Theory and Applications of Bijective Barycentric Mappings

Doctoral Dissertation submitted to the
Faculty of Informatics of the Università della Svizzera italiana
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

presented by
Teseo Schneider

under the supervision of
Kai Hormann

June 2017

Dissertation Committee

Kai Hormann	Università della Svizzera italiana, Lugano, Switzerland
Michael Bronstein	Università della Svizzera italiana, Lugano, Switzerland
Rolf Krause	Università della Svizzera italiana, Lugano, Switzerland
Mirela Ben-Chen	Technion, Haifa, Israel
Bert Jüttler	Johannes Kepler University, Linz, Austria

Dissertation accepted on 12 June 2017

Research Advisor	PhD Program Director
Kai Hormann	**Walter Binder**

I certify that except where due acknowledgement has been given, the work presented in this thesis is that of the author alone; the work has not been submitted previously, in whole or in part, to qualify for any other academic award; and the content of the thesis is the result of work which has been carried out since the official commencement date of the approved research program.

Teseo Schneider
Lugano, 12 June 2017

If debugging is the process of removing bugs, then programming must be the process of putting them in.

Edsger W. Dijkstra

Abstract

Barycentric coordinates provide a convenient way to represent a point inside a triangle as a convex combination of the triangle's vertices, and to linearly interpolate data given at these vertices. Due to their favourable properties, they are commonly applied in geometric modelling, finite element methods, computer graphics, and many other fields. In some of these applications it is desirable to extend the concept of barycentric coordinates from triangles to polygons. Several variants of such generalized barycentric coordinates have been proposed in recent years.

An important application of barycentric coordinates consists of barycentric mappings, which allow to naturally warp a source polygon to a corresponding target polygon, or more generally, to create mappings between closed curves or polyhedra. The principal practical application is image warping, which takes as input a control polygon drawn around an image and smoothly warps the image by moving the polygon vertices. A required property of image warping is to avoid fold-overs in the resulting image. The problem of fold-overs is a manifestation of a larger problem related to the lack of bijectivity of the barycentric mapping. Unfortunately, bijectivity of such barycentric mappings can only be guaranteed for the special case of warping between convex polygons or by triangulating the domain and hence renouncing smoothness. In fact, for any barycentric coordinates, it is always possible to construct a pair of polygons such that the barycentric mapping is not bijective. In the first part of this thesis we illustrate three methods to achieve bijective mappings.

The first method is based on the intuition that, if two polygons are sufficiently close, then the mapping is close to the identity and hence bijective. This suggests to "split" the mapping into several intermediate mappings and to create a composite barycentric mapping which is guaranteed to be bijective between arbitrary polygons, polyhedra, or closed planar curves. We provide theoretical bounds on the bijectivity of the composite mapping related to the norm of the gradient of the coordinates. The fact that the bound depends on the gradient implies that these bounds exist only if the gradient of the coordinates is bounded. We fo-

cus on mean value coordinates and analyse the behaviour of their directional derivatives and gradient at the vertices of a polygon.

The composition of barycentric mappings for closed planar curves leads to the problem of blending between two planar curves. We suggest to solve it by linearly interpolating the signed curvature and then reconstructing the intermediate curve from the interpolated curvature values. However, when both input curves are closed, this strategy can lead to open intermediate curves. We present a new algorithm for solving this problem, which finds the closed curve whose curvature is closest to the interpolated values. Our method relies on the definition of a suitable metric for measuring the distance between two planar curves and an appropriate discretization of the signed curvature functions.

The second method to construct smooth bijective mappings with prescribed behaviour along the domain boundary exploits the properties of harmonic maps. These maps can be approximated in different ways, and we discuss their respective advantages and disadvantages. We further present a simple procedure for reducing their distortion and demonstrate the effectiveness of our approach by providing examples.

The last method relies on a reformulation of complex barycentric mappings, which allows us to modify the "speed" along the edges to create complex bijective mappings. We provide some initial results and an optimization procedure which creates complex bijective maps.

In the second part we provide two main applications of bijective mapping. The first one is in the context of finite elements simulations, where the discretization of the computational domain plays a central role. In the standard discretization, the domain is triangulated with a mesh and its boundary is approximated by a polygon. We present an approach which combines parametric finite elements with smooth bijective mappings, leaving the choice of approximation spaces free. This approach allows to represent arbitrarily complex geometries on coarse meshes with curved edges, regardless of the domain boundary complexity. The main idea is to use a bijective mapping for automatically warping the volume of a simple parametrization domain to the complex computational domain, thus creating a curved mesh of the latter.

The second application addresses the meshing problem and the possibility to solve finite element simulations on polygonal meshes. In this context we present several methods to discretize the bijective mapping to create polygonal and piecewise polynomial meshes.

Acknowledgements

First of all I would like to express my gratitude to my advisor, Professor Kai Hormann for his support and advice during my PhD journey. By speaking with several colleagues I came to the conclusion that the success of a thriving PhD largely depends on the relationship with the advisor. In that sense Kai was also a great mentor. His precision, care for details and constructive criticism, allowed me to grow as a researcher. Finally he was always available (even while being the dean of the faculty) to give help, discuss, and have a beer.

Undoubtedly, I would also like to thank my committee members: Professor Rolf Krause, Professor Michael Bronstein, Professor Bert Jüttler, and Professor Mirela Ben-Chen. Thank you for agreeing to read this (long) document and evaluate my thesis, without your contribution I could not graduate.

I need to thank the Swiss National Fund (SNF) for having supported my work. They allowed me to literally travel worldwide to attend conferences, where I built a personal network and met other fellow researchers. I take the occasion to also thank all my co-authors for the opportunity to work with them. In particular Patrick, with whom I shared a long educational period (we started the bachelor together), for the many discussions, ideas, and scientific arguments.

Most certainly, I need to thank Mauro Prevostini, the program manager of the university. He allowed me to organize many miscellaneous didactic activities with the purpose to warm kids up to informatics. For instance, I had the possibility to teach in several local high schools, provide lectures for teachers, participate to many fairs and several other activities.

I also want to thank my three colleagues and "office fellows", Dmitry, Emiliano, and Elena for their friendship. All the many hours spent in the office tirelessly working on our publications were sweeter because of the occasional chats and coffees shared with them.

Last but not least I want to thank my family, in particular my wife Purnur for her presence, spell checking, and sacrifices in these many years. Most of all, I am grateful that she took care of our two small kids.

Contents

Figures

Overview

This dissertation is divided into two parts: theory and practice. In the first part, we start by explaining the contextualization of the problem of mapping between two domains (Section 1.1). While doing so we introduce the concept of barycentric coordinates for polygons and polyhedra, and explore its extension to the complex plane (Section 1.2) and to closed curves (Section 1.3). Chapter 2 gives an overview of existing methods for creating bijective maps, starting from barycentric mappings between convex polygons (Section 2.1.1) and then showing that bijectivity cannot be guaranteed for such mappings (Section 2.1.2). To overcome this limitation we illustrate several alternatives based on an optimization procedure (Section 2.2).

We then present different strategies for interpolating shapes (Chapter 3), focusing on the case of closed planar curves (Section 3.3), which is based on

Saba, Schneider, Scateni and Hormann [2014]. *Curvature-based blending of closed planar curves.* **76**(5): 263–272. Computer Aided Geometric Design, proceeding of Geometric Modelling and Processing.

Chapter 4 is based on two contributions

Schneider, Hormann and Floater [2013]. *Bijective composite mean value mappings.* **35**(5): 125–135. Computer Graphics Forum, proceeding of Symposium on Geometry Processing.

Schneider [2017]. *Generalized Barycentric Coordinates in Computer Graphics and Computational Mechanics.* (Chapter 4) In K. Hormann and N. Sukumar (eds), CRC Press, Florida, to appear in August.

These two contributions illustrate a method to create bijective mappings between arbitrary shapes based on a composition of barycentric mappings and shape interpolation.

Chapter 5 proposes an alternative approach for creating bijective maps using harmonic mappings. It is based on

Schneider and Hormann [2015]. *Smooth bijective maps between arbitrary planar polygons.* **35–36**(C): 243–354. Computer Aided Geometric Design, proceeding of Geometric Modelling and Processing.

As composite barycentric mappings require the norm of the gradient of the underlying barycentric coordinates to be bounded. This motivation leads to the study of the directional derivative of Wachspress (Section 6.1) and mean value (Section 6.2.1) coordinates which is based on

Anisimov, Hormann and Schneider [n.d.] *Behaviour of exponential three-point coordinates at the vertices of convex polygons.* Work in progress.

The chapter continues by providing some initial results on an actual bound of mean value coordinates obtained in collaboration with K. Hormann.

Finally, we exploit the expressiveness of complex coordinates (Chapter 7) to modify complex barycentric mappings for achieving bijectivity. This chapter is based on initial work performed together with E. Cirillo and K. Hormann.

The second part of the dissertation regards the applications of bijective mappings to concrete examples. We start by providing some classical examples and use-cases in computer graphics (Chapter 8). We then explain how these mappings can be used in the context of parametric finite elements (Section 9.1). The explanation is based on

Zulian, Schneider, Hormann and Krause [n.d.]. *Parametric finite elements with bijective mappings.* Accepted at BIT Numerical mathematics.

We conclude by giving an overview of the meshing challenges in finite elements (Chapter 10) and show how an approximation of the bijective map can be used as a meshing technique (Section 10.3), which is based on

Schneider, Zulian, Krause and Hormann [n.d.]. *Multigrid method with parametric finite elements for arbitrarily shaped 2D meshes.* Work in progress.

Part I

Theory

Chapter 1

Introduction

Let $\Omega \subset \mathbb{R}^d$ be an open domain, where by $\bar{\Omega}$ we denote the closure of Ω. The problem we address is the creation of the mapping

$$\boldsymbol{f} : \Omega_0 \to \Omega_1,$$

where Ω_0 is the source domain and Ω_1 is the target domain. In the following we consider the cases where the domains are described by polygons, polyhedra, or planar closed curves. Our main purpose is to construct bijective mappings, and we show different approaches to achieve bijectivity. Moreover, with the applications in sight, we discuss three main properties. First, the linearity of $\boldsymbol{f}$ along the boundary of the domain, which allows to achieve more natural results. Second, the natural distortion introduced by the mapping. Third, the symmetry of the construction; that is, if we construct the mapping

$$\boldsymbol{g} : \Omega_1 \to \Omega_0$$

by inverting the roles of Ω_0 and Ω_1, then $\boldsymbol{g} = \boldsymbol{f}^{-1}$.

For the two-dimensional case, let P be a polygon with $n \geq 3$ vertices $\boldsymbol{v}^1, \ldots, \boldsymbol{v}^n \in \mathbb{R}^2$ in anticlockwise order. Then the open domain Ω is the interior of P. Note that throughout this dissertation we consider vertex indices cyclically over $1, \ldots, n$, so that $\boldsymbol{v}^{n+1} = \boldsymbol{v}^1$ and $\boldsymbol{v}^0 = \boldsymbol{v}^n$, for instance. Similarly, let

$$\gamma : I \to \mathbb{R}^2$$

be a closed planar curve parametrized over the interval I, therefore Ω is the interior of $\gamma(s)$.

For $d = 3$, let $\mathcal{P}$ be a polyhedron with $n \geq 4$ vertices $\boldsymbol{v}^1, \ldots, \boldsymbol{v}^n \in \mathbb{R}^3$ and $l \geq 4$ triangular faces $\mathcal{F}_1, \ldots, \mathcal{F}_l$, where the domain Ω is the interior of $\mathcal{P}$.

1.1 Generalized barycentric coordinates

Barycentric coordinates were first discovered by August Ferdinand Möbius [Möbius, 1827], while considering the problem of defining a coordinate system with respect to simplices in $\mathbb{R}^d$. For instance, Möbius defined the *triangular barycentric coordinates* as the ratio of signed areas of a triangle

$$\phi_i(\boldsymbol{x}) = \frac{\mathcal{A}_i}{\mathcal{A}_1 + \mathcal{A}_2 + \mathcal{A}_3}, \qquad \text{for } i = \{1,2,3\} \tag{1.1}$$

where $\mathcal{A}_i = \mathcal{A}(\boldsymbol{v}, \boldsymbol{v}^{i-1}, \boldsymbol{v}^{i+1})$ is the signed area of the triangle opposite to $\boldsymbol{v}^i$. It turns out that these coordinates provide a natural way to interpolate function values at the vertices. Let h_i, $i = 1,\dots,3$ be some values defined at the vertices of the triangle, then

$$h(\boldsymbol{x}) = \sum_{i=1}^{3} \phi_i(\boldsymbol{x}) h_i,$$

interpolates the values on the interior.

This essential application led to the generalization of barycentric coordinates from simplices to arbitrary polyhedra. This generalization results in coordinates that are no longer unique, therefore their definition follows from the fundamental properties of simplicial barycentric coordinates.

Definition 1. A set of functions $\phi_1,\dots,\phi_n : \bar{\Omega} \to \mathbb{R}^d$ is called a set of *generalized barycentric coordinates*, if ϕ_i satisfy the three properties

$$\text{1) Partition of unity:} \qquad \sum_{i=1}^{n} \phi_i(\boldsymbol{x}) = 1, \qquad \boldsymbol{x} \in \bar{\Omega}, \tag{1.2a}$$

$$\text{2) Barycentric property:} \qquad \sum_{i=1}^{n} \phi_i(\boldsymbol{x}) \boldsymbol{v}^i = \boldsymbol{x}, \qquad \boldsymbol{x} \in \bar{\Omega}, \tag{1.2b}$$

$$\text{3) Lagrange property:} \qquad \phi_i(\boldsymbol{v}^j) = \delta_{ij}, \qquad j = 1,\dots,n, \tag{1.2c}$$

where δ_{ij} is the *Kronecker delta*.

As for simplices, they can be used to interpolate values at the vertices of the domain. Let $h_i \in \mathbb{R}$, $i = 1,\dots,n$ be some values defined at the vertices of the domain, then

$$h : \Omega \subset \mathbb{R}^d \to \mathbb{R} \qquad h(\boldsymbol{x}) = \sum_{i=1}^{n} \phi_i(\boldsymbol{x}) h_i, \tag{1.3}$$

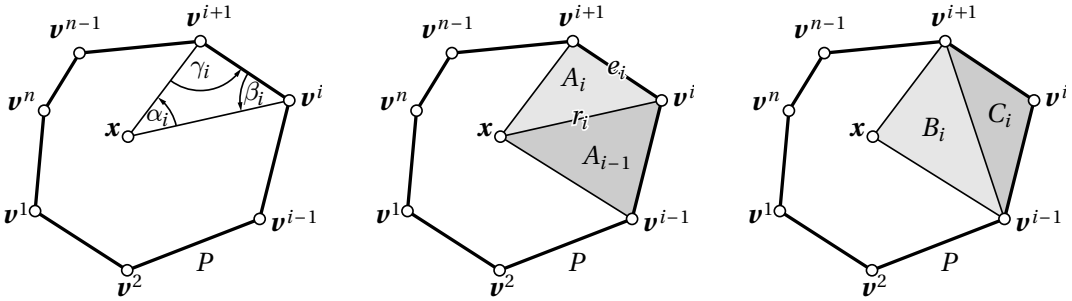


Figure 1.1. Notations for Wachspress, discrete harmonic, mean value coordinates, first figure. Last two figures illustrate the notation for three point family.

interpolates the function on the interior (1.2c). Moreover, if h_i is sampled from a linear function $\tilde{h}(\boldsymbol{x})$, then $h(\boldsymbol{x}) = \tilde{h}(\boldsymbol{x})$, by (1.2b).

The lack of uniqueness of generalized barycentric coordinates, drove researchers in the past few years to come up with many different definitions, striving for different properties. For instance, positivity is important for interpolation, as it guarantees that the interpolated function remains in the convex hull of the input data, that is

$$\max_{\boldsymbol{x}\in\bar{\Omega}} h(\boldsymbol{x}) \leq \max_{i=1,\dots,n} h_i \qquad \text{and} \qquad \min_{\boldsymbol{x}\in\bar{\Omega}} h(\boldsymbol{x}) \geq \min_{i=1,\dots,n} h_i.$$

For applications requiring efficient evaluation, the existence of a simple closed form is a fundamental requirement. Finally, a certain degree of smoothness (*e.g.*, C^1) is required when barycentric coordinates are used in optimization procedures.

Following a similar approach as in triangular coordinates (1.1), a common way to define barycentric coordinates consists of first defining a weight function w_i that satisfies (1.2b) and (1.2c). Then let

$$\phi_i(\boldsymbol{x}) = \frac{w_i(\boldsymbol{x})}{W(\boldsymbol{x})}, \quad \text{with} \quad W(\boldsymbol{x}) = \sum_{j=1}^{n} w_j(\boldsymbol{x}), \qquad i = 1,\dots,n. \tag{1.4}$$

One of the first two-dimensional generalizations of barycentric coordinates to arbitrary convex polygons was provided by Wachspress [1975]

$$w_i = \frac{\cot\gamma_{i-1} + \cot\beta_i}{r_i^2}, \qquad i = 1,\dots,n, \tag{1.5}$$

see Figure 1.1 for the notation. These coordinates are C^∞ and there is a simple formula to evaluate them. In the 90s Dziuk [1990], Eck et al. [1995], and Pinkall

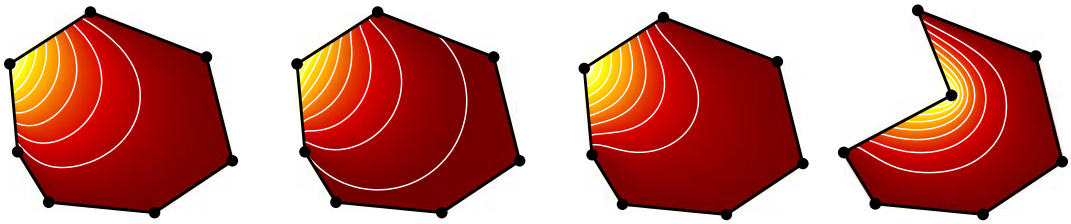


Figure 1.2. Visualization of Wachspress, discrete harmonic, and mean value coordinates, for a simple polygon.

and Polthier [1993] defined *discrete harmonic coordinates* as

$$w_i = \cot\beta_{i-1} + \cot\gamma_i, \qquad i = 1,\dots,n \tag{1.6}$$

which are based on the discretization of the Laplace operator. Note that these coordinates seems to have first appeared in Duffin [1959]. Similarly to Wachspress coordinates they are well defined (*i.e.*,they do not contain poles) only for convex polygons. Ten years later Floater [2003] discovered *mean value coordinates*

$$w_i = \frac{\tan(\alpha_{i-1}/2) + \tan(\alpha_i/2)}{r_i}, \qquad i = 1,\dots,n \tag{1.7}$$

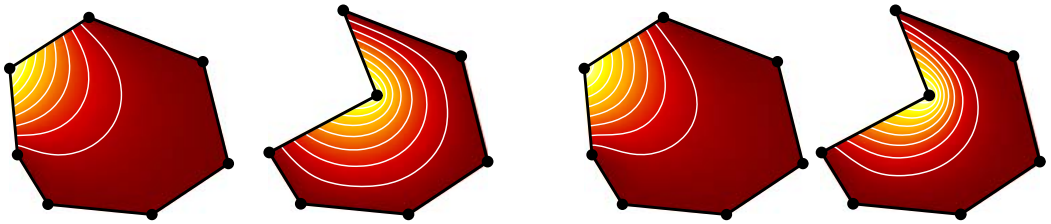
which are well defined for any polygon and for any point in the plane [Hormann and Floater, 2006]. Figure 1.2 shows an example of these three coordinates for the first vertex.

In 2006 a general recipe for constructing barycentric coordinates called *three point family* was found [Floater et al., 2006]

$$w_i^p = \frac{r_{i+1}^p A_{i-1} - r_i^p B_i + r_{i-1}^p A_i}{A_{i-1}A_i}. \tag{1.8}$$

It turns out that the aforementioned coordinates fall in this generalization, that is, $p = 0$ leads to Wachspress, $p = 1$ to mean value and $p = 2$ to harmonic coordinates.

These three main coordinates have also been extended to three dimensions. For Wachspress coordinates Wachspress [1975], Warren [1996]; Ju, Schaefer, Warren and Desbrun [2005], and Warren et al. [2007] provided the first definition. Note that the constructions in [Warren, 1996; Warren et al., 2007] also extend to higher dimensions. For discrete harmonics Meyer et al. [2003] introduced their formula for three-dimensional shapes. Finally, 3D mean value



harmonic maximum entropy

Figure 1.3. Visualization of harmonic and maximum entropy coordinates for a simple polygon.

coordinates are based on spherical triangles [Floater et al., 2005; Ju, Schaefer and Warren, 2005; Langer et al., 2006].

The aforementioned coordinates have either the disadvantage of being well defined only for a particular set of shapes or becoming negative inside the domain. One possibility to overcome these problems is to renounce the closed-form formula and to define coordinates based on an optimization procedure, at the price of making their evaluation and construction more costly. The first example of such coordinates is *harmonic coordinates* [Joshi et al., 2007], which are the solution of the Laplace equation

$$\Delta \phi_i = 0, \tag{1.9}$$

subject to suitable Dirichlet boundary conditions. That is, $\phi_i(\boldsymbol{v}^i) = 1$, the Lagrange property on the vertices and linear on the edges.

A second case of computational coordinates is maximum entropy coordinates [Hormann and Sukumar, 2008]. It requires to maximize an entropy function by enforcing the coordinates to be positive

$$\max_{\phi_1(\boldsymbol{x}),\dots,\phi_n(\boldsymbol{x})\in\mathbb{R}_+} -\sum_{i=1}^{n} \phi_i(\boldsymbol{x}) \ln \frac{\phi_i(\boldsymbol{x})}{\omega_i(\boldsymbol{x})} \qquad \forall \boldsymbol{x} \in \Omega, \tag{1.10}$$

where $\omega_i : \Omega \to \mathbb{R}_+$ is a prior estimate. Note that both of these coordinates can be naturally extended to higher dimensions. Figure 1.3 shows an example of these coordinates for a simple polygon.

All of these, as well as the Poisson coordinates [Li and Hu, 2013], positive mean value [Lipman et al., 2007], and Green coordinates [Lipman et al., 2008], are piece-wise linear along $\partial\Omega$, but giving up this property can be beneficial if conformal mappings are preferred [Weber et al., 2009].

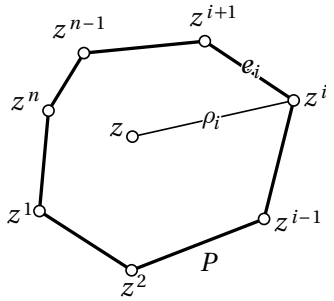


Figure 1.4. Notations for complex barycentric coordinates.

1.2 Complex barycentric coordinates

If we interpret the two-dimensional plane as the complex plane we can define complex barycentric coordinates. Let the coordinates of $\boldsymbol{v}^i = (x^i, y^i)$, then we have the complex polygon P with n vertices $z^i = x^i + \mathrm{i}y^i \in \mathbb{C}$, $i = 1, \ldots, n$. With this interpretation we define the generalized complex barycentric coordinates [Weber et al., 2009] as complex functions $\phi_i : \mathbb{C} \to \mathbb{C}$ that satisfy

$$\text{1) Partition of unity:} \qquad \sum_{i=1}^{n} \phi_i(z) = 1, \qquad z \in \bar{\Omega}, \tag{1.11a}$$

$$\text{2) Barycentric property:} \qquad \sum_{i=1}^{n} \phi_i(z) z^i = z, \qquad z \in \bar{\Omega}. \tag{1.11b}$$

Notice that these two properties are the complex equivalent of (1.2a) and (1.2b). Similarly to the real case, these coordinates can be used to interpolate values $h_i \in \mathbb{C}$, $i = 1, \ldots, n$ defined at the vertices by the function

$$h : \Omega \subset \mathbb{C} \to \mathbb{C} \qquad h(z) = \sum_{i=1}^{n} \phi_i(z) h_i.$$

Following the same idea as in the real case, for defining complex barycentric coordinates we first introduce the weight functions

$$w_i : \Omega \subset \mathbb{C} \to \mathbb{C} \qquad i = 1, \ldots, n,$$

that satisfy property (1.11b). Then, to ensure partition of unity (1.11a) let

$$\phi_i : \Omega \subset \mathbb{C} \to \mathbb{C} \qquad \phi_i(z) = \frac{w_i(z)}{\sum_{j=1}^{n} w_j(z)}, \qquad i = 1, \ldots, n.$$

It turns out that there is a simple recipe for creating such weight functions. For any set of complex functions $\gamma_i : \Omega \to \mathbb{C}$, $i = 1, \ldots, n$ the weight functions

$$w_i(z) = \gamma_i \frac{\rho_{i+1}}{e_i} - \gamma_{i-1} \frac{\rho_{i-1}}{e_{i-1}}, \qquad i = 1, \ldots, n$$

satisfy (1.11b), see Figure 1.4 for the notation.

An interesting aspect of complex coordinates [Weber et al., 2011] is that the whole three point family can be expressed in complex settings by letting

$$\gamma_i = \frac{e_i}{\Im(\bar{\rho}_i \rho_{i+1})} \left(\frac{r_{i+1}^p}{\rho_{i+1}} - \frac{r_i^p}{\rho_i} \right) = \frac{e_i}{\Im(\bar{\rho}_i \rho_{i+1})} \left(\frac{|\rho_{i+1}|^p}{\rho_{i+1}} - \frac{|\rho_i|^p}{\rho_i} \right).$$

By using this notation [Weber et al., 2011], we can rewrite complex barycentric interpolation as

$$h(z) = \frac{\sum_{j=1}^n \gamma_j(z) s_j(z)}{\sum_{j=1}^n \gamma_j(z)}, \qquad \text{with} \qquad s_j(z) = \frac{\rho_{j+1}(z) h_j - \rho_j(z) h_{j+1}}{e_j}. \tag{1.12}$$

This reformulation allows us to change the "parametrization" of h from γ_i to the set of monotonic polynomial "speed" functions

$$\sigma_i : [0, 1] \to \mathbb{R} \qquad \sigma_i(0) = 0 \qquad \text{and} \qquad \sigma_i(1) = 1, \tag{1.13}$$

with

$$s_i = h_i + (h_{i+1} - h_i)\big(\sigma_i(\Re(-\rho_i/e_i)) + \Im(-\rho_i/e_i)\mathrm{i}\big)$$

and

$$\gamma_i = \frac{e_j}{\Im(\bar{\rho}_i \rho_{i+1})} \left(\frac{|\rho_{j+1}|}{\rho_{j+1}} - \frac{|\rho_j|}{\rho_j} \right)$$

This reformulation provides a convenient way to construct bijective mappings (Chapter 5).

1.3 Transfinite barycentric coordinates

In this section we consider the continuous counterparts of generalized barycentric coordinates. As shown in (1.3), generalized barycentric coordinates can be used to interpolate function values defined at the vertices of a polygon or a polyhedron. Transfinite barycentric coordinates are the equivalent of discrete coordinates for interpolation over smooth domains defined by closed planar curves. Similarly to the discrete case, let us consider the curve $\gamma : I = [a, b] \subset \mathbb{R} \to \mathbb{R}^2$

which encloses the domain Ω. A smooth function $\phi(\boldsymbol{x},s)$, $\boldsymbol{x} \in \Omega$ and $s \in I$ that satisfies the two properties

$$\text{1) Partition of unity:} \qquad \int_{\partial\Omega} \phi(\boldsymbol{x},s)ds = 1, \qquad \boldsymbol{x} \in \bar{\Omega}, \tag{1.14a}$$

$$\text{2) Barycentric property:} \qquad \int_{\partial\Omega} s\phi(\boldsymbol{x},s)ds = \boldsymbol{x}, \qquad \boldsymbol{x} \in \bar{\Omega}, \tag{1.14b}$$

is called transfinite barycentric coordinate. Note that these two conditions are the continuous analogue of (1.2a) and (1.2b). Following this analogy, we define the transfinite interpolant. For any $\bar{h}\colon \partial\Omega \to \mathbb{R}$, the function

$$h\colon \Omega \subset \mathbb{R}^2 \to \mathbb{R} \qquad h(\boldsymbol{x}) = \int_{\partial\Omega} \bar{h}(s)\phi(\boldsymbol{x},s)ds,$$

interpolates the values of $\bar{h}$ to any point in the domain. An analogous definition of the three main barycentric coordinates exists for transfinite settings [Belyaev, 2006] as well as for the three point family [Schaefer et al., 2007].

Chapter 2

Bijective mappings

2.1 Barycentric mappings

A *barycentric mapping* between a *source polygon* $\bar{\Omega}_0 \subset \mathbb{R}^2$ with $n \geq 3$ *source vertices* $\boldsymbol{v}_0^i$, $i = 1, \ldots, n$, ordered anticlockwise, and a *target polygon* $\bar{\Omega}_1 \subset \mathbb{R}^2$ with the same number of *target vertices* $\boldsymbol{v}_1^i$, is a mapping

$$\boldsymbol{f} : \bar{\Omega}_0 \to \bar{\Omega}_1, \qquad \boldsymbol{f}(\boldsymbol{x}) = \sum_{i=1}^{n} \phi_i(\boldsymbol{x}) \boldsymbol{v}_1^i, \tag{2.1}$$

where the functions $\phi_i : \bar{\Omega}_0 \to \mathbb{R}$, $i = 1, \ldots, n$ are *barycentric coordinates* with respect to $\bar{\Omega}_0$ (Section 1.1). Because $\boldsymbol{f}$ depends on the particular choice of coordinates we denote the mapping $\boldsymbol{f}$ with the name of the coordinate generating it. For instance, a *Wachspress mapping* is a barycentric mapping based on Wachspress coordinates.

Because of the Lagrange property (1.2c) it is clear that $\boldsymbol{f}(\boldsymbol{v}_0^i) = \boldsymbol{v}_1^i$. Moreover,

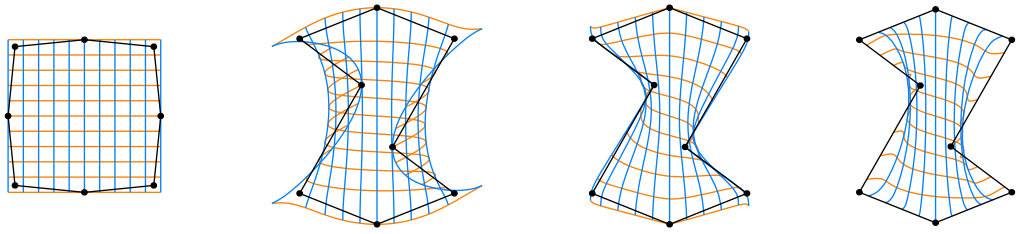


Figure 2.1. Example of a barycentric mapping based on different barycentric coordinates. For the discrete-harmonic mapping we only show the image of the interior of $\bar{\Omega}_0$.

if a point $\boldsymbol{x} = (1-\mu)\boldsymbol{v}_0^i + \mu\boldsymbol{v}_0^{i+1}$ lies on the edge $[\boldsymbol{v}_0^i, \boldsymbol{v}_0^{i+1}]$ the only non-zero coordinates are $\phi_i(\boldsymbol{x}) = 1-\mu$ and $\phi_{i+1}(\boldsymbol{x}) = \mu$, if the coordinates are linear along the edges of the polygon. Hence the mapping is $\boldsymbol{f}(\boldsymbol{x}) = (1-\mu)\boldsymbol{v}_1^i + \mu\boldsymbol{v}_1^{i+1}$, which means that it is also linear along the edges. Finally, the mapping $\boldsymbol{f}$ is always surjective, because the coordinates ϕ_i are continuous, which implies that $\boldsymbol{f}$ is continuous on $\bar{\Omega}_0$, and edges are mapped to edges.

Figure 2.1 shows an example of a barycentric mapping for different coordinates. We see that the mapping is linear along the edges, regardless of the choice of coordinates, and that it is not bijective, that is, the grid folds over in concave regions.

Barycentric mappings [Hormann and Floater, 2006; Weber et al., 2011] naturally provide smooth solutions with piecewise linear boundary behaviour, but they are not necessarily bijective. This limitation can be overcome by restricting the set of possible domains. Wachspress mappings are known to be bijective as long as both the source and the target polygon are convex [Floater and Kosinka, 2010], and barycentric mappings based on harmonic coordinates [Joshi et al., 2007] are bijective as long as the target polygon is convex. The latter is not surprising, once we notice that they are in fact *harmonic maps* and hence bijective as a consequence of the Radó–Kneser–Choquet theorem [Choquet, 1945; Kneser, 1926; Radó, 1926].

Image warping has first been mentioned as an application of barycentric mappings in the context of mean value [Hormann and Floater, 2006] and transfinite Wachspress coordinates [Warren et al., 2007], but neither these nor any of the follow-up work [Jacobson et al., 2011; Manson and Schaefer, 2010; Weber and Gotsman, 2010] is guaranteed to avoid fold-overs in the warped image. Even ad-hoc methods based on radial basis functions [Arad et al., 1994] fail to guarantee bijectivity. Modifying the interpolation speed along the boundary of Ω has the potential to overcome this problem [Weber et al., 2011], but it is unclear if this approach works for arbitrary source and target polygons.

2.1.1 Convex polygons

So far, bijectivity can only be guaranteed for the special case of Wachspress mappings between convex polygons [Floater and Kosinka, 2010] or by triangulating Ω and computing a piecewise linear mapping which preserves the orientation of each triangle (Section 2.2), but this solution comes at the price of giving up smoothness.

Figure 2.2 shows an extreme example of a barycentric mapping between two convex polygons. In the close-up we see that the mean value mapping is not

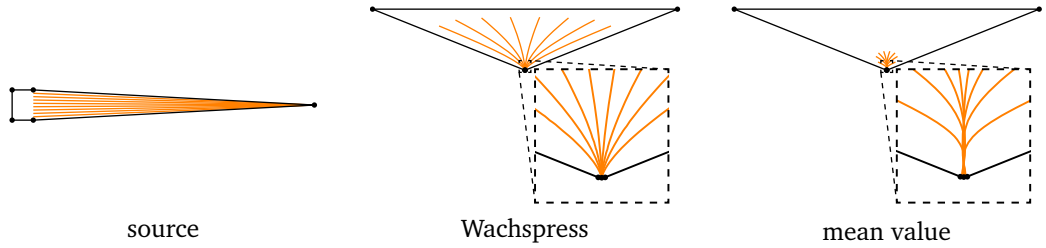


Figure 2.2. Example of a convex source and a convex target polygon with 5 vertices, taken from [Floater and Kosinka, 2010], for which the mean value mapping (right) is not bijective, whereas the Wachspress (middle) is.

bijective, whereas the Wachspress mapping is.

2.1.2 Arbitrary polygons

For any choice of barycentric coordinates, it is possible to construct a source and a target polygon such that the barycentric mapping is not bijective [Jacobson, 2012]. In order to construct such a counterexample, let us consider the barycentric mapping between a square and a deformed square, as shown in Figure 2.3.

Because we only move $\boldsymbol{v}_0^3$, we know that $\boldsymbol{v}_1^i - \boldsymbol{v}_0^i = 0$ for $i = 1, 2, 4$, and we can rewrite the mapping $\boldsymbol{f}$, evaluated at the origin, as

$$\boldsymbol{f}(\mathbf{0}) = \sum_{i=1}^{4} \phi_i(\mathbf{0})(\boldsymbol{v}_1^i - \boldsymbol{v}_0^i + \boldsymbol{v}_0^i) = \sum_{i=1}^{4} \phi_i(\mathbf{0})(\boldsymbol{v}_1^i - \boldsymbol{v}_0^i) = \phi_3(\mathbf{0})(\boldsymbol{v}_1^3 - \boldsymbol{v}_0^3) = \phi_3(\mathbf{0})(\boldsymbol{v}_1^3 + \boldsymbol{v}_0^1),$$

where we exploit the fact that $\boldsymbol{v}_0^3 = -\boldsymbol{v}_0^1$ in the last step.

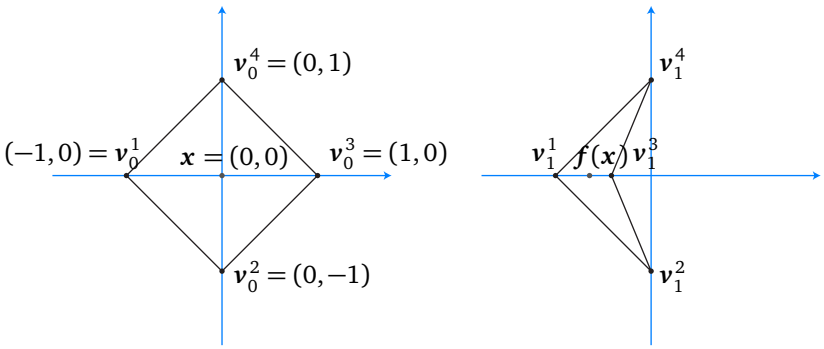


Figure 2.3. Source and target polygon for the construction of a non-bijective barycentric mapping.

We first assume that $\phi_3(\mathbf{0}) > 0.5$ and show that there exists a choice of $\boldsymbol{v}_1^3$ such that $\boldsymbol{f}(\mathbf{0}) = \boldsymbol{v}_1^1$, which contradicts the bijectivity of the mapping. To this end, observe that

$$x = \frac{\phi_3(\mathbf{0}) - 1}{\phi_3(\mathbf{0})} > -1$$

by assumption and choose $\boldsymbol{v}_1^3 = (x, 0) = -x\boldsymbol{v}_0^1$. With this choice,

$$\boldsymbol{f}(\mathbf{0}) = \phi_3(\mathbf{0})(1-x)\boldsymbol{v}_0^1 = \boldsymbol{v}_1^1.$$

Assuming next that $\phi_3(\mathbf{0}) < 0.5$, we conclude with similar considerations that $\boldsymbol{f}(\mathbf{0}) = \boldsymbol{v}_1^3$, again contradicting the bijectivity of $\boldsymbol{f}$.

Finally, consider the case when $\phi_3(\mathbf{0}) = 0.5$. Rotating the source polygon by 90 degrees, keeping in mind that $\boldsymbol{f}$ is invariant under rotations, and repeating the same reasoning as in the previous two cases, we conclude that $\phi_i(\mathbf{0}) = 0.5$ for all i, which contradicts the partition of unity property.

2.2 Optimization

To overcome this limitation, one possibility consists of renouncing to the use of barycentric mappings and considering the more general problem of mapping between polygons or polyhedra. In these settings, a common approach consists of discretizing the polygon Ω with a triangulation and formulating an optimization process that prevents the triangles from flipping. For instance, Alexa et al. [2000] propose to first interpolate all triangles in the triangulation by decomposing the morphing into rotation and scale-shear. Then they propose a global optimization problem that "glues" all triangles together. Fu and Liu [2016] extended the previous method to also include the possibility to minimize a certain energy.

To simplify this problem, the concept of polygon mappings can be relaxed to positional constraints. The advantage is that the algorithm attempts to satisfy them while maintaining bijectivity. Schüller et al. [2013] and Poranne and Lipman [2014] formulate an energy that encapsulates the mapping distortion, the positional constraints, and a barrier function to prevent triangles from flipping.

Surface parametrization methods [Floater and Hormann, 2005; Sheffer et al., 2006] can be used to create bijective maps that are piecewise linear along the boundary by interpreting $\bar{\Omega}_0$ as a (planar) mesh and parametrizing it over the domain $\bar{\Omega}_1$. However, they require triangulating $\bar{\Omega}_0$ and provide only piecewise linear solutions [Weber and Zorin, 2014]. The same restriction holds for quasi-conformal maps [Weber et al., 2012] and variational methods that minimize the distortion of the map [Aigerman et al., 2014; Lipman, 2012].

Finally, a continuous bijective mapping can be obtained trough simplicial foliations [Campen et al., 2016]. In other words, the mesh is decomposed in 1D sub-manifolds and the mapping consists of parametrizing them. In contrast with the “mesh-based” approaches, this method constructs “a bijection” without providing flexibility for optimization.

Chapter 3

Shape interpolation

Shape blending or *shape morphing* is a very active research field in computer graphics, which deals with the mathematical theory and the algorithms for constructing a gradual and continuous transformation between two planar or solid shapes. The problem is typically divided into two steps: the *vertex correspondence problem*, which establishes a correspondence between the two shapes [Belongie et al., 2002; Liu et al., 2004; Sederberg and Greenwood, 1992; Xu et al., 2009; Zhang, 1996], and the *vertex path problem*, which actually determines the interpolated shape.

3.1 Polygons

Given two planar polygons $P_0 \subset \mathbb{R}^2$ and $P_1 \subset \mathbb{R}^2$, the problem of blending between these two polygons consists of finding for any $t \in [0,1]$ a polygon $P_t \subset \mathbb{R}^2$ such that the mapping $t \to P_t$ is at least continuous in t and reproduces P_0 for $t = 0$ and P_1 for $t = 1$. One can then interpolate continuously between P_0 and P_1 by varying the parameter t.

