular relativesphere used for generating the examples.

7.1 Curvature Lines - Influence on Shape Perception

In [14] the authors emphasize that the direction of lines on a surface may have an important influence on shape perception. An example for this can be seen in figures 7.1 and 7.2. The figures show a hyperbolic paraboloid $\mathcal{S}$ with different types of conjugate curve families. In the first plot the classical principal curvature lines can be seen. In contrast, the second plot shows parameter lines of the parameterisation $z = x^2 - y^2$, which can be obtained as relative principal curvature lines using a relativesphere that partly consists of the elliptic paraboloid $z = x^2 + y^2$. Such a relativesphere can be generated using the graph of $z = x^2 + y^2 \leq r^2$, where r is chosen big enough such that the image of $\mathcal{S}$ under $\boldsymbol{n}^{\mathcal{R}}$ lies within this graph. This surface is then *closed* using a sphere. The third plot shows the case where r is chosen such that $\boldsymbol{n}^{\mathcal{R}}$ partly maps to the spherical part of $\mathcal{R}$. Consequently the curvature lines near the center are parameter lines and change to principal curvature lines towards the boundary.

The figure to the right shows a sectional drawing of the rotationally symmetric relativesphere. Its support function h can easily be calculated and was approximated according to 6.2. The radii r for the second and third plot were chosen as 2 and 0.75 respectively. The support functions were approximated up to degree $l_{max} = 30$ for the second plot and $l_{max} = 20$ for the third plot.



7.2 Architectural Examples

The following examples document the steps of the remeshing process using surfaces from architectural projects. They demonstrate the possibilities of the accompanying software and provide input for future improvements.

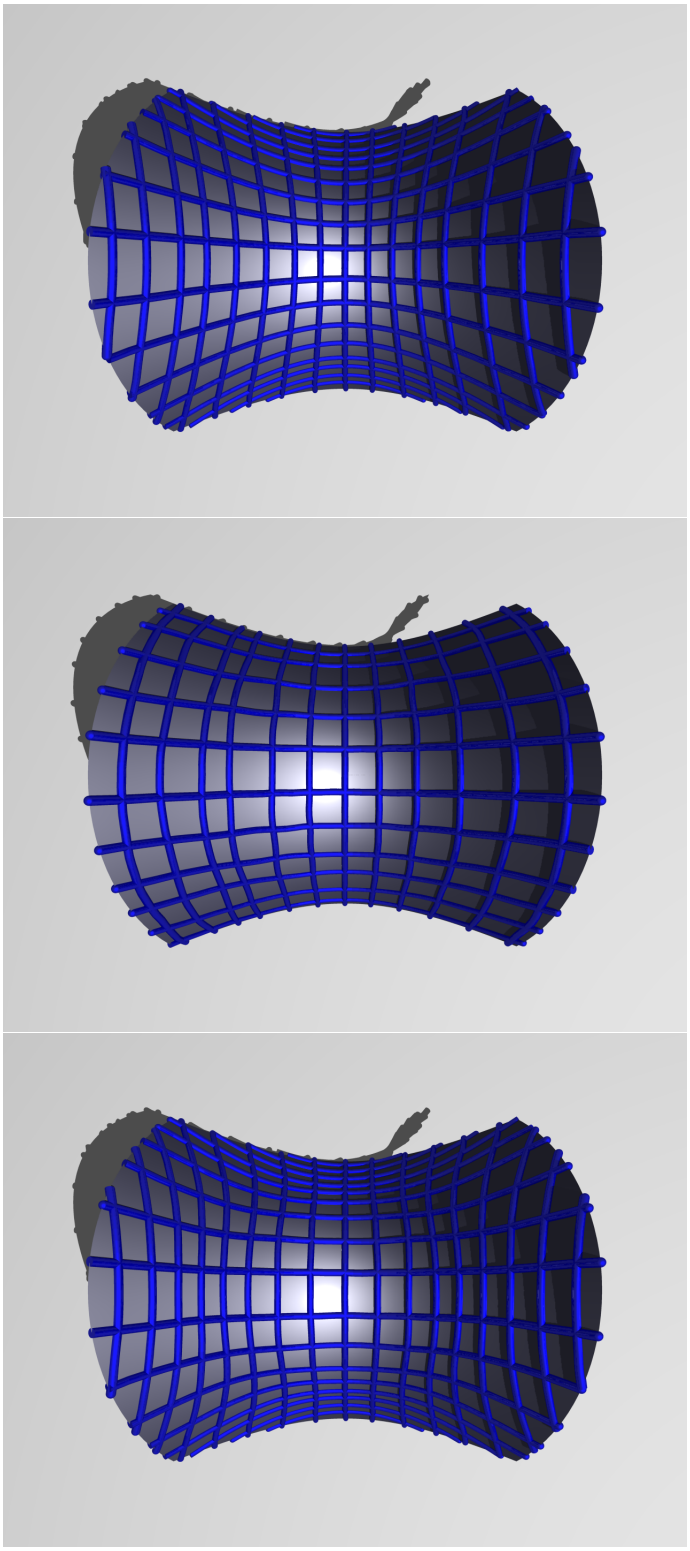


Figure 7.1: Saddle: Comparison of relative curvature lines for different $\mathcal{R}$