The simplest approach consists of linearly interpolating the vertices from source to target. This method has two main disadvantages: first it depends on the global position of the two polygons; second, it is likely to produce self-intersecting polygons. Instead, the intermediate polygons can be computed by linearly interpolating the turning angle and the edge lengths [Sederberg et al., 1993], which generates more natural results. Moreover, this method is intrinsic, since edges and angles are invariant with respect to similarity transformations.

An alternative approach consists of creating a multi-resolution representation of the two input curves (or polygons) and to interpolate between these multi-resolution representations to create the intermediate shape [Goldstein and Gots-

man, 1995].

Other methods propose to formulate the problem as a shape interpolation by either using skeletons [Michal and Ari, 1995; Mortara and Spagnuolo, 2001] or triangular meshes [Alexa et al., 2000]. The main advantage of this reformulation is that the whole body is considered in the algorithm, which makes it is really hard to produce self-intersecting intermediate polygons.

Unfortunately, none of these methods guarantees that the intermediate shapes are intersection-free. To overcome this limitation in two dimensions, initial results deal with pairs of polygons that have corresponding parallel edges [Guibas et al., 1999] or by interpolating simple polylines [Alon et al., 2001].

A more general method embeds the two polygons inside a convex region, generates a pair of compatible triangulations and interpolates the stochastic matrices whose eigenvectors encode the geometry [Gotsman and Surazhsky, 2001].

An alternative idea is to create the intermediate polygons by "unfolding" the source polygon and "refolding" it back to the target polygon [Iben et al., 2009].

3.2 Polyhedra

The problem is the same in three dimensions. However, instead of considering a pair of polygons, we consider a pair of polyhedra. In other words, let $\mathcal{P}_0 \subset \mathbb{R}^3$ and $\mathcal{P}_1 \subset \mathbb{R}^3$ source and target polyhedra, we seek for a mapping $t \in [0,1] \rightarrow \mathcal{P}_t \subset \mathbb{R}^3$ that is at least continuous in t and reproduces $\mathcal{P}_0$ and $\mathcal{P}_1$ for $t = 0$ and $t = 1$ respectively.

By combining two-dimensional ideas in 3D, we first interpolate edges and dihedral angles, and then we follow a hierarchical shape-matching approach, to derive a globally coherent solution [Winkler et al., 2010]. An alternative and simpler approach involves decomposing the global interpolation problem into local affine transformations and splitting them into a rotational part and a scale-shear part [Sumner and Popović, 2004; Sumner et al., 2005]. While the scale-shear part can then be interpolated linearly, the rotational part should be treated in log-space. However, the problem is that this method cannot properly handle large global rotations.

3.3 Curves

Given two planar parametric curves $\gamma_0 : I_0 \rightarrow \mathbb{R}^2$ and $\gamma_1 : I_1 \rightarrow \mathbb{R}^2$, the problem of blending between these two curves is to find for any $t \in [0,1]$ a curve $\gamma_t : I_t \rightarrow \mathbb{R}^2$

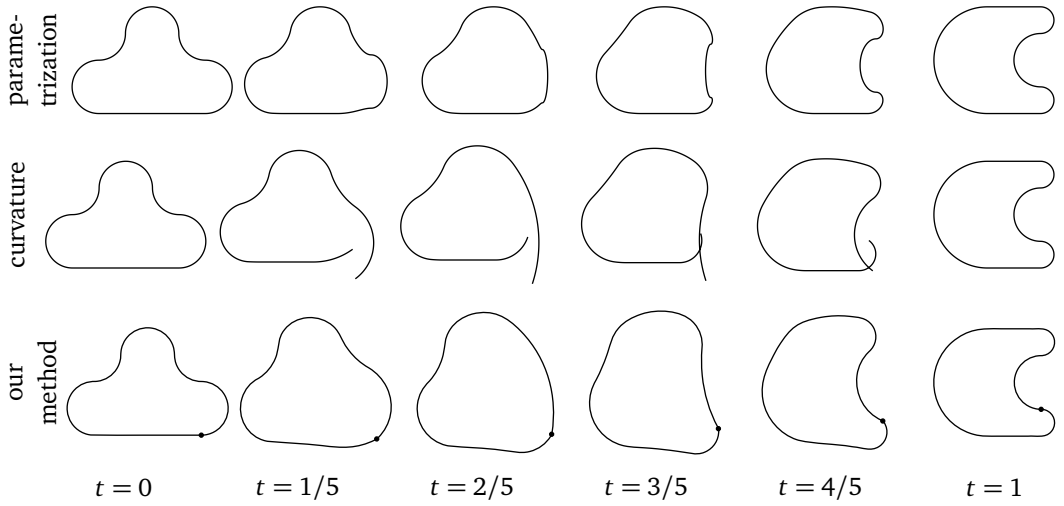


Figure 3.1. Deformation of the source curve γ_0 (left) into the target curve γ_1 (right). The top row shows the parametrization-based interpolant, the middle row the curvature-based interpolant, and the third row the results of our closing process. The dot indicates the start and end point of the curves.

such that the mapping $t \to \gamma_t$ is at least continuous in t and reproduces γ_0 for $t = 0$ and γ_1 for $t = 1$. One can then interpolate continuously between γ_0 and γ_1 by varying the parameter t.

If we assume that γ_0 and γ_1 are parametrized over a common interval $I = I_0 = I_1$, then the simplest blending is given by $\gamma_t : I \to \mathbb{R}^2$, $\gamma_t(s) = (1-t)\gamma_0(s)+t\gamma_1(s)$, but this choice is undesirable for two reasons. First, it depends on the particular parametrizations of γ_0 and γ_1 and second, it can lead to either naturally (Figure 3.1, first row) or unnaturally looking intermediate curves (Figure 3.2, first row).

A more promising approach [Surazhsky and Elber, 2002] consists of defining the intermediate curve by linearly interpolating the signed curvature functions of γ_0 and γ_1 and to reconstruct the intermediate curve γ_t from the interpolated curvature values. However, if γ_0 and γ_1 are closed, then this procedure can result in an open curve γ_t, which is again undesirable. Surazhsky and Elber [2002] fix this problem by adapting the strategy of Sederberg et al. [1993] to close γ_t in a post-processing step. Note that the idea of working in *curvature space* is also useful for computing isometric curvature flow [Crane et al., 2013].

We propose to interpolate two smooth closed curves in curvature space, using an appropriate discretization of the curvature space. Since the interpolated curves happen to be open, we introduce and use a new distance to assure that

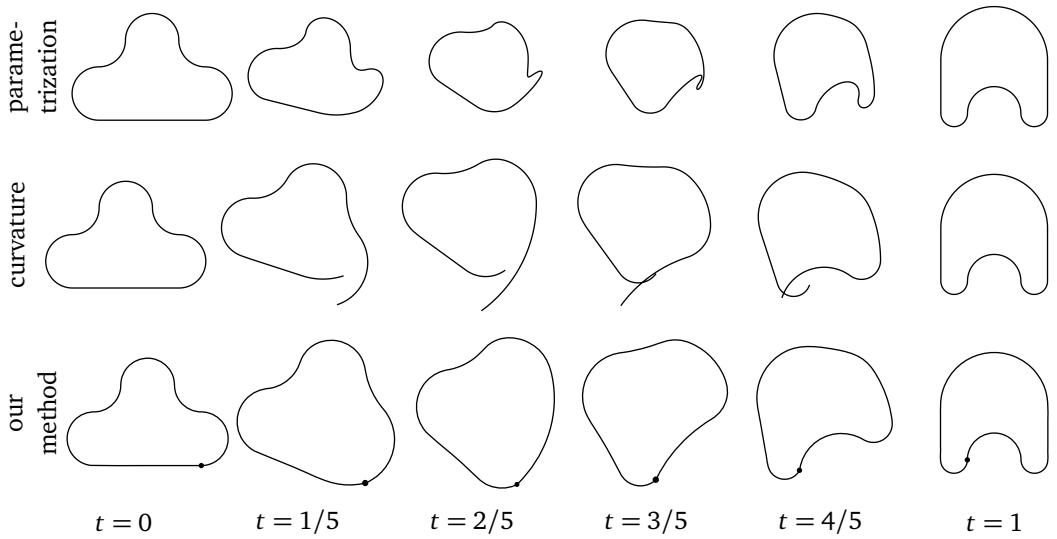


Figure 3.2. Deformation of the same input curves as in Figure 3.1, except that the target curve γ_1 is rotated clockwise by ninety degrees. The rows shows the same approaches as in Figure 3.1.

any interpolated curve can be approximated by the closest closed piecewise linear curve. We then fit a spline in a least square sense to the sampled points of the piecewise linear curve, to obtain the final smooth result. Our method lets the user choose the desired degree of approximation via three parameters: the number of samples on the curve, the degree of the fitting spline, and the number of its control points.

Let the two given curves γ_0 and γ_1 be C^2-continuous and closed, and assume without loss of generality that they are parametrized with respect to *arc length* over the intervals $I_0 = [0, L_0]$ and $I_1 = [0, L_1]$, respectively, where L_i is the *length* of the curve γ_i. That is, $\gamma_i(0) = \gamma_i(L_i)$ and $\|\gamma_i'(s)\| = 1$ for any $s \in I_i$.

It is well known [do Carmo, 1976] that for any given signed curvature function $k\colon [0, L] \to \mathbb{R}$ there exists a *unique* regular planar curve $\gamma\colon [0, L] \to \mathbb{R}^2$ up to rigid motions, such that the signed curvature of γ with respect to arc length is exactly k, and there even exists an explicit way of constructing γ. We simply let

$$\theta_s = \int_0^s k(u)du + \theta_0, \tag{3.1a}$$

and then

$$\gamma(s) = \left(\int_0^s \cos\theta_u du + x_0, \int_0^s \sin\theta_u du + y_0 \right)^T, \tag{3.1b}$$

where x_0, y_0, and θ_0 are the constants of integration which determine the aforementioned rigid motion, that is $\gamma(0) = (x_0, y_0)$ and $\gamma'(0) = (\cos\theta_0, \sin\theta_0)$.

Motivated by this property, Surazhsky and Elber [2002] propose to blend between γ_0 and γ_1 by linearly interpolating k_0 and k_1 and let γ_t be the curve that corresponds to the signed curvature function $(1-t)k_0 + tk_1$ and an appropriately chosen rigid motion. Since curvature is an *intrinsic* property of a curve, this approach has the advantage of being independent of the particular positions, orientations, and parametrizations of γ_0 and γ_1, which results in a very intuitive interpolation of their *shapes*. Unfortunately, the curve γ_t is not necessarily closed, as shown in the middle row of Figures 3.1 and 3.2.

To overcome this problem, we propose the following approach. Let $I = [0,1]$ be the unit interval and consider the set

$$\Gamma = \{\gamma : I \to \mathbb{R}^2 \mid \|\gamma'(s)\| = 1, \forall s \in I\}$$

of all planar C^2 curves with unit length, parametrized with respect to arc length, as well as the subset

$$\Gamma_\circ = \{\gamma \in \Gamma \mid \gamma(0) = \gamma(1), \gamma'(0) = \gamma'(1), \gamma''(0) = \gamma''(1)\}$$

of all closed C^2 curves.

After scaling the given curves uniformly by $1/L_0$ and $1/L_1$, respectively, we can assume without loss of generality that $\gamma_0, \gamma_1 \in \Gamma_\circ$ and $k_0, k_1 \in C[0,1]$. To account for this simplification we simply re-scale all intermediate curves $\gamma_t \in \Gamma_\circ$ uniformly by the linearly interpolated length $L_t = (1-t)L_0 + tL_1$, so as to get a smooth blend from γ_0 to γ_1.

For any $t \in [0,1]$, let

$$\bar{k}_t = (1-t)k_0 + tk_1 \ \in \ C[0,1]$$

be the linearly interpolated signed curvature function and $\bar{\gamma}_t$ be the unique curve that corresponds to $\bar{k}_t$ with

$$\begin{aligned} \bar{\gamma}_t(0) &= (1-t)\gamma_0(0) + t\gamma_1(0), \\ \bar{\gamma}_t'(0) &= (1-t)\gamma_0'(0) + t\gamma_1'(0). \end{aligned}$$

Since $\bar{\gamma}_t$ is not necessarily closed, we would like to take as intermediate curve the closed curve $\gamma_t \in \Gamma_\circ$ that is closest to $\bar{\gamma}_t$ with respect to some metric, that is

$$d(\gamma, \tilde{\gamma}) = \|k - \tilde{k}\|_2,$$

where $\|\cdot\|_2$ is the classical L^2-norm and $k, \tilde{k} \in C[0,1]$ are the signed curvature.

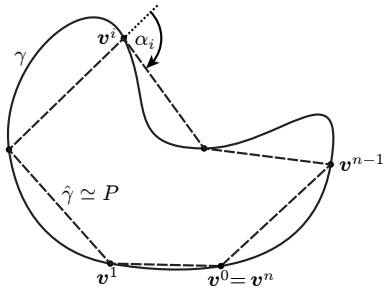


Figure 3.3. Approximation of a smooth curve γ by a polygon P, which can also be seen as a piecewise linear curve $\hat{\gamma}$. The exterior angle at vertex $\boldsymbol{v}^i$ of P is denoted by α_i.

Interestingly, this choice guarantees not only that γ is close to $\tilde{\gamma}$ with respect to curvature, but also with respect to the parametric distance

$$\bar{d}(\gamma, \tilde{\gamma}) = \max_{s \in [0,1]} \|\gamma(s) - \tilde{\gamma}(s)\|,$$

at least once they have been aligned properly as proven in [Saba et al., 2014]. The final problem consists of finding a closed curve γ_t such that

$$d(\gamma_t, \tilde{\gamma}_t) = \min_{\gamma \in \Gamma_\circ} d(\gamma, \tilde{\gamma}_t), \tag{3.2}$$

is minimal.

3.3.1 Discretization

Solving the optimization problem (3.2) in the smooth setting is difficult for two reasons. The first problem is that parametric curves are usually not given with respect to arc length and an analytic form of the signed curvature function with respect to arc length usually does not exist. The second problem is that there is no easy way to see if the curve γ which corresponds to some given signed curvature function k is closed. But minimizing over the set $\Gamma_\circ$, without having at hand a simple criterion for determining if $\gamma \in \Gamma_\circ$, is infeasible. Hence, we propose to use an approximate solution in practice.

Computing the signed curvature function

To tackle the first problem, we sample an arbitrary given curve $\gamma\colon [a, b] \to \mathbb{R}^2$ at $n+1$ uniformly distributed parameter values $u_i = a + (b-a)i/n$, giving the

polygon $P = [\boldsymbol{v}^0, \dots, \boldsymbol{v}^n]$ with points $\boldsymbol{v}^i = \gamma(u_i)$ for $i = 0, \dots, n$ (Figure 3.3). If the curve γ is closed, then so is P with $\boldsymbol{v}^0 = \boldsymbol{v}^n$.

Now consider the partition $\sigma = (s_0, \dots, s_n)$ of $[0,1]$ with $s_0 = 0$ and

$$s_i = \sum_{j=1}^{i} \|\boldsymbol{v}^j - \boldsymbol{v}^{j-1}\| \Big/ \sum_{j=1}^{n} \|\boldsymbol{v}^j - \boldsymbol{v}^{j-1}\|$$

for $i = 1, \dots, n$. The polygon P can then also be seen as the curve $\hat{\gamma}\colon [0,1] \to \mathbb{R}^2$, which is piecewise linear over σ with $\hat{\gamma}(s_i) = \boldsymbol{v}^i$ for $i = 0, \dots, n$. The curve $\hat{\gamma}$ is an arc length parametrized approximation of γ and the Hausdorff distance between γ and $\hat{\gamma}$ is of order $1/n^2$ [Floater, 2005].

We also use the sample points $\boldsymbol{v}^i$ to estimate the signed curvature of γ at $\boldsymbol{v}^i$ as

$$\kappa_i = \frac{2\alpha_i}{\|\boldsymbol{v}^i - \boldsymbol{v}^{i-1}\| + \|\boldsymbol{v}^{i+1} - \boldsymbol{v}^i\|} \tag{3.3}$$

for $i = 1, \dots, n-1$, where α_i is the signed angle between $\boldsymbol{v}^i - \boldsymbol{v}^{i-1}$ and $\boldsymbol{v}^{i+1} - \boldsymbol{v}^i$ (Figure 3.3).

For closed curves we also compute $\kappa_0 = \kappa_n$ by the same formula, using $\boldsymbol{v}^{-1} = \boldsymbol{v}^{n-1}$ and $\boldsymbol{v}^{n+1} = \boldsymbol{v}^1$. The function $\hat{k}\colon [0,1] \to \mathbb{R}$, which is piecewise linear over σ with $\hat{k}(s_i) = \kappa_i$ for $i = 0, \dots, n$ is then an approximation of the signed curvature function k of γ with respect to arc length.

The polygon P or the curve $\hat{\gamma}$ can be reconstructed up to a uniform scaling and a rigid motion from $\hat{k}$ and the parametrization σ, because the edge lengths of P are proportional to the distances $s_i - s_{i-1}$ and the turning angles α_i at the vertices of P can be recovered from the values $\hat{k}(s_i)$ using (3.3). We first specify $\boldsymbol{v}^0$ and α_0 to fix the rigid motion and then proceed in the spirit of (3.1) by defining the rotation of the edge $[\boldsymbol{v}^i, \boldsymbol{v}^{i+1}]$ as $\theta_i = \sum_{j=0}^{i} \alpha_j$ and the vertices of P as

$$\boldsymbol{v}^i = \boldsymbol{v}^{i-1} + |s_i - s_{i-1}| \begin{pmatrix} \cos\theta_{i-1} \\ \sin\theta_{i-1} \end{pmatrix}, \tag{3.4}$$

for $i = 1, \dots, n$.

Solving the optimization problem

To solve the optimization problem (3.2) we start by approximating the two given curves γ_0 and γ_1 as described above and considering the linearly interpolated signed curvature function

$$\hat{k}_t = (1-t)\hat{k}_0 + t\hat{k}_1.$$

With K denoting the space of all functions that are piecewise linear over the *joint partition* $\sigma_t = \sigma_0 \cup \sigma_1$ with nodes $s_0, \dots, s_m$, for some $m < 2n$, we have $\hat{k}_t \in K$.

We can now reconstruct a corresponding piecewise linear curve $\hat{\gamma}_t$ as explained at the end of the previous section, but as in the continuous case, this curve will usually not be closed. Therefore, our goal is to change $\hat{k}_t$ in the least possible way, so as to close the curve. That is, we want to find values $\delta_0, \dots, \delta_m \in \mathbb{R}$ with $\delta_0 = \delta_m$, such that the piecewise linear curve $\tilde{\gamma}_t$, which corresponds to $\tilde{k}_t \in K$ with $\tilde{k}_t(s_i) = \hat{k}_t(s_i) + \delta_i$ for $i = 0, \dots, m$, is closed. To solve (3.2), we thus need to minimize

$$\|\tilde{k}_t - \hat{k}_t\|_2 = \left(\sum_{i=1}^{m} {\delta_i}^2 \right)^{\frac{1}{2}}.$$

A slightly more intuitive understanding of this optimization procedure is the following. First note that the two initial piecewise linear curves $\hat{\gamma}_0$ and $\hat{\gamma}_1$, which we assume to have unit length, can also be seen as piecewise linear curves over the joint partition σ_t. For the corresponding closed polygons P_0 and P_1 this means that we have to refine them by inserting some points on the edges. For instance, if $s_i \in \sigma_t$ is a node in σ_1 but not in σ_0 then we add the point $\hat{\gamma}_0(s_i)$ to P_0 and vice versa. The refined closed polygons $\hat{P}_0 = [\boldsymbol{v}_0^0, \dots, \boldsymbol{v}_0^m]$ and $\hat{P}_1 = [\boldsymbol{v}_1^0, \dots, \boldsymbol{v}_1^m]$ then have the same number of points and matching edge lengths

$$e_i = \|\boldsymbol{v}_0^i - \boldsymbol{v}_0^{i-1}\| = \|\boldsymbol{v}_1^i - \boldsymbol{v}_1^{i-1}\|$$

for $i = 1, \dots, m$, because they share the same parametrization σ_t.

The procedure above, which keeps the parametrization, then searches for a closed intermediate polygon $\hat{P}_t = [\boldsymbol{v}_t^0, \dots, \boldsymbol{v}_t^m]$ with edge lengths $\|\boldsymbol{v}_t^i - \boldsymbol{v}_t^{i-1}\| = e_i$ for $i = 1, \dots, m$ and exterior angles which are as-close-as-possible to the linearly interpolated target values $\hat{\alpha}_t^i$ for $i = 0, \dots, m$. Note that we cannot use the angles $(1-t)\hat{\alpha}_0^i + t\hat{\alpha}_1^i$ as $\hat{\alpha}_t^i$, because about half of the exterior angles $\hat{\alpha}_0^i$ and $\hat{\alpha}_1^i$ at the vertices of $\hat{P}_0$ and $\hat{P}_1$ are zero, and this would introduce unexpected artefacts in the result. Instead, we use the linearly interpolated signed curvature function $\hat{k}_t$ and (3.3) to convert the curvature values $\hat{k}_t(s_i)$ into $\hat{\alpha}_t^i$.

In order to close the polygon we have to ensure that $\boldsymbol{v}_t^0 = \boldsymbol{v}_t^m$. One possible approach for closing the curve is to follow Sederberg et al. [1993] who keep the linearly interpolated angles fixed and modify the edge lengths to ensure that the polygon is closed. While this procedure also modifies the curvature as defined in (3.3), the big advantage is that it leads to a closed-form solution. However, in our setting we want to keep the edge lengths fixed, because we want to preserve the common arc length parametrization σ_t, and hence modify the angles.

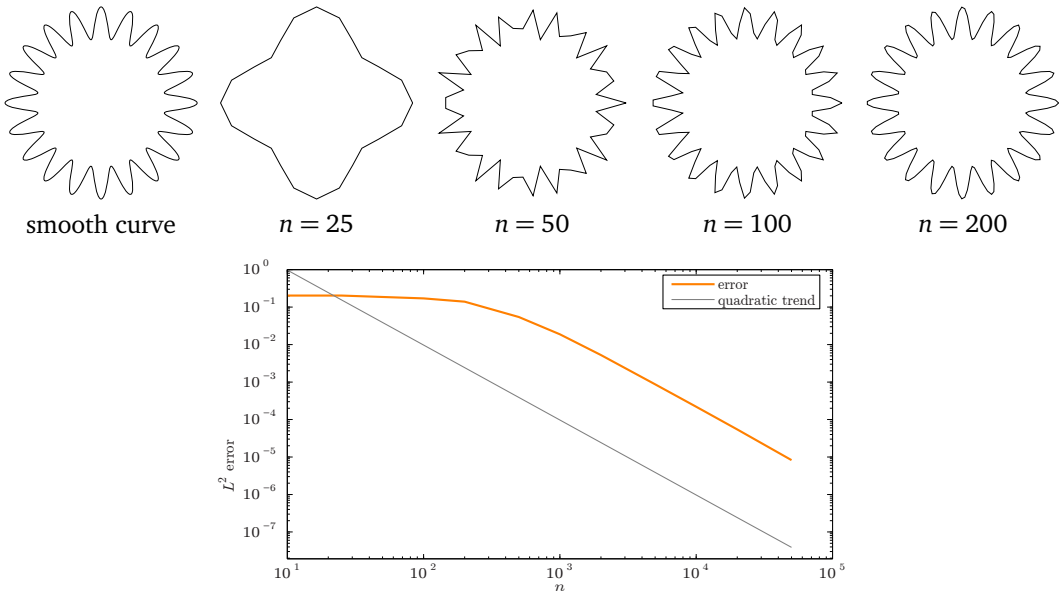


Figure 3.4. Convergence of the piecewise linear signed curvature function in the L^2-norm.

Unfortunately this leads to the non-linear optimization problem

$$\min_{\tilde{\alpha}_t} \sum_{i=0}^{n-1} \left\| \frac{2}{e_{i-1} + e_i} (\tilde{\alpha}_t^i - \hat{\alpha}_t^i) \right\|^2,$$

subject to $\boldsymbol{v}_t^0 = \boldsymbol{v}_t^m$. In order to find $\tilde{\alpha}_t$ we use the off-the-shelf solver `fmincon` in MATLAB and implement the closing constraint by constructing $\boldsymbol{v}_t^m$ with (3.4) and for the initial guess we use $\hat{\alpha}_t$.

To finally obtain a smooth closed intermediate curve, we compute $\gamma_t \in \Gamma_\circ$ by fitting a closed B-spline curve in the least squares sense to the vertices of the piecewise linear curve $\tilde{\gamma}_t$ (or rather the polygon $\hat{P}_t$). For this fitting procedure, we use the nodes of σ_t as initial parameter values, but perform several iterations of parameter optimization [Hoschek, 1988] to improve the result and to get a B-spline curve that is very close to being arc-length parametrized. Note that this would be difficult to achieve if we directly optimized the B-spline coefficients.

3.3.2 Results

In Section 3.3.1 we discuss how to approximate the signed curvature function of a smooth curve with respect to arc length by a piecewise linear function. To verify the approximation order of this approach we consider the following example.

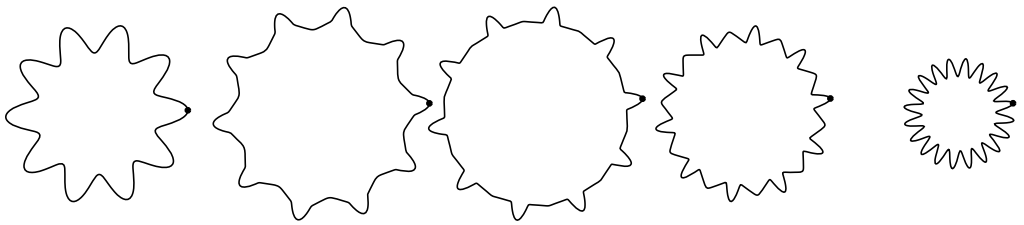


Figure 3.5. Interpolation results using initial polygons P_0 and P_1 with 500 vertices and fitting quintic B-Spline curves with 100 control points to the interpolated closed polygons $\hat{P}_t$. The dot indicates the start and end point of the curves.

We sample the smooth curve in the upper left of Figure 3.4 with n sample points, with n ranging from 10 to 50 000, construct the approximate piecewise linear signed curvature functions as described in (3.3), and compute the L^2-norm of the difference to the exact signed curvature function. Figure 3.4 shows that the error behaves like $\mathcal{O}(1/n^2)$.

Figure 3.5 shows some results of the non-linear optimization procedure described in Section 3.3.1 for interpolating between two cosine functions in polar coordinates with different periods. For this example we need initial polygons P_0 and P_1 with 500 vertices in order to accurately approximate the input curves and quintic B-splines with 100 control points to capture the details of the interpolated polygons $\hat{P}_t$ during the fitting step.

Figure 3.6 shows another example, where the input curves are arc-splines with piecewise constant signed curvature functions. Hence, the linearly interpolated target curvature is piecewise linear, too, but our method still interpolates successfully between the two curves. Figure 3.7 illustrates how the parameters of our method (that is, the number of samples used to create the initial polygons

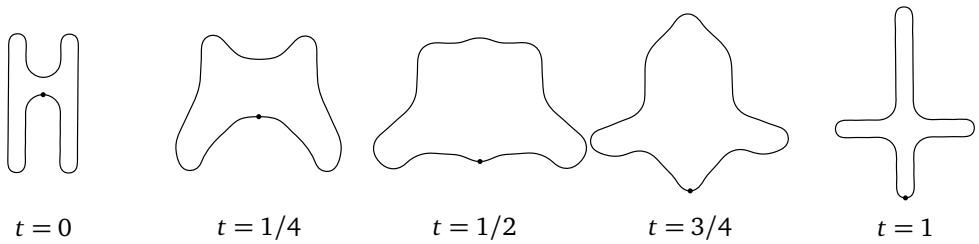


Figure 3.6. Interpolation results using initial polygons P_0 and P_1 with 500 vertices and fitting quintic B-Spline curves with 50 control points to the interpolated closed polygons $\hat{P}_t$. The dot indicates the start and end point of the curves.

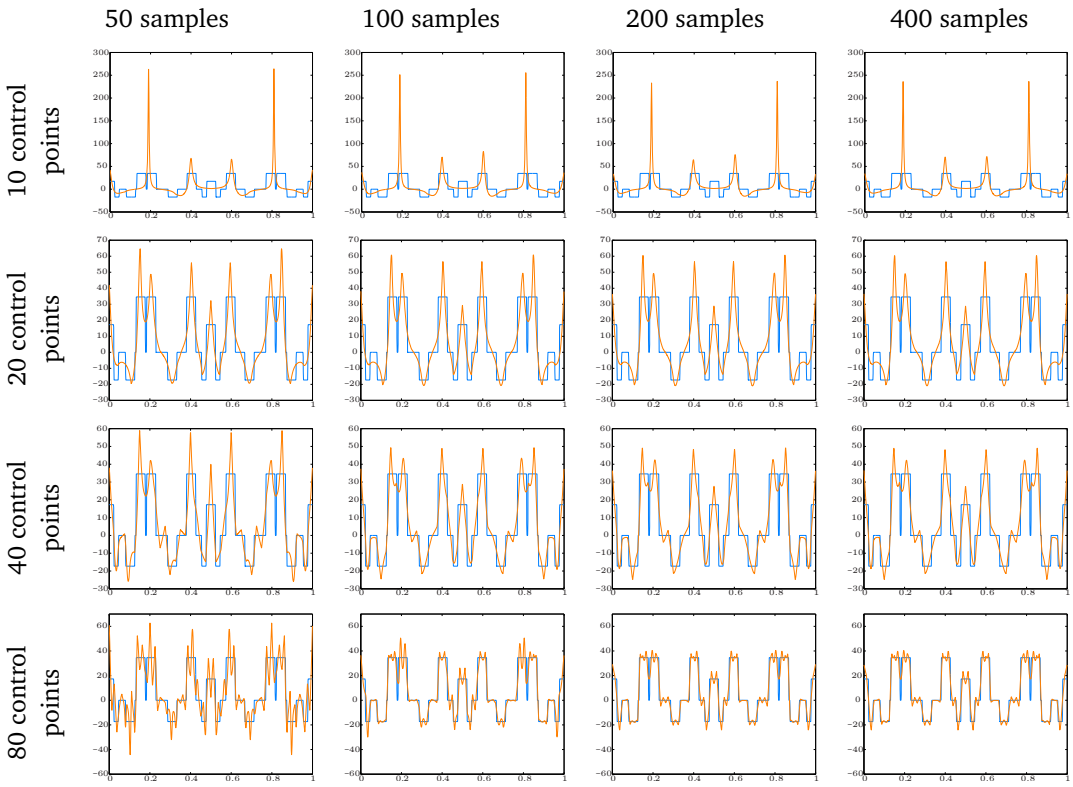


Figure 3.7. Target curvature (blue) and curvature of the interpolated closed curve for the example in Figure 3.6 at $t = 1/2$, varying the number of sample points used for creating P_0 and P_1 and the number of control points of the B-spline curve.

and the number of control points of the B-spline curve in the fitting step) influence the interpolated closed curve. We notice that increasing both the number of samples and the number of control points helps to reduce the difference between the target curvature and the curvature of the interpolated curve.

Figures 3.8 and 3.9 show a comparison of different methods for interpolating between two given B-spline curves with the same degree and the same number of control points. The simplest approach is to linearly interpolate the coordinates of the control points, with the obvious disadvantage that the result depends on the orientation of the initial curves and is not very intuitive. A much better and orientation-independent solution is obtained by using the algorithm of Sederberg et al. [1993] to interpolate between the control polygons of the initial curves and to take the resulting polygon as the control polygon of the interpolated curve. Following the idea of Surazhsky and Elber [2002], another approach is to fit a B-

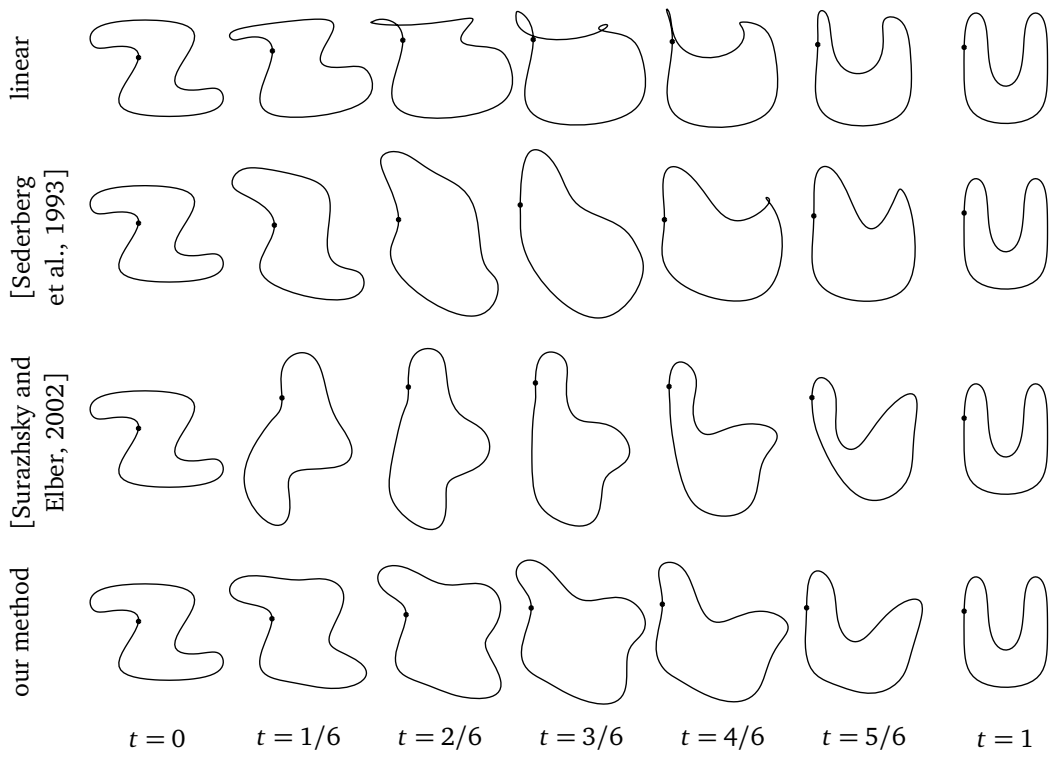


Figure 3.8. Comparison of the interpolation results by different methods. Source, target, and intermediate curves are quintic B-splines with 10 control points, and we used 100 samples for the initial polygons. The dot indicates the start and end point of the curves.

spline curve to the open intermediate curve $\bar{\gamma}_t$, which corresponds to the linearly interpolated signed curvature function, and to then apply a *closing procedure* to the control polygon to close the B-spline curve.

Even though it is not entirely clear which of the results is visually best, Figures 3.10 and 3.11 clearly show that the signed curvature functions of the closed intermediate curves generated by our method are closer to the linearly interpolated target curvature than for the other methods. Note that the L^2-error in Figure 3.11 for the curves generated by the approach of Surazhsky and Elber [Surazhsky and Elber, 2002] does not converge to 0 as t approaches 0 or 1, because the fitting step always generates an open B-spline curve, even if $\bar{\gamma}_t$ happens to be closed. Hence, the initial curves are not reproduced. Further, note that the "spikes" in the black, blue, and green plots in Figure 3.11 result from the intermediate curves having cusps, that is, containing singular points with infinite curvature.

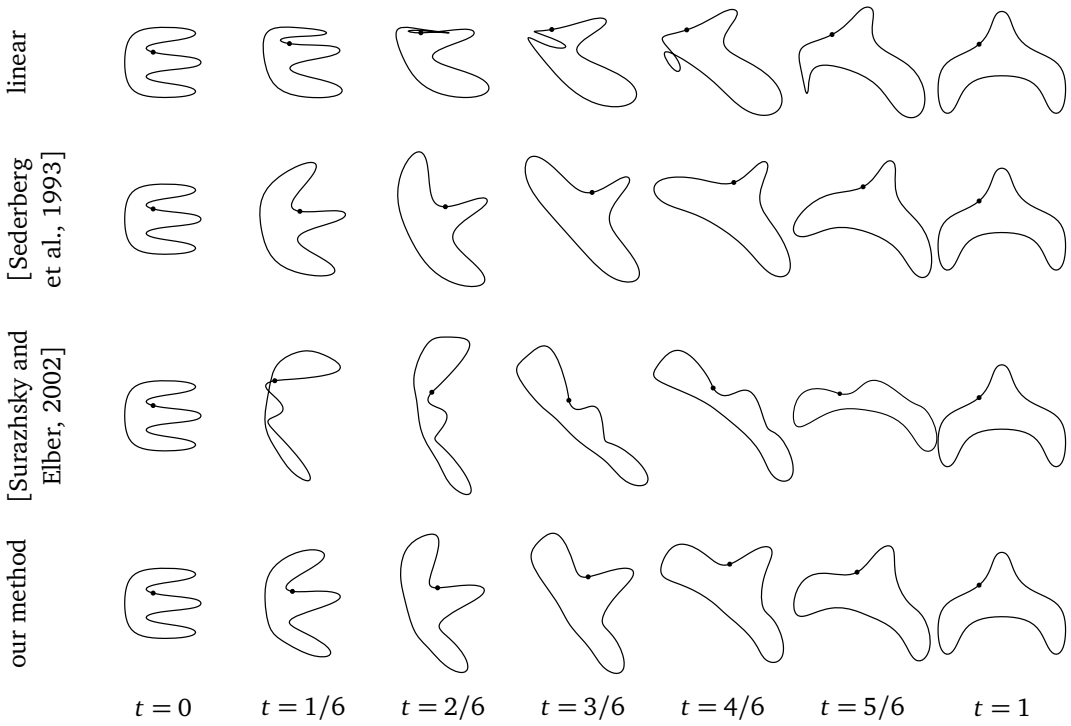


Figure 3.9. Comparison of the interpolation results by different methods. Source, target, and intermediate curves are quintic B-splines with 16 control points, and we used 200 samples for the initial polygons. The dot indicates the start and end point of the curves.

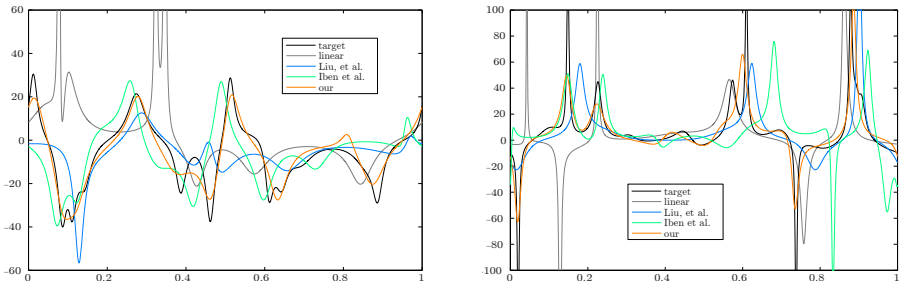


Figure 3.10. Comparison of the target curvature (black) and the curvatures of the interpolated closed curves at $t = 1/2$ for the examples in Figures 3.8 (left) and 3.9 (right).

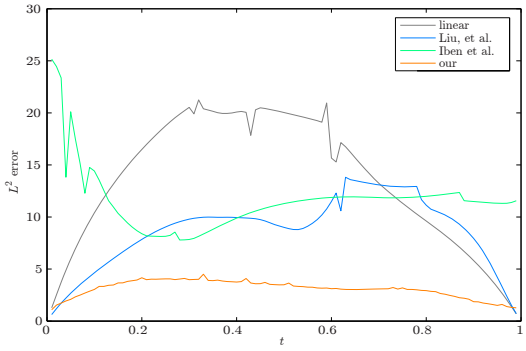


Figure 3.11. L^2-norms of the differences between target curvatures and curvatures of the interpolated curves for the example in Figure 3.8.

3.3.3 Limitations

When the two source and target curves are simple and closed, it is desirable to have all the interpolants both closed and simple. Our method always generates closed intermediate curves, but is not able to ensure the absence of self-intersections. For this it is necessary to either take intersections explicitly into account [Iben et al., 2009] or to consider the curve as the boundary of a (possibly triangulated) shape [Gotsman and Surazhsky, 2001; Chen et al., 2013].

It is hard to tell what should be the correct behaviour of a method when blending two curves with different turning numbers. This is challenging for testing our approach of wanting to be as compliant as possible with the "natural" deformation of the curves, since it involves one or more foldings or un-foldings. We experimented by blending a circle into an "8-shaped" curve. The results are reported in Figure 3.12. One can see how all methods, including ours, have a rather unpredictable behaviour.

Another limitation of our current implementation is that we cannot guarantee *temporal smoothness*, that is, smoothness of the mapping $t \to \gamma_t$, because the non-linear solver may run into a slightly different local minimum if t is changed by some small ε. It remains future work to study this behaviour in theory and to see if it can be fixed by implementing a better solver. However, a practical solution is to use our approach to compute intermediate curves for $t = k/N$, $k = 1, \ldots, N-1$ (with $N = 20$, for example) and to then use the method of Sederberg et al. [1993] for interpolating between the resulting curves.

We further believe that *shape matching* (that is, taking into account the semantics of the shapes and not only the geometry) will further improve our results, but it is beyond the scope of this chapter to properly address this issue, which

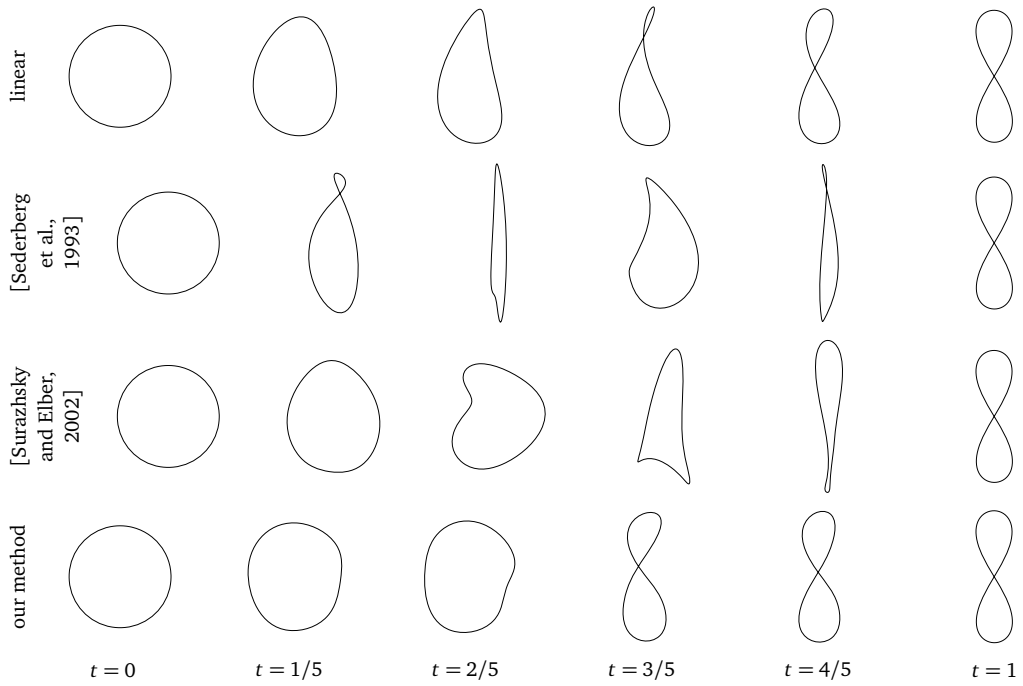


Figure 3.12. Comparison of the methods when interpolating two curves with different turning numbers. In this case, γ_0 (left) has turning number 1 and γ_1 (right) has turning number 0.

is a major problem in computer vision and pattern recognition [Belongie et al., 2002; Xu et al., 2009].

For example, we could apply our method to topologically similar shapes, using their topological skeleton, thus not being limited to examining the curve, but taking also into account its orientation (that is, what is inside or outside the curve) to define which shape it bounds. This will extend our method from curves to surfaces. If we want to blend two shapes having the same topology (that is, they have the same graphs of the skeleton), it is possible to use information contained in the skeleton [Michal and Ari, 1995] to avoid self-intersections, and, more generally speaking, to keep track of the shape while blending them.

Chapter 4

Composite barycentric mapping

Let us denote the *partial derivatives* of the barycentric mapping $\boldsymbol{f} = (f_1, f_2)$ in (2.1) at $\boldsymbol{x} = (x_1, x_2) \in \bar{\Omega}_0$ by $\partial_k \boldsymbol{f}(\boldsymbol{x}) = \partial \boldsymbol{f} / \partial x_k$, $k = 1, 2$, and the *gradients* of its two components f_i by $\nabla f_i = (\partial_1 f_i, \partial_2 f_i)$.

Since we consider only source and target polygons without self-intersections and assume that the barycentric coordinates ϕ_i are at least continuously differentiable, a sufficient condition for the injectivity of $\boldsymbol{f}$ is that its Jacobian determinant

$$J_f = \begin{vmatrix} \partial_1 f_1 & \partial_2 f_1 \\ \partial_1 f_2 & \partial_2 f_2 \end{vmatrix}$$

is strictly positive in $\bar{\Omega}_0$ [Meisters and Olech, 1963]. As it follows from (2.1) that a barycentric mapping between identical source and target polygons is the identity with $J_f(\boldsymbol{x}) = 1$ for all $\boldsymbol{x} \in \bar{\Omega}_0$, it is reasonable to expect that a small perturbation of the target vertices keeps J_f positive and the mapping bijective, as shown in Figure 4.1.

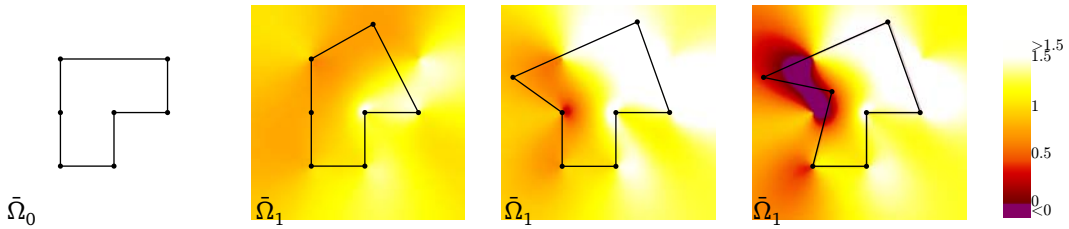


Figure 4.1. Colour-coded plots of J_f for the mean value mapping $\boldsymbol{f} : \bar{\Omega}_0 \to \bar{\Omega}_1$ for different target polygons, which remains positive when the perturbation is small, but becomes negative for large deformations.

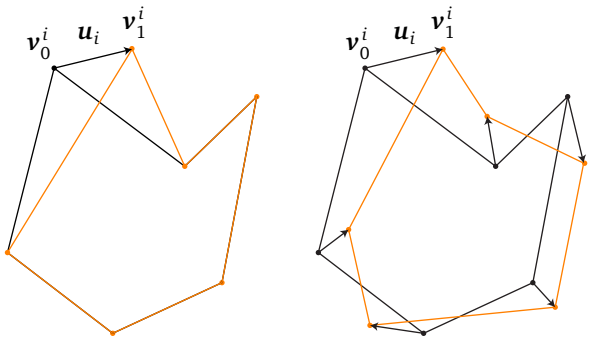


Figure 4.2. Perturbation of one vertex (left) and all vertices (right) of the target polygon.

4.1 Perturbed target polygons

Consider first a target polygon with a single perturbed vertex, as shown in Figure 4.2. Formally, the target vertices are $\boldsymbol{v}_1^i = \boldsymbol{v}_0^i + \boldsymbol{u}$ for some i and $\boldsymbol{v}_1^j = \boldsymbol{v}_0^j$ for $j \neq i$. Substituting these target vertices $\boldsymbol{v}_1^i$ in (2.1) and recalling the linear reproduction property, we obtain

$$\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{x} + \phi_i(\boldsymbol{x})\boldsymbol{u}$$

for any $\boldsymbol{x} \in \bar{\Omega}_0$ and further

$$J_f(\boldsymbol{x}) = \begin{vmatrix} 1 + \partial_1\phi_i(\boldsymbol{x})\boldsymbol{u}_1 & \partial_2\phi_i(\boldsymbol{x})\boldsymbol{u}_1 \\ \partial_1\phi_i(\boldsymbol{x})\boldsymbol{u}_2 & 1 + \partial_2\phi_i(\boldsymbol{x})\boldsymbol{u}_2 \end{vmatrix} = 1 + \nabla\phi_i(\boldsymbol{x}) \cdot \boldsymbol{u}.$$

Therefore,

$$J_f(\boldsymbol{x}) \geq 1 - |\nabla\phi_i(\boldsymbol{x}) \cdot \boldsymbol{u}| \geq 1 - \|\boldsymbol{u}\|\|\nabla\phi_i(\boldsymbol{x})\|,$$

which is strictly positive for $\|\boldsymbol{u}\| < 1/M_i$ with

$$M_i = \sup_{\boldsymbol{x} \in \bar{\Omega}_0} \|\nabla\phi_i(\boldsymbol{x})\|.$$

This result nicely extends to a perturbation of all vertices, where we consider a target polygon with vertices $\boldsymbol{v}_1^i = \boldsymbol{v}_0^i + \boldsymbol{u}_i$ for $i = 1, \ldots, n$ (Figure 4.2). Using again the linear reproduction property, we reformulate the mapping as

$$\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{x} + \sum_{i=1}^{n} \phi_i(\boldsymbol{x})\boldsymbol{u}_i$$

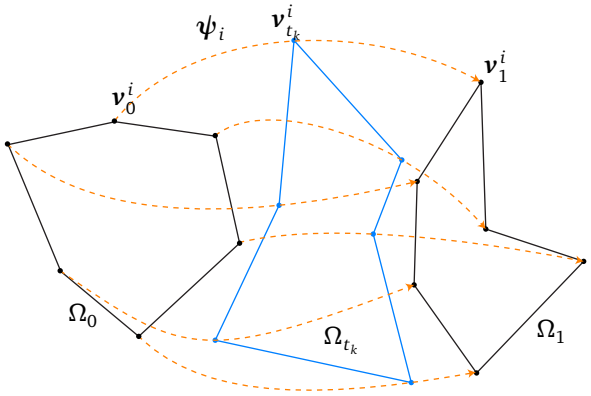


Figure 4.3. Construction of the intermediate polygon $\bar{\Omega}^{t_k}$ using the vertex paths $\boldsymbol{\psi}_i$.

for any $\boldsymbol{x} \in \bar{\Omega}_0$, hence

$$\begin{aligned} J_f(\boldsymbol{x}) &= \begin{vmatrix} 1+\sum_i \partial_1\phi_i(\boldsymbol{x})u_{i,1} & \sum_i \partial_2\phi_i(\boldsymbol{x})u_{i,1} \\ \sum_i \partial_1\phi_i(\boldsymbol{x})u_{i,2} & 1+\sum_i \partial_2\phi_i(\boldsymbol{x})u_{i,2} \end{vmatrix} \\ &= 1+\sum_i \nabla\phi_i(\boldsymbol{x})\cdot\boldsymbol{u}_i + \sum_i\sum_j \partial_1\phi_i(\boldsymbol{x})\partial_2\phi_j(\boldsymbol{x})(\boldsymbol{u}_i\times\boldsymbol{u}_j), \end{aligned}$$

where $\boldsymbol{u}_i = (u_{i,1}, u_{i,2})$ and the sums range from 1 to n. Therefore,

$$J_f(\boldsymbol{x}) \geq 1 - Md - M^2d^2$$

with $M = M_1 + \cdots + M_n$ and $d = \max_{1\leq i\leq n} \|\boldsymbol{u}_i\|$. Overall, this implies that the mapping $\boldsymbol{f}$ is injective if

$$d < \frac{\sqrt{5}-1}{2M}.$$

4.2 Bijective composite barycentric mapping

Section 4.1 suggests that if the source and target polygons are sufficiently close, the mapping is close to the identity and hence bijective. Therefore, by "splitting" the barycentric mapping from source to target polygon into a finite number of intermediate steps, where each step perturbs the vertices only slightly, it should be possible to obtain a bijective composite mapping.

To this end, suppose that $\boldsymbol{\psi}_i : [0,1] \to \mathbb{R}^2$, $i = 1,\ldots,n$ are a set of continuous *vertex paths* between each source vertex $\boldsymbol{v}_0^i = \boldsymbol{\psi}_i(0)$ and its corresponding target vertex $\boldsymbol{v}_1^i = \boldsymbol{\psi}_i(1)$, as shown in Figure 4.3. Let $\tau = (t_0, t_1, \ldots, t_m)$ with $t_0 = 0$,

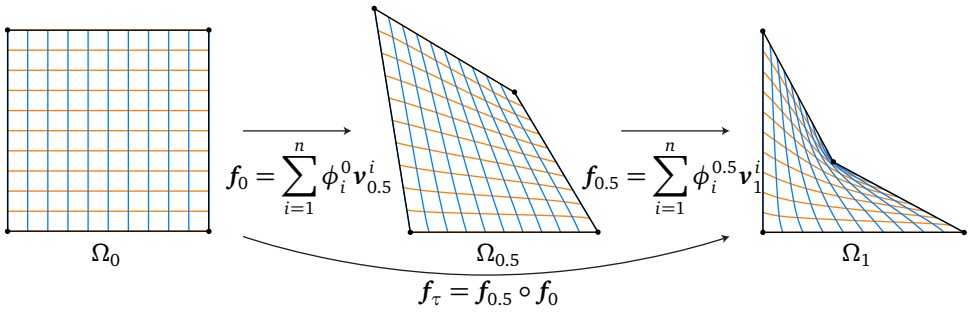


Figure 4.4. Construction of a *composite barycentric mapping* for $\tau = [0, 0.5, 1]$.

$t_m = 1$, and $t_k < t_{k+1}$ for $k = 0, \ldots, m-1$ be a *partition* of $[0,1]$ and $\boldsymbol{f}_k$ be the barycentric mapping from $\bar{\Omega}_{t_k}$ to $\bar{\Omega}_{t_{k+1}}$, based on the barycentric coordinates $\phi_i^{t_k}$. The mapping

$$\boldsymbol{f}_\tau = \boldsymbol{f}_{m-1} \circ \boldsymbol{f}_{m-2} \circ \cdots \circ \boldsymbol{f}_0$$

is called a *composite barycentric mapping* from $\bar{\Omega}_0$ to $\bar{\Omega}_1$ [Schneider et al., 2013]. An example of a composite barycentric mapping between a square and a concave quadrilateral is shown in Figure 4.4.

Denoting the maximum displacement distance between $\bar{\Omega}_{t_k}$ and $\bar{\Omega}_{t_{k+1}}$ by

$$d_k = \max_{1 \le i \le n} \left\| \boldsymbol{v}_{t_k}^i - \boldsymbol{v}_{t_{k+1}}^i \right\|,$$

it follows from the previous results that $\boldsymbol{f}_\tau$ is bijective if

$$d_\tau = \max_{0 \le k < m} d_k < \frac{\sqrt{5}-1}{2nM^*}, \tag{4.1}$$

where

$$M^* = \max_{1 \le i \le n} \sup_{t \in [0,1]} \sup_{\boldsymbol{x} \in \bar{\Omega}_t} \|\nabla \phi_i^t(\boldsymbol{x})\|.$$

For the special case of mean value mappings, this bound can provably be satisfied for mappings between any convex polygons [Floater, 2014]. Figure 4.5 shows that with enough intermediate steps the mapping $\boldsymbol{f}$ becomes bijective.

In particular, there exists some $m \in \mathbb{N}$ such that the *uniform partition* τ_m with $t_k = k/m$ gives a bijective composite barycentric mapping f_{τ_m}. Figure 4.6 shows an example of a composite mean value mapping for two nested squares, where the interior square rotated by 90 degrees in the target configuration using the uniform partition τ_{1000}.

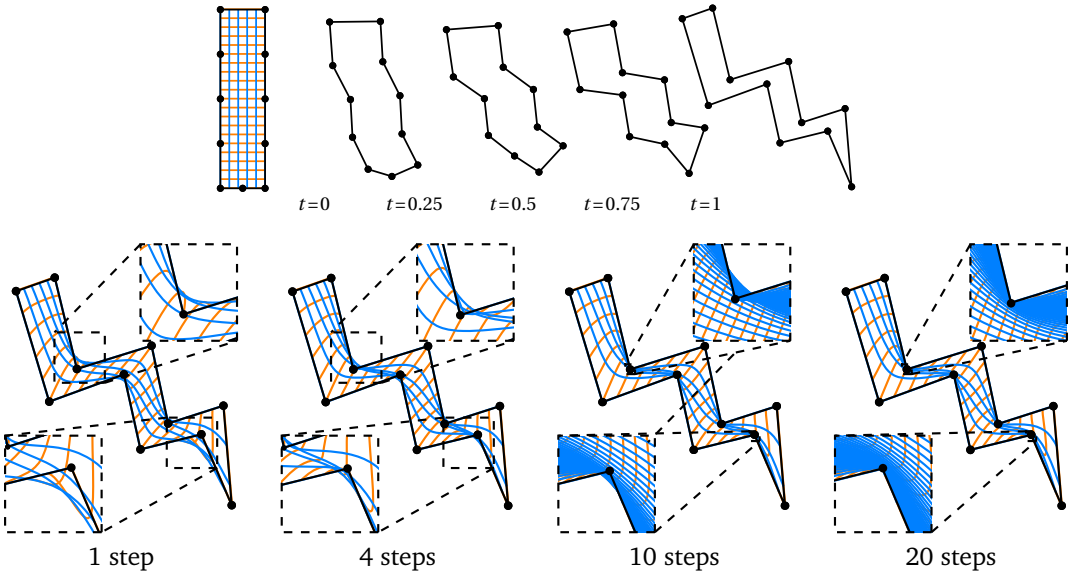


Figure 4.5. Examples of uniform composite mean value mappings for different numbers of uniform steps.

4.3 Limit of composite barycentric mappings

The idea of uniform composite barycentric mappings leads to the interesting question of the behavior in the limit. To this end, we consider the *infinite composite barycentric mapping* $\boldsymbol{f}_\infty = \lim_{m\to\infty} \boldsymbol{f}_{\tau_m}$ and its *backward mapping* $\boldsymbol{g}_\infty : \bar{\Omega}_1 \to \bar{\Omega}_0$. Figure 4.7 shows the result of mapping a star with a uniform composite mapping $\boldsymbol{f}_{\tau_m}$ composed with its backward mapping $\boldsymbol{g}_{\tau_m} : \bar{\Omega}_1 \to \bar{\Omega}_0$ with the same number of steps. Computing the maximum distance $\|\boldsymbol{x} - \boldsymbol{g}_{\tau_m}(\boldsymbol{f}_{\tau_m}(\boldsymbol{x}))\|$ for one million random points $\boldsymbol{x} \in \bar{\Omega}_0$ indicates that this distance converges to zero and so $\boldsymbol{g}_{\tau_m} = \boldsymbol{f}_{\tau_m}^{-1}$ as $m \to \infty$. Consequently, the inverse of an infinite barycentric

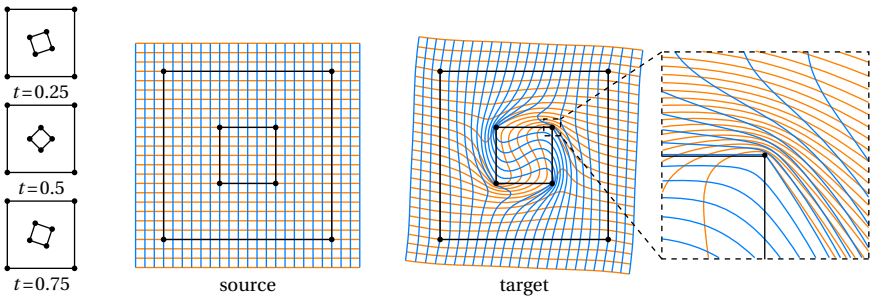


Figure 4.6. Example of a composite mean value mapping with 1000 uniform steps. The resolution of the grid is increased by a factor of four in the close-up.

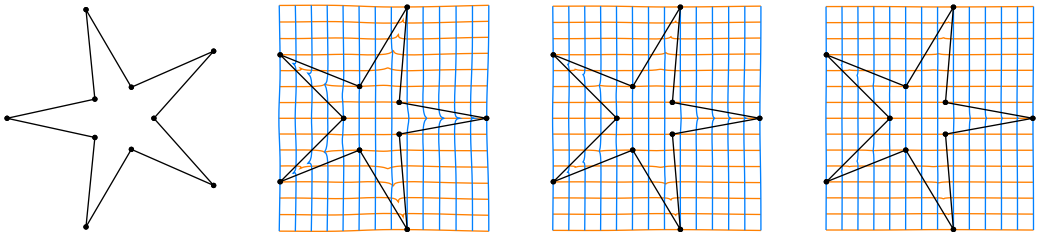


Figure 4.7. Composing the mapping $\boldsymbol{f}_{\tau_m}$ and the backward mapping $\boldsymbol{g}_{\tau_m}$ converges to the identity as the number of uniform steps m increases.

mapping is likely to be an infinite barycentric mapping itself, which is not the case for standard barycentric mappings.

This observation can be proven by considering the difference between two successive steps and taking its limit [Floater, 2015]. For any point $\boldsymbol{x}^0 \in \bar{\Omega}_0$ we evaluate the first step $\boldsymbol{f}_0$ of the composite mapping $\boldsymbol{f}$,

$$\boldsymbol{x}^{t_1} = \boldsymbol{f}_0(\boldsymbol{x}^0) = \sum_{i=1}^{n} \phi_i^{t_0}(\boldsymbol{x}^0)\boldsymbol{\psi}_i(t_1).$$

Because of the linear reproduction property,

$$\boldsymbol{x}^0 = \sum_{i=1}^{n} \phi_i^{t_0}(\boldsymbol{x}^0)\boldsymbol{\psi}_i(t_0),$$

hence

$$\boldsymbol{x}^{t_1} - \boldsymbol{x}^0 = \sum_{i=1}^{n} \phi_i^{t_0}(\boldsymbol{x}^0)(\boldsymbol{\psi}_i(t_1) - \boldsymbol{\psi}_i(t_0)).$$

Dividing by $t_1 - t_0$ and taking the limit for $t_1 \to t_0$, yields

$$\boldsymbol{x}'(t) = \sum_{i=1}^{n} \phi_i(\boldsymbol{x}(t), t)\boldsymbol{\psi}_i'(t),$$

which is a first-order differential equation in $\boldsymbol{x}(t)$ of the form

$$\boldsymbol{x}'(t) = F(t, \boldsymbol{x}(t)).$$

with initial condition $\boldsymbol{x}(0) = \boldsymbol{x}^0$.

If the barycentric coordinates ϕ_i and the vertex paths $\boldsymbol{\psi}_i$ are Lipschitz-continuous in x and t, then F is Lipschitz-continuous, too, which is a sufficient condition for

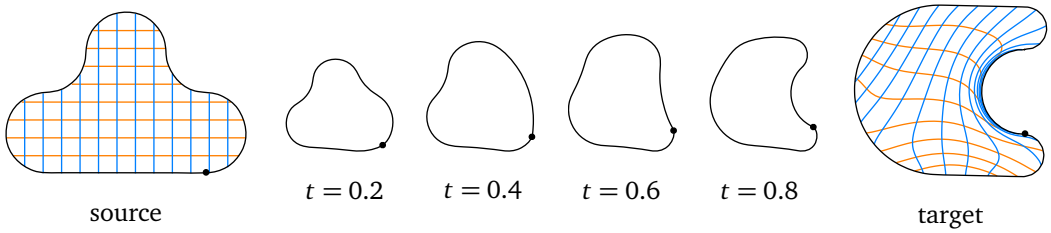


Figure 4.8. Example of a composite mean value mapping between two closed planar curves. The grid shows how the interior of the source curve is morphed to the interior of the target curve.

the existence of a local unique solution, according to the Picard–Lindelöf Theorem [Picard, 1893; Lindelöf, 1894]. The solution of the differential equation is also a global solution, since $\boldsymbol{x}(t)$ stays inside all intermediate polygons. This is the case because the mapping is bijective and edges are mapped to edges.

In order to show that $\boldsymbol{g}^{-1} = \boldsymbol{f}$ we consider, for any point $\boldsymbol{x}^1 = \boldsymbol{x}(1) \in \bar{\Omega}_1$

$$\boldsymbol{y}'(t) = \sum_{i=1}^{n} \phi_i(\boldsymbol{x}(1-t), 1-t)\bar{\boldsymbol{\psi}}_i'(t), \qquad \bar{\boldsymbol{\psi}}_i(t) = \boldsymbol{\psi}_i(1-t)$$

with $\boldsymbol{y}(0) = \boldsymbol{x}(1)$ as the mapping of $\boldsymbol{x}^1$ from $\bar{\Omega}_1$ to $\bar{\Omega}_0$. Because $\bar{\boldsymbol{\psi}}_i'(t) = -\boldsymbol{\psi}_i'(1-t)$, we conclude that $\boldsymbol{y}(t) = \boldsymbol{x}(1-t)$. Therefore, $\boldsymbol{y}(1) = \boldsymbol{x}^0$ and $\boldsymbol{g}$ has an inverse generated by the previous equation with $\boldsymbol{g}^{-1} = \boldsymbol{f}$.

4.4 Extensions

The sufficient condition for guaranteeing bijectivity of a composite barycentric mapping between polygons can be naturally extended to mappings between closed curves and polyhedra.

4.4.1 Closed planar curves

We define the *barycentric mapping* between a closed planar source curve $\gamma_0(s)$ and a closed-planar target curve $\gamma_1(s)$ with $s \in [0,1]$, as

$$\boldsymbol{f}(\boldsymbol{x}) = \int_0^1 \phi(s, \boldsymbol{x})\gamma_1(s)ds,$$

where $\phi(s, \boldsymbol{x})$ are the transfinite barycentric coordinates with respect to γ_0 (Section 1.3). Analogously to the polygonal case, we now consider a perturbed target

curve $\gamma_1(s) = \gamma_0(s) + \boldsymbol{u}(s)$ and assume that the maximum displacement distance satisfies

$$d = \sup_{s\in[0,1]} \|\boldsymbol{u}(s)\| < \frac{\sqrt{5}-1}{2M},$$

where

$$M = \sup_{s\in[0,1]} \sup_{\boldsymbol{x}\in\gamma_0} \|\nabla\phi(s,\boldsymbol{x})\|.$$

We exploit the linear reproduction property to rewrite

$$\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{x} + \int_0^1 \phi(s,\boldsymbol{x})\boldsymbol{u}(s)ds$$

and compute

$$\begin{aligned} J_f(\boldsymbol{x}) &= \begin{vmatrix} 1+\int \partial_1\phi(s,\boldsymbol{x})u_1(s)ds & \int \partial_2\phi(s,\boldsymbol{x})u_1(s)ds \\ \int \partial_1\phi(s,\boldsymbol{x})u_2(s)ds & 1+\int \partial_2\phi(s,\boldsymbol{x})u_2(s)ds \end{vmatrix} \\ &= 1 + \int \nabla\phi(s,\boldsymbol{x})\cdot\boldsymbol{u}(s)ds + \iint \partial_1\phi(s,\boldsymbol{x})\partial_2\phi(r,\boldsymbol{x})(\boldsymbol{u}(s)\times\boldsymbol{u}(r))ds\,dr, \end{aligned}$$

where the integrals range from zero to one. Therefore,

$$J_f(\boldsymbol{x}) \geq 1 - Md - M^2d^2,$$

which is strictly positive for $Md < (\sqrt{5}-1)/2$. Hence the mapping is injective as long as $d < \frac{\sqrt{5}-1}{2M}$. It is interesting to note that this bound is the same as in the polygonal case.

Now suppose that $\boldsymbol{\psi}$ is a continuous *closed curve interpolation* function between the source curve $\gamma_0 = \boldsymbol{\psi}(0)$ and the target curve $\gamma_1 = \boldsymbol{\psi}(1)$. Again we let $\tau = (t_0, t_1, \ldots, t_m)$ with $t_0 = 0$, $t_m = 1$, and $t_k < t_{k+1}$ for $k = 0, \ldots, m-1$ be a *partition* of $[0,1]$ and let $\boldsymbol{f}_k$ be the barycentric mapping from γ_{t_k} to $\gamma_{t_{k+1}}$, based on the barycentric coordinates ϕ^{t_k}. The mapping

$$\boldsymbol{f}_\tau = \boldsymbol{f}_{m-1} \circ \boldsymbol{f}_{m-2} \circ \cdots \circ \boldsymbol{f}_0,$$

is called a *composite barycentric mapping* from γ_0 to γ_1. An example of a composite barycentric mapping between two closed curves is shown in Figure 4.8.

Denoting the maximum displacement distance between γ_{t_k} and $\gamma_{t_{k+1}}$ by

$$d_k = \sup_{s\in[0,1]} \left\|\gamma_{t_k}(s) - \gamma_{t_{k+1}}(s)\right\|,$$

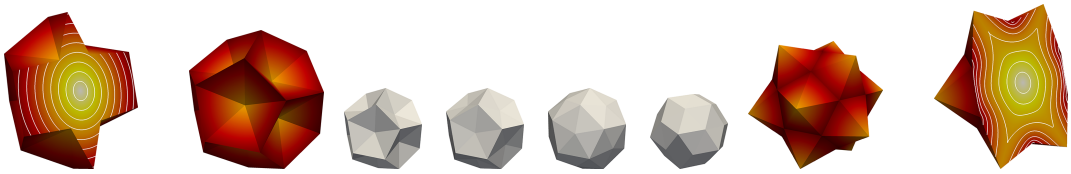


Figure 4.9. Example of a composite mean value mapping between two polyhedra. The color shows how the interior of the source polyhedron is morphed to the interior of the target polyhedron.

it follows from the previous result that $\boldsymbol{f}_\tau$ is bijective if

$$d_\tau = \max_{0 \le k < m} d_k < \frac{\sqrt{5}-1}{2nM^*},$$

where

$$M^* = \sup_{s\in[0,1]} \sup_{t\in[0,1]} \sup_{\boldsymbol{x}\in\gamma_t} \|\nabla \phi^t(s,\boldsymbol{x})\|.$$

4.4.2 Polyhedra

A barycentric mapping between two polyhedra with the same number of vertices n and the same topology is defined as a function

$$\boldsymbol{f} : \bar{\Omega}_0 \to \bar{\Omega}_1, \qquad \boldsymbol{f}(\boldsymbol{x}) = \sum_{i=1}^{n} \phi_i(\boldsymbol{x}) \boldsymbol{v}_1^i,$$

where $\boldsymbol{v}_1^i$ are the vertices of the target polyhedron and $\phi_i(\boldsymbol{x})$ are 3D barycentric coordinates (Section 1.1). An example of such a mapping is illustrated in Figure 4.9.

As in the polygonal case we first perturb only one vertex and consider a target polyhedron with vertices $\boldsymbol{v}_1^i = \boldsymbol{v}_0^i + \boldsymbol{u}$ for some i and $\boldsymbol{v}_1^j = \boldsymbol{v}_0^j$ for $j \neq i$. Then,

$$J_f(\boldsymbol{x}) = \begin{vmatrix} 1+\partial_1\phi_i(\boldsymbol{x})u_1 & \partial_2\phi_i(\boldsymbol{x})u_1 & \partial_3\phi_i(\boldsymbol{x})u_1 \\ \partial_1\phi_i(\boldsymbol{x})u_2 & 1+\partial_2\phi_i(\boldsymbol{x})u_2 & \partial_3\phi_i(\boldsymbol{x})u_2 \\ \partial_1\phi_i(\boldsymbol{x})u_3 & \partial_2\phi_i(\boldsymbol{x})u_3 & 1+\partial_3\phi_i(\boldsymbol{x})u_3 \end{vmatrix} = 1 + \nabla\phi_i(\boldsymbol{x}) \cdot \boldsymbol{u}$$

and

$$|J_f(\boldsymbol{x})| \ge 1 - |\phi_i(\boldsymbol{x}) \cdot \boldsymbol{u}| \ge 1 - \|\phi_i(\boldsymbol{x})\|\|\boldsymbol{u}\|,$$

which is positive for $\|\boldsymbol{u}\| < 1/M_i$ with

$$M_i = \sup_{\boldsymbol{x}\in\Omega} \|\nabla b_i(\boldsymbol{x})\|.$$

Again, we perturb all vertices of the polyhedron and consider a target polyhedron with vertices $\boldsymbol{v}_1^i = \boldsymbol{v}_0^i + \boldsymbol{u}_i$ for $i = 1, \ldots, n$, to get

$$\begin{aligned} J_f(\boldsymbol{x}) &= \begin{vmatrix} 1+\sum_i \partial_1\phi_i(\boldsymbol{x})u_{i,1} & \sum_i \partial_2\phi_i(\boldsymbol{x})u_{i,1} & \sum_i \partial_3\phi_i(\boldsymbol{x})u_{i,1} \\ \sum_i \partial_1\phi_i(\boldsymbol{x})u_{i,2} & 1+\sum_i \partial_2\phi_i(\boldsymbol{x})u_{i,2} & \sum_i \partial_3\phi_i(\boldsymbol{x})u_{i,2} \\ \sum_i \partial_1\phi_i(\boldsymbol{x})u_{i,3} & \sum_i \partial_2\phi_i(\boldsymbol{x})u_{i,3} & 1+\sum_i \partial_3\phi_i(\boldsymbol{x})u_{i,3} \end{vmatrix} \\ &= 1 + \sum_i \nabla\phi_i(\boldsymbol{x}) \cdot \boldsymbol{u} + \sum_i \sum_j \sum_k \partial_1\phi_i(\boldsymbol{x})\, \partial_2\phi_j(\boldsymbol{x})\, \partial_3\phi_k(\boldsymbol{x})\, |(\boldsymbol{u}_i\ \boldsymbol{u}_j\ \boldsymbol{u}_k)| \\ &+ \sum_i \sum_j \Big(\partial_1\phi_i(\boldsymbol{x})\, \partial_2\phi_j(\boldsymbol{x})\, D_3 + \partial_1\phi_i(\boldsymbol{x})\, \partial_3\phi_j(\boldsymbol{x})\, D_2 + \partial_2\phi_i(\boldsymbol{x})\, \partial_3\phi_j(\boldsymbol{x})\, D_1 \Big), \end{aligned}$$

where the sums range from 1 to n and D_k is the k-th component of $\boldsymbol{u}_i \times \boldsymbol{u}_j$. Therefore,

$$J_f(\boldsymbol{x}) \geq 1 - Md - 3M^2d^2 - M^3d^3,$$

where

$$d = \max_{1 \leq i \leq n} \|\boldsymbol{u}_i\| \qquad \text{and} \qquad M = M_1 + \cdots + M_n,$$

which is positive if $Md < \sqrt{2}-1$, implying that $\boldsymbol{f}$ is injective for $d < \frac{\sqrt{2}-1}{M}$.

Now, suppose that $\boldsymbol{\psi}_i : [0,1] \to \mathbb{R}^3$, $i = 1, \ldots, n$ are a set of continuous *vertex paths* between each source vertex $\boldsymbol{v}_0^i = \boldsymbol{\psi}_i(0)$ and its corresponding target vertex $\boldsymbol{v}_1^i = \boldsymbol{\psi}_i(1)$. We define the *composite barycentric mapping* from $\bar{\Omega}_0$ to $\bar{\Omega}_1$ as

$$\boldsymbol{f}_\tau = \boldsymbol{f}_{m-1} \circ \boldsymbol{f}_{m-2} \circ \cdots \circ \boldsymbol{f}_0,$$

where $\boldsymbol{f}_k : \bar{\Omega}_{t_k} \to \bar{\Omega}_{t_{k+1}}$ are barycentric mappings based on the partition $\tau = (t_0, t_1, \ldots, t_m)$ of the interval $[0,1]$.

Denoting the maximum displacement distance between $\bar{\Omega}_{t_k}$ and $\bar{\Omega}_{t_{k+1}}$ by

$$d_k = \max_{1 \leq i \leq n} \left\| \boldsymbol{v}_{t_k}^i - \boldsymbol{v}_{t_{k+1}}^i \right\|,$$

it follows that $\boldsymbol{f}_\tau$ is bijective if

$$d_\tau = \max_{0 \leq k < m} d_k < \frac{\sqrt{2}-1}{2nM^*},$$

where

$$M^* = \max_{1 \leq i \leq n} \sup_{t \in [0,1]} \sup_{\boldsymbol{x} \in \bar{\Omega}_t} \|\nabla \phi_i^t(\boldsymbol{x})\|.$$

The composite barycentric mapping $\boldsymbol{f}_\tau$ can also be extended to closed smooth surfaces, and by following a similar reasoning we can conclude that the mapping $\boldsymbol{f}_\tau$ is bijective if $d_\tau < (\sqrt{2}-1)/(2nM^*)$.

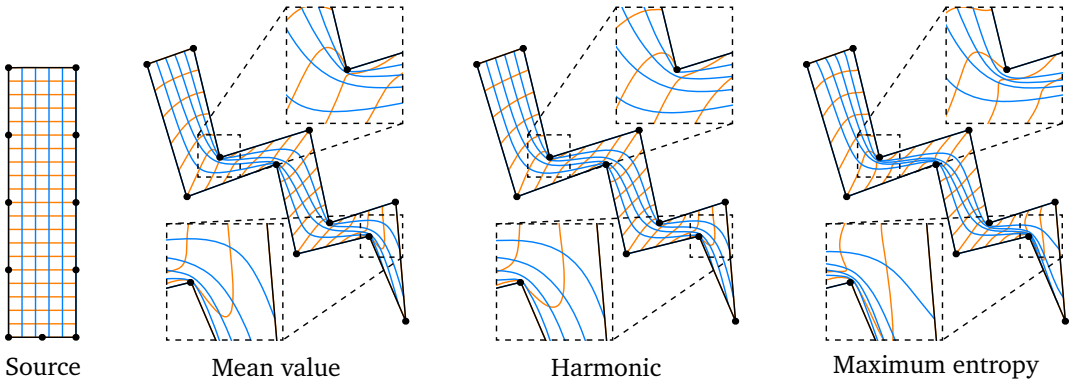


Figure 4.10. Composite barycentric map for different types of barycentric coordinates for 100 uniform steps.

4.5 Practical considerations

Composite barycentric mappings depend on two main choices: the underlying barycentric coordinates ϕ and the vertex paths ψ. In this section we illustrate the influence of these choices on the composite mapping.

4.5.1 Choosing the coordinates

In general, the vertex paths ψ_i produce an arbitrary intermediate shape, which may be concave or weakly convex. For this reason we need barycentric coordinates that are well-defined for arbitrary simple polygons. We suggest to use mean value 1.7, harmonic 1.9, or maximum entropy coordinates 1.10. Figure 4.10 shows an example of a composite barycentric mapping for these three different coordinates.

4.5.2 Choosing the vertex paths

To create intermediate shapes, the natural vertex path that works in any dimension and for any shape, is to linearly interpolate the shapes from source to target. However, this interpolation is not invariant with respect to similarity transformations, while the mapping is. Moreover, linear interpolation of the vertices is more likely to produce self-intersecting polygons in case of rotations, which invalidates the mapping since barycentric coordinates are well-defined only for non-self-intersecting polygons. For these reasons linear interpolation is not well-suited for creating intermediate shapes for the composite barycentric mappings. In Chapter 3 we provide an extensive list of methods to define the vertex path.

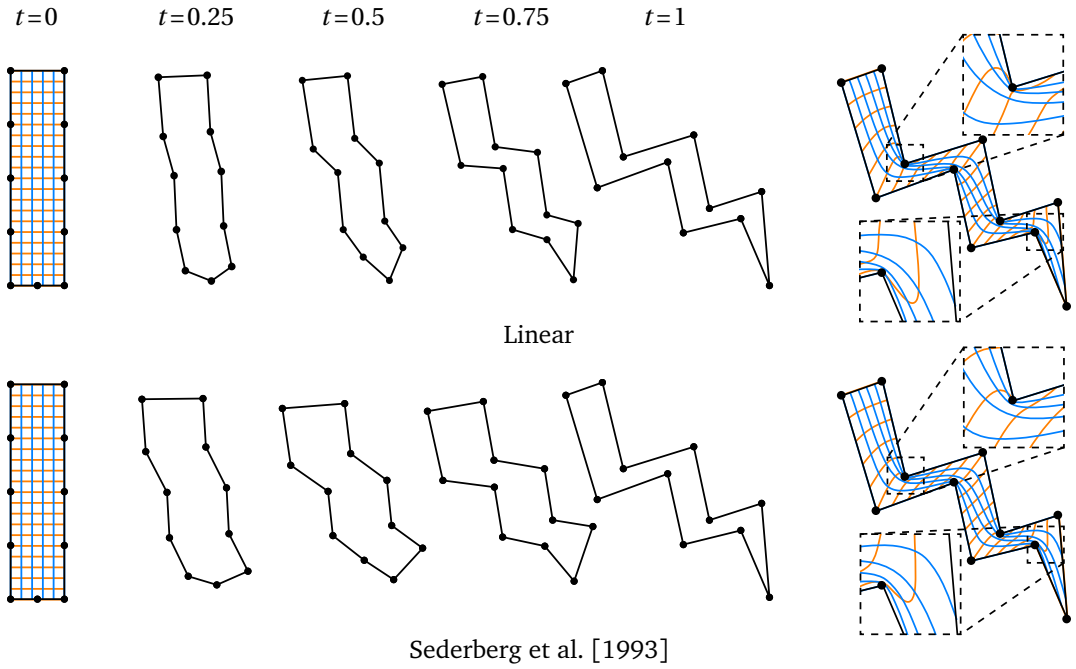


Figure 4.11. Example of a composite barycentric map for different vertex paths for 100 uniform steps.