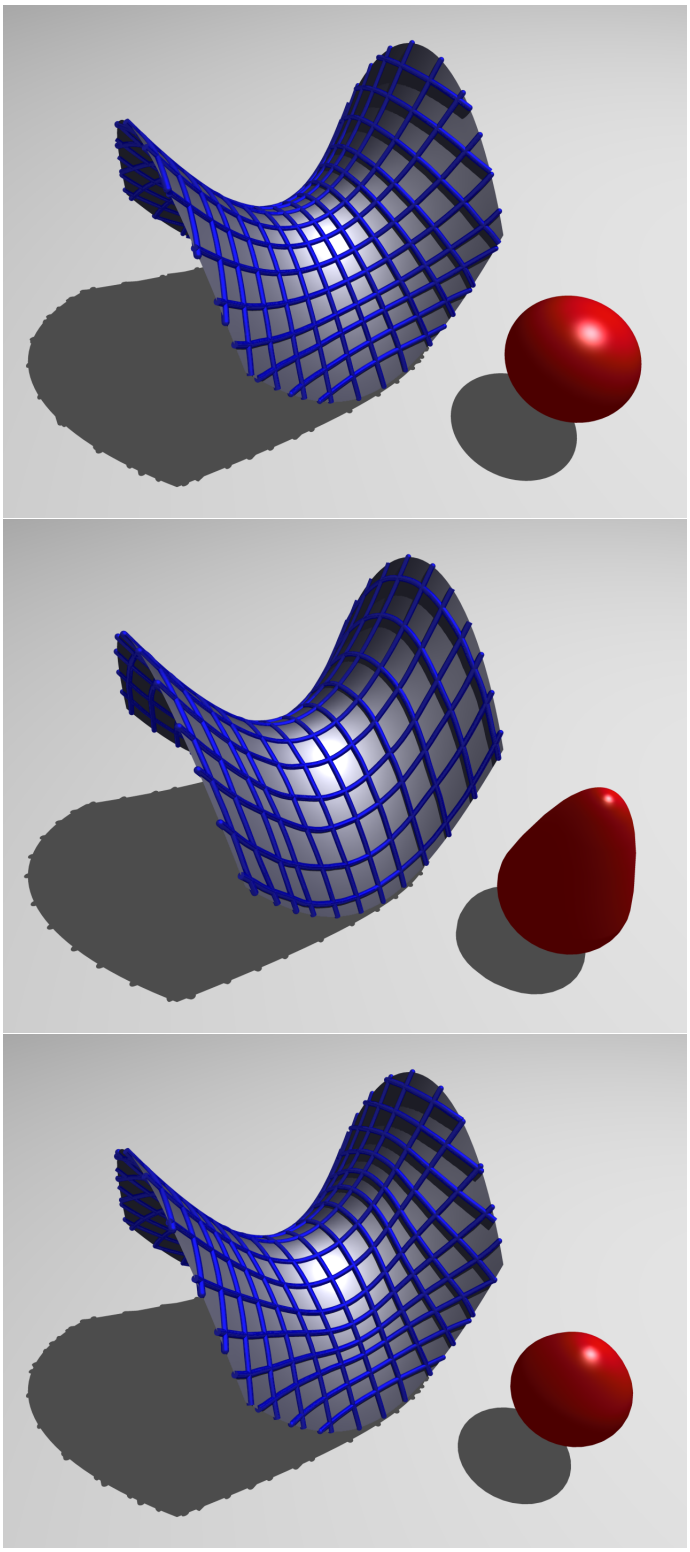


Figure 7.2: Saddle: Comparison of relative curvature lines for different $\mathcal{R}$

7.2.1 Architectural Example 1

The given surface $\mathcal{S}$ was the easiest to cope with out of the three examples. It consists of two nearly cylindrical parts and two parts that are close to a surface of revolution, which are connected by a central part. Figure 7.3 top shows classical principal curvature lines and umbilic points[1]. The umbilic points clearly cumulate in the central part of $\mathcal{S}$ and consequently would cause problems for the generation of a planar quad mesh. A possible approach is to choose a relativesphere $\mathcal{R}$ such that the central part exhibits less umbilic points. Figure 7.3 bottom shows a choice of $\mathcal{R}$ that drastically reduces the number of umbilic points in the central region. An approximation[2] up to degree $l_{max} = 35$ of the superellipsoid with $a_1 = 2, a_2 = 0.7, a_3 = 1.4$ and $l = 2.5$ was used for $\mathcal{R}$, see also example 5.5. The accompanying software was used to interactively rotate $\mathcal{R}$ relatively to $\mathcal{S}$, such that most of the umbilic points were removed. Figure 7.4 top shows a quad mesh generated using the curvature lines presented in 7.3. This quad mesh is not planar, as can be seen in the reflection of some faces. Planarization of the faces was done using the penalty method described in section 4.3, where the weights have been set to $w_{close} = w_{fair} = 1$ and the boundary vertices at corners have been fixed. The optimization was stopped at $\max_{f \in F} |c_f| = 5 \cdot 10^{-4}$. Figure 7.4 bottom shows the resulting planar quad mesh. A critical region of the mesh before and after planarization has already been shown in figure 4.12, see also example 4.4.

7.2.2 Architectural Example 2

The challenge in this example lies in the changing scale and orientation of similar features. Figure 5.1 shows the classical curvature lines and the umbilic points. While the features at the left and the lower side show the same geometry, they clearly differ in size from the right side. This leads to curvature lines becoming very close and eventually stopping, which can be recognized as singularities in a resulting quad mesh. The umbilic region on the right side can be reduced to isolated umbilic points. A possibility for doing this has been presented in section 5.3. Figure 5.1 and example 5.13 explain its application to this example.

Figure 7.6 shows two interesting examples using another relativesphere. In both cases $\mathcal{R}$ was chosen as an approximation up to degree $l_{max} = 50$ of the ellipsoid with $a_1 = 4, a_2 = 0.5, a_3 = 1$. As can be seen in figure 7.6, one can impose a main direction to one family of curves, which has been discussed in section 2.2.3. However this may lead to curvature lines crossing each other in sharp angles, which results in an undesirable appearance in the extracted quad meshes shown in figure 7.7.

Figure 7.5 shows a planar quad mesh obtained from the relative principal curvature lines shown in figure 5.1. The planarization was done using the penalty method with fixed boundary vertices and weights $w_{fair} = 0, w_{close} = 1$. Due to the fixed boundary vertices, the planarization stagnated at $\max_{f \in F} |c_f| \approx 3 \cdot 10^{-3}$. Looking closely, the reflection of some faces reveals their unplanarity, as can be seen in figure 7.7.

[1] In figure 7.3 it can be seen that the recognition of umbilic points may fail, see remark 6.4.
[2] See section 6.2.