Figure 4.11 shows how the composite barycentric mapping changes when this method is used instead of linearly interpolating the vertices.

4.5.3 Efficient implementation

We show that for a sufficiently large number of steps m, we can guarantee bijectivity of a composite mean value mapping. However, it is important to choose m small, because the computations scale linearly with m. To find such a small m that is large enough to produce mappings which are bijective up to pixel accuracy, we adopt the following strategy, which relies on the minimum of the Jacobian

$$J_{\min} = \min_{x \in \Omega} \det J_f(x).$$

Suppose we start with the initial partition $\tau = (0, 1)$ and consider the classical mean value mapping $f_\tau : \Omega_0 \to \Omega_1$. We can now compute $J_{\min}$ for this mapping and check for positivity. If $J_{\min}$ is negative, and hence the mapping is not bijective, we can try to restore bijectivity by inserting 0.5 into the partition τ and considering the intermediate polygon $\Omega_{0.5}$. We then recalculate $J_{\min}$ for the two mappings $f_1 : \Omega_0 \to \Omega_{0.5}$ and $f_2 : \Omega_{0.5} \to \Omega_1$ and test both for positivity. We continue this

binary refinement of τ until $J_{\min}$ is positive for all mappings $\boldsymbol{f}_k$ or until the distance between two steps becomes numerically too small, that is, smaller than some threshold $\epsilon > 0$ (we used $\epsilon = 0.01$ in our examples). The pseudo-code 1 illustrates a recursive algorithm for finding such a binary partition. Given two domains Ω_a and Ω_b for $0 \leq a < b \leq 1$, the algorithm's core consists of computing $J_{\min}$ for the mapping between Ω_a and Ω_b. Then, by checking the positivity of the minimum, the algorithm decides whether to introduce an intermediate step. Overall, this produces a binary partition which guarantees that the corresponding composite mapping is bijective and consists of a small number of intermediate steps. Figure 4.12 shows an example of the result of this algorithm.

In order to compute $J_{\min}$, we need a formula for the partial derivatives of the mean value coordinates (1.7). To efficiently evaluate expression (1.7), we express the tangent t_i as

$$t_i = \tan(\alpha_i/2) = \frac{\sin\alpha_i}{1+\cos\alpha_i} = \frac{s_i \times s_{i+1}}{r_i r_{i+1} + s_i \cdot s_{i+1}},$$

where $s_i = v_i - v$. Therefore, the gradient of ϕ_i can now be derived from the gradients of the cross product,

$$\nabla(s_i \times s_{i+1}) = (\boldsymbol{v}^{i+1} - \boldsymbol{v}^i)^\perp = \begin{pmatrix} v_2^i - v_2^{i+1} \\ v_1^{i+1} - v_1^i \end{pmatrix},$$

the distance,

$$\nabla r_i = \frac{-s_i}{r_i}.$$

Algorithm 1: Pseudo-code of the algorithm for finding a binary partition with a small number of steps.

```
Function buildPartition(a, b)
    Data: Interval [a, b]
    Result: Partition τ

    J_min ← compute J_min for the mapping f : Ω_a → Ω_b

    if J_min ≤ 0 and |b − a| > ε
        c ← (a + b)/2
        τ ← τ ∪ c

        buildPartition(a, c)
        buildPartition(c, b)
```

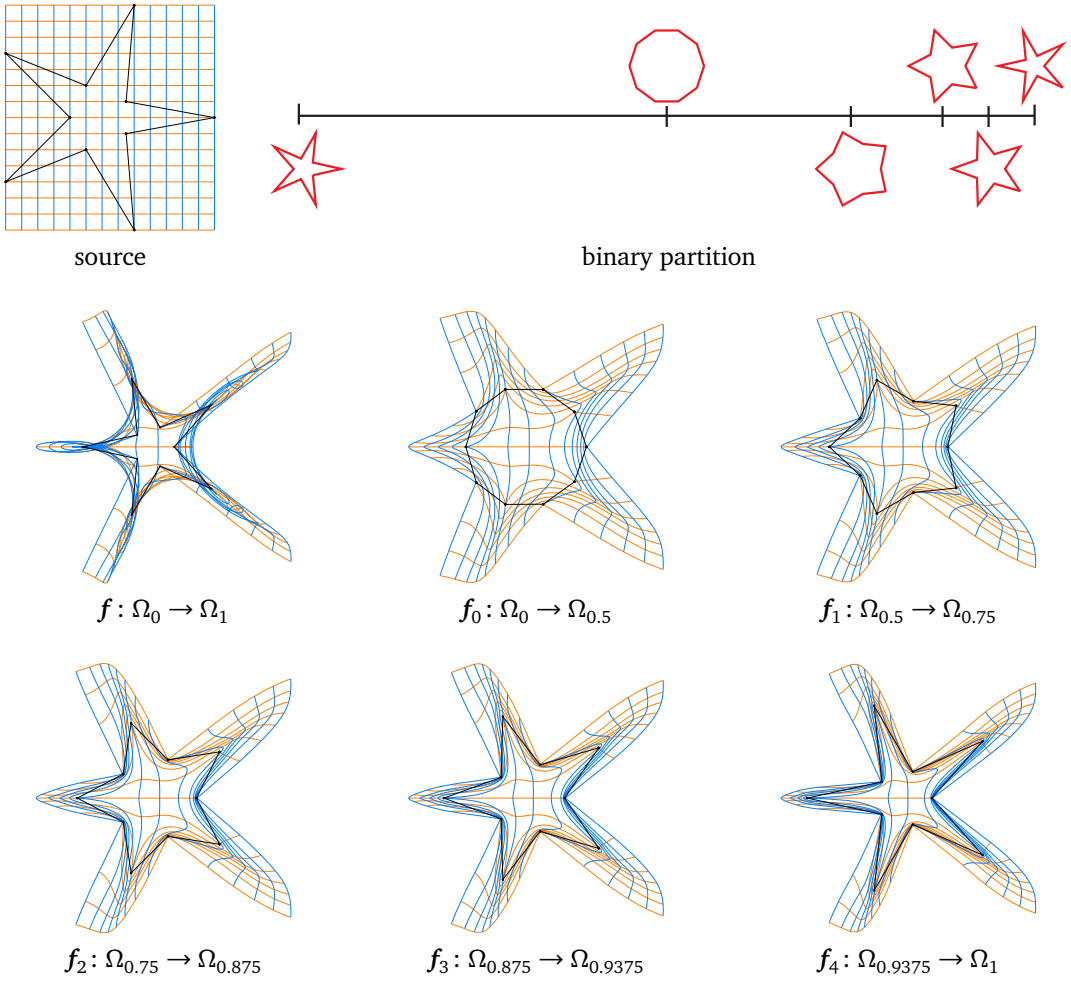


Figure 4.12. Example of a composite mean value mapping with a 5-step binary partition.

and the dot product

$$\nabla(s_i \cdot s_{i+1}) = -(s_i + s_{i+1}),$$

and by using the chain rule, which first gives

$$\nabla t_i = \frac{\nabla(s_i \times s_{i+1}) - t_i(\nabla r_i r_{i+1} + r_i \nabla r_{i+1} + \nabla(s_i \cdot s_{i+1}))}{r_i r_{i+1} + s_i \cdot s_{i+1}},$$

then

$$\nabla w_i = \frac{\nabla t_{i-1} + \nabla t_i - w_i \nabla r_i}{r_i},$$

and finally

$$\nabla\phi_i = \frac{\nabla w_i - \phi_i \nabla \sum_j w_j}{\sum_j w_j}. \tag{4.2}$$

A common technique for constructing fast algorithms is to parallelize them. In such algorithms the speed-up factor is given by the nature of the problem, as only unrelated computations can be performed in parallel. In our case, the computations of the determinants for several points $\boldsymbol{x} \in \Omega$ are completely independent of each other, hence our problem is perfectly suited for parallelization. There are three main practices for parallelizing an algorithm: using the CPU with threads (SIMD), cluster computing, and exploiting the parallel nature of the GPU. To compute $J_{\min}$, we need to compute the determinant for a large number of vertices. For example, if we want pixel resolution for an image of 512×512, then we need to consider $262,144$ points. An implementation with CPU threads becomes impractical for such large numbers, as we would need approximately the same number of threads as points. We also discard cluster computing, because this technique is not made for real-time applications. The most reasonable choice is to implement a GPU algorithm. Among the various possible GPU implementations, we opt for implementing the algorithm as a vertex shader. This shader receives the vertices of the source and target polygons, computes the Jacobian, and writes it as a colour. Figure 4.1 shows some examples of the values of J_f computed this way. The major drawback of this approach is that the output of the shader is a full resolution image, which must be read back and analysed by the CPU, which should be avoided whenever possible.

The vertex shader knows only the position of the currently processed vertex, hence finding the global minimum is impossible. To circumvent this problem we exploit the *z-buffer*. This particular OpenGL-buffer is used to determine which object is closer to the camera. Using this trick we modify the shader to "collapse" all the points to the same (x, y) position and we artificially set the z-component to the value of the Jacobian (clamped to zero if negative). In this way the generated image contains only one pixel which is the one closest to the camera, and the colour of this pixel corresponds to $J_{\min}$. At the end, the CPU reads only one pixel and produces a sequence of domains Ω_i, $i = 1, \ldots, n$ such that the mapping between two successive domains is visually bijective.

Once we have computed a suitable binary partition, the last problem to solve is the evaluation of the composite mapping. Again, this problem can be parallelized nicely, like the problem of computing the Jacobian, because the computations for each point are independent of the computations in other points.

The shader receives two successive domains Ω_k and Ω_{k+1} and moves the ver-

tex position according to the mapping f_k. In this case the main challenge is how to pass the moved vertices, without passing them back to the CPU, so the GPU can compute the next step of the visually bijective composition. We then exploit the *transform feedback buffer*, a technique that allows to alter the rendering pipeline of OpenGL such that the primitives, in our case the vertex positions, are written into a buffer object. Hence the algorithm becomes a "ping pong" between two *transform feedback buffers*. We use the first buffer as output (where the vertex position is written) and the second as input (from where the positions are read). In the next step the role of the two buffers is inverted and so on. At the end, the last output buffer contains the vertex positions of the composite mapping. Figure 4.12 shows the warping of the star for different choices of intermediate steps. When the partition is sufficiently refined, the morphing is visually bijective and prevents fold-overs.

Chapter 5

Composite harmonic mappings

An alternative approach to build bijective maps consists of combining two harmonic maps to define a smooth map between two arbitrary planar polygons. In contrast with respect to the composite barycentric mapping (Chapter 4), this construction is symmetric. However, the mapping can be only evaluated numerically and can be approximated in different ways, each with its own pros and cons. The different implementations provide a high degree of flexibility, ranging from an efficient computation to the exact point-wise evaluation. While these methods are guaranteed to generate bijective maps only in the limit, as they converge to the exact solution, our experiments show that they produce bijective results in practice.

The behaviour of the map along the boundary can be prescribed (for example, to be piecewise linear), giving the user an intuitive way of controlling the map in applications such as image warping and surface cross-parametrization (Figure 5.8).

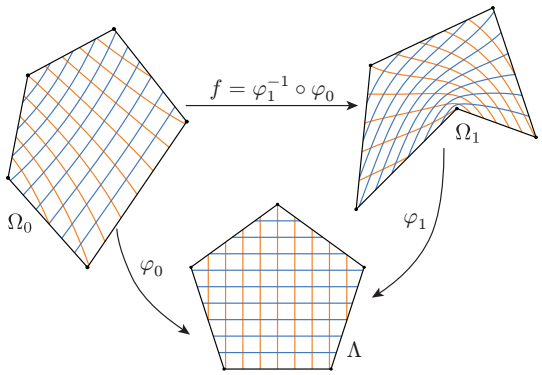


Figure 5.1. Main idea for defining smooth bijective maps.

The simplicity of this construction allows for several extensions (Sections 5.3 and 5.4), including a simple non-linear optimization procedure to reduce the distortion of the map without compromising the other properties and a strategy which appears to give almost conformal results. Moreover, this method trivially extends to smooth volumetric maps between polyhedral domains, and although the theoretical guarantees on bijectivity are lost [Snyder Laugesen, 1996], our experiments demonstrate that the maps are bijective even for non-trivial examples.

Given two planar domains $\bar{\Omega}_0$, $\bar{\Lambda}$ and a bijective boundary mapping $g : \partial\bar{\Omega}_0 \to \partial\bar{\Lambda}$, the unique map $\varphi : \bar{\Omega}_0 \to \bar{\Lambda}$ that solves the Laplace equation

$$\Delta\varphi = 0 \tag{5.1a}$$

subject to the continuous Dirichlet boundary condition

$$\varphi|_{\partial\bar{\Omega}_0} = g \tag{5.1b}$$

is called a *harmonic map*. The map φ is smooth, and it follows from the Radó–Kneser–Choquet theorem [Choquet, 1945; Kneser, 1926; Radó, 1926] that φ is bijective if $\bar{\Lambda}$ is convex.

In order to define the smooth bijective map $\boldsymbol{f}$ between $\bar{\Omega}_0$ and $\bar{\Omega}_1$, we simply introduce an intermediate convex polygon $\bar{\Lambda}$ with n vertices, combine the two harmonic maps $\varphi_0 : \bar{\Omega}_0 \to \bar{\Lambda}$ and $\varphi_1 : \bar{\Omega}_1 \to \bar{\Lambda}$ as shown in Figure 5.1, and let

$$\boldsymbol{f} = \varphi_1^{-1} \circ \varphi_0. \tag{5.2}$$

For simplicity, we can use a regular n-gon as $\bar{\Lambda}$ and piecewise linear boundary constraints g_0 and g_1, but it is possible to deviate from this default choice (Sections 5.3 and 5.4).

The concept of constructing bijective maps in this way has already been used in the context of mesh parametrization [Weber and Zorin, 2014], mesh morphing [Kanai et al., 1997], and surface correspondences [Lipman and Funkhouser, 2009], but only in the discrete setting. Instead, we consider smooth maps, and we also discuss two extensions Sections 5.3 and 5.4.

This approach poses two practical challenges. First, we need to solve the boundary value problem (5.1), and the exact solution is known only for simple domains like rectangles. However, an approximate solution can be computed in different ways (Section 5.1). Second, we have to invert the map φ_1, and again we treat this problem numerically (Section 5.2).

5.1 Solving the Laplace equation

There are three main methods for solving the Laplace equation: the finite element method [Strang and Fix, 2008] (FEM), the boundary element method [Hall, 1994; Rustamov, 2007] (BEM), and the method of fundamental solutions [Fairweather and Karageorghis, 1998; Martin et al., 2008] (MFS). Each method comes with certain advantages and disadvantages. Since these methods are designed to find *harmonic functions* $\varphi : \Omega \to \mathbb{R}$, we apply them to both components of the harmonic maps φ_0, φ_1 separately.

5.1.1 Finite element method

By calculus of variations, φ solves (5.1a), if and only if

$$\int_\Omega \Delta\varphi \, \psi = 0 \tag{5.3}$$

for a certain set of *test functions* ψ. In the finite element approach this set contains all functions that vanish on the boundary of Ω. Using this fact and integration by parts, the weak form (5.3) can be rewritten as

$$\int_\Omega \nabla\varphi \, \nabla\psi = 0. \tag{5.4}$$

In our implementation, we consider the space of piecewise linear functions over a constrained Delaunay triangulation of Ω, spanned by the hat functions

$$B_1, \ldots, B_m : \Omega \to \mathbb{R},$$

and the approximation

$$\varphi \approx \sum_{i=1}^{m} c_i B_i.$$

Assuming that the first $k < m$ basis functions correspond to the interior nodes and thus vanish at the boundary, the weak form (5.4) becomes

$$\sum_{i=1}^{m} c_i \int_\Omega \nabla B_i \nabla B_j = 0, \qquad j = 1 \ldots, k.$$

The coefficients $c_{k+1}, \ldots, c_m$ are given by the boundary condition (5.1b), and the others can be found by solving a sparse linear system of size k, where the off-diagonal elements are the well-known *cotangent weights* [Eck et al., 1995; Pinkall and Polthier, 1993; Strang and Fix, 2008].

Figure 5.2. Nodes used by BEM (left) and nodes and sites used by MFS (right).

5.1.2 Boundary element method

The principal idea of the boundary element method is to first solve a small problem on the boundary and then extend this solution to the interior. To this end, we use the divergence theorem to rewrite (5.3) as

$$\int_{\partial\Omega} \frac{\partial\varphi}{\partial \boldsymbol{n}}\psi - \int_{\Omega} \nabla\varphi\,\nabla\psi = 0$$

and then apply integration by parts to the last term to obtain

$$\int_{\partial\Omega} \frac{\partial\varphi}{\partial \boldsymbol{n}}\psi - \int_{\partial\Omega} \frac{\partial\psi}{\partial \boldsymbol{n}}\varphi + \int_{\Omega} \varphi\,\Delta\psi = 0,$$

where $\boldsymbol{n}$ is the unit normal to $\partial\Omega$. In the boundary element approach, the set of test functions is $\{G_{\boldsymbol{x}} : \boldsymbol{x} \in \Omega\}$, where

$$G_{\boldsymbol{x}}(\boldsymbol{y}) = -\frac{1}{2\pi}\log(\|\boldsymbol{x}-\boldsymbol{y}\|) \tag{5.5}$$

are 2D Green's functions. Since $\Delta G_{\boldsymbol{x}}(\boldsymbol{y}) = \delta(\boldsymbol{x}-\boldsymbol{y})$, we obtain the *boundary integral equation*

$$\int_{\partial\Omega} \frac{\partial\varphi}{\partial \boldsymbol{n}} G_{\boldsymbol{x}} - \int_{\partial\Omega} \frac{\partial G_{\boldsymbol{x}}}{\partial \boldsymbol{n}}\varphi + \omega(\boldsymbol{x})\varphi(\boldsymbol{x}) = 0, \qquad \boldsymbol{x} \in \Omega, \tag{5.6}$$

where

$$\omega(\boldsymbol{x}) = \begin{cases} \frac{\alpha(\boldsymbol{x})}{2\pi}, & \text{for } \boldsymbol{x} \in \partial\Omega, \\ 1, & \text{otherwise,} \end{cases}$$

with $\alpha(\boldsymbol{x})$ denoting the exterior angle at $\boldsymbol{x} \in \partial\Omega$, that is, $\alpha(\boldsymbol{x}) = \pi$ along the open edges of Ω.

In our implementation, we sample the boundary of Ω with the *nodes* $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m$, which include the vertices $\boldsymbol{v}^i$ (Figure 5.2), and consider the corresponding space

of piecewise linear functions, spanned by the one-dimensional hat functions $B_1, \dots, B_m \colon \partial\Omega \to \mathbb{R}$. This allows us to use

$$\varphi(\boldsymbol{x}) = \sum_{i=1}^{m} c_i B_i(\boldsymbol{x}), \qquad \frac{\partial \varphi}{\partial \boldsymbol{n}}(\boldsymbol{x}) \approx \sum_{i=1}^{m} d_i B_i(\boldsymbol{x})$$

for all $\boldsymbol{x} \in \partial\Omega$, so that (5.6), evaluated at $\boldsymbol{x}_j$, turns into

$$\sum_{i=1}^{m} d_i \int_{\partial\Omega} B_i G_{\boldsymbol{x}_j} \approx \sum_{i=1}^{m} c_i \left(\int_{\partial\Omega} \frac{\partial G_{\boldsymbol{x}_j}}{\partial \boldsymbol{n}} B_i + \omega(\boldsymbol{x}_j) B_i(\boldsymbol{x}_j) \right)$$

for $j = 1, \dots, m$. Since c_i are given by the boundary condition (5.1b), we then determine the coefficients d_i by solving a dense linear system of size m. After solving this small system on the boundary, we are ready to evaluate φ at any $\boldsymbol{x} \in \Omega$ by rearranging (5.6),

$$\varphi(\boldsymbol{x}) \approx \frac{1}{\omega(\boldsymbol{x})} \sum_{i=1}^{m} \left(d_i \int_{\partial\Omega} B_i G_{\boldsymbol{x}} - c_i \int_{\partial\Omega} \frac{\partial G_{\boldsymbol{x}}}{\partial \boldsymbol{n}} B_i \right).$$

We employ Gaussian quadrature for evaluating all the required boundary integrals numerically, because they do not have closed forms in general. Note that for points near to the boundary the integrals my be numerically unstable due to the poles of the Green functions. To avoid this problem we "snap" these points to the boundary where the value of the function is given by the boundary conditions.

5.1.3 Method of fundamental solutions

The main idea of the method of fundamental solutions is to exploit the fact that Green's functions (5.5) and linear functions are harmonic by construction. We approximate φ by

$$\hat{\varphi} = \sum_{i=1}^{m} w_i \, G_{\boldsymbol{s}_i} + A$$

for certain *sites* $\boldsymbol{s}_1, \dots, \boldsymbol{s}_m$ (Figure 5.2) and some linear function A. In order to avoid singularities inside the domain Ω, the sites need to be placed outside the polygon, and we follow the strategy in [Martin et al., 2008] to determine their positions. The unknown weights w_i and the coefficients of A are then determined by minimizing

$$\sum_{j=1}^{k} |\hat{\varphi}(\boldsymbol{x}_j) - b(\boldsymbol{x}_j)|^2$$

for $k \gg m$ uniformly spaced nodes $\boldsymbol{x}_j \in \partial\Omega$, which can be done by solving a dense over-determined linear system of size $k \times (m+3)$ in the least squares sense.

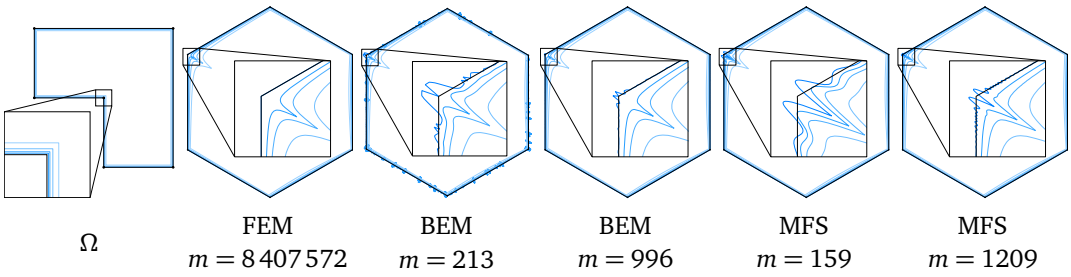


Figure 5.3. Behaviour of the different approximations to the harmonic mapping φ from Ω to the regular hexagon $\bar{\Lambda}$ near the boundary. The curves correspond to the images of the boundary (darkest blue) and boundary offsets at distances 0.001, 0.002, 0.004, and 0.008 (from dark to light blue), relative to the size of the bounding box of Ω.

5.1.4 Comparison

The main advantages of FEM are speed and robustness, but it provides only piecewise linear approximations of the exact solution. Consequently, the first derivatives are constant per triangle and higher derivatives vanish, which is a disadvantage for applications that rely on these quantities being smooth. Another disadvantage is that FEM requires choosing the triangulation a priori, resulting in a fixed resolution of the result. If later a higher resolution is needed, the problem needs to be solved again from scratch. Finally, the result is not guaranteed to be bijective, because the cotangent weights can be negative. Although non-bijectivity usually does not occur in practice, especially at high resolution, it can be prevented by replacing the cotangents with positive weights [Weber and Zorin, 2014]. The resulting piecewise linear mapping is then guaranteed to be bijective, but it does no longer approximate the harmonic solution.

The principal benefits of BEM and MFS are that they give smooth closed-form approximations of φ and that derivatives can be computed analytically. For BEM, every function evaluation requires calculating $2m$ boundary integrals, which is rather slow, whereas evaluating the MFS approximation and its derivatives is fast, because they have simple closed forms. The main disadvantage of both methods is that the boundary conditions are not satisfied exactly and hence they do not guarantee to linearly map the edges from source to target polygon. Instead, the image of $\partial\Omega$ can exhibit oscillations, especially near concave corners, as shown in Figure 5.3. This effect extends to the interior, but disappears quickly with increasing distance to the boundary. Moreover, these oscillations shrink as we raise the number m of basis functions. In the case of BEM, this behaviour stems from the fact that the normal derivative at the boundary of Ω is only approximated

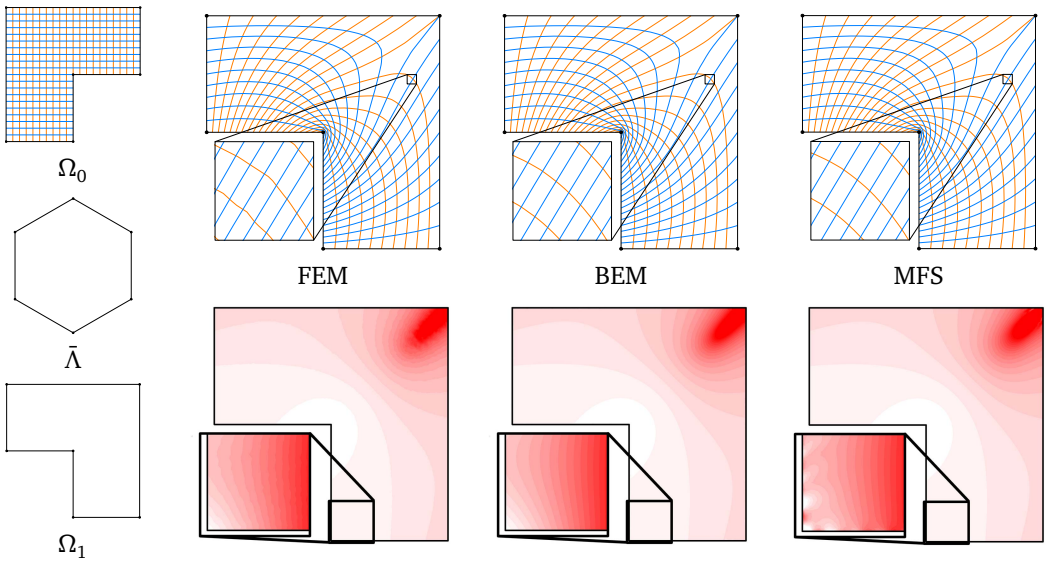


Figure 5.4. Comparison between different methods to approximate a smooth bijective map between Ω_0 and Ω_1. The middle row visualizes the mapping itself, while the determinant of its Jacobian is colour-coded in the bottom row.

by a piecewise linear function. For MFS, it is a consequence of enforcing the boundary constraints only in a least squares sense and at k discrete nodes.

On a theoretical level, this lack of linear precision along the boundary means that the target domain is not convex, and even though both the BEM and the MFS approximations are harmonic by construction, the Radó–Kneser–Choquet theorem cannot be used to conclude their bijectivity. However, as m increases, both methods converge to the exact harmonic and bijective solution, and so they are bijective for a sufficiently large m. In all our examples we observe that even moderately large values of m (less than 1000) are enough to obtain mappings which are practically bijective, in the sense that the Jacobian is positive at a dense set of sample points. However, BEM is usually more precise near the boundary than MFS and seems to converge faster.

Figure 5.4 shows an example of a smooth bijective map and the determinant of its Jacobian, approximated with the different methods. The close-ups confirm that the FEM solution is only piecewise linear, while the others are smooth, and that the MFS solution has artefacts near the boundary due to the oscillations mentioned before. Table 5.1 summarizes the advantages and disadvantages of the methods.

	smooth	meshless	exact on $\partial\Omega$	precise near $\partial\Omega$	fast
FEM	↯	↯	✓	✓	✓
BEM	✓	✓	↯	✓	↯
MFS	✓	✓	↯	↯	✓

Table 5.1. Pros and cons of the different methods.

5.2 Approximation of the inverse of the second harmonic mapping

Inverting the FEM approximation of φ_1 is simple, because it is a piecewise linear map between a Delaunay triangulation of Ω_1 and a corresponding triangulation of $\bar{\Lambda}$. Hence, for any $\boldsymbol{x} \in \bar{\Lambda}$ we search for the triangle T in $\bar{\Lambda}$ that contains $\boldsymbol{x}$ using a k-d tree, compute the local coordinates of $\boldsymbol{x}$ with respect to T, and apply them to the corresponding triangle in Ω_1. This gives an approximation of $\varphi_1^{-1}(\boldsymbol{x})$, where the accuracy depends on the size of T. However, even if the triangles in Ω_1 are small, their images under φ_1 in $\bar{\Lambda}$ can be large and the only way to improve the accuracy is to recompute the approximation of φ_1 with a finer mesh.

For BEM and MFS we can proceed similarly by first triangulating Ω_1 and then mapping the nodes, which induces a triangulation of $\bar{\Lambda}$. Since both methods provide smooth solutions, we can now employ an adaptive refinement strategy that finds all the large triangles in $\bar{\Lambda}$, refines the corresponding triangles in Ω_1, and repeats this process until all triangles in $\bar{\Lambda}$ are sufficiently small (Figure 5.5). In practice this requires only few steps of refinement.

A better option is to exploit the fact that BEM and MFS provide gradients and Hessians of the solution. Hence, we can use efficient numerical solvers to approximate $\varphi_1^{-1}(\boldsymbol{x})$ more accurately. In our implementation we use IPOPT [Wächter

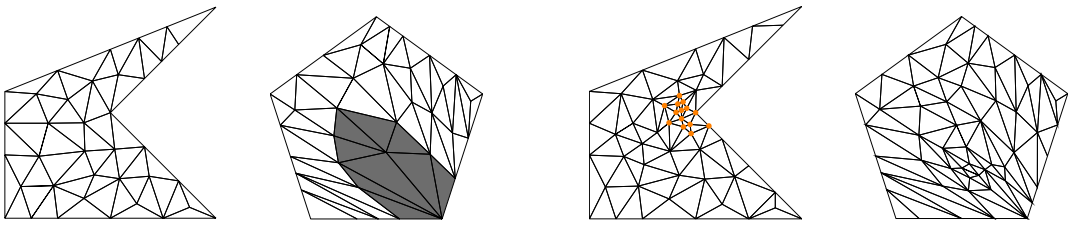


Figure 5.5. One step of the adaptive refinement strategy: regions with large triangles in $\bar{\Lambda}$ are detected (grey) and the corresponding regions in Ω_1 are refined by inserting new points (orange), so that the large triangles are removed from $\bar{\Lambda}$.

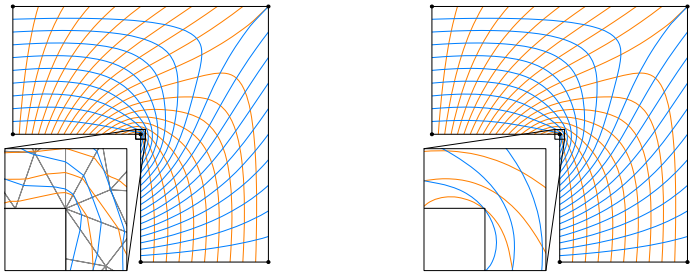


Figure 5.6. Inverting φ_1 with a piecewise linear approximation (left) and pointwise minimization (right).

and Biegler, 2006] to minimize the function

$$d(\boldsymbol{y}) = \|\boldsymbol{x} - \varphi_1(\boldsymbol{y})\|^2,$$

which is convex and guaranteed to be zero at the optimal solution $\boldsymbol{y}^*$. This minimization needs to constrain $\boldsymbol{y}$ to the interior of Ω, because the harmonic function φ is undefined outside Ω. This is no problem for convex domains, but we need to be careful if Ω is concave. In that case we decompose Ω into triangles and constrain the optimization to the triangle T whose image $\varphi(T)$ contains $\boldsymbol{x}$. If the solver finds a minimum at some $\boldsymbol{y}$ with $d(\boldsymbol{y}) > 0$, then we know that the optimal $\boldsymbol{y}^*$ needs to be found in some other triangle. But when this happens, $\boldsymbol{y}$ lies on one of the edges of T and we keep searching in the neighbouring triangle which shares this edge. However, this approach is much slower, because it requires several evaluations of φ_1 and its derivatives.

Figure 5.6 illustrates the results obtained by both inversion procedures. The close-ups clearly show that the first method provides only piecewise linear results, while the result of the second approach is smooth. However, the minimization procedure takes several minutes, whereas the other procedure can be computed in a few seconds.

Figure 5.7. Deforming a source image (left) by means of smooth bijective maps (centre) and composite mean value maps (right). The results are globally similar, but smooth bijective map has less distortion (grey boxes).

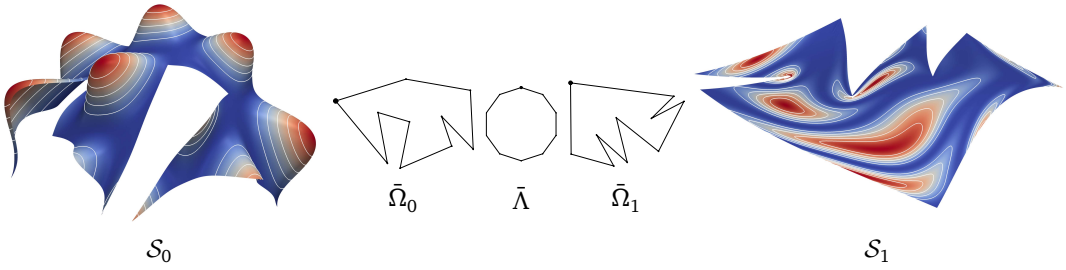


Figure 5.8. Using a smooth surface cross-parametrization to transfer a colour-valued signal from $\mathcal{S}_0$ to $\mathcal{S}_1$.

5.3 Reducing the distortion

An interesting fact to observe is that the mapping $\boldsymbol{f}$ depends on the shape of $\bar{\Lambda}$, as illustrated in Figure 5.9. In this example we compare the results obtained by using a regular polygon versus an irregular cyclic polygon with edge lengths proportional to the average lengths of corresponding source and target edges.

This flexibility can actually be used to reduce the distortion of the map. We implemented a simple strategy, which minimizes a given distortion energy by moving one vertex of $\bar{\Lambda}$ at a time under the constraint that $\bar{\Lambda}$ remains convex. More specifically, to optimize the position of a vertex $\boldsymbol{v}$ of $\bar{\Lambda}$, we approximate the gradient of the distortion energy with respect to $\boldsymbol{v}$, using finite differences, resulting in a displacement vector $\Delta\boldsymbol{v}$. We then move from $\boldsymbol{v}$ to $\boldsymbol{v} + \Delta\boldsymbol{v}$ and check if this new position violates the convexity constraints of $\bar{\Lambda}$. If that is the case, then we iteratively halve $\Delta\boldsymbol{v}$ until the constraints are met.

An example of this optimization procedure, where we reduce the overall conformal distortion of the map by about 10%, is shown in Figure 5.10. A similar improvement can be obtained for the isometric distortion of the map, as illus-

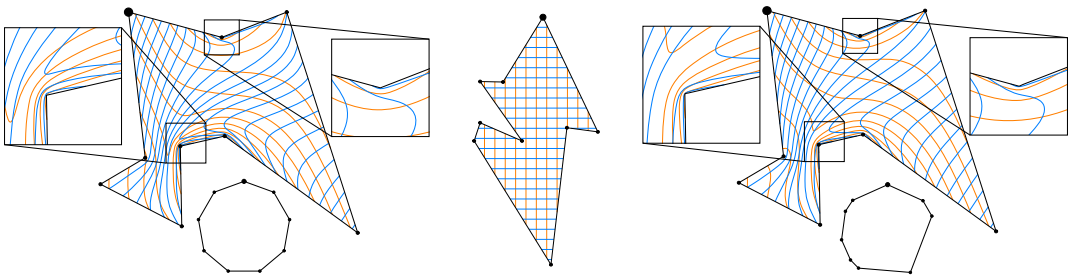


Figure 5.9. Effect of mapping from $\bar{\Omega}_0$ (centre) to $\bar{\Omega}_1$ using a regular (left) and an irregular (right) intermediate polygon.

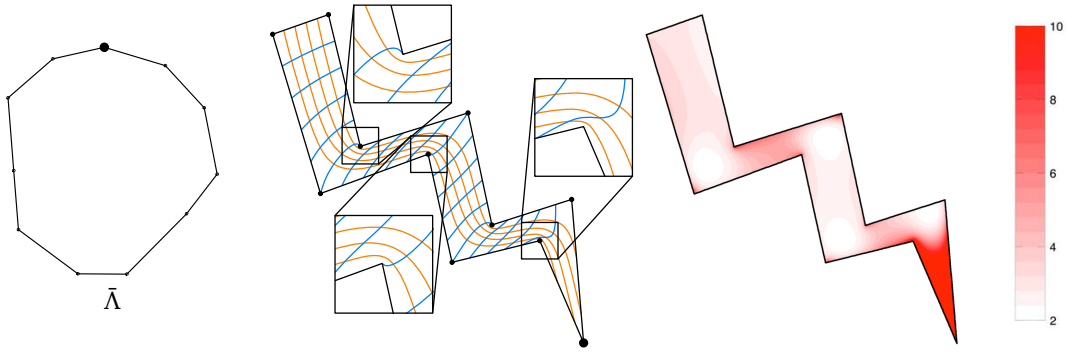


Figure 5.10. Optimizing the shape of $\bar{\Lambda}$ for the example in Figure 5.16 reduces the overall conformal distortion from 7.03 to 6.45.

trated in Figure 5.11, which also compares the result to the method in Chapter 4. Table 5.2 summarizes the distortions obtained with this method, using regular and optimized intermediate polygons, and with the method shown in Chapter 4 for all examples shown in this chapter.

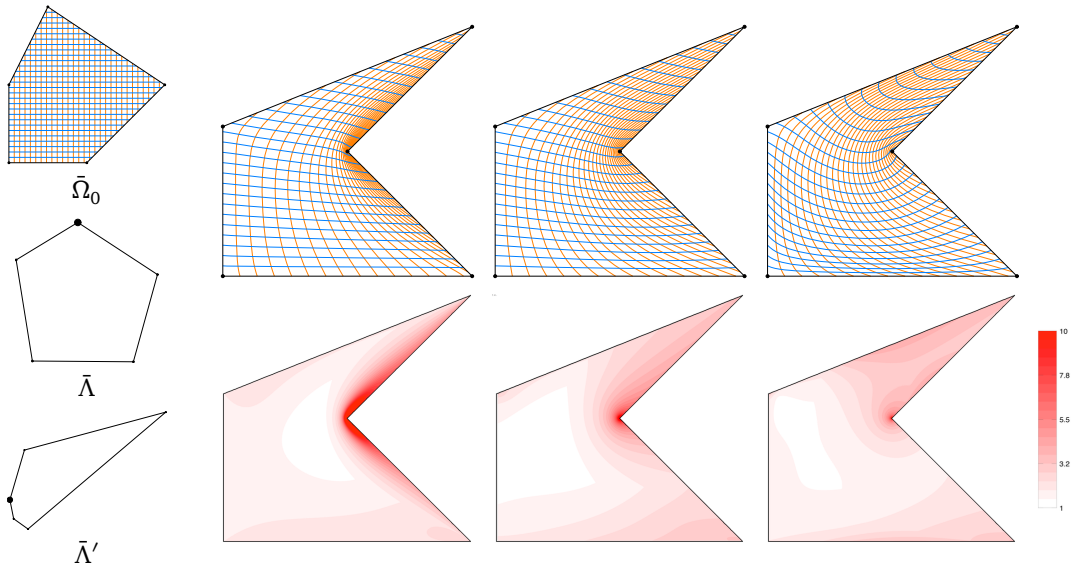


Figure 5.11. Comparison of smooth bijective maps generated the method in Chapter 4 (left) and this method for an irregular (centre) and the optimized (right) intermediate polygon. The respective overall isometric distortions are 7.34, 3.02, and 2.49.