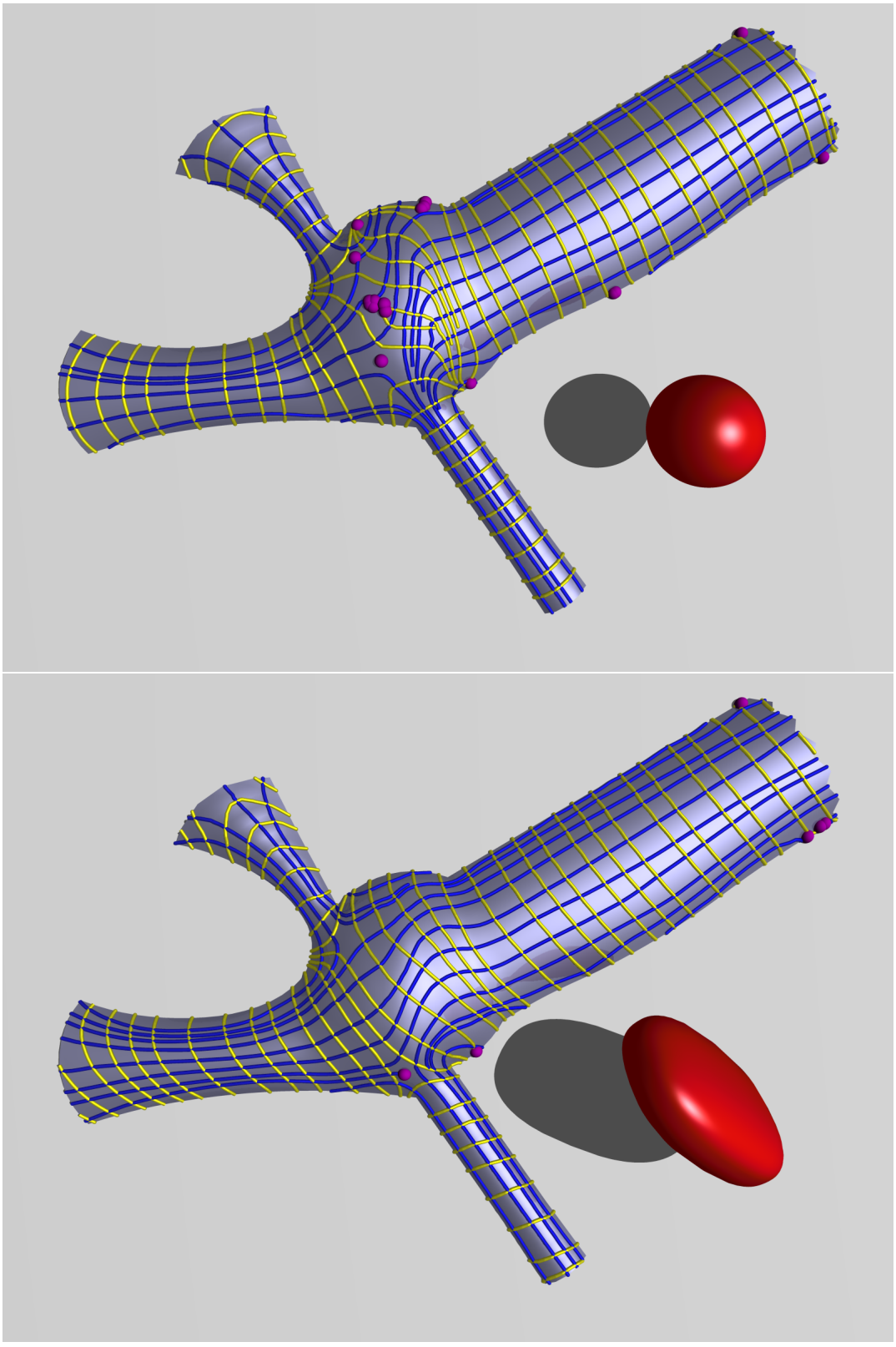


Figure 7.3: Architectural Example 1: Classical and relative curvature lines

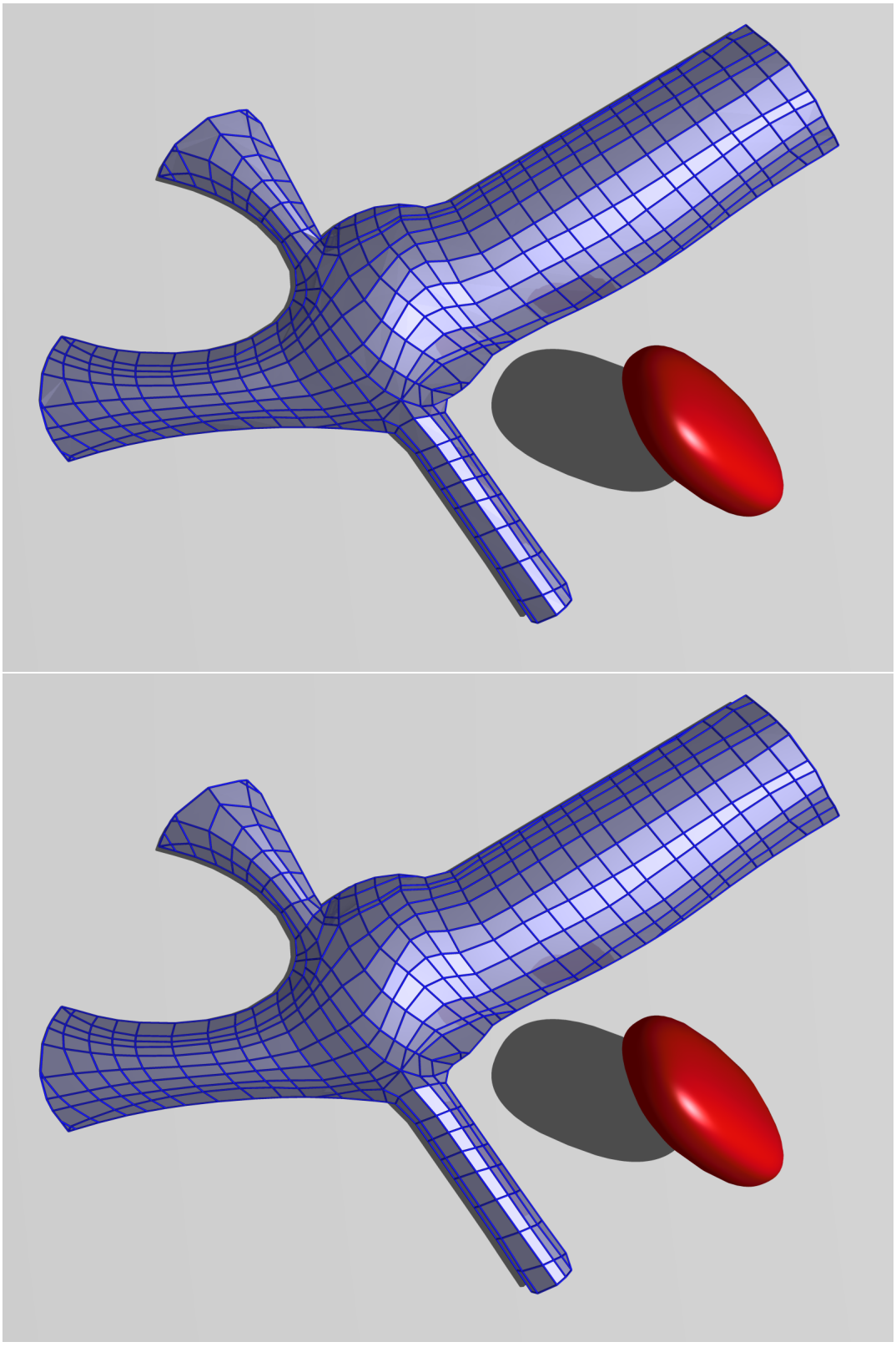


Figure 7.4: Architectural Example 1: Quad mesh before and after planarization

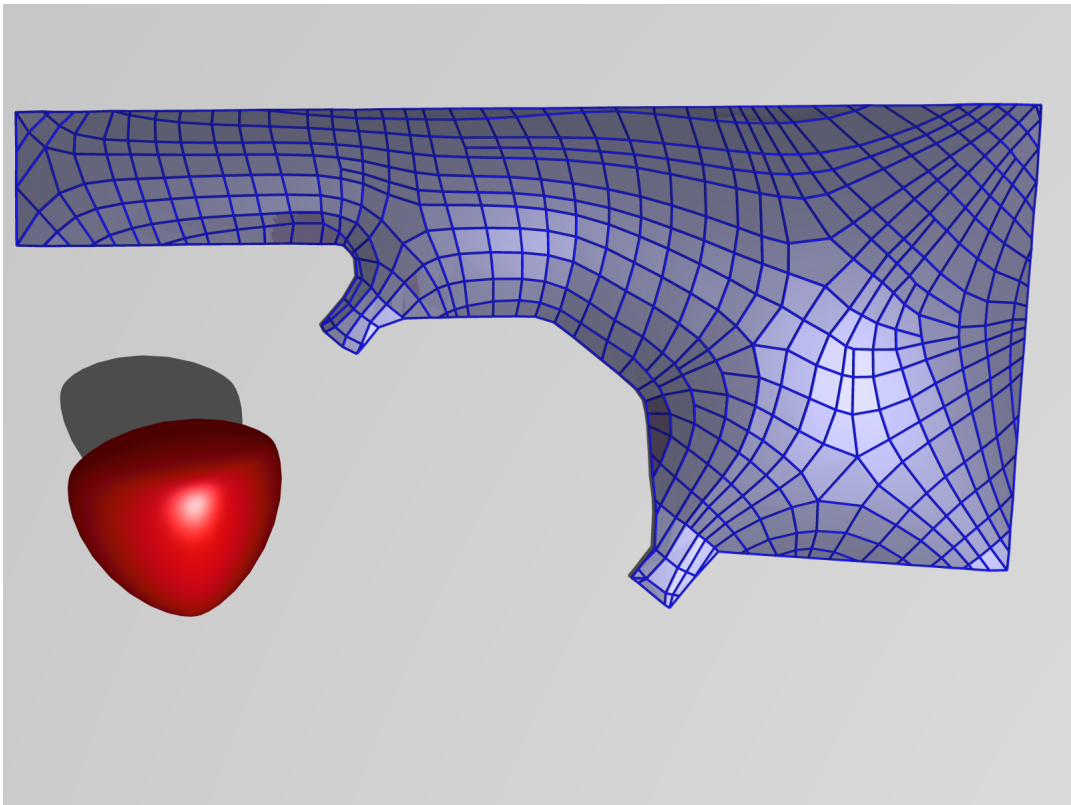


Figure 7.5: Architectural Example 2: PQ mesh from relative curvature lines shown in figure 5.1