Figure	conformal			isometric		
	Chapter 4	$\bar{\Lambda}$	$\bar{\Lambda}'$	Chapter 4	$\bar{\Lambda}$	$\bar{\Lambda}'$
Average distortion						
5.4	2.93	2.84	2.78	2.85	2.60	2.15
5.7	2.91	2.85	—	2.10	1.95	—
5.9	7.99	6.41	—	3.63	3.59	—
5.10	203.77	7.03	6.45	232.05	5.57	5.18
5.11	9.32	4.85	4.53	7.34	3.02	2.49
5.12	9.26	6.13	—	8.28	5.02	—
Maximum distortion						
5.4	104.86	89.53	75.01	57.53	96.80	83.17
5.7	96.59	96.07	—	88.14	68.61	—
5.9	99.96	99.41	—	99.25	94.30	—
5.10	107 402	2 481.40	1 968.75	122 576	1 883.76	1 679.25
5.11	99.76	99.86	89.14	99.98	97.79	84.67
5.12	99.61	83.26	—	100.00	82.35	—

Table 5.2. Average and maximum conformal and isometric distortions for composite mean value maps (Chapter 4) and composite harmonic mapping based on regular intermediate polygons $\bar{\Lambda}$ and optimized polygons $\bar{\Lambda}'$.

5.4 Boundary conditions

Most applications require linearity on the boundary, but with this method we can also impose a different boundary behaviour, similar to how it is suggested in [Weber et al., 2011]. In the example in Figure 5.12 we specify a linear behaviour for the edges of $\bar{\Omega}_0$ and a quadratic behaviour for the edges of $\bar{\Omega}_1$. That is, the boundary condition b_1 for any point $\boldsymbol{x} = (1-t)\boldsymbol{v}_1^1 + t\boldsymbol{v}_1^2$ on the edge $[\boldsymbol{v}_1^1, \boldsymbol{v}_1^2]$

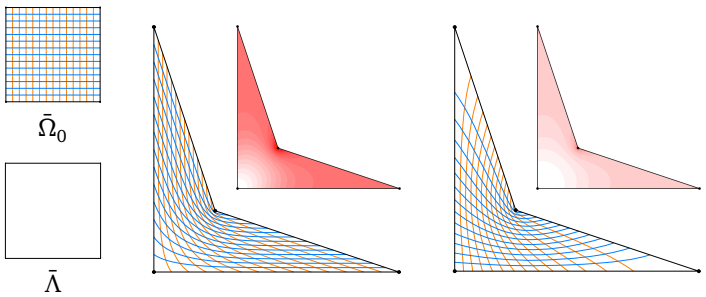


Figure 5.12. Effect of changing the behaviour of the boundary conditions for φ_1 from linear (left) to quadratic (right).

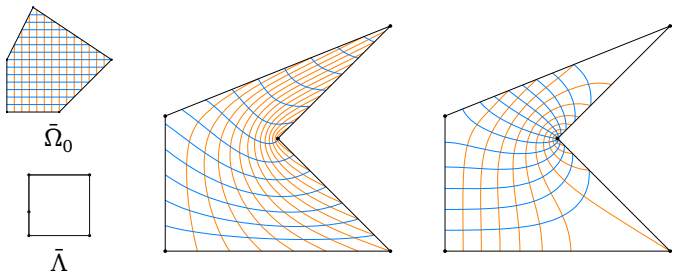


Figure 5.13. Effect of replacing Dirichlet boundary conditions (left) with mixed Neumann/Dirichlet boundary conditions (right).

of $\bar{\Omega}_1$ is set to $b_1(\boldsymbol{x}) = (1-t^2)\boldsymbol{w}^1 + t^2\boldsymbol{w}^2$, where $[\boldsymbol{w}^1, \boldsymbol{w}^2]$ is the corresponding edge of $\bar{\Lambda}$, and similarly for the other edges. This increases the density of the grid lines near the concave vertex as well as the lower left target vertex and also happens to reduce the conformal distortion. It remains future work to exploit this flexibility to further reduce the distortion of the mapping.

Another extension consists of imposing natural boundary conditions, which requires $\bar{\Lambda}$ to be the unit square, possibly with side nodes. This particular choice allows us to interpret the components of φ as two coordinate functions

$$x, y : \bar{\Omega}_0 \to [0,1].$$

We then solve for x by specifying Dirichlet boundary conditions on the two horizontal edges and Neumann boundary conditions on the two vertical edges of the unit square, and vice versa when solving for y. For the example in Figure 5.13, this approach produces more conformal results, but we have not further investigated this behaviour, yet.

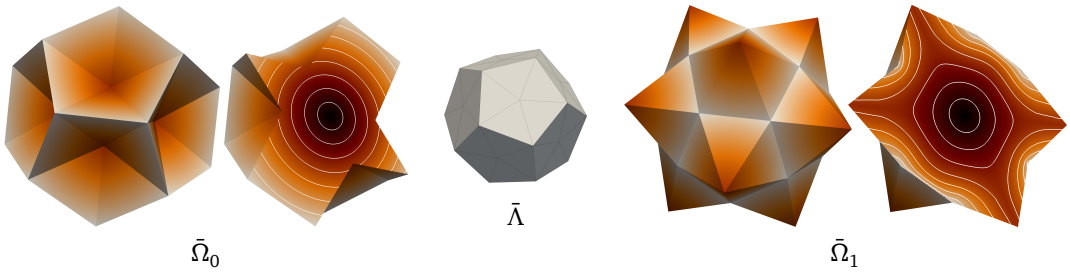


Figure 5.14. Using a smooth volumetric mapping to transfer a colour-valued signal from one polyhedron to another.

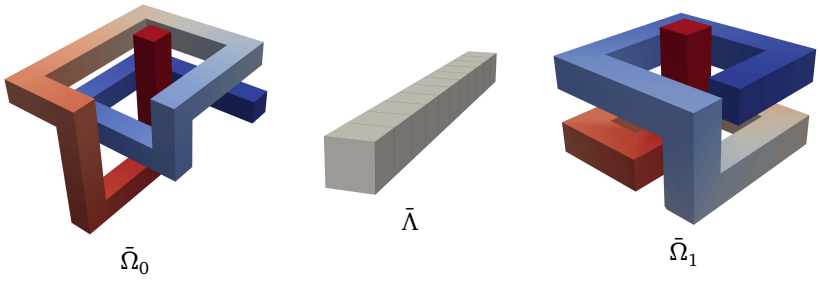


Figure 5.15. Smooth volumetric mapping between two polyhedral domains with quadrilateral faces.

5.5 Three dimensional mappings

This approach easily extends to volumetric mappings between two polyhedral domains in $\mathbb{R}^3$ with the same topology, as shown in Figure 5.14. Unfortunately, the Radó–Kneser–Choquet theorem does not hold in $\mathbb{R}^3$, and there is no mathematical guarantee that this mapping is bijective. However, in our experiments we did not observe any problems, and it even works for the rather extreme example in Figure 5.15.

5.6 Comparison to composite barycentric mappings

We now compare this method with the one described in Chapter 4. The example in Figure 5.7 suggests that this method gives maps with lower distortion. To support this hypothesis, we measured and compared the actual distortion of both maps.

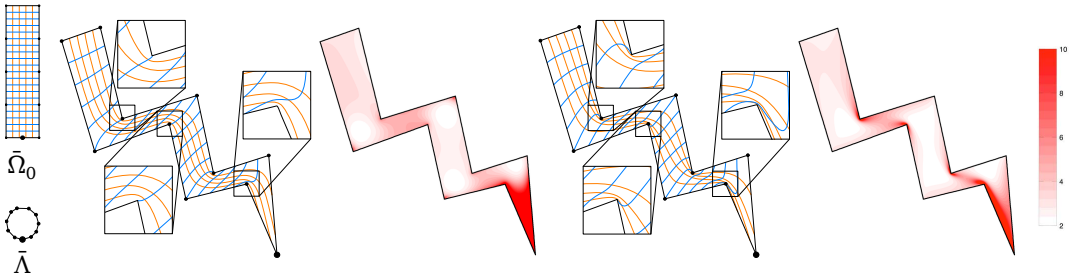


Figure 5.16. Comparison of the method in Chapter 5 (left) with the one in Chapter 4 (right). The energy plots show the conformal distortion per point, clamped at 10. The overall L_2-distortion of this map is 7.03, compared to 203.77 for the method in Chapter 4.

In Figure 5.16 we visualize the *conformal distortion* $\sigma_1/\sigma_2+\sigma_2/\sigma_1$ [Hormann and Greiner, 2000], where σ_1 and σ_2 are the singular values of the Jacobian of the map. The energy plot and the close-ups illustrate that this map behaves better around the concave corners of the target polygon, and the overall distortion is much lower. A similar behaviour can be observed in Figure 5.11, where the *isometric distortion* $\max(\sigma_1, 1/\sigma_2)$ [Sorkine et al., 2002] is considered. We performed similar tests with different polygons and energies, and since the results are all similar, we decided to show only a few prototypical examples.

For these comparisons we use BEM, because it provides the most reliable derivative data, which is needed for computing the distortion.

Chapter 6

Smoothness of barycentric coordinates

The bound (4.1) that guarantees bijectivity of the composite barycentric mappings exists only if the norm of gradient of the barycentric coordinates is bounded. It turns out that a large part of the three point family (1.8) is C^1 continuous for convex polygons (Section 6.1), which implies that the norm of the gradient is bounded. The only case ($p = 1$) in this family which is well defined for concave polygons is mean value coordinates, which are not smooth [Hormann and Floater, 2006]. However, we show that their directional derivative is bounded (Section 6.2.1) and provide numerical evidence of the boundedness of their gradient (Section 6.2.2).

The first two sections of this chapter are part of a larger work where we show that the whole three point family (1.8) is linear along the edges and well-defined for convex polygons. Since the weights w_i diverge to infinity as $\boldsymbol{x}$ approaches the edges, we introduce the products

$$\mathcal{A} = \prod_{j=1}^{n} A_j, \qquad \mathcal{A}_i = \prod_{\substack{j=1 \\ j\neq i}}^{n} A_j, \qquad \mathcal{A}_{i-1,i} = \prod_{\substack{j=1 \\ j\neq i-1,i}}^{n} A_j, \qquad i = 1,\dots,n,$$

of all areas A_j and those with one or two terms missing, respectively, and consider the weights

$$\tilde{w}_i = w_i\mathcal{A} = r_{i+1}^p \mathcal{A}_i - r_i^p B_i \mathcal{A}_{i-1,i} + r_{i-1}^p \mathcal{A}_{i-1}, \qquad i = 1,\dots,n, \tag{6.1}$$

and

$$\tilde{W} = W\mathcal{A} = \sum_{i=1}^{n} \tilde{w}_i. \tag{6.2}$$

Since $\mathcal{A}$ is well-defined and does not vanish over Ω, it is clear that the functions

$$\tilde{\phi}_i = \frac{\tilde{w}_i}{\tilde{W}}, \qquad i = 1, \dots, n, \tag{6.3}$$

coincide with ϕ_i on Ω, but they have the advantage of being well-defined over the open edges of P. We then proceed with a similar trick to extend the definition to the vertices for $p < 0$. In this case, the distance r_j converges to zero, so that r_j^p diverge to infinity. Therefore we introduce the products

$$\mathcal{R} = \prod_{j=1}^{n} r_j^{-p}, \qquad \mathcal{R}_i = \prod_{\substack{j=1 \\ j \neq i}}^{n} r_j^{-p}, \qquad i = 1, \dots, n,$$

and consider the functions

$$\hat{w}_i = \tilde{w}_i \mathcal{R} = \mathcal{R}_{i+1} \mathcal{A}_i - \mathcal{R}_i B_i \mathcal{A}_{i-1,i} + \mathcal{R}_{i-1} \mathcal{A}_{i-1}, \qquad i = 1, \dots, n, \tag{6.4}$$

and

$$\hat{W} = \tilde{W} \mathcal{R} = \sum_{i=1}^{n} \hat{w}_i.$$

Since $\mathcal{R}$ is well-defined and does not vanish over $\bar{\Omega}$, it is clear that the functions

$$\hat{\phi}_i = \frac{\hat{w}_i}{\hat{W}}, \qquad i = 1, \dots, n,$$

coincide with the $\tilde{\phi}_i$ on $\bar{\Omega}$, but have the advantage of being well-defined at the vertices of P.

6.1 Differentiability at the vertices

Since $\hat{\phi}_i$ are C^0 and linear along the edges, it implies that the directional derivative along the two adjacent edges to $\boldsymbol{v}^i = (x_i, y_i)$ is

$$[\boldsymbol{v}^{i-1}, \boldsymbol{v}^i] = \frac{y_i - y_{i-1}}{x_i - x_{i-1}}, \quad [\boldsymbol{v}^i, \boldsymbol{v}^{i+1}] = \frac{y_i - y_{i+1}}{x_i - x_{i+1}}.$$

To find the gradient $\nabla \phi_i(\boldsymbol{v}^i) = (\partial_x, \partial_y)^T$, we exploit the fact that the directional derivative at a point can be represented as the gradient at this point times the normalized direction. This leads to

$$\nabla \phi_i(\boldsymbol{v}^i) = -\frac{\nabla A_{i-1} + \nabla A_i}{C_i} = -\frac{\nabla B_i}{C_i}.$$

and similarly

$$\nabla\phi_i(\boldsymbol{v}^{i-1}) = \frac{\nabla A_{i-2}}{C_{i-1}}, \quad \nabla\phi_i(\boldsymbol{v}^{i+1}) = \frac{\nabla A_{i+1}}{C_{i+1}}, \quad \nabla\phi_i(\boldsymbol{v}^j) = 0, \quad j \neq i-1,\ i,\ i+1.$$

However, to prove C^1 continuity we first need to show that $\phi_i(\boldsymbol{v}^j)$ are totally differentiable at the vertices and the total derivative corresponds to the edge behaviour.

For the following proofs we recall some partial results in [Anisimov et al., n.d.]; for $p < 1$ we have

$$\lim_{\boldsymbol{v}\to\boldsymbol{v}^1} \frac{A_1(\boldsymbol{v})}{r_1(\boldsymbol{v})^p} = 0, \qquad \lim_{\boldsymbol{v}\to\boldsymbol{v}^1} \frac{A_n(\boldsymbol{v})}{r_1(\boldsymbol{v})^p} = 0, \tag{6.5}$$

$$0 \leq \frac{A_1}{r_1} \leq \frac{e_1}{2}, \qquad 0 \leq \frac{A_n}{r_1} \leq \frac{e_n}{2}, \tag{6.6}$$

and for $0 \leq p < 1$

$$0 \leq \frac{A_1}{r_1^p} = \frac{r_1 e_1 \sin\tau_1}{2r_1^p} \leq \frac{r_1^{1-p} e_1}{2} \tag{6.7}$$

for any $\boldsymbol{v} \in \Omega$.

Lemma 1. *Three-point coordinates are totally differentiable at the vertices for $p < 1$ and the derivatives are*

$$\nabla\phi_i(\boldsymbol{v}^j) = \frac{1}{C_j}\begin{cases} \nabla A_{i-2}, & j = i-1, \\ -\nabla B_i, & j = i, \\ \nabla A_{i+1}, & j = i+1, \\ 0, & \textit{otherwise}. \end{cases}$$

Proof. Without loss of generality, we consider only the first coordinate $\phi_1(\boldsymbol{v}_j)$, for which we need to show that

$$\lim_{\|\boldsymbol{v}\|\to 0} \frac{\phi_1(\boldsymbol{v}^j+\boldsymbol{v}) - \phi_1(\boldsymbol{v}^j) - \langle\nabla\phi_1(\boldsymbol{v}^j), \boldsymbol{v}\rangle}{\|\boldsymbol{v}\|} = \lim_{\|\boldsymbol{v}\|\to 0} \frac{\phi_1(\boldsymbol{v}^j+\boldsymbol{v}) - \phi_1(\boldsymbol{v}^j)}{\|\boldsymbol{v}\|} - \frac{\langle\nabla\phi_1(\boldsymbol{v}^j), \boldsymbol{v}\rangle}{\|\boldsymbol{v}\|} = 0.$$

We first observe that

$$\begin{aligned} \lim_{r_1\to 0} \frac{B_2}{r_1} &= \lim_{r_1\to 0} \frac{\|\boldsymbol{v}^1 - \boldsymbol{v}^3\| r_1 \sin\alpha_2}{2r_1} < r_3 \qquad \text{and} \\ \lim_{r_1\to 0} \frac{B_n}{r_1} &= \lim_{r_1\to 0} \frac{\|\boldsymbol{v}^1 - \boldsymbol{v}^{n-1}\| r_1 \sin\alpha_n}{2r_1} < r_{n-1}, \end{aligned} \tag{6.8}$$

where α_2 is the angle between the segments $(\boldsymbol{\nu}, \boldsymbol{\nu}^1)$ and $(\boldsymbol{\nu}^3, \boldsymbol{\nu}^1)$ and similarly for α_n. Using (6.5) and (6.6), we conclude that

$$\lim_{r_1 \to 0} \frac{A_1 A_n}{r_1^{p+1}} = \lim_{r_1 \to 0} \frac{A_1}{r_1^p} \frac{A_n}{r_1} = 0, \tag{6.9}$$

and

$$\begin{aligned} \lim_{r_1 \to 0} \frac{B_2 A_n}{r_1^{p+1}} &= \lim_{r_1 \to 0} \frac{B_2}{r_1} \frac{A_n}{r_1^p} = 0, \\ \lim_{r_1 \to 0} \frac{B_n A_1}{r_1^{p+1}} &= \lim_{r_1 \to 0} \frac{B_n}{r_1} \frac{A_1}{r_1^p} = 0. \end{aligned} \tag{6.10}$$

Finally we obtain

$$\lim_{r_1 \to 0} \frac{\mathcal{R}_1\big(A_1 \mathcal{A}_{n,1} + A_n \mathcal{A}_{n,1}\big)}{r_1 \hat{W}} \overset{(6.7)}{=} \lim_{r_1 \to 0} \frac{\mathcal{R}_1\big(\mathcal{A}_{n,1} e_1 \sin\tau_1 + \mathcal{A}_{n,1} e_n \sin\tau_n\big)}{2\hat{W}} = \frac{e_1 \sin\tau_1 + e_n \sin\tau_n}{2C_1}, \tag{6.11}$$

because $\lim_{\|\nu\| \to 0} \hat{W} = \mathcal{R}_1 C_1 \mathcal{A}_{n,1}$.

We first consider the case $j = 1$ and assume that $\boldsymbol{\nu}^1$ is at the origin, hence $\|\boldsymbol{\nu}\| = r_1$, by noticing that $\phi_1(\boldsymbol{\nu}^1) = 1$, we rewrite

$$\begin{aligned} &\lim_{\|\boldsymbol{\nu}\| \to 0} \frac{\phi_1(\boldsymbol{\nu}^1 + \boldsymbol{\nu}) - 1}{\|\boldsymbol{\nu}\|} \\ &= \lim_{r_1 \to 0} -\Bigg(r_1^{-p} \Bigg(\sum_{i=3}^{n-1} \big(\mathcal{R}_{1,i+1} \mathcal{A}_i - \mathcal{R}_{1,i} B_i \mathcal{A}_{i-1,i} + \mathcal{R}_{1,i-1} \mathcal{A}_{i-1} \big) + \mathcal{R}_{1,3} \mathcal{A}_2 + \mathcal{R}_{1,n-1} \mathcal{A}_{n-1} \Bigg) \\ &\quad - r_1^{-p} (\mathcal{R}_{1,2} B_2 \mathcal{A}_{1,2} + \mathcal{R}_{1,n} B_n \mathcal{A}_{n-1,n}) + \mathcal{R}_1\big(\mathcal{A}_n + \mathcal{A}_1\big) \Bigg) \big(r_1 \hat{W} \big)^{-1} \\ &= -\lim_{r_1 \to 0} A_1 A_n \frac{\sum_{i=3}^{n-1} (\mathcal{R}_{1,i+1} \mathcal{A}_{1,i,n} - \mathcal{R}_{1,i} B_i \mathcal{A}_{1,i-1,i,n} + \mathcal{R}_{1,i-1} \mathcal{A}_{1,i-1,n}) + \mathcal{R}_{1,3} \mathcal{A}_{1,2,n} + \mathcal{R}_{1,n-1} \mathcal{A}_{1,n-1,n}}{r_1^{p+1} \hat{W}} \\ &\quad + \lim_{r_1 \to 0} \frac{\mathcal{R}_{1,2} B_2 \mathcal{A}_{1,2} + \mathcal{R}_{1,n} B_n \mathcal{A}_{n-1,n}}{r_1^{p+1} \hat{W}} - \lim_{r_1 \to 0} \frac{\mathcal{R}_1\big(A_1 \mathcal{A}_{n,1} + A_n \mathcal{A}_{n,1}\big)}{\|\nu\| \hat{W}} \\ &\overset{(6.9)}{=} \lim_{r_1 \to 0} \frac{\mathcal{R}_{1,2} B_2 \mathcal{A}_{1,2} + \mathcal{R}_{1,n} B_n \mathcal{A}_{n-1,n}}{r_1^{p+1} \hat{W}} - \lim_{r_1 \to 0} \frac{\mathcal{R}_1\big(A_1 \mathcal{A}_{n,1} + A_n \mathcal{A}_{n,1}\big)}{\|\nu\| \hat{W}} \\ &\overset{(6.10)}{=} \lim_{r_1 \to 0} \frac{\mathcal{R}_1\big(A_1 \mathcal{A}_{n,1} + A_n \mathcal{A}_{n,1}\big)}{\|\nu\| \hat{W}} \overset{(6.11)}{=} \frac{e_1 \sin\tau_1 + e_n \sin\tau_n}{2C_1}. \end{aligned}$$

If we let the point $\boldsymbol{\nu}$ approach the origin along the unit direction $\boldsymbol{d}$, then the previous equation can be seen as the sum of the projections of the vectors

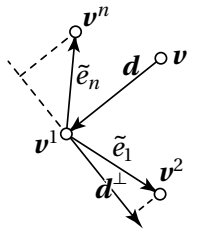


Figure 6.1. Illustration that $e_1 \sin\tau_1 + e_n \sin\tau_n$ is equivalent of projecting $\boldsymbol{v}^2 - \boldsymbol{v}^n$ on $\boldsymbol{d}^\perp$.

$\tilde{\boldsymbol{e}}_1 = \boldsymbol{v}^2 - \boldsymbol{v}^1$ and $\tilde{\boldsymbol{e}}_n = \boldsymbol{v}^n - \boldsymbol{v}^1$ on the vector $\boldsymbol{d}^\perp$. Since the projection is a linear operator, this is equivalent to projecting $\tilde{\boldsymbol{e}}_1 - \tilde{\boldsymbol{e}}_n = \boldsymbol{v}^2 - \boldsymbol{v}^n$ on the same direction, see Figure 6.1. Note that the minus sign comes from the fact that $\tilde{\boldsymbol{e}}_1$ and $\tilde{\boldsymbol{e}}_n$ lie on the opposite side of $\boldsymbol{d}$. We remark that $\nabla B_1 = (\boldsymbol{v}^n - \boldsymbol{v}^2)^\perp/2$ and compute

$$\lim_{\|\boldsymbol{v}\|\to 0} -\frac{\langle \nabla B_1, \boldsymbol{v}\rangle}{C_1\|\boldsymbol{v}\|} = -\frac{\langle \nabla B_1, \boldsymbol{d}\rangle}{C_1} = -\frac{\langle (\boldsymbol{v}_n - \boldsymbol{v}_2)^\perp, \boldsymbol{d}\rangle}{2C_1} = \frac{\langle \boldsymbol{v}_2 - \boldsymbol{v}_n, \boldsymbol{d}^\perp\rangle}{2C_1},$$

which coincides with the previous result and shows total differentiability for $j = 1$.

For the case $j = 2$ (and similarly for $j = n$) we again assume that $\boldsymbol{v}^2$ is at the origin, hence $\|\boldsymbol{v}\| = r_2$. We also recall that $\phi_1(\boldsymbol{v}^2) = 0$ hence

$$\frac{\phi_1(\boldsymbol{v}^2 + \boldsymbol{v})}{\|\boldsymbol{v}\|} = \frac{\mathcal{R}_2\mathcal{A}_1}{\|\boldsymbol{v}\|\hat{W}} - \frac{\mathcal{R}_{1,2}B_2\mathcal{A}_{1,2}}{r_2^{p+1}\hat{W}} + \frac{\mathcal{R}_{n,2}\mathcal{A}_n}{r_2^{p+1}\hat{W}}.$$

Again the last two terms converge to zero, hence

$$\lim_{\|\boldsymbol{v}\|\to 0} \frac{\phi_1(\boldsymbol{v}^2 + \boldsymbol{v})}{\|\boldsymbol{v}\|} = \lim_{\|\boldsymbol{v}\|\to 0} \frac{\mathcal{R}_2\mathcal{A}_1}{\|\boldsymbol{v}\|\hat{W}} \overset{(6.7)}{=} \lim_{\|\boldsymbol{v}\|\to 0} \frac{\mathcal{R}_2\mathcal{A}_{1,2}e_2\sin\tau_1}{2\hat{W}} = \frac{e_2\sin\tau_1}{2C_2},$$

for the same reason as before. We remark that $\nabla A_2 = \tilde{\boldsymbol{e}}_2^\perp/2$ hence

$$\lim_{\|\boldsymbol{v}\|\to 0} \frac{\langle \nabla A_2, \boldsymbol{v}\rangle}{C_1\|\boldsymbol{v}\|} = \frac{\langle \nabla A_2, \boldsymbol{d}\rangle}{C_1} = \frac{\langle \tilde{\boldsymbol{e}}_2^\perp, \boldsymbol{d}\rangle}{2C_1} = \frac{\langle \tilde{\boldsymbol{e}}_2, \boldsymbol{d}^\perp\rangle}{2C_1},$$

which again confirms that $\phi_1(\boldsymbol{v}^j)$ is totally differentiable for $j = 2$ and $j = n$.

For the remaining cases, that is $j = 3, \ldots, n-1$, we again assume that $\boldsymbol{v}^j$ is at the origin, hence

$$\frac{\phi_1(\boldsymbol{v}^j + \boldsymbol{v})}{\|\boldsymbol{v}\|} = \frac{\mathcal{R}_{2,j}\mathcal{A}_1}{r_j^{p+1}\hat{W}} - \frac{\mathcal{R}_{1,j}B_1\mathcal{A}_{n,1}}{r_j^{p+1}\hat{W}} + \frac{\mathcal{R}_{n,j}\mathcal{A}_n}{r_j^{p+1}\hat{W}},$$

where all terms converge to zero because all terms contain the product $A_j A_{j-1}$ which counterbalances the zero denominator r_j^{p+1}. □

Despite the fact that these results coincide with the edge behaviour, it is not enough to conclude that $\phi_i(\boldsymbol{v})$ is C^1 for $p < 1$. It remains only to show that the limit

$$\lim_{\boldsymbol{v}\to\boldsymbol{v}_j} \nabla\phi_i(\boldsymbol{v})$$

also converges to the same results since in the interior the function does not have poles.

Theorem 1. *Three-point coordinates are C^1 functions for $p < 1$.*

Proof. We remark that for any point in the interior and the open edges, ϕ_i has no poles, hence it is C_1. It remains to establish

$$\lim_{\boldsymbol{v}\to\boldsymbol{v}^j} \nabla\phi_i(\boldsymbol{v}) = \frac{1}{C_j}\begin{cases} \nabla A_{i-2}, & j = i-1, \\ -\nabla B_i, & j = i, \\ \nabla A_{i+1}, & j = i+1, \\ 0, & \text{otherwise}, \end{cases}$$

since in Lemma 1 we show total differentiability.

Since the weights w_i are not well defined at the vertices, we use

$$\hat{w}_i = \tilde{w}_i\mathcal{R} = \mathcal{R}_{i+1}\mathcal{A}_i - \mathcal{R}_i B_i \mathcal{A}_{i-1,i} + \mathcal{R}_{i-1}\mathcal{A}_{i-1}, \qquad i = 1,\dots,n,$$

in the definition of ϕ_i.

Without loss of generality we focus on the case $j = 1$. We first note that, since $\mathcal{A}_{l,k} = 0$ for $l \neq n$ and $l \neq 1$, we have

$$\mathcal{R}_k \nabla\mathcal{A}_l = \mathcal{R}_k \sum_{\substack{j=1 \\ j\neq l}}^{n} \nabla A_j \mathcal{A}_{l,j},$$

and we conclude that

$$\begin{aligned} &\lim_{\boldsymbol{v}\to\boldsymbol{v}_1} \mathcal{R}_1\nabla\mathcal{A}_1 = \mathcal{R}_1\nabla A_n\mathcal{A}_{n,1}, \\ &\lim_{\boldsymbol{v}\to\boldsymbol{v}_1} \mathcal{R}_1\nabla\mathcal{A}_n = \mathcal{R}_1\nabla A_1\mathcal{A}_{n,1}, \text{ and} \\ &\lim_{\boldsymbol{v}\to\boldsymbol{v}_1} \mathcal{R}_k\nabla\mathcal{A}_l = 0 \quad \text{for } l = 2,\dots,n-1. \end{aligned} \tag{6.12}$$

Similarly, because of (6.8), we have

$$\begin{aligned} \lim_{\nu\to\nu_1} \mathcal{R}_2 B_2 \nabla \mathcal{A}_{1,2} &= \lim_{\nu\to\nu_1} \mathcal{R}_2 B_2 \nabla A_n \mathcal{A}_{1,2,n} = 0 \qquad \text{and} \\ \lim_{\nu\to\nu_1} \mathcal{R}_n B_n \nabla \mathcal{A}_{n-1,n} &= \lim_{\nu\to\nu_1} \mathcal{R}_n B_n \nabla A_1 \mathcal{A}_{1,n-1,n} = 0. \end{aligned} \tag{6.13}$$

We now focus on the quantity

$$\nabla \mathcal{R}_k \mathcal{A}_l = \sum_{\substack{j=1 \\ j\neq k}}^{n} \nabla r_j^{-p} \mathcal{R}_{j,k} \mathcal{A}_l = p \sum_{\substack{j=1 \\ j\neq k}}^{n} \frac{s_j}{r_j^{p+2}} \mathcal{R}_{j,k} \mathcal{A}_l,$$

where for the case $k = 1$, since $\mathcal{R}_1$ does not contain r_1 and $\mathcal{A}_i$ converges to zero, we conclude that

$$\lim_{\nu\to\nu_1} \nabla \mathcal{R}_1 \mathcal{A}_l = 0. \tag{6.14}$$

For $k = 2, \ldots, n$ we split the sum in

$$\nabla \mathcal{R}_k \mathcal{A}_l = p \sum_{\substack{j=2 \\ j\neq k}}^{n} \frac{s_j}{r_j^{p+2}} \mathcal{R}_{j,k} \mathcal{A}_l + p \frac{s_1}{r_1^{p+2}} \mathcal{R}_{1,k} \mathcal{A}_l, \tag{6.15}$$

where the first term can be rewritten as

$$p \sum_{\substack{j=2 \\ j\neq k}}^{n} \frac{s_j}{r_j^{p+2}} \mathcal{R}_{j,k} \mathcal{A}_l = p \sum_{\substack{j=2 \\ j\neq k}}^{n} \frac{s_j}{r_j^{p+2}} \mathcal{R}_{1,j,k} \begin{cases} \mathcal{A}_{n,l} A_n r_1^{-p}, & \text{for } l \neq n \\ \mathcal{A}_{1,l} A_1 r_1^{-p}, & \text{for } l = n \end{cases}$$

from which we can deduce that

$$\lim_{\nu\to\nu_1} p \sum_{\substack{j=2 \\ j\neq k}}^{n} \frac{s_j}{r_j^{p+2}} \mathcal{R}_{j,k} \mathcal{A}_l = 0. \tag{6.16}$$

For the second term in (6.15) we first remark that even if s_1/r_1 is not defined at v_1, its limit is bounded. Following a similar idea as in (6.7) we have

$$0 \leq \frac{A_1}{r_1^{p+1}} = \frac{r_1 e_1 \sin\tau_1}{2 r_1^{p+1}} \leq \frac{e_1}{2 r_1^p} \qquad \text{and} \qquad 0 \leq \frac{A_n}{r_1^{p+1}} = \frac{r_1 e_n \sin\tau_n}{2 r_1^{p+1}} \leq \frac{e_n}{2 r_1^p}$$

which converge to zero for $p < 0$. These results allow us to conclude that for $p < 0$ and for $l \neq n$ (and similarly for $l = n$)

$$\lim_{\nu\to\nu_1} p \frac{s_1}{r_1^{p+2}} \mathcal{R}_{1,k} \mathcal{A}_l = \lim_{\nu\to\nu_1} p \frac{s_1}{r_1} \frac{A_n}{r_1^{p+1}} \mathcal{R}_{1,k} \mathcal{A}_{n,l} = 0.$$

For $0 < p < 1$ and $l = 2, \dots, n-1$ we rewrite the second term in (6.15) as

$$p \frac{s_1}{r_1^{p+2}} \mathcal{R}_{1,k} \mathcal{A}_l = p \frac{s_1}{r_1} \frac{A_n}{r_1^p} \frac{A_1}{r_1} \mathcal{R}_{1,k} \mathcal{A}_{n,1,l}, \tag{6.17}$$

which again converges to zero. With the previous result we conclude that for $l = 2, \dots, n-1$ we have

$$\lim_{v \to v_1} \nabla \mathcal{R}_k \mathcal{A}_l \overset{(6.16)}{=} 0. \tag{6.18}$$

By using the product rule on $\nabla \hat{w}_i$ we get

$$\nabla \hat{w}_i = \nabla \mathcal{R}_{i+1} \mathcal{A}_i + \mathcal{R}_{i+1} \nabla \mathcal{A}_i - \nabla \mathcal{R}_i B_i \mathcal{A}_{i-1,i} - \mathcal{R}_i \nabla B_i \mathcal{A}_{i-1,i} - \mathcal{R}_i B_i \nabla \mathcal{A}_{i-1,i} + \nabla \mathcal{R}_{i-1} \mathcal{A}_{i-1} + \mathcal{R}_{i-1} \nabla \mathcal{A}_{i-1}.$$

For $i = 2$

$$\begin{aligned}
&\lim_{v \to v_1} \nabla \hat{w}_2 \\
&= \lim_{v \to v_1} \left(\nabla \mathcal{R}_3 \mathcal{A}_2 + \mathcal{R}_3 \nabla \mathcal{A}_2 - \nabla \mathcal{R}_2 B_2 \mathcal{A}_{1,2} - \mathcal{R}_2 \nabla B_2 \mathcal{A}_{1,2} - \mathcal{R}_2 B_2 \nabla \mathcal{A}_{1,2} + \nabla \mathcal{R}_1 \mathcal{A}_1 + \mathcal{R}_1 \nabla \mathcal{A}_1 \right) \\
&\overset{(6.12)}{=} \lim_{v \to v_1} \left(\nabla \mathcal{R}_3 \mathcal{A}_2 - \nabla \mathcal{R}_2 B_2 \mathcal{A}_{1,2} - \mathcal{R}_2 B_2 \nabla \mathcal{A}_{1,2} - \mathcal{R}_2 B_2 \nabla \mathcal{A}_{1,2} + \nabla \mathcal{R}_1 \mathcal{A}_1 \right) + \mathcal{R}_1 \nabla A_n \mathcal{A}_{1,n} \\
&\overset{(6.13)}{=} \lim_{v \to v_1} \left(\nabla \mathcal{R}_3 \mathcal{A}_2 - \nabla \mathcal{R}_2 B_2 \mathcal{A}_{1,2} - \mathcal{R}_2 B_2 \nabla \mathcal{A}_{1,2} + \nabla \mathcal{R}_1 \mathcal{A}_1 \right) + \mathcal{R}_1 \nabla A_n \mathcal{A}_{1,n} \\
&\overset{(6.14)}{=} \lim_{v \to v_1} \left(\nabla \mathcal{R}_3 \mathcal{A}_2 - \nabla \mathcal{R}_2 B_2 \mathcal{A}_{1,2} - \mathcal{R}_2 B_2 \nabla \mathcal{A}_{1,2} \right) + \mathcal{R}_1 \nabla A_n \mathcal{A}_{1,n} \\
&\overset{(6.18)}{=} \lim_{v \to v_1} \left(- \mathcal{R}_2 \nabla B_2 \mathcal{A}_{1,2} \right) + \mathcal{R}_1 \nabla A_n \mathcal{A}_{1,n} \\
&\overset{(6.5)}{=} \mathcal{R}_1 \nabla A_n \mathcal{A}_{1,n}
\end{aligned}$$

and analogously for $i = 2, \dots, n$, we conclude that

$$\lim_{v \to v_1} \nabla \hat{w}_i = \begin{cases} \mathcal{R}_1 \nabla A_n \mathcal{A}_{n,1} & i = 2, \\ 0 & i = 3, \dots, n-1, \\ \mathcal{R}_1 \nabla A_1 \mathcal{A}_{n,1} & i = n. \end{cases} \tag{6.19}$$

This result and the fact that $\lim_{v \to v_1} B_1 = -C_1$ and $\nabla A_1 + \nabla A_n = \nabla B_1$ allow us to establish

$$\lim_{v \to v_1} \nabla \phi_1 = \lim_{v \to v_1} \frac{\nabla \hat{w}_1 - \phi_1 \sum_{j=1}^n \nabla \hat{w}_j}{\hat{W}} = - \lim_{v \to v_1} \frac{\sum_{j=2}^n \nabla \hat{w}_j}{\hat{W}} \overset{(6.19)}{=} - \frac{\nabla A_1 + \nabla A_n}{C_1} = - \frac{\nabla B_1}{C_1}.$$

Since $\lim_{v \to v_1} \nabla \hat{w}_1$ goes to infinity, we consider

$$\lim_{v \to v_1} \nabla \hat{w}_1 \hat{w}_m$$

for $m \neq 1$, where it remains to show that $\lim_{v \to v_1} \hat{w}_m / r_1$ is bounded. We first consider the case $m = 2$ ($m = n$ is similar) and recall (6.5), (6.6), and (6.8) to get

$$\lim_{v \to v_1} \frac{\hat{w}_2}{r_1} = \lim_{v \to v_1} \mathcal{R}_{1,3}\mathcal{A}_{1,2,n}\frac{A_1}{r_1^p}\frac{A_n}{r_1} - \mathcal{R}_{1,2}\mathcal{A}_{1,2,n}\frac{B_2}{r_1}\frac{A_n}{r_1^p} + \mathcal{R}_1\mathcal{A}_{1,n}\frac{A_n}{r_1} < \mathcal{R}_1\mathcal{A}_{1,n}e_n.$$

For $m = 3, \ldots, n-1$ by using the same arguments we obtain

$$\lim_{v \to v_1} \frac{\hat{w}_m}{r_1} = \lim_{v \to v_1} \mathcal{R}_{1,m+1}\mathcal{A}_{1,m,n}\frac{A_1}{r_1^p}\frac{A_n}{r_1} - \mathcal{R}_{1,m}B_m\mathcal{A}_{1,m-1,m,n}\frac{A_1}{r_1^p}\frac{A_n}{r_1} + \mathcal{R}_{1,m-1}\mathcal{A}_{1,m-1,n}\frac{A_1}{r_1^p}\frac{A_n}{r_1} = 0.$$

We now consider the term in (6.12) multiplied by $\hat{w}_m$ for $l = 1$ and $k \neq 1$,

$$\lim_{v \to v_1} \mathcal{R}_k \nabla \mathcal{A}_1 \hat{w}_m = \lim_{v \to v_1} \mathcal{R}_{1,k} \nabla \mathcal{A}_1 \frac{\hat{w}_m}{r_1^p} = \lim_{v \to v_1} \mathcal{R}_{1,k} \nabla \mathcal{A}_1 r_1^{1-p} \frac{\hat{w}_m}{r_1} = 0.$$

because of the previous result. Finally we consider the second term in (6.15) multiplied by $\hat{w}_m$ for $l = 1$ ($l = n$ follows) and $k \neq 1$

$$\lim_{v \to v_1} p \frac{s_1}{r_1^{p+2}} \mathcal{R}_{1,k}\mathcal{A}_1 \hat{w}_m = \lim_{v \to v_1} p \frac{s_1}{r_1}\frac{A_n}{r_1^p}\mathcal{R}_{1,k}\mathcal{A}_{1,n}\frac{\hat{w}_m}{r_1} = 0,$$

for the same reason as before.