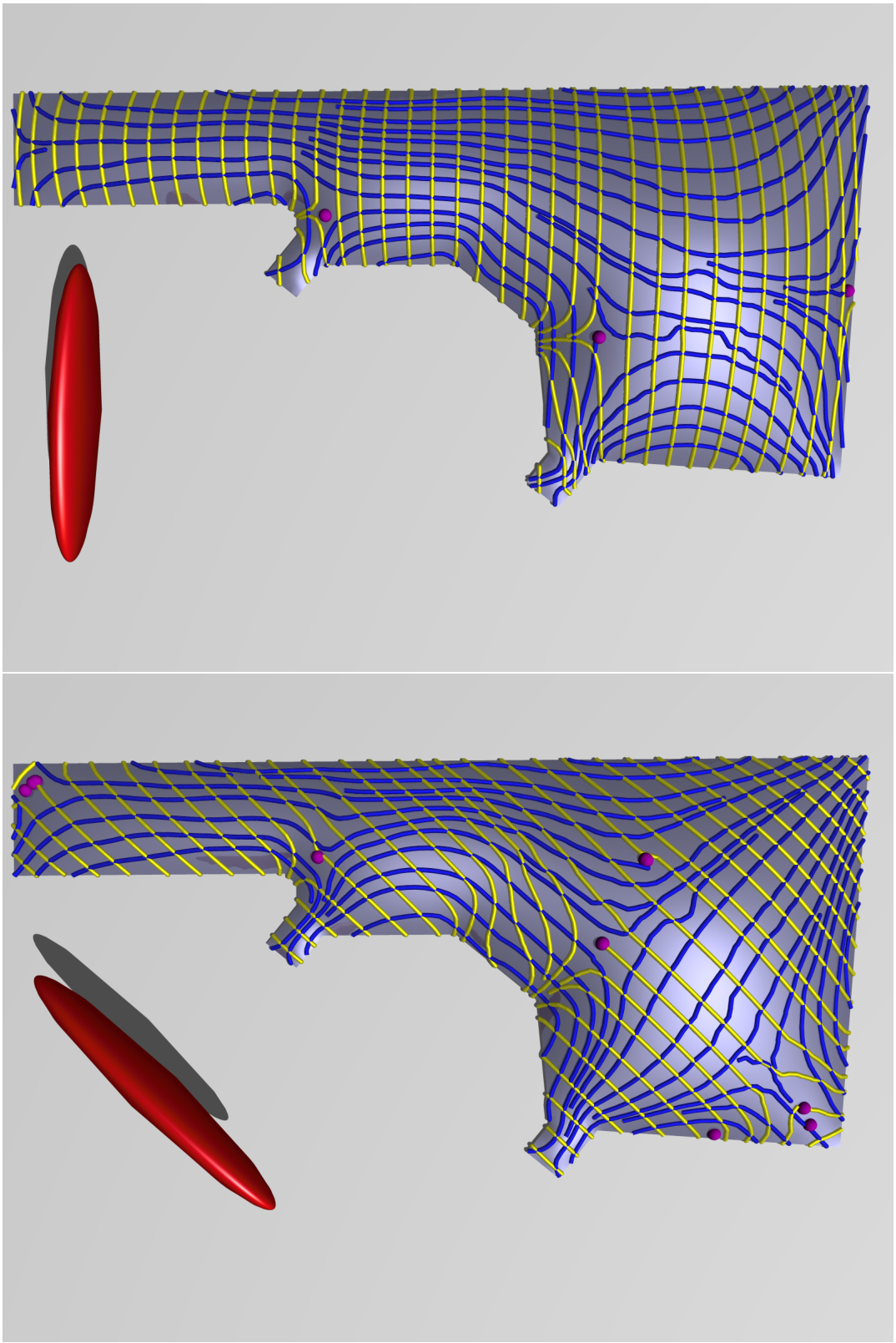


Figure 7.6: Architectural Example 2: Relative curvature lines and umbilic points for different orientations of $\mathcal{R}$

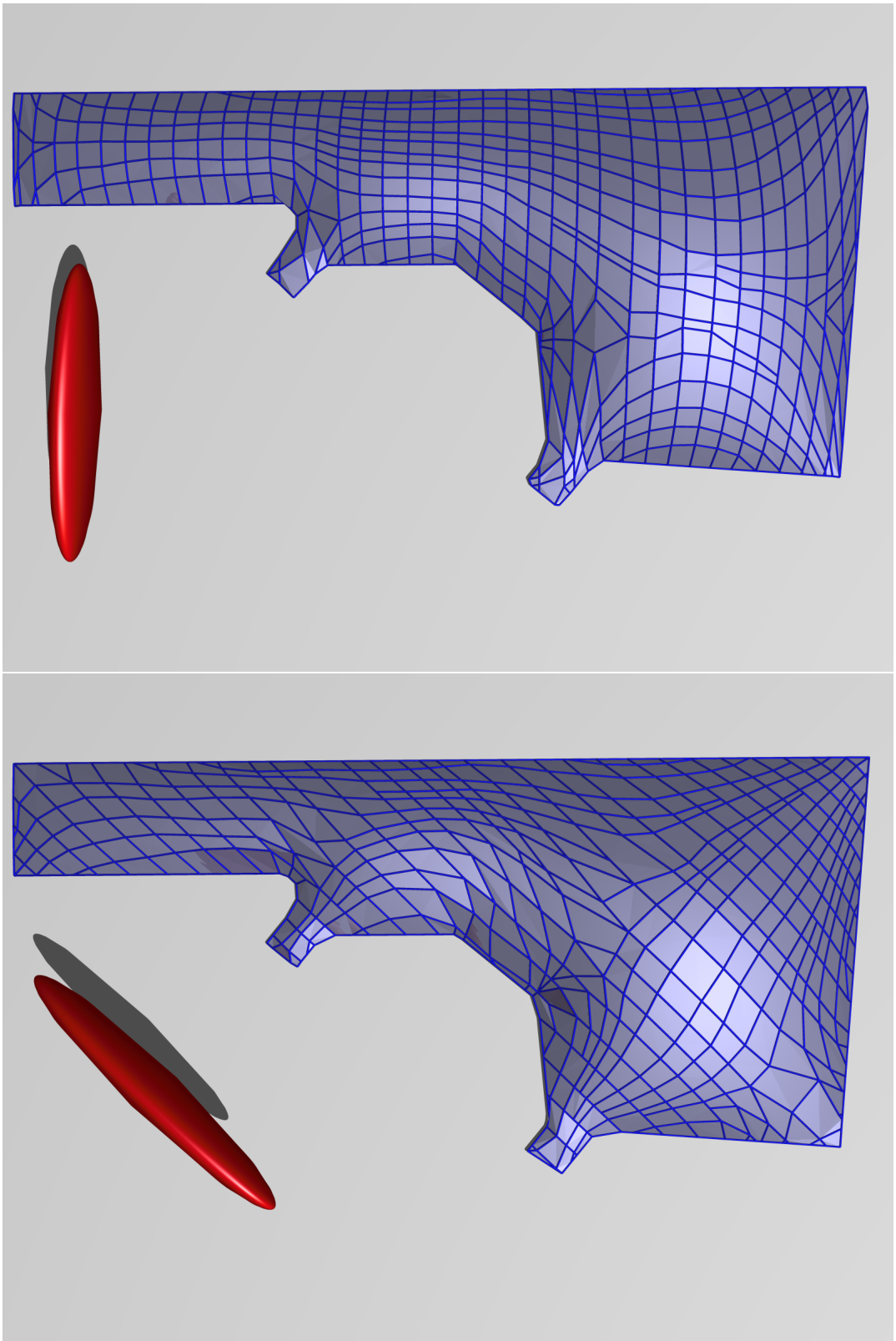


Figure 7.7: Architectural Example 2: Quad meshes corresponding to figure 7.6. Note that the faces of these quad meshes are not planar.