We can now summarize the results for $m \neq 1$ using all previous findings

$$\lim_{v \to v_1} \nabla \hat{w}_1 \phi_m = \lim_{v \to v_1} \nabla \hat{w}_1 \frac{\hat{w}_m}{\hat{W}} = 0. \tag{6.20}$$

This final result allows us to establish for $i = 2$ (and similarly $i = n$)

$$\lim_{v \to v_1} \nabla \phi_2 = \lim_{v \to v_1} \frac{\nabla \hat{w}_2 - \phi_1 \sum_{j=1}^n \nabla \hat{w}_j}{\hat{W}} \overset{(6.20)}{=} \lim_{v \to v_1} \frac{\nabla \hat{w}_2}{\hat{W}} \overset{(6.19)}{=} \frac{\nabla A_n}{C_1}$$

and for $i = 3, \ldots, n-1$

$$\lim_{v \to v_1} \nabla \phi_i = \lim_{v \to v_1} \frac{\nabla \hat{w}_i - \phi_1 \sum_{j=1}^n \nabla \hat{w}_j}{\hat{W}} \overset{(6.20)}{=} \lim_{v \to v_1} \frac{\nabla \hat{w}_i}{\hat{W}} \overset{(6.19)}{=} 0.$$

□

The reasoning in the previous theorem does not extend to the case $p \geq 1$, because (6.17) diverges for $p \geq 1$, which is in line with the result in [Hormann and Floater, 2006]. The following examples show that the three-point coordinates for $p \geq 1$ are not C^1 continuous over Ω. Note that the polygon in the example is chosen to keep the calculations as simple as possible, but that we observed the same phenomena for all other polygons that we tested.

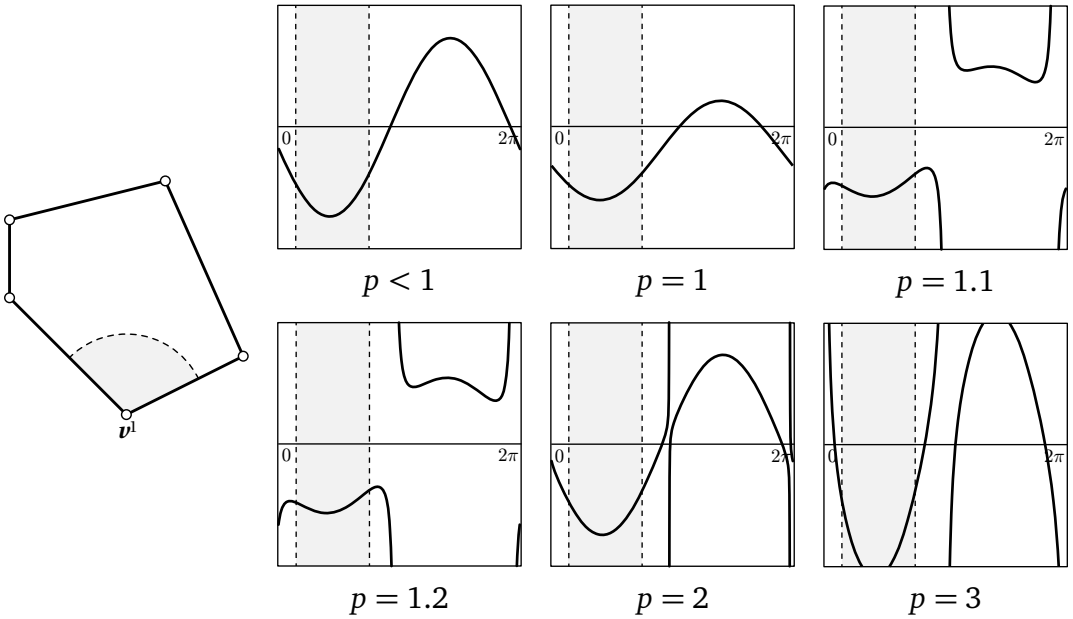


Figure 6.2. Directional derivatives of ϕ_1 at $\boldsymbol{v}_1$ for different values of $\boldsymbol{x}$. The grey area shows the portion of the angle inside P. Note that the intersections of the dashed lines across the different plots is at the same points, because it corresponds to the edge directions where ϕ_1 is linear independently from $\boldsymbol{x}$.

Example 1. Let us examine the gradient of the three-point coordinates for $p \geq 1$ over the quadrilateral P with vertices $\boldsymbol{v}^1 = (0,0)$, $\boldsymbol{v}^2 = (0,1)$, $\boldsymbol{v}^3 = (-1/4,1)$, and $\boldsymbol{v}^4 = (-1,0)$, and study the convergence of the x partial derivative along the x-axis

$$\lim_{x \to 1^-} \frac{\partial \phi_1}{\partial x}(0,x).$$

According to the behaviour on the edges and the previous results the limit should converge to 1 independently of p. However for $p = 1$ it turns out that

$$\frac{\partial \phi_1}{\partial x}(0,x) = 2\frac{16x^3 + 16x^2\sqrt{x^2+1} + 8x^2 + 8x\sqrt{x^2+1} + 21x + 11\sqrt{x^2+1} + 5}{\sqrt{x^2+1}(3x + 2 + 5\sqrt{x^2+1})^2},$$

whose limit for $x \to 1^-$ is $(14 + 10\sqrt{2})/(15 + 10\sqrt{2}) \neq 1$. Similarly for $p = 2$

$$\frac{\partial \phi_1}{\partial x}(0,x) = \frac{8(6x^2 + 20x + 11)}{5(3x+5)^2},$$

which converges to $37/40$ for $x \to 1^-$.

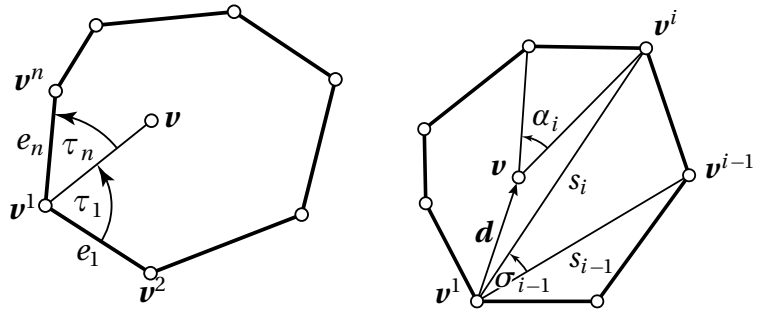


Figure 6.3. Notation for angles and edge lengths and directional derivative.

Despite the fact that the coordinates are not C^1 for $p \geq 1$ at the vertices, their directional derivative shows a reasonable behaviour, see Figure 6.2. In fact, for $p = 1$ the directional derivative is a shifted sine function (Section 6.2.1).

6.2 Mean value coordinate gradient behaviour

6.2.1 Directional derivative

Since mean value coordinates are not C^1 [Hormann and Floater, 2006] at the vertices, we now consider the directional derivative

$$\mu_i = \nabla_d \phi_i(\boldsymbol{v}^1) = \lim_{t \to 0} \frac{\phi_i(\boldsymbol{v}^1 + \boldsymbol{d} \cdot t) - \phi_i(\boldsymbol{v}^1)}{t}$$

of the coordinate ϕ_i along a direction $\boldsymbol{d}$ at $\boldsymbol{v}^1$.

By exploiting the properties (1.2a) and (1.2c), μ_i can be split into two cases

$$\mu_i = \begin{cases} \lim_{t \to 0} \dfrac{\phi_1(\boldsymbol{v}^1 + \boldsymbol{d} \cdot t) - 1}{t} = -\lim_{t \to 0} \dfrac{\sum_{j=2}^{n} w_j(\boldsymbol{v}^1 + \boldsymbol{d} \cdot t)}{tW} & \text{if } i = 1, \\ \lim_{t \to 0} \dfrac{\phi_i(\boldsymbol{v}^1 + \boldsymbol{d} \cdot t)}{t} = \lim_{t \to 0} \dfrac{w_i(\boldsymbol{v}^1 + \boldsymbol{d} \cdot t)}{tW} & \text{if } i \neq 1. \end{cases} \tag{6.21}$$

Without loss of generality we assume that the first edge e_1 of P is aligned with the x-axis and the first vertex lies at the origin. Let $\Sigma = \sum_{i=2}^{n-1} \sigma_i$ (Figure 6.3), then it is easy to see that $\Sigma = \tau_1 + \tau_n$, too. Since $\tau_1 = \alpha$ (Figures 6.3) it follows that

$$\tau_n - \tau_1 = (\Sigma - \tau_1) - \tau_1 = \Sigma - 2\tau_1 = \Sigma - 2\alpha.$$

Similarly, when $\boldsymbol{v}$ approaches $\boldsymbol{v}^1$ we conclude that

$$\begin{aligned}\lim_{\boldsymbol{v}\to\boldsymbol{v}_1}\alpha_i(\boldsymbol{v}) &= \begin{cases}\pi-\tau_1 & \text{if } i=1,\\ \pi-\tau_n & \text{if } i=n,\\ \sigma_i & \text{if } i=2,\dots,n-1,\end{cases}\\ \lim_{\boldsymbol{v}\to\boldsymbol{v}_1} r_i(\boldsymbol{v}) &= s_i, \quad i=2,\dots,n,\end{aligned} \tag{6.22}$$

where $s_i = \|\boldsymbol{v}^i-\boldsymbol{v}^1\|$. For simplicity, instead of using the formula (1.8), we use the trigonometric representation [Floater et al., 2006, Equation (30)] of the weight functions

$$w_i = \frac{2}{r_i}\left(\frac{r_{i+1}^{p-1}-r_i^{p-1}\cos\alpha_i}{\sin\alpha_i}+\frac{r_{i-1}^{p-1}-r_i^{p-1}\cos\alpha_{i-1}}{\sin\alpha_{i-1}}\right) \tag{6.23}$$

where the angles α_i and the distances r_i, $i=1,\dots,n$ are depicted in Figure 6.3.

Proposition 1. *For all $i\neq 1$ the following limit holds*

$$\lim_{t\to 0} w_i(\boldsymbol{v}^1+\boldsymbol{d}\cdot t) = \begin{cases} 2s_i^{-1}\big(\tan(\sigma_i/2)+\tan(\sigma_{i-1}/2)\big) & \text{if } i=3\dots n-1,\\ 2s_2^{-1}\big(\tan(\sigma_2/2)+\cot(\tau_1/2)\big) & \text{if } i=2,\\ 2s_n^{-1}\big(\cot(\tau_n/2)+\tan(\sigma_{n-1}/2)\big) & \text{if } i=n.\end{cases}$$

Proof. For $p=1$ we rewrite (6.23) as

$$w_i = \frac{2}{r_i}\left(\frac{1-\cos\alpha_i}{\sin\alpha_i}+\frac{1-\cos\alpha_{i-1}}{\sin\alpha_{i-1}}\right) = \frac{2}{r_i}(\tan(\alpha_i/2)+\tan(\alpha_{i-1}/2)).$$

We recall that $\tan(\pi/2-\alpha)=\cot(\alpha)$ hence by using (6.22) the result follows. □

Now we study the behaviour of the denominator.

Proposition 2. *The following limit holds*

$$\lim_{t\to 0} t w_i(\boldsymbol{v}^1+\boldsymbol{d}\cdot t) = \begin{cases} 2\big(\cot(\tau_1/2)+\cot(\tau_n/2)\big) & \text{if } i=1\\ 0 & \text{if } i=2,\dots,n.\end{cases}$$

Proof. For $i=2,\dots,n$ it trivially follows from Proposition 1. For $i=1$ we have

$$\lim_{t\to 0} t w_1(\boldsymbol{v}^1+\boldsymbol{d}\cdot t) = 2\Big(\frac{1+\cos\tau_1}{\sin\tau_1}+\frac{1+\cos\tau_n}{\sin\tau_n}\Big) = 2\big(\cot(\tau_1/2)+\cot(\tau_n/2)\big).$$

□

We are now ready to summarise the result and show the behaviour of the directional derivative at the vertices.

Theorem 2. *The directional derivative μ_i with $i = 1, \dots, n$ of mean value coordinate at the first vertex of $\bar{\Omega}$ is of the form*

$$\mu_i = a + b \sin(\alpha + \varphi)$$

where a, b, and φ are constant.

Proof. For $i = 1$ we exploit the previous propositions to obtain

$$\mu_1 = -\lim_{t\to 0} \frac{\sum_{j=2}^{n} w_j}{tW} = -\frac{s_2^{-1}\cot(\tau_1/2) + s_n^{-1}\cot(\tau_n/2)}{\cot(\tau_1/2) + \cot(\tau_n/2)} - \frac{\sum_{i=2}^{n-1}(s_i^{-1} + s_{i+1}^{-1})\tan(\sigma_i/2)}{\cot(\tau_1/2) + \cot(\tau_n/2)}.$$

By using trigonometric identities and previous results, the first term can be simplified to $a_1 + b_1 \sin(\Delta\tau/2)$ with

$$a_1 = \frac{s_2 + s_n}{2s_2 s_n} \qquad \text{and} \qquad b_1 = \frac{s_n - s_2}{2s_2 s_n \sin(\Sigma/2)}.$$

For the second term we note that the numerator $\mathcal{N}$ is independent from the angle α, hence the term can be simplified as $a_2 + b_2 \sin(\Delta\tau/2)$ with

$$a_2 = -\mathcal{N}/2\cot(\Sigma/2) \qquad \text{and} \qquad b_2 = \frac{\mathcal{N}}{2\sin(\Sigma/2)}.$$

Using these two results and the formula for the linear combination of sine functions we conclude that $\mu_1 = a + b\sin(\alpha + \varphi)$ with

$$a = -(a_1 + a_2), \qquad b = \sqrt{b_1^2 + b_2^2}, \qquad \text{and} \qquad \varphi = -(\Sigma/2 + \arctan(b_2/b_1)).$$

For $i = 2$, and similarly for $i = n$, we have

$$\mu_2 = \lim_{t\to 0} \frac{w_2}{tW} = \frac{\cot(\tau_1/2) + \tan(\sigma_2/2)}{2s_2(\cot(\tau_1/2) + \cot(\tau_n/2))} = \sin(\tau_1/2)\sin(\tau_n/2)\frac{\cot(\tau_1/2) + \tan(\sigma_2/2)}{2s_2\sin(\Sigma/2)}.$$

Again this formula can be simplified using trigonometric identities and the rule of linear combination of sine functions to $\mu_2 = a + b\sin(\alpha + \varphi)$, where

$$a = \frac{1 - \cot(\Sigma/2)\tan(\sigma_2/2)}{2s_2}, \qquad b = -\frac{1}{2s_2\sin(\Sigma/2)|\cos(\sigma_2/2)|}, \qquad \text{and} \qquad \varphi = -\frac{\Sigma + \sigma_2}{2}.$$

For the remaining cases (*i.e.*, $i = 3, \dots, n-1$)

$$\mu_i = \lim_{t\to 0} \frac{w_i}{tW} = \frac{\tan(\sigma_{i-1}/2) + \tan(\sigma_i/2)}{s_i(\cot(\tau_1/2) + \cot(\tau_n/2))} = \frac{(\tan(\sigma_{i-1}/2) + \tan(\sigma_i/2))\sin(\tau_1/2)\sin(\tau_n/2)}{s_i\sin(\Sigma/2)}$$

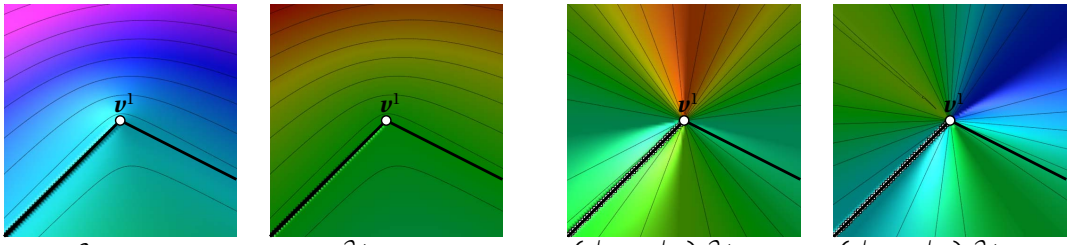


Figure 6.4. Numerical evidence of boundedness of the two terms in the gradient of mean value coordinates.

Using the same rules we conclude that $\mu_i = a + b\sin(\alpha + \varphi)$ with

$$a = -\cot(\Sigma/2)\frac{\tan(\sigma_{i-1}/2) + \tan(\sigma_i/2)}{2s_i}, \text{ and}$$
$$\varphi = -\frac{\Sigma + \pi}{2}.$$

□

6.2.2 Gradient bound

We present preliminary unpublished results and numerical evidence that suggests that the norm of gradient of mean value coordinates is bounded. Following the result in (4.2) and noticing that

$$-\frac{w_i \nabla r_i}{r_i W} = \frac{\phi_i s_i}{r_i^2}$$

we can rewrite

$$\nabla\phi_i = \sum_{j\neq i}\left(\frac{\phi_j}{r_i}\frac{\phi_i s_i}{r_i} - \frac{\phi_i}{r_j}\frac{\phi_j s_j}{r_j}\right) + \sum_{j\neq i-1,i}\left(\frac{\phi_j \nabla t_{i-1}}{W r_i} - \frac{\phi_i \nabla t_j}{W r_j}\right) + \left(\frac{\phi_{i-1}}{r_i} - \frac{\phi_i}{r_{i-1}}\right)\frac{\nabla t_{i-1}}{W}$$
$$+ \sum_{j\neq i,i+1}\left(\frac{\phi_j \nabla t_i}{W r_i} - \frac{\phi_i \nabla t_{j-1}}{W r_j}\right) + \left(\frac{\phi_{i+1}}{r_i} - \frac{\phi_i}{r_{i+1}}\right)\frac{\nabla t_i}{W}, \tag{6.24}$$

where all the terms seem bounded when $\boldsymbol{x}$ converges to the vertices.

Without loss of generality we consider the case $i = 1$, therefore, for $j \neq 1$

$$\lim_{x\to v_1}\frac{\phi_j}{r_1} = \lim_{x\to v_1}\frac{t_{j-1} + t_j}{\sum_{k=1}^{n}\frac{r_j r_1}{r_k}(t_{k-1} + t_k)} = \frac{t_{j-1} + t_j}{r_j(t_n + t_1)} = \frac{w_j}{t_n + t_1},$$

since all the terms in the sum converge to zero. In what concerns $\phi_j s_j / r_j$, we remark that its norm is equal to $|\phi_j|$ for any j. From these two results we can conclude that the first sum in (6.24) is bounded.

For the remaining terms we need to show that for $j = 2, \ldots, n-1$

$$\lim_{x \to v_1} \frac{\nabla t_j}{W} = \lim_{x \to v_1} \frac{\nabla(s_j \times s_{j+1}) - t_j\big(\nabla r_j r_{j+1} + r_j \nabla r_{j+1} + \nabla(s_j \cdot s_{j+1})\big)}{W(r_j r_{j+1} + s_j \cdot s_{j+1})}$$

and

$$\lim_{x \to v_1} \left(\frac{\phi_2}{r_1} - \frac{\phi_1}{r_2} \right) \frac{\nabla t_1}{W} \qquad \text{and} \qquad \lim_{x \to v_1} \left(\frac{\phi_n}{r_1} - \frac{\phi_1}{r_n} \right) \frac{\nabla t_n}{W}$$

are bounded, for which we have only numerical evidence, so far (Figure 6.4).

Chapter 7

Bijective complex mappings

We exploit the complex formulation of barycentric coordinates (Section 1.2) and barycentric mapping to create bijective mappings. Note that the results presented in this chapter are unpublished and in an early stage.

Our idea is to use higher order polynomials to "increase the resolution" of the complex barycentric mapping (1.12) near the regions where it is not bijective. In practice we can modify the "speed" of the mapping along the edges by renouncing the linear function $\sigma_i(t)$ in (1.13).

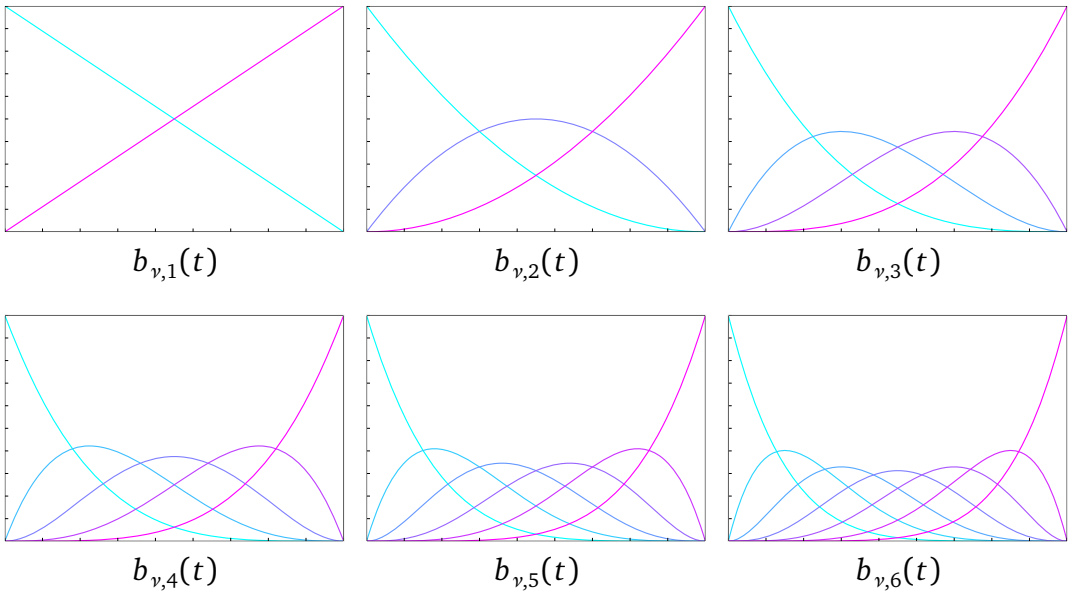


Figure 7.1. Bernstein basis function for different degrees.

We use the Bernstein basis [Lorentz, 1953] (Figure 7.1) of degree d

$$b_{\nu,d}(t) = \binom{d}{\nu} t^{\nu}(1-t)^{d-\nu}$$

to express the functions σ_i as

$$\sigma_i(t) = \sum_{\nu=0}^{d} b_{\nu,d}(t) c_{\nu}^{i}.$$

This formulation has two advantages. First, by letting $c_0^i = 0$ and $c_d^i = 1$ we ensure that $\sigma_i(0) = 0$ and $\sigma_i(1) = 1$. Second, by imposing that

$$0 < c_{\nu-1}^{i} < c_{\nu}^{i} < 1 \qquad \nu = 1, \ldots, d-1 \tag{7.1}$$

we can enforce monotonicity of σ_i.

As for the real case, to ensure bijectivity we need that

$$\min_{z \in P} \det(J_h(z)) > 0, \tag{7.2}$$

where J_h is the Jacobian of the complex mapping. To compute it we suggest to express γ_i and s_i in the complex polar form and to calculate the partial derivatives with respect to the radius ρ and angle ψ. For instance,

$$\gamma_i = \frac{|e_i| e^{\mathrm{i}\alpha_i}}{|z_i - \rho| \sin(\alpha_i - \psi_i)} \left(e^{-\mathrm{i}\psi_{j+1}} - e^{-\mathrm{i}\psi_j} \right),$$

where α_i is the angle of the i-th edge.

This trick leads to

$$\nabla h(z) = \begin{pmatrix} \dfrac{\partial h(z)}{\partial \Re(z)} \\ \dfrac{\partial h(z)}{\partial \Im(z)} \end{pmatrix} = \begin{pmatrix} -\dfrac{\partial h(\rho,\psi)}{\partial \psi} \dfrac{\Im(z)}{\rho^2} + \dfrac{\partial h(\rho,\psi)}{\partial \rho} \dfrac{\Re(z)}{\rho} \\ \dfrac{\partial h(\rho,\psi)}{\partial \psi} \dfrac{\Re(z)}{\rho^2} + \dfrac{\partial h(\rho,\psi)}{\partial \rho} \dfrac{\Im(z)}{\rho} \end{pmatrix},$$

and therefore

$$\det(J_h(z)) = \Re\left(\frac{\partial h(z)}{\partial \Re(z)} \right) \Im\left(\frac{\partial h(z)}{\partial \Im(z)} \right) - \Re\left(\frac{\partial h(z)}{\partial \Im(z)} \right) \Im\left(\frac{\partial h(z)}{\partial \Re(z)} \right).$$

In order to solve this problem we formulate an optimization procedure. Let $C = \{c_{\nu}^{i}\} \in \mathbb{R}^{n,d-1}$ be the matrix of coefficients of the functions σ_i, $i = 1, \ldots, n$; we wish to find the optimal $C^{\star}$ that minimizes (7.2) under the constrains (7.1).

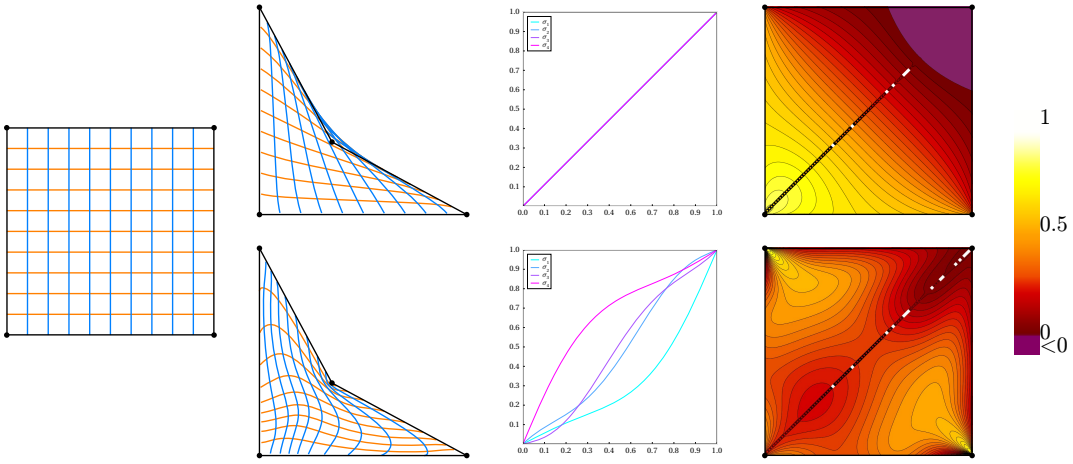


Figure 7.2. Visualization of a complex mapping before and after optimization.

Unfortunately, this problem is highly non-linear and non-smooth, therefore we need to reformulate it.

We introduce the linear function k and sample the polygon with a dense grid of m points $\bar{z}_j$. Thus our optimization problem becomes

$$\begin{aligned} \underset{k}{\text{minimize}} \quad & -k \\ \text{subject to} \quad & 0 < c^i_{\nu-1} < c^i_\nu < 1 \qquad i = 1,\dots,n,\ \nu = 1,\dots,d-1 \\ & \det(J_h(\bar{z}_j)) \geq k \qquad\qquad j = 1,\dots,m, \end{aligned} \tag{7.3}$$

which produces a bijective map when $k > 0$.

The advantage of this formulation is that both the objective function and the non-linear constraints are smooth. We initialize this optimization process with $c^i_{\nu-1} = \nu/d$ and $k = \min_{j=,\dots,m} \det(J_h(\bar{z}_j))$. Note that the constraints (7.1) allow to minimize only concave functions, therefore we “invert” σ_i as

$$\sigma_i(t) = 1 - \sum_{\nu=1}^{d-1} b_{\nu,d}(1-t)c^i_\nu + b_{d,d}(1-t).$$

Then we solve the optimization problem (7.3) for all 2^n possible combinations of “inverted” and not-inverted σ_i and choose the one with the largest k.

Figure 7.2 shows how this optimization procedure restores bijectivity for the given resolution for a mapping between two quadrilaterals.

Part II

Applications

Chapter 8

Application to computer graphics

The definition of a map $\boldsymbol{f}$ between the interior of two polygons (or more in general between two polyhedra) is a basic problem that arises in different fields of computer graphics, including image warping, parametrization and finite element simulations.

The main application in computer graphics is *image warping*, where a *source image* is deformed into a *target image* by modifying an initial control polygon $\bar{\Omega}_0$ to obtain a target polygon $\bar{\Omega}_1$, as shown in Figure 8.1. Image warping can be naturally extended to three dimensions, so called *shape deformation*, where an object is contained in a source polyhedron $\bar{\Omega}_0$ called "cage". As for the two-dimensional application, the "cage" is deformed into a target polyhedron $\bar{\Omega}_1$ and the contained object is deformed accordingly. In other words, the application consists of mapping the volume of $\bar{\Omega}_0$ to $\bar{\Omega}_1$, as illustrated in Figure 8.2.

In image warping, *bijectivity* of $\boldsymbol{f}$ prevents fold-overs, which cause undesirable artefacts in almost all applications, Figure 8.1 shows an example of lack of bijectivity in image warping.

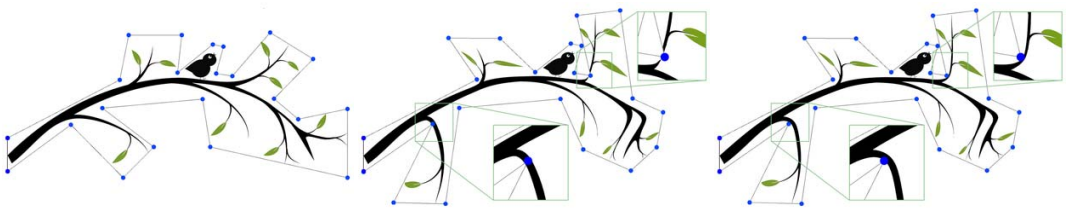


Figure 8.1. Image warping between a source and a target polygon. If the warping is not bijective the image contains fold-overs, visible in the close-ups in the middle picture.

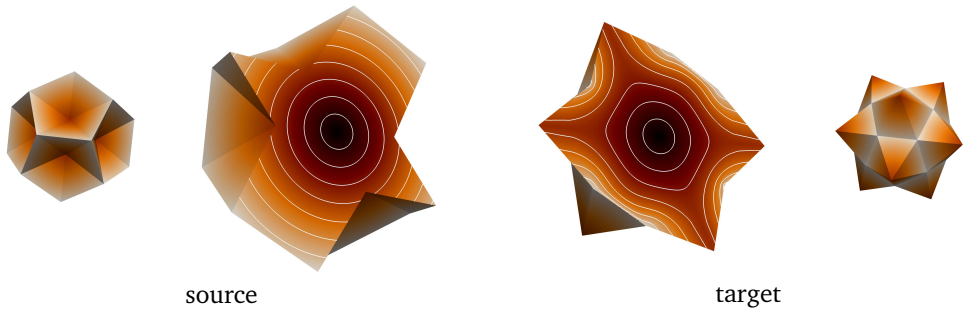


Figure 8.2. Bijective mapping between two polyhedra.

Another application consists of *surface cross parameterization*, where we want to transfer quantities (*e.g.*, textures, functions, vector fields, *etc.*) from a continuous parametric surface patch $\gamma_0 \colon \bar{\Omega}_0 \subset \mathbb{R}^2 \to \mathbb{R}^3$ to another patch $\gamma_1 \colon \bar{\Omega}_1 \subset \mathbb{R}^2 \to \mathbb{R}^3$. In this case we create a smooth bijective map $\boldsymbol{f}$ between the two parametric domains $\bar{\Omega}_0$ and $\bar{\Omega}_1$ and construct the overall mapping $h = \gamma_1 \circ \boldsymbol{f} \circ \gamma_0^{-1}$. Figure 8.3 shows an example of surface cross parameterization, where γ_0 is a paraboloid and γ_1, is a surface of revolution and we transfer colour information.

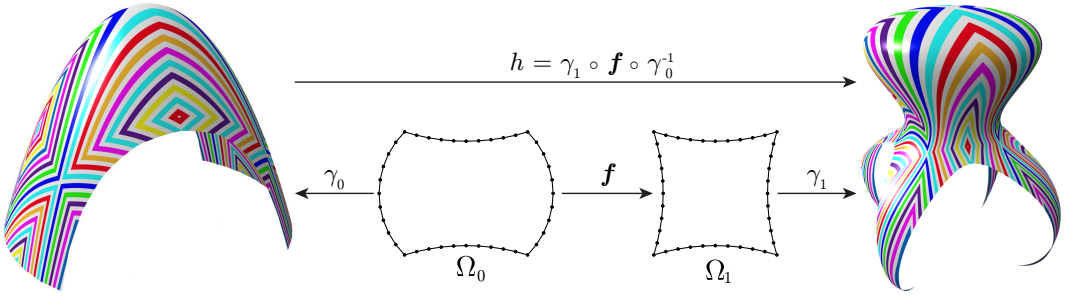


Figure 8.3. Bijective mapping between two smooth parametric surface patches.

Chapter 9

Parametric finite elements

Computational mechanics and computational science in general combine aspects from mathematics, physics, engineering, and computer science. Their application consists of simulating physical phenomena. These simulations are applied in a large variety of fields such as medicine, aeronautics, or mechanics. For instance, the finite element method has been used in cardiac surgery [Hashim and Richens, 2006], car crash test simulations [Teng et al., 2008], or flight simulations [Richards and NASA Dryden Flight Research Center, 1997].

Simulating physical phenomena often requires dealing with complex geometric objects, generated, for instance, by computer aided design (CAD) software or captured from real life objects or organisms (*e.g.*, 3D scans, MRI, *etc.*). When focusing on the finite element method (FEM), such highly complex geometric descriptions need to be represented in a sufficiently accurate way. This is the case because the accuracy of the geometric representation influences the approximation error of the discrete solution of a partial differential equation. For instance, in a fabrication simulation the shape of the mechanical part determines the result of the simulation.

Due to the central nature of this geometric approximation, its influence on the approximation error has been studied for curved boundary of iso-parametric discretizations [Ciarlet and Raviart, 1972; Scott, 1973, 1975] and for contact problems [Kikuchi and Oden, 1988]. More recent research focuses on numerical studies for elliptic and Maxwell problems [Xue et al., 2005], and for different approximation spaces [Bertrand et al., 2014b,a].

Unfortunately, the ability to reproduce the exact geometry is not considered in most state-of-the-art tools, since they are based on a one-way connection between geometric information and simulation environment. In fact, the detailed geometric description is used only in a pre-processing phase where a mesh is

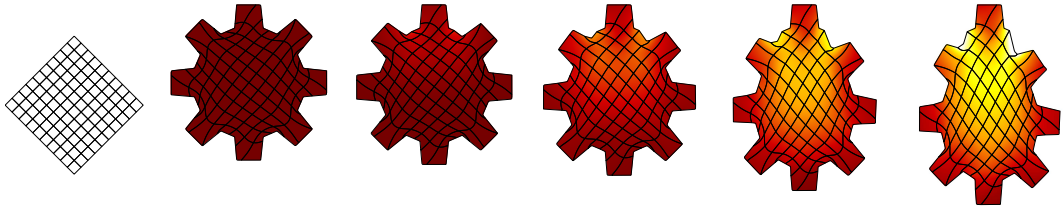


Figure 9.1. Transient non-linear elasticity simulation for a warped quad-mesh with compressible-neo-Hookean material. The elastic gear is subject to vertical body forces (gravity) and has a fixed tooth on the top boundary. The colour represents the von Mises stress for the solution at different time-steps t.

generated. This pre-processing phase usually consists of creating an approximate representation of the input geometry. For CAD surfaces, this consists of meshing the smooth surface to create a polygon-mesh representation, whereas for captured surfaces the pre-processing phase consists of mesh coarsening, because the mesh resolution influences the size of the system describing the model problem and affects the overall computational cost.

During a simulation, the approximation of the solution might not be accurate enough to represent large variations. This problem is usually solved by means of adaptive refinement strategies, such as h-refinement [Bey, 1995; Bramble et al., 2002] (*e.g.*, adding new triangles to the mesh) and p-refinement [Melenk and Wohlmuth, 2001] (*e.g.*, increasing the order of the local basis functions). When using such strategies, the original surface should be taken into consideration and recovered while refining [Dörfler and Rumpf, 1998]. However, the geometric information is usually discarded in the simulation environment. In other words, the mesh is generated within a modelling software and used in the simulation environment without considering the original surface, thus preventing a better surface approximation. Consequently, adaptive refinement is rarely accompanied by an increase in the accuracy of the shape.

The most famous example of parametric finite elements is isogeometric analysis [Hughes et al., 2005] (IGA). This method was born to narrow the gap between the analysis and the modelling in CAD software. Its main advantage is that it avoids the meshing phase, since the problem is solved directly on the CAD geometry, such as non-uniform rational B-splines (NURBS). Moreover, the method implies using splines as basis functions for both the geometry and the finite element space, hence the name isogeometric. This construction has many advantages, such as the exact and efficient representation of the CAD geome-

try, the easy h and p refinement, its use of smooth basis functions with compact support, and its direct employment in multigrid solvers.

In fact this method uses parametric elements. Most CAD geometries are made by patches and each patch is the image of a spline map from a parametric domain, which usually is a (trimmed) unit square. With this set-up the basis functions (which are also splines) are defined over the parametric domain.

One of the main drawbacks of this method is that the geometries arising from the modelling software may be invalid, for instance they may contain holes or self-intersections [Lian et al., 2012]. One possibility to deal with this problem is to use T-splines [Sederberg et al., 2004; Buffa et al., 2010; da Veiga et al., 2011], however, CAD modelling software does not implement them. A second advantage of T-splines regards the refinement, which remains local. Another important open problem of IGA regards the construction of volumetric spline parametrization from the modelled surface [Aigner et al., 2009; Martin and Cohen, 2010; Li et al., 2013]. Moreover, IGA approximations, similarly to many mesh-free methods, lead to complications in the imposition of essential boundary conditions, which can be either imposed in a weak sense [Bazilevs et al., 2010], or least-squares satisfied in the strong sense [Hughes et al., 2005].

Additionally, when dealing with three-dimensional shapes, CAD models usually describe only the boundary. Creating a NURBS volume parameterization is a complex task, for which many different approaches exist. For instance, some of them require particular shapes [Aigner et al., 2009], need special geometric information [Martin and Cohen, 2010], or do not reproduce the surface exactly [Li et al., 2013].

An alternative to IGA is the NURBS-enhanced finite element method (NEFEM) [Sevilla et al., 2011] that allows exploiting CAD geometries to describe exactly the boundary of the geometry. However, this method requires creating a parameterization mesh and a special handling of the boundary, which according to [Sevilla et al., 2011] is still an open problem.

The problem of dealing with exact geometries has been extensively studied for CAD geometries by the IGA community. Unfortunately, a similar study for surface meshes is missing. For this reason, we focus on the *exact representation* provided by surface meshes, and present the construction of a bijective volume parametrization from arbitrarily shaped domains to arbitrarily shaped meshes.

One possible solution is to exploit the bijective mappings $\boldsymbol{f}$ into the finite elements simulation, as shown in Figure 9.1. This new discretization enables exploiting exact geometric descriptions (*e.g.*, splines or surface meshes) together with strategies employed in standard finite element simulations. This discretization has the advantage of decoupling the geometry and the approximation space

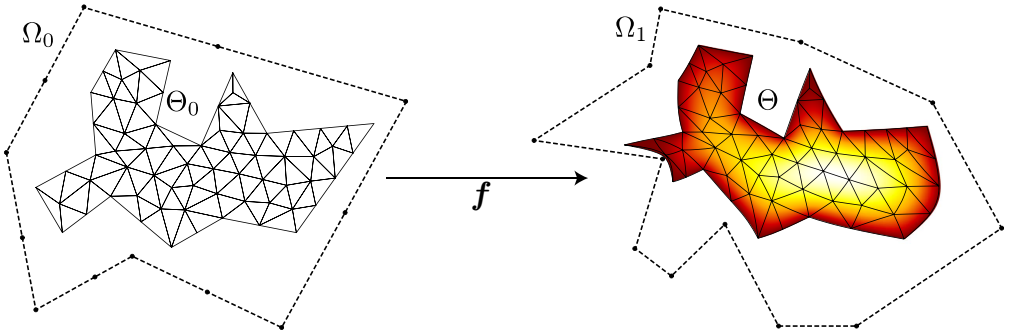


Figure 9.2. Overview of parametric finite elements with bijective mappings, with colour-coded solution of the Poisson problem (9.1) on a 2D warped domain Θ, with zero boundary conditions and constant right-hand side.

allowing for sub/iso/super-parametric elements.

9.1 Parametric finite elements with bijective mappings

Let us consider the standard Poisson problem

$$-\Delta u = g, \qquad u|_{\partial\Theta} = h, \tag{9.1}$$

where Θ is the *computational domain*, and h describes the boundary values. In contrast with the classical construction,

$$\Theta = \boldsymbol{f}(\Theta_0) \subseteq \Omega_1$$

is given by the image of a sufficiently smooth *bijective mapping*

$$\boldsymbol{f} : \Omega_0 \to \Omega_1.$$

In this context we call $\Theta_0 \subseteq \Omega_0$ the *source domain*, $\Omega_0 \subset \mathbb{R}^d$ the *parametrization domain*, and $\Omega_1 \subset \mathbb{R}^d$ the *parametrization image*. Figures 9.2 and 9.3 show an overview of our construction and the solution of the Poisson problem (9.1).

Using the same derivations as in Section 5.1.1, we rewrite (9.1) in its weak form, which is: find u such that

$$\int_\Theta \nabla u \cdot \nabla v = \int_\Theta g v \qquad \forall v.$$

Using $\boldsymbol{f}$, we express the previous integral with respect to the *source domain* Θ_0. Considering that $u(\boldsymbol{x}) = u(\boldsymbol{f}(\boldsymbol{x}_0))$ and $v(\boldsymbol{x}) = v(\boldsymbol{f}(\boldsymbol{x}_0))$ where $\boldsymbol{x} \in \Theta$ and

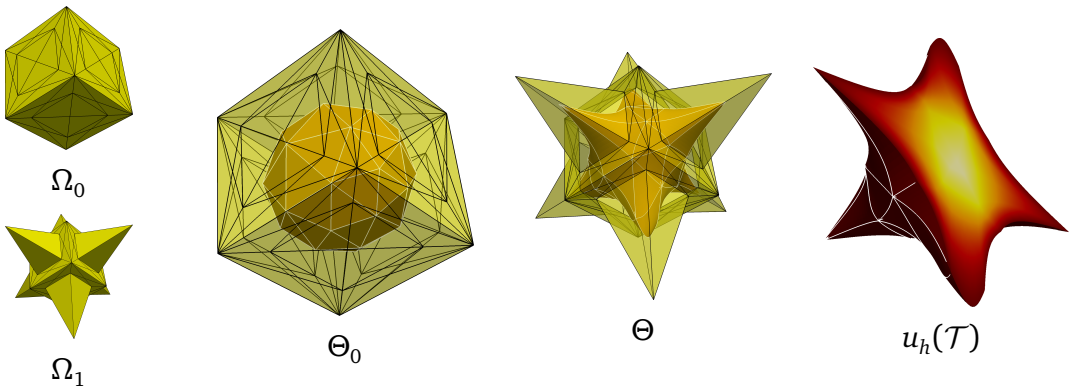


Figure 9.3. Overview of the parametric finite elements with bijective mappings, with colour-coded solution of the Poisson problem (9.1) on a 3D warped domain Θ, with zero boundary conditions and constant right-hand side.

$\boldsymbol{x}_0 = \boldsymbol{f}^{-1}(\boldsymbol{x}) \in \Theta_0$, and applying change of variables in the integrals, we rewrite the weak form: find u such that

$$\int_{\Theta_0} J_f^{-T} \nabla u \cdot J_f^{-T} \nabla v \ \det(J_f) = \int_{\Theta_0} g v \ \det(J_f) \qquad \forall v, \tag{9.2}$$

where J_f is the Jacobian matrix of the mapping $\boldsymbol{f}$.

In order to solve this problem, we represent the computational domain Θ by a warped mesh $\mathcal{T} = \boldsymbol{f}(\mathcal{T}_0)$. Note that, as described in (9.2), the bijective mapping warps the entire volume, creating warped elements. Again, as in Section 5.1.1, we introduce the basis

$$B_1, \ldots, B_m : \Theta \to \mathbb{R},$$

over $\mathcal{T}$ and approximate

$$u \approx u_h = \sum_{i=1}^{m} c_i B_i,$$

where c_i are real coefficients. By choosing the test space as the function space, we discretize (9.2) as

$$\sum_{i=0}^{m} u_i \int_{\Theta} J_f^{-T} \nabla B_i \cdot J_f^{-T} \nabla N_j \ \det(J_f) = \sum_{i=0}^{m} g_i \int_{\Theta} B_i B_j \ \det(J_f) \qquad \forall j = 1, \ldots, m,$$

which can be represented in the classical matrix form

$$\boldsymbol{L}\boldsymbol{u} = \boldsymbol{M}\boldsymbol{f}, \tag{9.3}$$

with $\boldsymbol{u} = [u_1, \ldots, u_m]^T$ and $\boldsymbol{g} = [g_1, \ldots, g_m]^T$

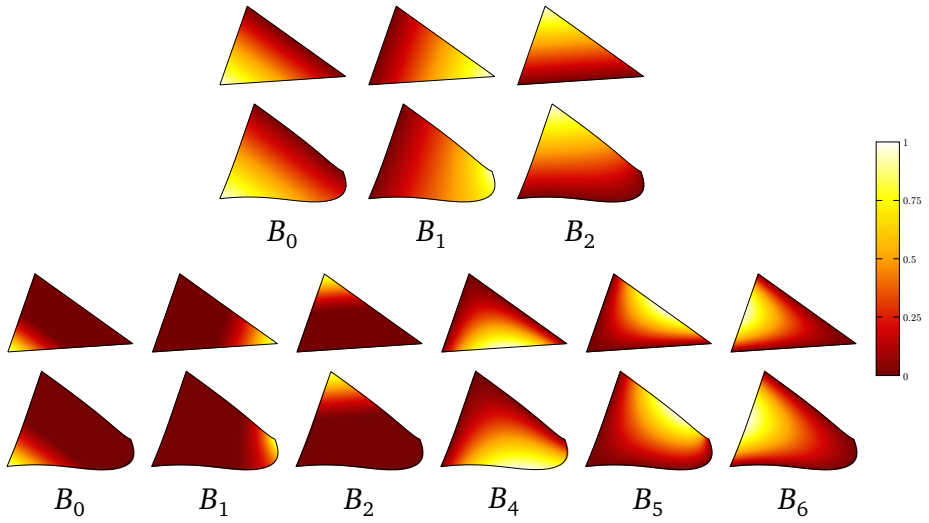


Figure 9.4. The standard linear and quadratic shape functions N_i on the element of the *source mesh* and the corresponding warped element.

As usual, we introduce the reference element $\hat{E}$ and the transformation G from the reference element to the corresponding element E_0 in the source domain. We perform the quadrature in $\hat{E}$, using quadrature points $\hat{\boldsymbol{x}}_k \in \hat{E}, \boldsymbol{x}_k = G(\hat{\boldsymbol{x}}_k)$ with the respective *quadrature weights* $\alpha_k \in \mathbb{R}$, $k = 1, \dots, K$. Figure 9.5 shows all the geometric transformations from the reference element $\hat{E}$ to the warped element E. We denote by $\hat{B}_i$ the basis functions on the reference element and by J_G the Jacobian of G. This allows assembling the local matrices for the element E

$$\begin{aligned} L^E_{i,j} &= \sum_{k=1}^{K} \beta_k \, J^{-T}(\boldsymbol{x}_k) \, \nabla \hat{B}_i(\hat{\boldsymbol{x}}_k) \cdot J^{-T}(\boldsymbol{x}_k) \nabla \hat{B}_j(\hat{\boldsymbol{x}}_k), \\ M^E_{i,j} &= \sum_{k=1}^{K} \beta_k \, \hat{B}_i(\hat{\boldsymbol{x}}_k) \, \hat{B}_j(\hat{\boldsymbol{x}}_k), \end{aligned} \tag{9.4}$$

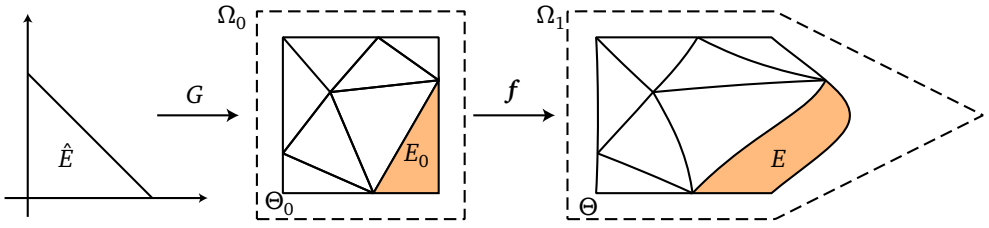


Figure 9.5. Overview of the geometric transformations from the reference element $\hat{E}$ to the source element $E_0 \in \mathcal{T}_0$ and to the warped element $E \in \mathcal{T}$.

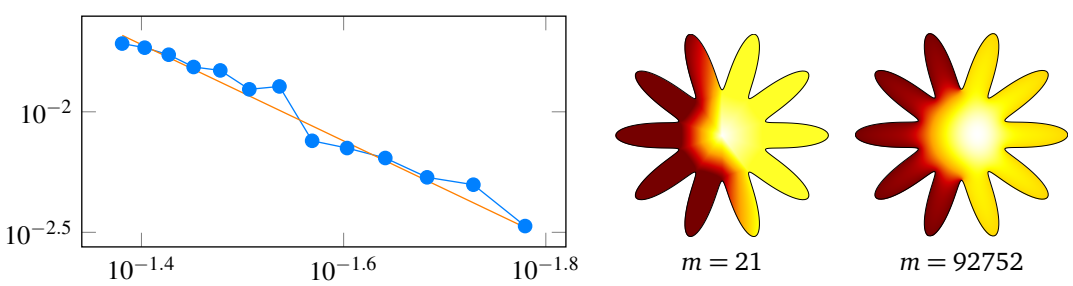


Figure 9.6. Left: visualization of $e(u_h)$ against the mesh size h, where the straight line shows the quadratic trend. Right: solution of the Poisson problem for different numbers of nodes m.

where $J(\boldsymbol{x}_k) = J_f(\boldsymbol{x}_k)J_G(\hat{\boldsymbol{x}}_k)$ and $\beta_k = \alpha_k \det(J(\boldsymbol{x}_k))|\hat{E}|$, with $|\hat{E}|$ the volume of $\hat{E}$. These local contributions are then gathered to compute the matrices $\boldsymbol{L}$ and $\boldsymbol{M}$.

Note that the weak formulation and the assembly procedures are very similar to classical finite elements. In fact, the only difference is the usage of the geometric terms depending on the bijective mapping $\boldsymbol{f}$, such as J_f, which contributes to $J = J_f J_G$. As in standard the FEM, the choice of the basis is independent from the geometric description, leading to super/sub/iso-parametric approximations. In our method the geometric description is given by the mapping $\boldsymbol{f}$, which is usually non-linear, so that our discretization falls into the category of super-parametric elements.

If we assume that $\boldsymbol{f}(\mathcal{T}_0)$ describes the exact geometry, then the geometric error is zero. However, the error in the solution is also connected to the choice of the approximation space and the shape of the elements. This error is influenced by the Jacobian J_f of the bijective mapping. We estimate it by means of the condition number

$$\kappa = \sup_{\boldsymbol{x}_0 \in \Theta_0, \boldsymbol{x} \in \Theta} \|J_f(\boldsymbol{x}_0)\| \, \|J_f^{-1}(\boldsymbol{x})\| \tag{9.5}$$

as in standard parametric finite elements estimates [Braess, 2007; Brenner and Scott, 2008].

9.2 Numerical experiments

We restrict our study to super-parametric discretizations based on composite mean value mappings (Chapter 4) with linear Lagrange elements ($\mathbb{P}_1$). For our experiments the analytical solution is unknown, hence we estimate it by computing a reference solution u on a very fine mesh $\mathcal{T}_f$. To evaluate the quality of our

discretization and of the standard discretization, we compute different solutions u_h for several mesh sizes h.

9.2.1 Convergence

The solution is expected to converge quadratically in $L^2(\Omega)$ to the exact one with respect to the mesh size h for classical FEM with linear elements for H^2-regular problems. Hence, we study the convergence related to our approach by measuring the approximation error as

$$e(u_h) = \|P(u_h) - u\|_{L^2(\mathcal{T}_f)},$$

where P is the L^2-projection operator [Wohlmuth, 1998; Krause and Zulian, 2015] (the assembly of P is performed in the parametrization domain). Similar to the standard FEM, our method shows a quadratic convergence behaviour for the Poisson problem, as illustrated in the plot in Figure 9.6. Despite the fact that the computation is always performed in the exact geometry, the approximation error is not zero because of the piecewise polynomial approximation of the solution, which is visible for a mesh with small m and disappears for larger m.

9.2.2 Comparison

We compare our method with the standard finite element discretization for a simple 2D problem (Figure 9.8), an extreme 2D problem (Figure 9.9), and for a realistic 3D shape (Figure 9.10). Since for the standard finite element discretization, the boundary of $\mathcal{T}$ differs from Θ, we measure

$$r(u_h) = \left| \frac{\|u_h\|_{L^2(\mathcal{T})}}{\|u\|_{L^2(\mathcal{T}_f)}} - 1 \right|$$

to estimate the approximation error [Luo et al., 2001].

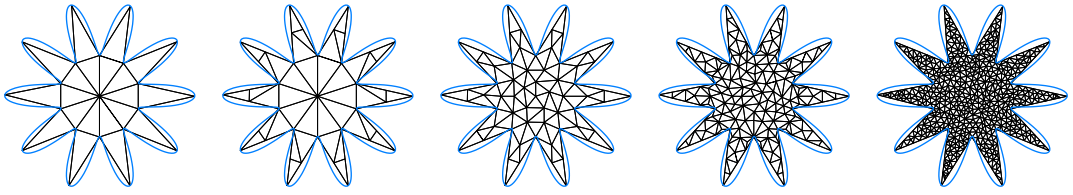


Figure 9.7. Mesh refinement without shape recovery. Even at fine resolution (last image) we do not recover the original shape (blue polygon).

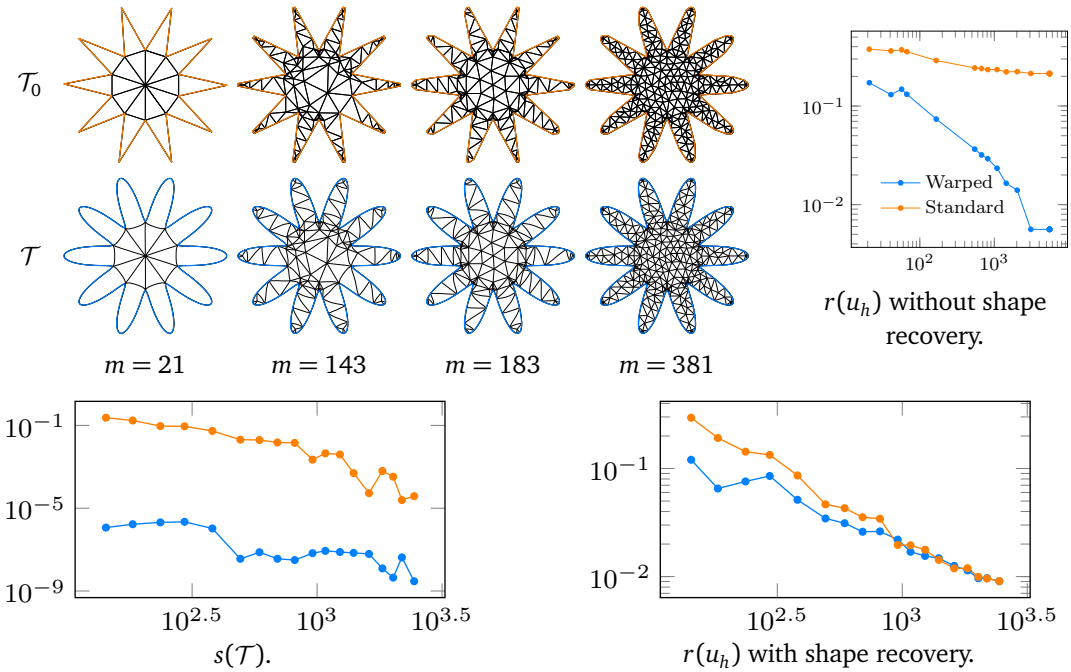


Figure 9.8. Source meshes $\mathcal{T}_0$ with boundary Ω_0 (first row), warped meshes $\mathcal{T}$ used by our method (second row), and convergence plots against different numbers of degrees of freedom m (last row).

In classical finite element simulations the original shape is usually not recovered when performing mesh refinement as shown in Figure 9.7. For this reason, $r(u_h)$ does not converge to zero for the standard solution, while our approach converges (left plots in Figures 9.8, 9.9, and 9.10).

In order to better understand this behaviour, we measure the actual geometric deviation with

$$s(\mathcal{T}) = \|1\|_{L^2(\mathcal{T})},$$

which corresponds to the volume of the mesh (note that $s(\mathcal{T})$ is equivalent to the square root of the sum of the entries of the mass-matrix). We compute the volume by means of numerical quadrature, which might introduce errors, since our discretization consists of warped elements. For the standard discretization, when refining the mesh without recovering the shape, the volume trivially stays constant. Hence, in order to have a fair comparison, we increase the shape accuracy while refining the mesh to ensure that the shape of the domain also converges to the exact one. The behaviour of $s(\mathcal{T})$ shows that our discretization has almost zero geometrical error independently of h, while the standard discretization has higher geometrical error (middle plots in Figures 9.8, 9.9, and 9.10).

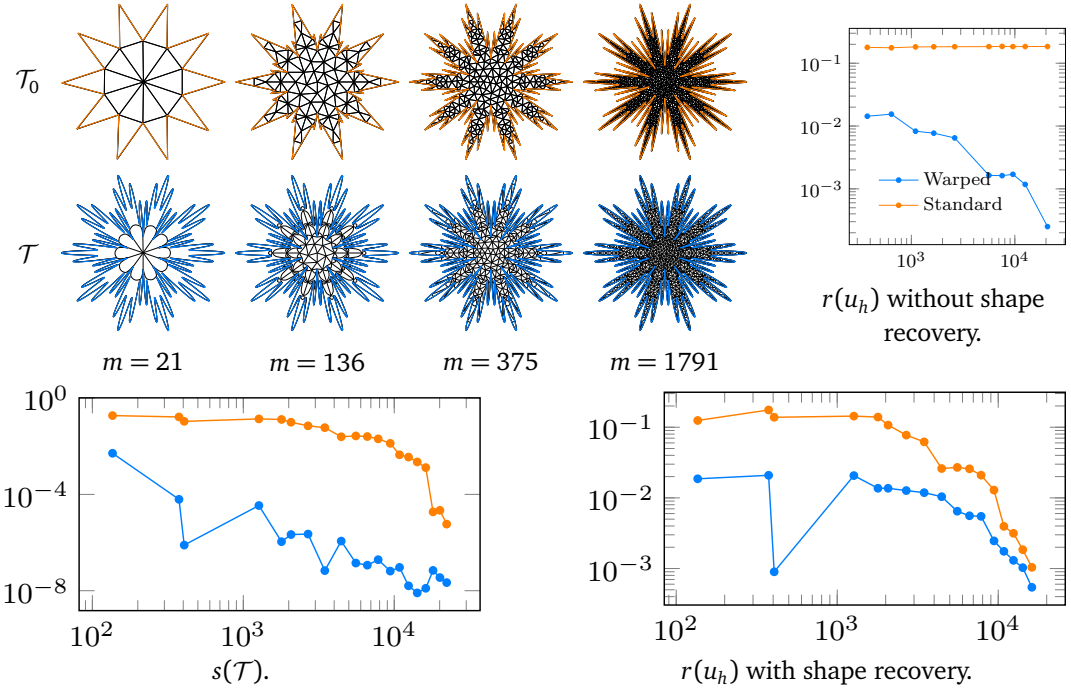


Figure 9.9. Source meshes $\mathcal{T}_0$ with boundary Θ_0 (first row), warped meshes $\mathcal{T}$ used by our method (second row), and convergence plots against different numbers of degrees of freedom m (last row).

In order to investigate how the approximation error is influenced by the geometrical error, we measure $r(u_h)$ for our method and classical finite elements with shape recovery. Our discretization always has a smaller approximation error compared to the standard discretization (right plots in Figures 9.8, 9.9, and 9.10). This is due to the fact that our approach allows solving the problem in the exact geometry, even at low resolutions.

9.2.3 Conditioning

For solution methods such as iterative solvers, the condition number κ of the stiffness matrix plays an important role for the convergence rate [Bathe and Wilson, 1976]. In order to understand how our discretization affects the condition number, we compute κ for the discrete Laplace operator $\boldsymbol{L}$ with respect to different mesh sizes h for both our discretization and the standard one. Because of the influence of the bijective mapping b, as shown in (9.5), our discretization has a slightly larger condition number. Figure 9.11 shows that $\kappa(\boldsymbol{L})$ behaves similarly for both discretizations which suggests that iterative solvers perform nearly as

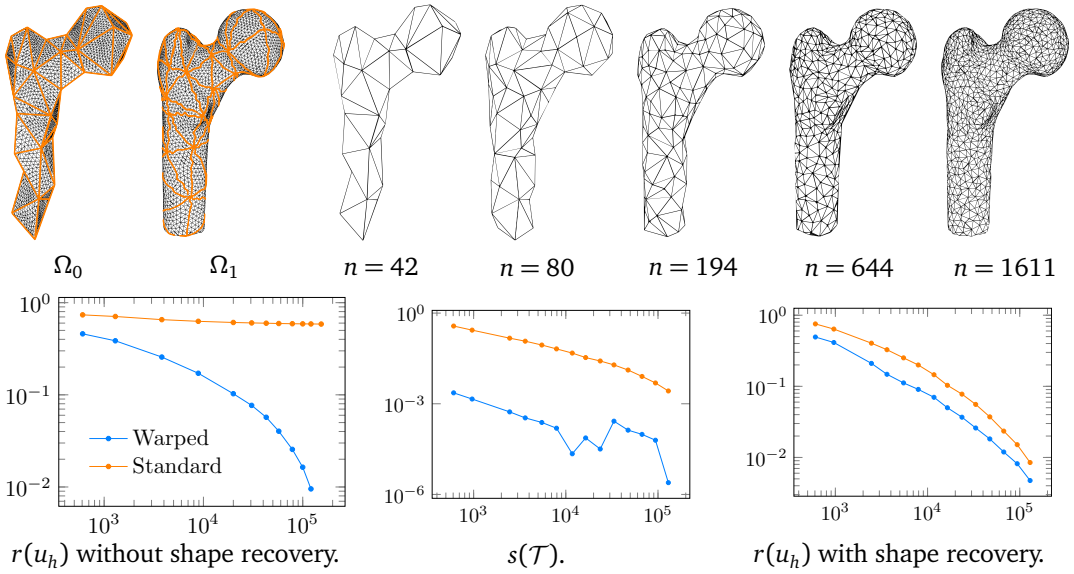


Figure 9.10. Convergence plots against different numbers of degrees of freedom m for a 3D experiment.

well for our discretization as for the standard one.

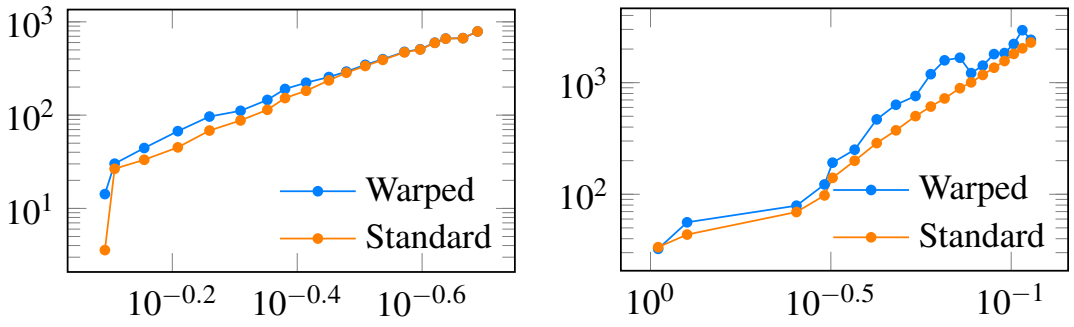


Figure 9.11. Condition number of the discrete Laplace operator $\kappa(\boldsymbol{L})$ against the mesh size h for the examples in Figure 9.8 (left) and Figure 9.9 (right).

Chapter 10

Meshing and polygonal finite elements

To represent the physical domain, the classical approach consists of tetrahedralizing [Si, 2015] (or triangulating [Shewchuk, 1996]) the domain, where the mesh size (*i.e.*, number of tetrahedra) determines the accuracy of the domain description. This strategy has two main advantages: first, it is easy to implement; second, simplices are easier to treat (*e.g.*, the transformation between two tetrahedra is affine and unique). However, linear tetrahedra elements are known to be more prone to the so-called *mesh locking* effect due to their linear behaviour [Babuška and Suri, 1992]. Besides, hexagons have better properties for finite element analysis [D'Azevedo, 2000; Shepherd and Johnson, 2008] and better describe tubular shapes such as body parts [Bommes et al., 2012; Panozzo, 2015]. To avoid such problems efforts are made to generate hexahedral meshes, but pure hexahedral meshes are very difficult, if not impossible, to create [Shepherd and Johnson, 2008].

Additionally, for complex cases such as micro structures it is impossible to generate pure hexahedral meshes, which makes simulations based on such structures infeasible. Another typical example consists of transient simulations, where the geometry undergoes large deformations (*e.g.*, large non-linear elastic deformations). In these cases re-meshing is often required to restore mesh quality. In the case of hexahedral meshes this aspect poses a critical problem, since many meshing techniques are (partially) manual.

One possible solution to deal with this problem consists of using parametric finite elements (Chapter 9) which allow to "move" the meshing problems into the parametrization domain.

An alternative approach consists of using meshes composed by polyhedral

elements, thus deviating from the standard choice of "hat" basis functions and using barycentric coordinates (Section 1.1) as basis [Gout, 1985; Sukumar and Tabarraei, 2004]. In this case the classical assembly of the physical operators is not applicable since it is difficult to create a bijective map from the reference element (a regular polyhedron with n faces) to the physical domain. Therefore, the assembly is performed directly on the physical domain.

10.1 Polygonal finite elements

The classical formulation of the finite elements method with tetrahedral elements and polynomial basis functions has been generalized in recent years. One direction consists of enriching the finite element basis and using the so-called *extended finite element method* [Belytschko et al., 2009]. For instance, by enriching the standard basis with jumping functions we can describe discontinuous functions which are usually used in crack modelling.

The second direction consists of giving up the simplices and defining a basis for polyhedral elements which is no longer unique. Many approaches have been proposed, the most famous ones being *mimetic finite difference* [Lipnikov et al., 2014], its generalization the *virtual element method* [Beirão Da Veiga et al., 2013], and the *polygonal finite element* method [Sukumar and Malsch, 2006; Manzini et al., 2014; Chi et al., 2016; Khoei et al., 2015; Talischi et al., 2015]. The last approach uses barycentric coordinates as a basis. In particular, it proposes to use Wachspress (1.5) or discrete harmonic (1.6) coordinates when all elements in the mesh are convex or mean value coordinates (1.7) for general, possibly concave, elements.

10.2 Meshing

As already explained, hexahedra have many advantages over tetrahedra. For this reason the creation of hexahedral and hex-dominant meshes has become an active field in itself. One of the earliest attempts to achieve automatic hexahedral meshing is paving and sweeping [Owen and Saigal, 2000; Yamakawa and Shimada, 2003; Staten et al., 2005; Shepherd and Johnson, 2008]. The first one consists of inserting regular layers of cubes aligned with a boundary quad mesh. The second one extrudes a partial quad mesh. The major problem with these attempts is their extremely challenging implementation, due to the large number of special cases. One simplification consists of considering only tubular meshes

and using the skeleton for the sweeping phase [Livesu et al., 2016].

10.2.1 Spatial Partitioning

A more popular approach is spatial partitioning, which can be used to discretize shapes in regular collections of cubes, which coarsely approximate the input shape [Su et al., 2004; Zhang and Bajaj, 2006; Zhang et al., 2007], in particular combined with octrees [Maréchal, 2009; Ito et al., 2009; Zhang et al., 2013]. The main advantage of these methods is their robustness, which makes them standard for hex-mesh generation. However, these methods can represent only features that are well-aligned with the principal axes and place all singularities on the shape boundary, which is unfortunate since this is often the region of highest interest. These disadvantages are, however, compensated by the high robustness of these methods, making them the de facto standard for automatic hex mesh generation.

10.2.2 Polycube Parametrization

Polycube methods [Gregson et al., 2011; Li et al., 2013; Livesu et al., 2013; Huang et al., 2014; Fang et al., 2016; Fu et al., 2016] parametrize the interior of a closed surface mesh into a polycube, which is trivially subdivided in a hexahedral mesh and warped back into the input geometry. These methods produce a better distribution of the boundary singularities. However, they are not guaranteed to produce a valid hex-mesh. Similarly to spatial partitioning methods, all singularities are located on the surface boundary. However, in contrast to spatial partitioning schemes, polycube methods distribute them in a superior way to account for surface features, obtaining both higher quality elements and a lower total element count. These methods are unfortunately not guaranteed to produce a valid polycube and can fail on complex inputs, limiting their practical applicability.

10.2.3 Field-Aligned Methods

Field-aligned methods [Nieser et al., 2011; Huang et al., 2011; Li et al., 2012; Jiang et al., 2014] compute a hex-mesh in three stages. These methods start by estimating the gradients of a volumetric parametrization using a directional field [Vaxman et al., 2016]. Then they compute a parametrization aligned with the estimated gradients. Finally, they trace the cubes' edges in parametric space [Lyon et al., 2016]. Their main disadvantage is that producing a parametrization that induces a pure hex-mesh remains an unsolved problem, and currently used

heuristics tend to fail on complex inputs. For instance, the method [Sokolov et al., 2016] is one of the few existing field-aligned parametrization methods that targets hex-dominant meshes. It can process complex CAD models with alignment to surface features, creating meshes composed of hexahedra, tetrahedra, triangle-based prisms, and quad-based pyramids. However, the meshes produced are not conforming (*i.e.*, there are interfaces where a hexahedron is connected with two tetrahedra).

10.2.4 Existing Software

Due to the applicative aspect of these techniques, many of them have led to actual software implementations. Boundary-aligned techniques exist to automatically mesh special geometries, such as cylinders, boxes and sweepable solids [*PAMGEN*, 2016; *ANSYSTurbogrid*, 2016; *HyperMesh*, 2016; *SiemensPLM*, 2016]. Numerous software packages exist for octree-based hexahedral meshes [*Cart3D*, 2016; *LBIE*, 2016; *Harpoon*, 2016; *HEXPress*, 2016; *Bolt*, 2016; *Hexotic*, 2016; MeshGems, 2015; *Kubrix*, 2016; *XBX*, 2016]. Other techniques manually decompose the shape into simpler pieces that are then meshed while ensuring that compatible interfaces are introduced [Sandia, National Lab, 2016; *Trelis*, 2016; *Apex*, 2016]. Mixed meshes, containing not only hexahedra, are easier to generate. Therefore, many commercial codes have been developed for this task [*Autodesk Simulation-Mechanical*, 2016; *MeshGemsHybrid*, 2016; *AMPS*, 2016; *BETA CAE*, 2016; *TexMesher*, 2016]. These tools are closer to our goals. Unfortunately, it is difficult to find free or open-source implementation. Moreover, the limitations of these techniques are difficult to quantify precisely, even though they are informally understood. Finally, none of the software can produce meshes aligned with a given volumetric orientation field.

The proposed approach will lift the global conditions that make the hexahedral meshing problem hard, opening the door to simpler and more robust meshing techniques that are specifically designed to use the proposed new discretizations.

10.3 Parametric finite elements approximation

We extended our parametric finite element method (Section 9.1) for creating polygonal or piece-wise polynomial meshes. The basic idea is to approximate the warped mesh produced by the bijective map to create a polygonal or piecewise-polynomial mesh. Note that, to maintain bijectivity, the approximation is per-

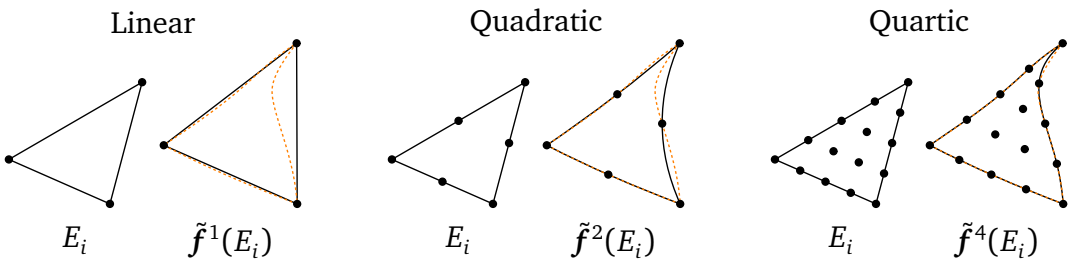


Figure 10.1. Comparison of f (orange dashed line) with its polynomial approximations $\tilde{f}^k$ (black solid line) for an element E_i.

formed directly in Θ and can be done with arbitrary accuracy.

To obtain piece-wise polynomial approximations of the warped mesh, we sample a parametrization element E_i with the necessary interpolation nodes (*e.g.*, for a quadratic approximation we add to the element the edge mid-points) as shown in Figure 10.1. We then warp with f these nodes and fit a Lagrange polynomial to create the element-wise approximation $\tilde{f}^k(E_i)$ of order k of the mapping. The overall approximation is then obtained as a union of the element approximations, see the third, fourth, and fifth rows of Figure 10.7. Unfortunately, this approximation is not guaranteed to create a valid mesh as visible on the close-ups in Figure 10.7, and bijectivity cannot be restored by refinement.

To overcome this limitation, one possibility is to approximate the mapping with a piecewise polygonal approximation. As for the polynomial approximation, this procedure is performed per element. We sample every side of an element E_i with k uniformly sampled points $\boldsymbol{x}_i$, $i = 1, \ldots, k$. Then we compute $f(\boldsymbol{x}_i)$, which gives a densely sampled polygonal approximation of $f(E_i)$. Finally, for efficiency reasons, we discard all approximately collinear points by checking if the angle between two successive segments is approximately flat, thus creating polygons with fewer vertices, Figure 10.2. This procedure produces the approximation $\tilde{f}$

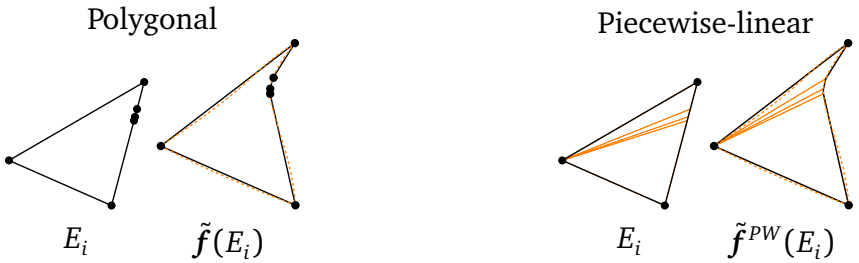


Figure 10.2. Comparison of f (orange dashed line) with its polygonal $\tilde{f}$ and piecewise-linear approximations $\tilde{f}^{PW}$ (black solid line) for an element E_i.

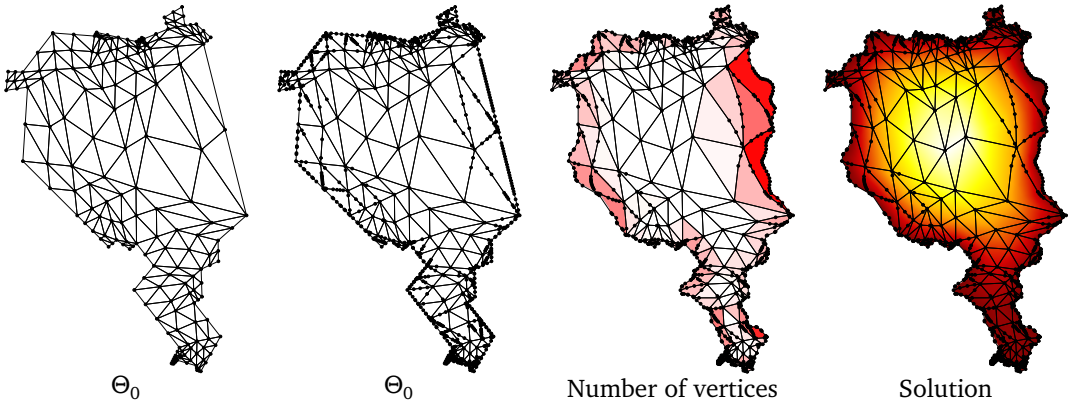


Figure 10.3. Example of discretization of f for a triangular mesh. The third image illustrates the number of vertices of each element, from white (triangles) to red (icosagon). The last image shows the solution of the Poisson equation using mean value coordinates as basis functions.

which is a mesh with polygonal elements which is always bijective, last row of Figure 10.7.

Since the mapping f is rather local, this strategy naturally generates triangles away from the boundary, as shown in Figure 10.3. This effect becomes more visible as we refine the mesh, see Figure 10.4. Note that, to solve the physical problem, one can employ mean value coordinates as in [Sukumar and Malsch, 2006].

Using barycentric coordinates as basis function is a non-standard technique to solve physical problems. To avoid this discretization we suggest to triangulate the polygons, thus creating the piecewise approximation $\tilde{f}^{PW}$ of the map-

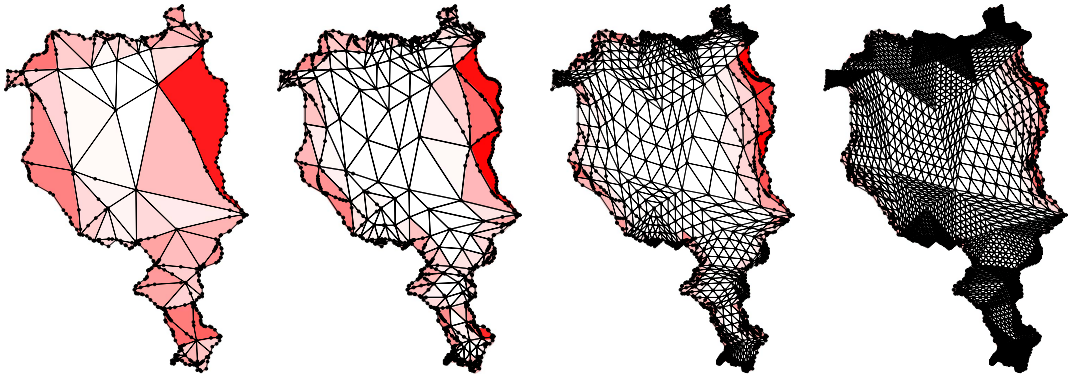
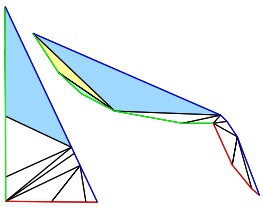
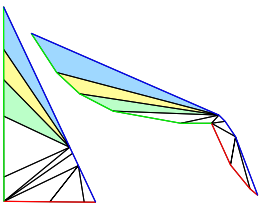


Figure 10.4. The number of vertices per element becomes more and more localized on the boundary under refinements.



Degenerate triangulation, yellow and green elements have zero area in the domain.



Valid triangulation, all elements have positive area in the domain.

Figure 10.5. Invalid and valid triangulations for constructing the mapping $\tilde{f}^{PW}$.

ping, Figure 10.2. Certain applications require to have the same mesh on the parametrization domain. For instance, multi level methods require a hierarchy of nested spaces for optimal convergence [Hackbusch, 1985; Briggs et al., 2000] which can be achieved by constructing this hierarchy in the parametrization domain [Dickopf and Krause, 2011]. For this reason, when triangulating the polygonal approximation, we forbid inserting additional points and having a triangle which is connected to the same edge, see the yellow triangle in Figure 10.5 (left). Figure 10.6 shows different approximations of f and the image of the inverse in the parametrization domain.

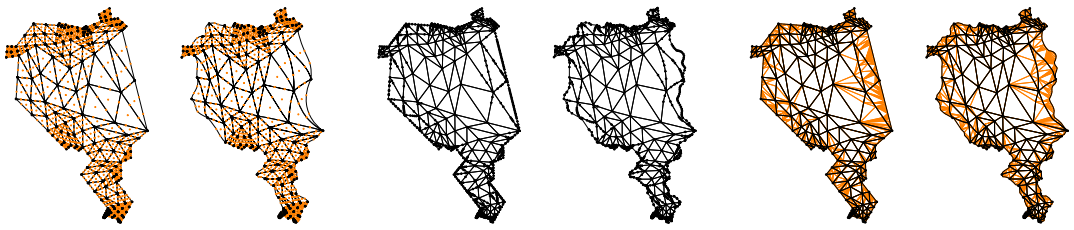


Figure 10.6. Mesh hierarchies for different discretizations.