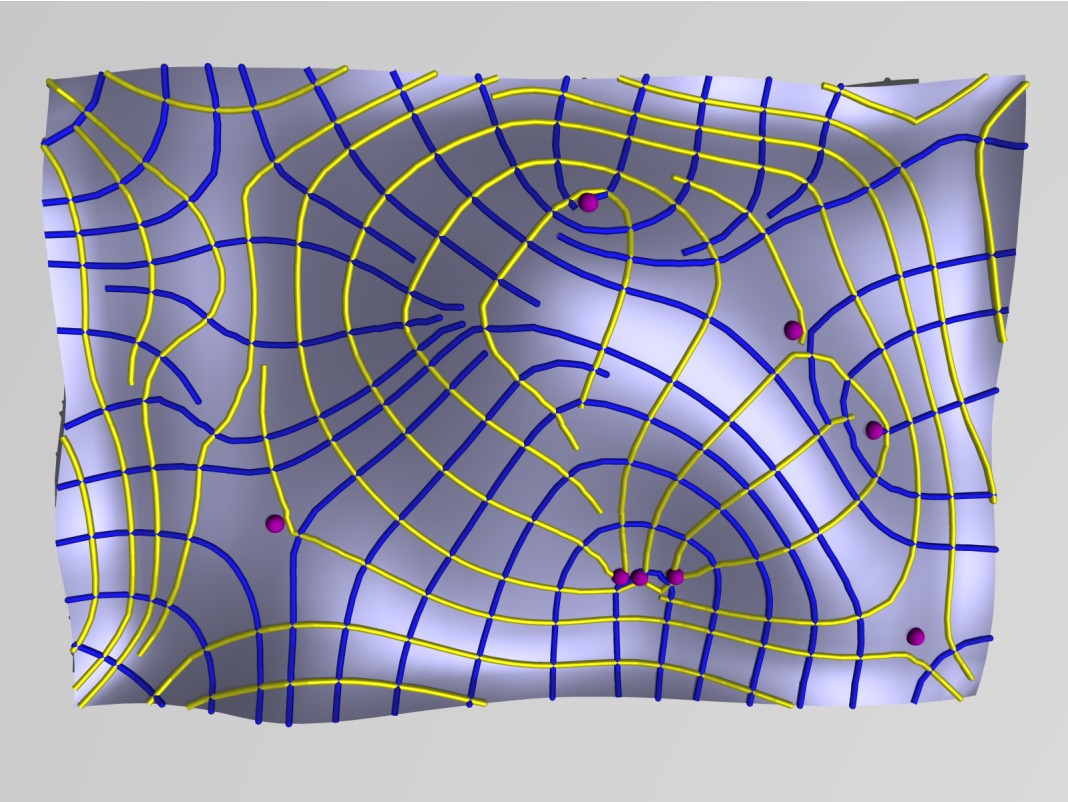


Figure 7.8: Architectural Example 3: Classical curvature lines and umbilic points

7.2.3 Architectural Example 3

The third surface mimics a flying carpet. It can be considered as the graph of a function defined on a rectangle. Figure 7.8 shows the umbilic points as well as classical curvature lines. As can be seen, the surface possesses a rich geometry with lots of *valleys*, *saddles* and *hilltops*. Consequently, choosing the relativesphere such that the number of umbilic points is reduced and curvature lines are spread more equally, becomes a game of luck. Figure 7.9 top shows a choice that resulted in the relative curvature lines orbiting around two umbilic points. The lower plot shows a planar quad mesh obtained from these. Note that some hand editing was done to the quad mesh before planarization. The penalty method was used for planarization using fixed vertices at the corners and $w_{fair} = 0, w_{close} = 1$.