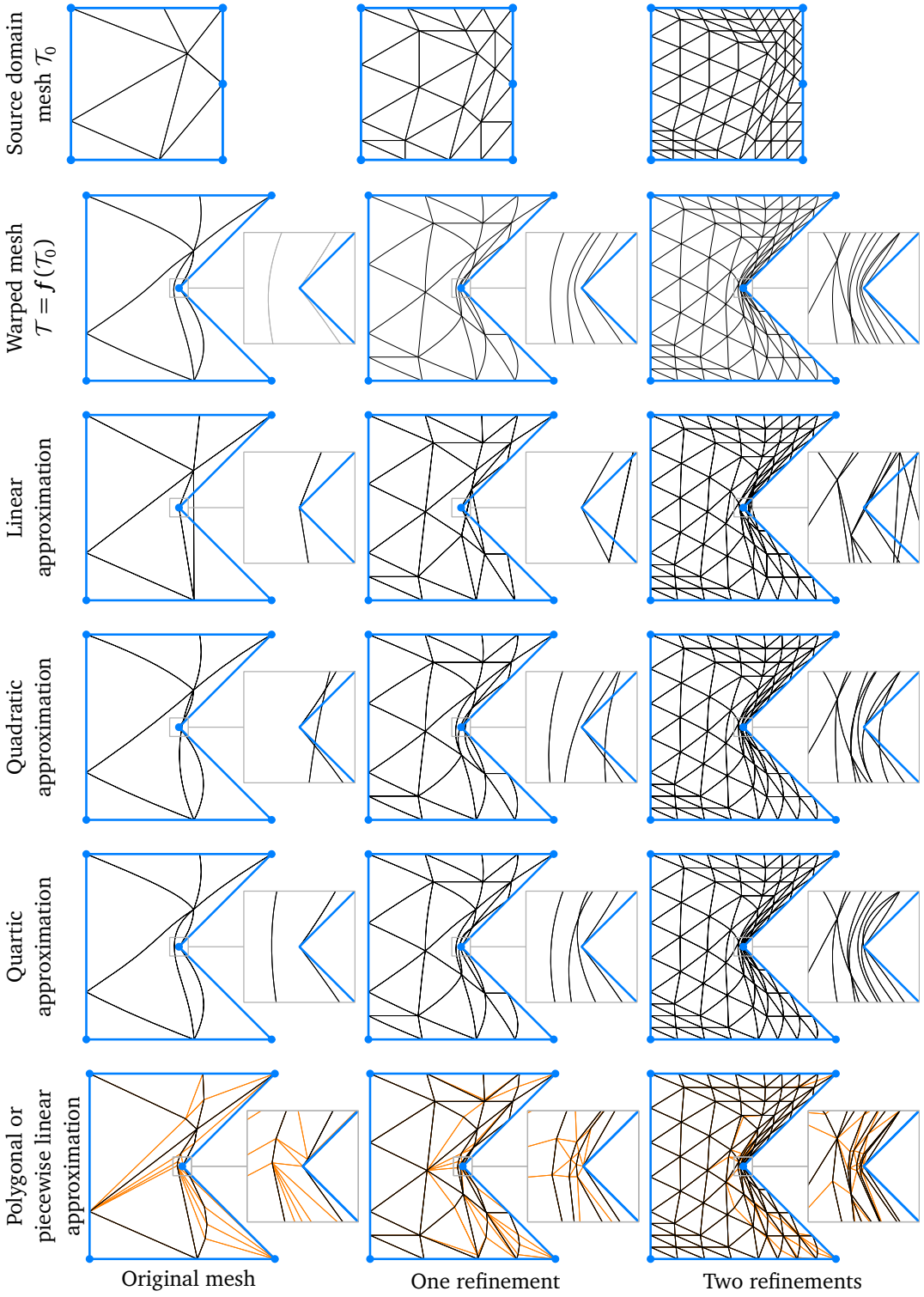


Figure 10.7. Comparison between $\boldsymbol{f}$ and its different approximations.

Conclusion

This dissertation presents and discusses three main methods to create bijective mappings and their application to finite elements and meshing.

Since it is difficult or even impossible (*e.g.*, in the case of barycentric mappings) to create bijective mappings directly, we propose to achieve bijectivity through composition (Chapters 4 and 5). The first method is based on the intuition that if two polygons (or polyhedra) are sufficiently close then the barycentric mapping (Section 2.1) is bijective. Following this intuition we propose to decompose the overall mapping into a finite number of intermediate steps, where the mapping between each pair of successive polygons is bijective (Section 4.2). An interesting remark is that if we let the number of steps converge to infinity then the bijective mapping becomes symmetric (Section 4.3), that is, if we invert the roles of source and target polygons we construct the inverse mapping.

The second method for creating bijective mappings (Chapter 5) is also based on composition. The principal idea follows from the Radó–Kneser–Choquet theorem which states that a harmonic map to a convex target polygon is always bijective. Therefore for creating a bijective mapping between two polygons with n vertices we propose to create two harmonic mappings φ_0 and φ_1 that map from the source and target polygons to a common regular n-gon respectively. Then the bijective mapping can be simply created by inverting φ_1 and composing it with φ_0. Since the Radó–Kneser–Choquet theorem only requires the intermediate polygon to be convex, we illustrate how to change the shape of the intermediate polygons to reduce the distortion of the mapping (Section 5.3).

The condition that guarantees bijectivity is linked to the norm of the gradient of barycentric coordinates (4.1). We show that a large part of the so-called three-point family coordinates [Floater et al., 2006] are C_0 through the study of their directional derivative (Section 6.1). Moreover, we show that the directional derivative of mean value coordinates (1.7) behaves like a shifted sine function (Section 6.2.1) and we provide some initial results and numerical evidence on the boundedness of their gradient (Section 6.2.2).

The composition of barycentric mappings can be applied to closed planar

curves by means of transfinite barycentric coordinates (Section 1.3). We discuss a method that allows to create intermediate curves (Section 3.3) based on the interpolation of the curvature, which is an intrinsic property and produces natural results. The main challenge consists of closing the curve, which we resolve through an optimization procedure (Section 3.3.1).

The last method for creating bijective mappings (Chapter 7) is based on complex barycentric coordinates (Section 1.2). It follows the intuition acquired by changing the boundary conditions of the harmonic mappings (Section 5.4). The idea is to relinquish the linear behaviour of the mapping along the edges of the polygons to achieve bijectivity. We present some initial results and an optimization procedure that finds the coefficients of a Bernstein polynomial that describes the "speed" of the mapping along the edges.

The first application we present regards parametric finite elements (Chapter 9). We use the bijective mappings to warp a mesh from a parametrization domain to the physical domain (Section 9.1). The main advantage is that the parametrization domain can be arbitrarily simple (*e.g.*, a triangle) and therefore meshed with arbitrary resolution. The bijective mapping will then take care of warping this domain into the physical domain, which ensures that the shape of interest is preserved. An interesting discovery is that even if closed form coordinates are fast and efficient to evaluate and easy to parallelise, with polygons with many sides (as the one in Figure 9.7) they become a bottleneck. One possibility would be to use blended coordinates [Anisimov et al., 2017], however they are more complicated to parallelise. It would be interesting to compare highly optimized versions of both coordinates to increase the speed of our parametric FEM.

The second application considers meshing (Chapter 10). We propose different strategies (polynomial, polygonal, and piece-wise linear) to approximate the bijective mappings (Section 10.3). These strategies allow to create different meshes, such as polygonal or piece-wise quadratic meshes. It is interesting to note that naively refining the mesh does not restore bijectivity. In fact, our experiments show that there are situations in which even a large number of refinements does not solve the problem, indicating that bijectivity can be achieved only in the limit.

Future work

In my short five year experience I have noticed that at the conclusion of a research, there are more questions and open problems than at the start. For instance, my work on composite barycentric mappings (Chapter 4) was the driving force for studying the directional derivative of the three-point family, since the bound (4.1) exists only if $\|\nabla\phi_i\|$ is bounded. Standing from there, we show that the directional derivative of mean value coordinates is bounded, and therefore it suggests that the gradient might be bounded too. This consideration led to the study of the gradient of these coordinates. Unfortunately this work is incomplete, and proof of the boundedness of the gradient is missing. Moreover, it would be interesting to find a possibly simple tight bound that depends on the geometry of the polygons.

Also related to composite mean value mappings, it would be interesting to be able to estimate the number of intermediate steps necessary to guarantee the bijectivity of the mappings. This estimate would allow to avoid to the strategy described in Section 4.5.3. To achieve this result we need first to have the simple bound for the gradient of the coordinates and then to combine it with some conditions on the vertex paths. This result would have an impact for practical applications since it would allow to compute the optimal number of steps.

Chapter 7 illustrates some initial results and possibilities to achieve bijective mappings by means of complex coordinates. The optimization procedure is just a preliminary study to show the feasibility. On the one hand, it would be interesting to improve it (*e.g.*, find a better strategy for selecting the concavity of the functions on the edges) or to include distortion measures in the process. On the other hand, a sound mathematical analysis to find the necessary conditions to guarantee bijectivity is missing.

The parametric finite element method described in Section 9.1 requires integrating the Jacobian on the bijective mappings. One initial strategy consists of adaptively adding as many quadrature points as needed. This strategy is not optimal since it requires evaluating the mapping a large number of times. One solution would be to devise special quadrature rules for this application.

In Section 5.3 we started scratching the surface of the mapping optimization. A thoughtful study of the influence of the vertex path and coordinates on the composite barycentric mapping is missing. One can modify them to obtain mappings with lower distortion, following the ideas proposed by "energy-based" bijective mappings. Moreover, it would be interesting to design energy functions from the perspective of finite element applications. That is, to design bijective mappings optimized for parametric finite elements, by favouring, for instance, well-shaped elements or "as piece-wise polynomial as possible" behaviour to facilitate numerical integration.

We show how to use bijective mappings in the context of meshing. The strategy used for creating the polygons is local and creates many polygons with a large number of sides. However, one can design a global strategy that checks for self-intersections and produces more simplices. Moreover, it would be nice to extend this meshing strategy to three dimensions, where the local approximation becomes much more complicated.

Finally, most of the images and results presented in this document and in my papers arise from a complex software implemented in collaboration with Patrick Zulian, called MOONOLITH. We plan to clean it and extract a simple interface to provide the possibility to replicate our results and access many utility geometrical algorithms, such as meshing tools, eps exporters, and mesh format conversion. On a related note, I feel that with the many hours invested into searching and hunting for bugs in MOONOLITH, my PhD should be more in entomology than in computer science.

Bibliography

Aigerman, N., Poranne, R. and Lipman, Y. [2014]. Lifted bijections for low distortion surface mappings, *ACM Transactions on Graphics* **33**(4): Article 69, 12 pages.

Aigner, M., Heinrich, C., Jüttler, B., Pilgerstorfer, E., Simeon, B. and Vuong [2009]. *Swept volume parameterization for isogeometric analysis*, Springer.

Alexa, M., Cohen-Or, D. and Levin, D. [2000]. As-rigid-as-possible shape interpolation, *Proceedings of SIGGRAPH*, pp. 157–164.

Alon, E., Sariel, H.-P., Leonidas, G. J. and Murali, T. M. [2001]. Morphing between polylines, *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 680–689.

AMPS [2016]. `http://www.ampstech.com/ampstech/Asp/Solid.asp`.

Anisimov, D., Hormann, K. and Schneider, T. [n.d.]. Behaviour of exponential three-point coordinates at the vertices of convex polygons.

Anisimov, D., Panozzo, D. and Hormann, K. [2017]. Blended barycentric coordinates, *Computer Aided Geometric Design* **52–53**: 205–216. Proceedings of GMP.

ANSYSTurbogrid [2016]. `http://www.ansys.com/Products/Fluids/ANSYS-TurboGrid/ANSYS-TurboGrid-Features#1`.

Apex [2016]. `http://www.mscapex.com/apex-modeler/`.

Arad, N., Dyn, N., Reisfeld, D. and Yeshurun, Y. [1994]. Image warping by radial basis functions: Application to facial expressions, *CVGIP: Graphical Models and Image Processing* **56**(2): 161–172.

Autodesk Simulation-Mechanical [2016]. `http://www.autodesk.com/products/simulation-mechanical/features/all/gallery-view`.

Babuška, I. and Suri, M. [1992]. Locking effects in the finite element approximation of elasticity problems, *Numerische Mathematik* **62**(1): 439–463.

Bathe, K.-J. and Wilson, E. L. [1976]. *Numerical methods in finite element analysis*, Prentice-Hall civil engineering and engineering mechanics series, Prentice-Hall Englewood Cliffs, NJ.

Bazilevs, Y., Michler, C., Calo, V. and Hughes, T. [2010]. Isogeometric variational multiscale modeling of wall-bounded turbulent flows with weakly enforced boundary conditions on unstretched meshes, *Computer Methods in Applied Mechanics and Engineering* **199**(13âĂŞ16): 780–790.

Beirão Da Veiga, L., Brezzi, F., Cangiani, A., Manzini, G., Marini, L. D. and Russo, A. [2013]. Basic principles of virtual element methods, *Mathematical Models and Methods in Applied Sciences* **23**(01): 199–214.

Belongie, S., Malik, J. and Puzicha, J. [2002]. Shape matching and object recognition using shape contexts, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **24**(4): 509–522.

Belyaev, A. [2006]. On transfinite barycentric coordinates, *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, SGP '06, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp. 89–99.

Belytschko, T., Gracie, R. and Ventura, G. [2009]. A review of extended/generalized finite element methods for material modeling, *Modelling and Simulation in Materials Science and Engineering* **17**(4): 043001.

Bertrand, F., Muñzenmaier, S. and Starke, G. [2014a]. First-order system least squares on curved boundaries: Higher-order Raviart–Thomas elements, *SIAM Journal on Numerical Analysis* **52**(6): 3165–3180.

Bertrand, F., Muñzenmaier, S. and Starke, G. [2014b]. First-order system least squares on curved boundaries: Lowest-order Raviart–Thomas elements, *SIAM Journal on Numerical Analysis* **52**(2): 880–894.

BETA CAE [2016]. `http://meshgems.com/volume-meshing-meshgems-hybrid.html`.

Bey, J. [1995]. Tetrahedral grid refinement, *Computing* **55**: 355–378.

Bolt [2016]. `http://www.csimsoft.com/boltoverview`.

Bommes, D., Lévy, B., Pietroni, N., Puppo, E., Silva, C., Tarini, M. and Zorin, D. [2012]. State of the art in quad meshing, *Eurographics STARS*.

Braess, D. [2007]. *Finite Elements*, third edn, Cambridge University Press. Cambridge Books Online.

Bramble, J. H., Pasciak, J. E. and Steinbach, O. [2002]. On the stability of the L^2-projections in H^1, *Mathematics of Computation* **71**(237): 147–156.

Brenner, S. and Scott, R. [2008]. *The Mathematical Theory of Finite Element Methods*, Vol. 15 of *Texts in Applied Mathematics*, Springer-Verlag New York.

Briggs, W. L., Henson, V. E. and McCormick, S. F. [2000]. *A multigrid tutorial (2nd ed.)*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA.

Buffa, A., Cho, D. and Sangalli, G. [2010]. Linear independence of the t-spline blending functions associated with some particular t-meshes, *Computer Methods in Applied Mechanics and Engineering* **199**(23): 1437 – 1445.

Campen, M., Silva, C. T. and Zorin, D. [2016]. Bijective maps from simplicial foliations, *ACM Transactions on Graphics* **35**(4): 74:1–74:15.

Cart3D [2016]. `http://people.nas.nasa.gov/~aftosmis/cart3d/`.

Chen, R., Weber, O., Keren, D. and Ben-Chen, M. [2013]. Planar shape interpolation with bounded distortion, *ACM Transactions on Graphics* **32**(4): Article 108, 11 pages.

Chi, H., Talischi, C., Lopez-Pamies, O. and Paulino, G. H. [2016]. A paradigm for higher-order polygonal elements in finite elasticity using a gradient correction scheme, *Computer Methods in Applied Mechanics and Engineering* **306**: 216–251.

Choquet, G. [1945]. Sur un type de transformation analytique généralisant la représentation conforme et définie au moyen de fonctions harmoniques, *Bulletin des Sciences Mathématiques* **69**: 156–165.

Ciarlet, P. G. and Raviart, P.-A. [1972]. Interpolation theory over curved elements, with applications to finite element methods, *Computer Methods in Applied Mechanics and Engineering* **1**(2): 217–249.

Crane, K., Pinkall, U. and Schröder, P. [2013]. Robust fairing via conformal curvature flow, *ACM Transactions on Graphics* **32**(4): Article 61, 10 pages.

da Veiga, L. B., Buffa, A., Cho, D. and Sangalli, G. [2011]. Isogeometric analysis using t-splines on two-patch geometries, *Computer Methods in Applied Mechanics and Engineering* **200**(21): 1787 – 1803.

D'Azevedo, E. F. [2000]. Are bilinear quadrilaterals better than linear triangles?, *SIAM Journal on Scientific Computing* **22**(1): 198–217.

Dickopf, T. and Krause, R. [2011]. Monotone multigrid methods based on parametric finite elements, *Technical report*, ICS, USI. Submitted to Lecture Notes in Computational Science and Engineering.

do Carmo, M. P. [1976]. *Differential geometry of curves and surfaces*, Prentice-Hall.

Dörfler, W. and Rumpf, M. [1998]. An adaptive strategy for elliptic problems including a posteriori controlled boundary approximation, *Mathematics of Computation of the American Mathematical Society* **67**(224): 1361–1382.

Duffin, R. J. [1959]. Distributed and lumped networks, *Journal of Mathematics and Mechanics* **8**(5): 793–826.

Dziuk, G. [1990]. An algorithm for evolutionary surfaces, *Numerische Mathematik* **58**(1): 603–611.

Eck, M., DeRose, T., Duchamp, T., Hoppe, H., Lounsbery, M. and Stuetzle, W. [1995]. Multiresolution analysis of arbitrary meshes, *Proceedings of SIGGRAPH*, pp. 173–182.

Fairweather, G. and Karageorghis, A. [1998]. The method of fundamental solutions for elliptic boundary value problems, *Advances in Computational Mathematics* **9**(1–2): 69–95.

Fang, X., Xu, W., Bao, H. and Huang, J. [2016]. All-hex meshing using closed-form induced polycube, *ACM Transactions on Graphics* **35**(4): 124.

Floater, M. S. [2003]. Mean value coordinates, *Computer Aided Geometric Design* **20**(1): 19–27.

Floater, M. S. [2005]. Arc length estimation and the convergence of polynomial curve interpolation, *Bit Numerical Mathematics* **45**(4): 679–694.

Floater, M. S. [2014]. *Wachspress and Mean Value Coordinates*, Springer International Publishing, Cham, pp. 81–102.

Floater, M. S. [2015]. Generalized barycentric coordinates and applications, *Acta Numerica* **24**: 161–214.

Floater, M. S. and Hormann, K. [2005]. Surface parameterization: a tutorial and survey, *in* N. A. Dodgson, M. S. Floater and M. A. Sabin (eds), *Advances in Multiresolution for Geometric Modelling*, Springer, pp. 157–186.

Floater, M. S., Hormann, K. and Kós, G. [2006]. A general construction of barycentric coordinates over convex polygons, *Advances in Computational Mathematics* **24**(1–4): 311–331.

Floater, M. S., Kós, G. and Reimers, M. [2005]. Mean value coordinates in 3D, *Computer Aided Geometric Design* **22**(7): 623–631.

Floater, M. S. and Kosinka, J. [2010]. On the injectivity of Wachspress and mean value mappings between convex polygons, *Advances in Computational Mathematics* **32**(2): 163–174.

Fu, X., Bai, C. and Liu, Y. [2016]. Efficient volumetric polycube-map construction, *Computer Graphics Forum* **35**(7).

Fu, X. and Liu, Y. [2016]. Computing inversion-free mappings by simplex assembly, *ACM Transactions on Graphics* **35**(6): 216:1–216:12.

Goldstein, E. and Gotsman, C. [1995]. Polygon morphing using a multiresolution representation, *Proceedings of Graphics Interface '95*, Canadian Human-Computer Communications Socity, pp. 247–254.

Gotsman, C. and Surazhsky, V. [2001]. Guaranteed intersection-free polygon morphing, *Computers & Graphics* **25**(1): 67–75.

Gout, J. L. [1985]. Rational wachspress-type finite elements on regular hexagons, *IMA Journal of Numerical Analysis* **5**(1): 59.

Gregson, J., Sheffer, A. and Zhang, E. [2011]. All-hex mesh generation via volumetric polycube deformation, *Computer Graphics Forum* **30**(5): 1407–1416.

Guibas, L., Hershberger, J. and Suri, S. [1999]. Morphing simple polygons, *Discrete & Computational Geometry* **24**(1): 1–34.

Hashim, S. and Richens, D. [2006]. Finite element method in cardiac surgery, *Interactive CardioVascular and Thoracic Surgery* **5**(1): 5.

Hackbusch, W. [1985]. *Multi-grid methods and applications*, Vol. 4 of *Springer series in computational mathematics*, Springer-Verlag.

Hall, W. S. [1994]. *The Boundary Element Method*, Vol. 27, Springer.

Harpoon [2016]. `http://www.sharc.co.uk/index.htm`.

Hexotic [2016]. `https://www.rocq.inria.fr/gamma/gamma/Membres/CIPD/Loic.Marechal/Research/Hexotic.html`.

HEXPress [2016]. `http://www.numeca.com/en/products/automeshtm/hexpresstm`.

Hormann, K. and Floater, M. S. [2006]. Mean value coordinates for arbitrary planar polygons, *ACM Transactions on Graphics* **25**(4): 1424–1441.

Hormann, K. and Greiner, G. [2000]. MIPS: An efficient global parametrization method, *in* P.-J. Laurent, P. Sablonnière and L. L. Schumaker (eds), *Curve and Surface Design: Saint-Malo 1999*, Vanderbilt University Press, pp. 153–162.

Hormann, K. and Sukumar, N. [2008]. Maximum entropy coordinates for arbitrary polytopes, *Computer Graphics Forum* **27**(5): 1513–1520.

Hoschek, J. [1988]. Intrinsic parametrization for approximation, *Computer Aided Geometric Design* **5**(1): 27–31.

Huang, J., Jiang, T., Shi, Z., Tong, Y., Bao, H. and Desbrun, M. [2014]. L1-based construction of polycube maps from complex shapes, *ACM Transactions on Graphics* **33**(3): 25:1–25:11.

Huang, J., Tong, Y., Wei, H. and Bao, H. [2011]. Boundary aligned smooth 3d cross-frame field, *ACM Transactions on Graphics* **30**(6): 143:1–143:8.

Hughes, T. J., Cottrell, J. A. and Bazilevs, Y. [2005]. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement, *Computer Methods in Applied Mechanics and Engineering* **194**(39): 4135–4195.

HyperMesh [2016]. `http://www.altairhyperworks.com/product/HyperMesh`.

Iben, H. N., O'Brien, J. F. and Demaine, E. D. [2009]. Refolding planar polygons, *Discrete & Computational Geometry* **41**(3): 444–460.

Ito, Y., Shih, A. M. and Soni, B. K. [2009]. Octree-based reasonable-quality hexahedral mesh generation using a new set of refinement templates, *International Journal for Numerical Methods in Engineering* pp. 1809–1833.

Jacobson, A. [2012]. Bijective mappings with generalized barycentric coordinates: a counterexample, *Technical report*, ETH Zurich.

Jacobson, A., Baran, I., Popović, J. and Sorkine, O. [2011]. Bounded biharmonic weights for real-time deformation, *ACM Transactions on Graphics* **30**(4): Article 78, 8 pages.

Jiang, T., Huang, J., Wang, Y., Tong, Y. and Bao, H. [2014]. Frame field singularity correction for automatic hexahedralization, *IEEE Transactions on Visualization and Computer Graphics* **20**(8): 1189–1199.

Joshi, P., Meyer, M., DeRose, T., Green, B. and Sanocki, T. [2007]. Harmonic coordinates for character articulation, *ACM Transactions on Graphics* **26**(3): Article 71, 9 pages.

Ju, T., Schaefer, S. and Warren, J. [2005]. Mean value coordinates for closed triangular meshes, *ACM Transactions on Graphics* **24**(3): 561–566.

Ju, T., Schaefer, S., Warren, J. and Desbrun, M. [2005]. A geometric construction of coordinates for convex polyhedra using polar duals, *Symposium on Geometry Processing*, pp. 181–186.

Kanai, T., Suzuki, H. and Kimura, F. [1997]. 3D geometric metamorphosis based on harmonic map, *Proceedings of Pacific Graphics*, pp. 97–104.

Khoei, A. R., Yasbolaghi, R. and Biabanaki, S. O. R. [2015]. A polygonal finite element method for modeling crack propagation with minimum remeshing, *International Journal of Fracture* **194**(2): 123–148.

Kikuchi, N. and Oden, J. T. [1988]. *Contact problems in elasticity: a study of variational inequalities and finite element methods*, Vol. 8, siam.

Kneser, H. [1926]. Lösung der Aufgabe 41, *Jahresbericht der Deutschen Mathematiker-Vereinigung* **35**(2. Abteilung): 123–124.

Krause, R. and Zulian, P. [2015]. A parallel approach to the variational transfer of discrete fields between arbitrarily distributed finite element meshes. Accepted at SISC.

Kubrix [2016]. http://www.itascacg.com/software/kubrix.

Langer, T., Belyaev, A. and Seidel, H.-P. [2006]. Spherical barycentric coordinates, *Proceedings of the Fourth Symposium on Geometry Processing*, pp. 81–88.

LBIE [2016]. http://www.cs.utexas.edu/~bajaj/cvc/software/LBIE.shtml.

Li, B., Li, X., Wang, K. and Qin, H. [2013]. Surface mesh to volumetric spline conversion with generalized polycubes, *IEEE Transactions on Visualization and Computer Graphics* **19**(9): 1539–1551.

Li, X.-Y. and Hu, S.-M. [2013]. Poisson coordinates, *IEEE Transactions on Visualization and Computer Graphics* **19**(2): 344–352.

Li, Y., Liu, Y., Xu, W., Wang, W. and Guo, B. [2012]. All-hex meshing using singularity-restricted field, *ACM Transactions on Graphics* **31**(6): 177:1–177:11.

Lian, H., Bordas, S. P. A., Sevilla, R. and Simpson, R. N. [2012]. Recent Developments in CAD/analysis Integration, *ArXive-prints* .

Lindelöf, E. [1894]. Sur l'application de la méthode des approximations successives aux équations différentielles ordinaires du premier ordre, *Comptes rendus hebdomadaires des séances de l'Académie des sciences* **116**(3): 454–457.

Lipman, Y. [2012]. Bounded distortion mapping spaces for triangular meshes, *ACM Transactions on Graphics* **31**(4): Article 108, 13 pages.

Lipman, Y. and Funkhouser, T. [2009]. Möbius voting for surface correspondence, *ACM Transactions on Graphics* **28**(3): Article 72, 12 pages.

Lipman, Y., Kopf, J., Cohen-Or, D. and Levin, D. [2007]. GPU-assisted positive mean value coordinates for mesh deformations, *Proceedings of the Fifth Symposium on Geometry Processing*, pp. 117–123.

Lipman, Y., Levin, D. and Cohen-Or, D. [2008]. Green coordinates, *ACM Transactions on Graphics* **27**(3): Article78,10pages.

Lipnikov, K., Manzini, G. and Shashkov, M. [2014]. Mimetic finite difference method, *Journal of Computational Physics* **257, Part B**: 1163–1227. Physics-compatible numerical methods.

Liu, L., Wang, G., Zhang, B., Guo, B. and Shum, H.-Y. [2004]. Perceptually based approach for planar shape morphing, *Proceedings of Pacific Graphics*, pp. 111–120.

Livesu, M., Muntoni, A., Puppo, E. and Scateni, R. [2016]. Skeleton-driven adaptive hexahedral meshing of tubular shapes, *Computer Graphics Forum*, Vol. 35, pp. 237–246.

Livesu, M., Vining, N., Sheffer, A., Gregson, J. and Scateni, R. [2013]. Polycut: monotone graph-cuts for polycube base-complex construction, *ACM Transactions on Graphics* **32**(6): 171.

Lorentz, G. [1953]. *Bernstein polynomials*, Mathematical expositions, University of Toronto Press.

Luo, X., Shephard, M. S. and Remacle, J.-F. [2001]. The influence of geometric approximation on the accuracy of high order methods, *Rensselaer SCOREC report* **1**.

Lyon, M., Bommes, D. and Kobbelt, L. [2016]. Hexex: Robust hexahedral mesh extraction, *ACM Transactions on Graphics* **35**(4): 123:1–123:11.

Manson, J. and Schaefer, S. [2010]. Moving least squares coordinates, *Computer Graphics Forum* **29**(5): 1517–1524.

Manzini, G., Russo, A. and Sukumar, N. [2014]. New perspectives on polygonal and polyhedral finite element methods, *Mathematical Methods in the Applied Sciences* **24**(08): 1665–1699.

Maréchal, L. [2009]. Advances in octree-based all-hexahedral mesh generation: handling sharp features, *proceedings of the 18th International Meshing Roundtable*, Springer, pp. 65–84.

Martin, S., Kaufmann, P., Botsch, M., Wicke, M. and Gross, M. [2008]. Polyhedral finite elements using harmonic basis functions, *Computer Graphics Forum* **27**(5): 1521–1529.

Martin, T. and Cohen, E. [2010]. Volumetric parameterization of complex objects by respecting multiple materials, *Computers & Graphics* **34**(3): 187–197.

Meisters, G. H. and Olech, C. [1963]. Locally one-to-one mappings and a classical theorem on the schlicht functions, *Duke Mathematical Journal* **30**(1): 63–80.

Melenk, J. and Wohlmuth, B. [2001]. On residual-based a posteriori error estimation in hp-fem, *Advances in Computational Mathematics* **15**(1): 311–331.

MeshGems [2015]. Volume Meshing: MeshGems-Hexa, `http://meshgems.com/volume-meshing-meshgems-hexa.html`.

MeshGemsHybrid [2016]. `http://meshgems.com/volume-meshing-meshgems-hybrid.html`.

Meyer, M., Desbrun, M., Schröder, P. and Barr, A. H. [2003]. Discrete differential-geometry operators for triangulated 2-manifolds.

Michal, S. and Ari, R. [1995]. Shape blending using the star-skeleton representation, *IEEE Computer Graphics and Applications* **15**(2): 44–50.

Möbius, A. F. [1827]. *Der barycentrische Calcul*, Johann Ambrosius Barth, Leipzig.

Mortara, M. and Spagnuolo, M. [2001]. Similarity measures for blending polygonal shapes, *Computers & Graphics* **25**(1): 13–27.

Nieser, M., Reitebuch, U. and Polthier, K. [2011]. Cubecover- parameterization of 3d volumes, *Computer Graphics Forum* **30**(5): 1397–1406.

Owen, S. J. and Saigal, S. [2000]. H-morph: an indirect approach to advancing front hex meshing, *International Journal for Numerical Methods in Engineering* **49**(1–2): 289–312.

PAMGEN [2016]. `https://trilinos.org/packages/pamgen/`.

Panozzo, D. [2015]. Demystifying quadrilateral remeshing, *Edge* **1**(4): 44–51.

Picard, E. [1893]. Sur l'applicationdes méthodes d'approximations successives á l'étude de certaines équations différentielles ordinaires, *Journal de Mathématiques* p. 217.

Pinkall, U. and Polthier, K. [1993]. Computing discrete minimal surfaces and their conjugates, *Experimental Mathematics* **2**(1): 15–36.

Poranne, R. and Lipman, Y. [2014]. Provably good planar mappings, *ACM Transactions on Graphics* **33**(4): Article 76, 11 pages.

Radó, T. [1926]. Aufgabe 41, *Jahresbericht der Deutschen Mathematiker-Vereinigung* **35**(2. Abteilung): 49.

Richards, W. and NASA Dryden Flight Research Center [1997]. *Finite-Element Analysis of a Mach-8 Flight Test Article Using Nonlinear Contact Elements*, National Aeronautics and Space Administration, Office of Management, Scientific and Technical Information Program.

Rustamov, R. M. [2007]. Boundary element formulation of harmonic coordinates, *Technical report*, Purdue University.

Saba, M., Schneider, T., Scateni, R. and Hormann, K. [2014]. Curvature-based blending of closed planar curves, *Graphical Models* **76**.

Sandia, National Lab [2016]. Cubit, `https://cubit.sandia.gov/`.

Schaefer, S., Ju, T. and Warren, J. [2007]. A unified, integral construction for coordinates over closed curves, *Computer Aided Geometric Design* **24**(8): 481–493.

Schneider, T. [2017]. *Generalized Barycentric Coordinates in Computer Graphics and Computational Mechanics.*, CRC Press, chapter Barycentric Mappings. To appear in august.

Schneider, T. and Hormann, K. [2015]. Smooth bijective maps between arbitrary planar polygons, *Computer Aided Geometric Design* **35–36**: 243–354. Proceedings of GMP.

Schneider, T., Hormann, K. and Floater, M. S. [2013]. Bijective composite mean value mappings, *Computer Graphics Forum* **32**(5): 137–146.

Schneider, T., Zulian, P., Krause, R. and Hormann, K. [n.d.]. Multigrid method with parametric finite elements for arbitrarily shaped 2d meshes.

Schüller, C., Kavan, L., Panozzo, D. and Sorkine-Hornung, O. [2013]. Locally injective mappings, *Computer Graphics Forum* **32**(5): 125–135.

Scott, R. [1973]. *Finite element techniques for curved boundaries*, PhD thesis, Massachusetts Institute of Technology, Department of Mathematics.

Scott, R. [1975]. Interpolated boundary conditions in the finite element method, *SIAM Journal on Numerical Analysis* **12**(3): 404–427.

Sederberg, T. W., Cardon, D. L., Finnigan, G. T., North, N. S., Zheng, J. and Lyche, T. [2004]. T-spline simplification and local refinement, *ACM Transactions on Graphics* **23**(3): 276–283.

Sederberg, T. W., Gao, P., Wang, G. and Mu, H. [1993]. 2-D shape blending: an intrinsic solution to the vertex path problem, *Proceedings of SIGGRAPH*, pp. 15–18.

Sederberg, T. W. and Greenwood, E. [1992]. A physically based approach to 2–D shape blending, *ACM SIGGRAPH Computer Graphics* **26**(2): 25–34.

Sevilla, R., Fernández-Méndez, S. and Huerta, A. [2011]. Nurbs-enhanced finite element method (nefem), *Archives of Computational Methods in Engineering* **18**(4): 441–484.

Sheffer, A., Praun, E. and Rose, K. [2006]. Mesh parameterization methods and their applications, *Foundations and Trends in Computer Graphics and Vision* **2**(2): 105–171.

Shepherd, J. F. and Johnson, C. R. [2008]. Hexahedral mesh generation constraints, *Engineering with Computers* **24**(3): 195–213.

Shewchuk, J. R. [1996]. *Triangle: Engineering a 2D quality mesh generator and Delaunay triangulator*, Springer Berlin Heidelberg, pp. 203–222.

Si, H. [2015]. Tetgen, a delaunay-based quality tetrahedral mesh generator, *ACM Transactions on Mathematical Software* **41**(2): 11:1–11:36.

SiemensPLM [2016]. `https://www.plm.automation.siemens.com/en_us/products/lms/virtual-lab/structures/meshing.shtml`.

Snyder Laugesen, R. [1996]. Injectivity can fail for higher-dimensional harmonic extensions, *Complex Variables, Theory and Application: An International Journal* **28**(4): 357–369.

Sokolov, D., Ray, N., Untereiner, L. and Lévy, B. [2016]. Hexahedral-dominant meshing, *ACM Transactions on Graphics* **35**(5): 157:1–157:23.

Sorkine, O., Cohen-Or, D., Goldenthal, R. and Lischinski, D. [2002]. Bounded-distortion piecewise mesh parameterization, *Proceedings of IEEE Visualization*, IEEE Computer Society, pp. 355–362.

Staten, M. L., Owen, S. J. and Blacker, T. D. [2005]. *Unconstrained Paving & Plastering: A New Idea for All Hexahedral Mesh Generation*, Springer Berlin Heidelberg, pp. 399–416.

Strang, G. and Fix, G. [2008]. *An Analysis of The Finite Element Method*, Wellesley-Cambridge Press.

Su, Y., Lee, K. and Senthil Kumar, A. [2004]. Automatic hexahedral mesh generation for multi-domain composite models using a hybrid projective grid-based method, *Computer-Aided Design* **36**(3): 203–215.

Sukumar, N. and Malsch, E. A. [2006]. Recent advances in the construction of polygonal finite element interpolants, *Archives of Computational Methods in Engineering* **13**(1): 129.

Sukumar, N. and Tabarraei, A. [2004]. Conforming polygonal finite elements, *International Journal for Numerical Methods in Engineering* **61**(12): 2045–2066.

Sumner, R. W. and Popović, J. [2004]. Deformation transfer for triangle meshes, *ACM Transactions on Graphics* **23**(3): 399–405.

Sumner, R. W., Zwicker, M., Gotsman, C. and Popović, J. [2005]. Mesh-based inverse kinematics, *ACM Transactions on Graphics* **24**(3): 488–495.

Surazhsky, T. and Elber, G. [2002]. Metamorphosis of planar parametric curves via curvature interpolation, *International Journal of Shape Modelling* **8**(2): 201–216.

Talischi, C., Pereira, A., Menezes, I. F. M. and Paulino, G. H. [2015]. Gradient correction for polygonal and polyhedral finite elements, *International Journal for Numerical Methods in Engineering* **102**(3–4): 728–747. nme.4851.

Teng, T.-L., Chang, F.-A., Liu, Y.-S. and Peng, C.-P. [2008]. Analysis of dynamic response of vehicle occupant in frontal crash using multibody dynamics method, *Mathematical and Computer Modelling* **48**(11): 1724–1736.

TexMesher [2016]. `http://texmesher.com/tex.html`.

Trelis [2016]. `http://www.csimsoft.com/trelis.jsp`.

Vaxman, A., Campen, M., Diamanti, O., Panozzo, D., Bommes, D., Hildebrandt, K. and Ben-Chen, M. [2016]. Directional field synthesis, design, and processing, *Computer Graphics Forum*, Vol. 35, pp. 545–572.

Wachspress, E. L. [1975]. *A Rational Finite Element Basis*, Vol. 114, Academic Press.

Wächter, A. and Biegler, L. T. [2006]. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming, *Mathematical Programming* **106**(1): 25–57.

Warren, J. [1996]. Barycentric coordinates for convex polytopes, *Advances in Computational Mathematics* **6**(1): 97–108.

Warren, J., Schaefer, S., Hirani, A. N. and Desbrun, M. [2007]. Barycentric coordinates for convex sets, *Advances in Computational Mathematics* **27**(3): 319–338.

Weber, O., Ben-Chen, M. and Gotsman, C. [2009]. Complex barycentric coordinates with applications to planar shape deformation, *Computer Graphics Forum* **28**(2): 587–597.

Weber, O., Ben-Chen, M., Gotsman, C. and Hormann, K. [2011]. A complex view of barycentric mappings, *Computer Graphics Forum* **30**(5): 1533–1542.

Weber, O. and Gotsman, C. [2010]. Controllable conformal maps for shape deformation and interpolation, *ACM Transactions on Graphics* **29**(4): Article 78, 11 pages.

Weber, O., Myles, A. and Zorin, D. [2012]. Computing extremal quasiconformal maps, *Computer Graphics Forum* **31**(5): 1679–1689.

Weber, O. and Zorin, D. [2014]. Locally injective parametrization with arbitrary fixed boundaries, *ACM Transactions on Graphics* **33**(4): Article 75, 12 pages.

Winkler, T., Drieseberg, J., Alexa, M. and Hormann, K. [2010]. Multi-scale geometry interpolation, *Computer Graphics Forum* **29**(2): 309–318.

Wohlmuth, B. I. [1998]. A mortar finite element method using dual spaces for the Lagrange multiplier, *SIAM Journal on Numerical Analysis* **38**: 989–1012.

XBX [2016]. `http://texmesher.com/kbx.html`.

Xu, C., Liu, J. and Tang, X. [2009]. 2D shape matching by contour flexibility, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **31**(1): 180–186.

Xue, D., Demkowicz, L. et al. [2005]. Control of geometry induced error in hp finite element (fe) simulations. i. evaluation of fe error for curvilinear geometries, *International Journal of Numerical Analysis and Modeling* **2**(3): 283–300.

Yamakawa, S. and Shimada, K. [2003]. Fully-automated hex-dominant mesh generation with directionality control via packing rectangular solid cells, *International Journal for Numerical Methods in Engineering* **57**(15): 2099–2129.

Zhang, H., Zhao, G. and Ma, X. [2007]. Adaptive generation of hexahedral element mesh using an improved grid-based method, *Computer-Aided Design* **39**(10): 914–928.

Zhang, Y. [1996]. A fuzzy approach to digital image warping, *IEEE Computer Graphics and Applications* **16**(4): 34–41.

Zhang, Y. and Bajaj, C. [2006]. Adaptive and quality quadrilateral/hexahedral meshing from volumetric data, *Computer Methods in Applied Mechanics and Engineering* **195**(9): 942–960.

Zhang, Y. J., Liang, X. and Xu, G. [2013]. A robust 2-refinement algorithm in octree or rhombic dodecahedral tree based all-hexahedral mesh generation, *Computer Methods in Applied Mechanics and Engineering* **256**: 88–100.

Zulian, P., Schneider, T., Hormann, K. and Krause, R. [n.d.]. Parametric finite elements with bijective mappings. Accepted at BIT numerical mathematics.