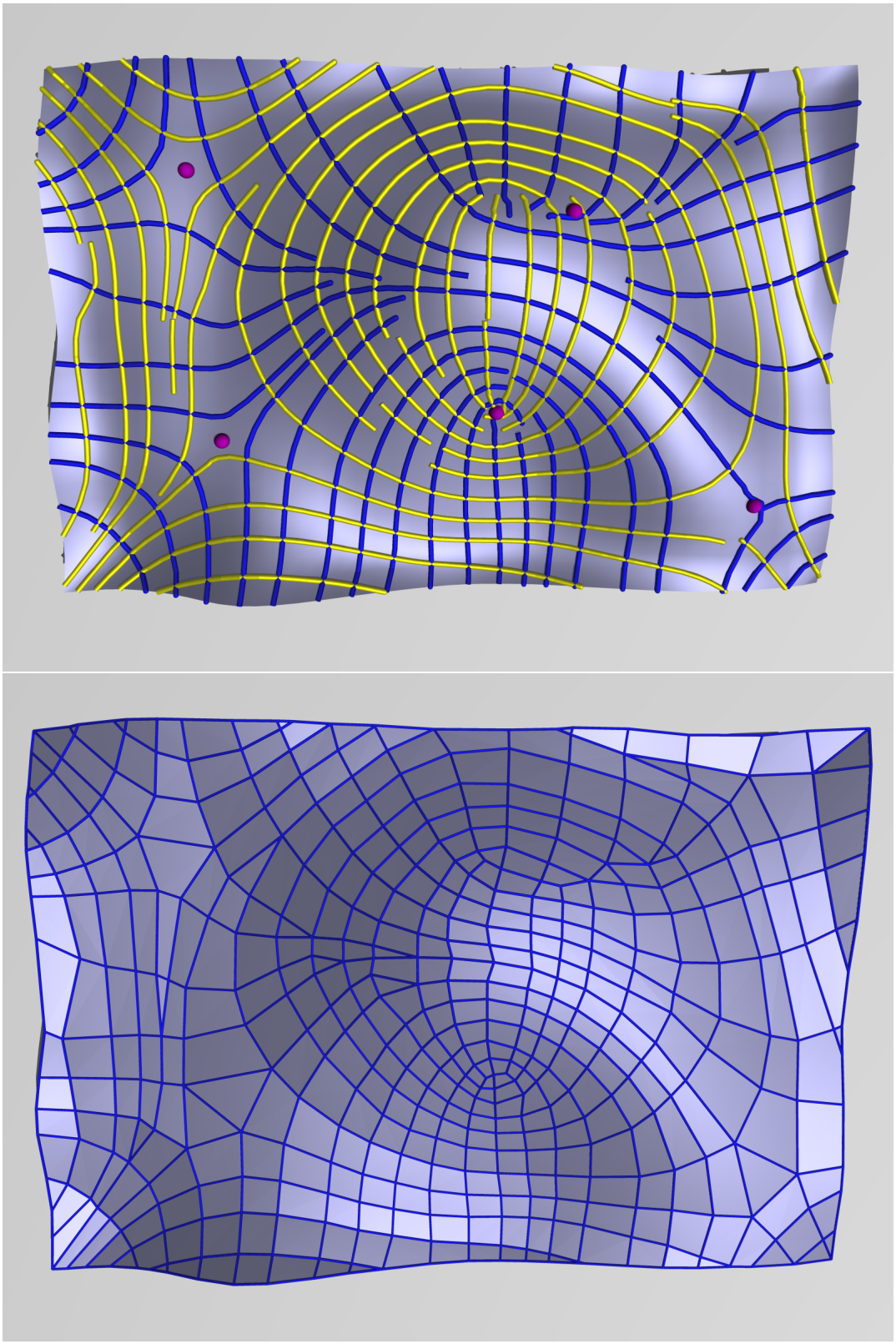


Figure 7.9: Architectural Example 3: Relative curvature lines and PQ mesh

8 Conclusion

In chapter 4 it was shown that it is essential to start from a conjugate curve network, if one wants to obtain a quad mesh with planar quadrilateral faces approximating $\mathcal{S}$. The use of relative principal curvature lines is one way to obtain a conjugate curve network. Representing the relativesphere $\mathcal{R}$ by its support function allows for a fast computation of the relative curvature tensor field from the classical curvature tensor field. Consequently the process of laying out the conjugate curve network can be done interactively. This has partly been implemented in the accompanying software, which allows for a fast change between different support functions as well as rotation of the represented relativesphere with respect to the given surface $\mathcal{S}$. The changes in the relative curvature tensor field as well as the relative umbilic points can be seen immediately. Apart from choosing $\mathcal{R}$ by trial and error, in sections 2.2.3 and 5.3 methods for influencing the direction of relative curvature lines, as well as for choosing the placement of relative umbilic points, were investigated. The approach described in section 5.3 has been implemented in the accompanying software, although some questions stated there remain open for further investigation. Concluding, the method described provides a powerful tool for the layout of conjugate curve networks and consequently for the generation of planar quad meshes. A drawback is the global choice of $\mathcal{R}$, which results in the requirement that it be strictly convex (cf. definition 2.15 of the relative Gaussian map $\boldsymbol{n}^{\mathcal{R}}$). This could be relaxed by requiring that $\boldsymbol{n}^{\mathcal{R}}$ can still be defined in a differentiable way and $\mathcal{R}$ being locally strictly convex in $\boldsymbol{n}^{\mathcal{R}}(\mathcal{S}) \subset \mathcal{R}$. Furthermore it would be interesting to investigate whether $\mathcal{R}$ can be chosen as a function of $p \in \mathcal{S}$.

The examples in the previous chapter show that a lot of work remains to be done in order to make the method useable for architectural purposes. Aesthetics of the resulting mesh plays a dominant role in this case, which results in the conflictive requirements of tracing long curvature lines while covering the surface uniformly.

Regarding the implementation of the described methods in the accompanying software, the following improvements could be done. The approach chosen for integration of curvature lines without a parameterisation of the surface has major drawbacks. First of all higher order integration schemes can not be used for curvature line integration, and the results depend strongly on the mesh which defines $\mathcal{S}$. This issues could be addressed by a global parameterisation. Moreover a global conformal parameterisation as described in [33] directly leads to a flattened curvature tensor field, which simplifies the computation of umbilic points. The construction of a quad-dominant output mesh from relative curvature principal lines would benefit from a global parameterisation too, because this can be done easier and more reliable in 2-dimensional parameter space. Finally, extensive editing capabilities of the curvature lines would prove very useful to improve the mesh construction.

List of Figures

Bibliography

[1] M. Abramowitz and I. A. Stegun. *Handbook of mathematical functions.* National Bureau of Standards, 1964.

[2] Pierre Alliez, David Cohen-Steiner, Olivier Devillers, Bruno Levy, and Mathieu Desbrun. Anisotropic polygonal remeshing. *ACM Transactions on Graphics. Special issue for SIGGRAPH conference*, pages 485–493, 2003.

[3] Herbert Amann and Joachim Escher. *Analysis II.* Birkhäuser, 1999.

[4] T. Bonnesen and W. Fenchel. *Theorie der konvexen Körper.* Springer, 1934.

[5] M. Botsch, S. Steinberg, S. Bischoff, and L. Kobbelt. Openmesh – a generic and efficient polygon mesh data structure, 2002.

[6] David Cohen-Steiner and Jean-Marie Morvan. Restricted delaunay triangulations and normal cycle. In *SCG '03: Proceedings of the nineteenth annual symposium on Computational geometry*, pages 312–321, New York, NY, USA, 2003. ACM Press.

[7] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms.* MIT Press and McGraw-Hill, second edition, 2001.

[8] Thierry Delmarcelle. *The Visualization of Second-Order Tensor Fields.* PhD thesis, Standford University, December 1994.

[9] Thierry Delmarcelle and Lambertus Hesselink. The topology of symmetric, second-order tensor fields. In *VIS '94: Proceedings of the conference on Visualization '94*, pages 140–147, Los Alamitos, CA, USA, 1994. IEEE Computer Society Press.

[10] M. Desbrun, E. Grinspun, E. Schröder, K. Polthier, and A. Stern. Discrete differential geometry: An applied introduction. In *SIGGRAPH Course Notes*, 2006.

[11] Mathieu Desbrun, Mark Meyer, Peter Schröder, and Alan H. Barr. Implicit fairing of irregular meshes using diffusion and curvature flow. In *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 317–324, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

[12] Manfredo P. do Carmo. *Differential Geometry of Curves and Surfaces.* Prentice Hall, 1976.

[13] S. Dong, S. Kircher, and M. Garland. Harmonic functions for quadrilateral remeshing of arbitrary manifolds. *Comput. Aided Geom. Des.*, 22(5):392–423, 2005.

[14] Ahna Girshick, Victoria Interrante, Steven Haker, and Todd Lemoine. Line direction matters: an argument for the use of principal directions in 3d line drawings. In *NPAR '00: Proceedings of the 1st international symposium on Non-photorealistic animation and rendering*, pages 43–52, New York, NY, USA, 2000. ACM Press.

[15] H. Groemer. *Geometric Applications of Fourier Series and Spherical Harmonics*. Cambridge University Press, 1996.

[16] J. Ivanic and K. Ruedenberg. Rotation matrices for real spherical harmonics. direct determination by recursion. *Journal of Physical Chemistry*, 100:6342–6347, 1996.

[17] J. Ivanic and K. Ruedenberg. Rotation matrices for real spherical harmonics. direct determination by recursion. additions and corrections. *Journal of Physical Chemistry*, 102(45):9099–9100, 1998.

[18] Z. Sir J. Gravesen and B. Jüttler. Curves and surfaces represented by polynomial support functions. *submitted*, 2007.

[19] Klaus Jänich. *Vektoranalysis*. Springer, 5th edition, 2005.

[20] Bruno Jobard and Wilfrid Lefer. Creating evenly-spaced streamlines of arbitrary density. In W. Lefer and M. Grave, editors, *Visualization in Scientific Computing '97. Proceedings of the Eurographics Workshop in Boulogne-sur-Mer, France*, pages 43–56, Wien, New York, 1997. Springer Verlag.

[21] D. Healy Jr., D. Rockmore, P. Kostelec, and S. Moore. Ffts for the 2-sphere - improvements and variations. *The Journal of Fourier Analysis and Applications*, 9:341 – 385, 2003.

[22] Peter J. Kostelec and Daniel N. Rockmore. S2kit: A lite version of spharmonickit. Technical report, Department of Mathematics, Dartmouth College, Hanover NH 03755, 2004.

[23] Yang Liu, Helmut Pottmann, Johannes Wallner, Yong-Liang Yang, and Wenping Wang. Geometric modeling with conical meshes and developable surfaces. *ACM Trans. Graphics*, 25(3):681–689, 2006. Proc. SIGGRAPH.

[24] K. Madsen, H. Nielsen, and O. Tingleff. Methods for nonlinear least squares problems. Lecture Notes, 2004.

[25] K. Madsen, H. Nielsen, and O. Tingleff. Optimization with constraints. Lecture Notes, 2004.

[26] Martin Marinov and Leif Kobbelt. Direct anisotropic quad-dominant remeshing. In *PG '04: Proceedings of the Computer Graphics and Applications, 12th Pacific Conference on (PG'04)*, pages 207–216, Washington, DC, USA, 2004. IEEE Computer Society.

[27] Robert Plato. *Numerische Mathematik kompakt*. Vieweg, 2nd edition, 2004.

[28] Ian R. Porteous. *Geometric Differentiation*. Cambridge University Press, 2nd edition, 2001.

[29] Helmut Pottmann. General offset surfaces. *Neural, Parallel & Scient. Comput.*, 5:55–80, 1997.

[30] Helmut Pottmann and Johannes Wallner. *Computational Line Geometry*. Springer, 2001.

[31] Helmut Pottmann, Johannes Wallner, Yong-Liang Yang, Yu-Kun Lai, and Shi-Min Hu. Principal curvatures from the integral invariant viewpoint. *Comput. Aided Geom. Design*, 2006. submitted.

[32] William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling. *Numerical Recipes in C : The Art of Scientific Computing*. Cambridge University Press, October 1992.

[33] Nicolas Ray, Wan Chiu Li, Bruno Lévy, Alla Sheffer, and Pierre Alliez. Periodic global parameterization. *ACM Trans. Graph.*, 25(4):1460–1485, 2006.

[34] Robert Sauer. *Differenzengeometrie*. Springer-Verlag Berlin Heidelberg New York, 1970.

[35] Xavier Tricoche. *Vector and Tensor Topology Simplification, Tracking, and Visualization*. PhD thesis, University of Kaiserslautern, 2002.

[36] Wikipedia. Associated legendre function — wikipedia, the free encyclopedia, 2007. [Online; accessed 19-March-2007].

[37] Wikipedia. Spherical harmonics — wikipedia, the free encyclopedia, 2007. [Online; accessed 19-March-2007].

[38] Yong-Liang Yang, Yu-Kun Lai, Shi-Min Hu, and Helmut Pottmann. Robust principal curvatures on multiple scales. In Konrad Polthier and Alla Sheffer, editors, *SGP 2006: 4th Eurographics Symposium on Geometry processing*, pages 223–226. Eurographics Association, 2006.