Die approbierte Originalversion dieser Dissertation ist an der Hauptbibliothek der Technischen Universität Wien aufgestellt (http://www.ub.tuwien.ac.at).

The approved original version of this thesis is available at the main library of the Vienna University of Technology (http://www.ub.tuwien.ac.at/englweb/).

Dissertation

Constrained Matching of Point Clouds and Surfaces

ausgeführt zum Zwecke der Erlangung
des akademischen Grades eines
Doktors der technischen Wissenschaften
unter der Leitung von

O.Univ.-Prof. Dr. Helmut Pottmann
Institut für Diskrete Mathematik und Geometrie (E104)

eingereicht an der Technischen Universität Wien,
Fakultät für Mathematik und Geoinformation,
von

Dipl.-Ing. Simon Flöry
Liniengasse 16/16, 1060 Wien

Zusammenfassung. Das Themengebiet dieser Dissertation umfasst die gegenseitige Ausrichtung und Anpassung (*matching*) geometrischer Objekte. Wir behandeln die Registrierung von zwei oder mehreren Punktwolken und die Rekonstruktion von B-spline Flächen aus diskreten Punktmengen. Die zu Grunde liegenden nichtlinearen und mit unter nicht glatten Optimierungsprobleme basieren auf einer Minimierung der nicht vorzeichenbehafteten beziehungsweise der quadratischen Distanzfunktion zwischen den Objekten. Verschiedene Anwendungen definieren Nebenbedingungen an diese Optimierungen. Die Registrierung von Punktwolken ohne gegenseitige Durchdringung führt zu einer realistischen Rekonstruktion zerbrochener Objekte. Für ein effektives Zusammenfügen zahlreicher Datensätze dreidimensionaler Oberflächenkoordinaten, gemessen in zeitlich kurzen Abständen, beschränken wir die Registrierung lokal auf kinematische Flächen. Der zweite Teil dieser Arbeit befasst sich mit der Approximation von Punktwolken durch Regelflächen. Zum einen verwenden wir die entwickelten Algorithmen zur Flächenrekonstruktion für die Bahnberechnung zylindrischer Fräser. Die Vermeidung von Unterschnitt lässt sich ebenso wie ein Voreilen des Fräskopfes als Nebenbedingung formulieren und erhöht die Qualität der erhaltenen Fräsbewegung. Für Anwendungen in der Architektur fügen wir mehrere Regelflächen zu Streifenmodellen zusammen und untersuchen Methoden zur Erzeugung glatter Übergänge zwischen den einzelnen Streifen.

Abstract. In this thesis we are concerned with the geometric matching of shapes. We consider the local and rigid alignment of point clouds and the reconstruction of ruled surfaces from point clouds within an active shape framework. The emerging non-linear and occasionally non-smooth optimization problems center around minimizations of the squared and unsigned distance function, respectively. Several application specific requirements impose side conditions on this optimization. The alignment of point clouds without mutual penetration reconstructs broken objects in a physical meaningful way. For an effective alignment of input data acquired at high frame rates, we constrain the registration of input shapes locally to a kinematic surface in a space-time model. We consider ruled surface approximations to compute optimal tool paths for production technologies. In particular, for cylindrical flank milling, we minimize undercut errors and stress along the cutting tool by introducing constraints. For architecture, we join multiple ruled surface patches and discuss ways to achieve smooth and visually pleasing designs.

Preface

This thesis collects the major results of my research between the years 2006 and 2009. I was mostly concerned with the registration of point clouds and the B-spline surface reconstruction from discrete point sets. It seems appropriate to summarize both fields under the common hood of *shape matching*. As my particular interest is in constraints of applied as well as of theoretical nature, I prefixed this shape matching as *constrained*.

Several parts of this thesis have been published in conference and journal papers that I (co-)authored. Chapter 2 contributed to (Huang et al., 2006) and (Thuswaldner et al., 2009), Chapter 3 was presented in (Flöry and Hofer, 2010) and Chapter 4 in (Mitra et al., 2007). The second part comprising Chapters 5 and 6 was primarily developed for two companies, Evolute GmbH and ModuleWorks GmbH. The first ideas for Section 5.3 originate from work for (Kilian et al., 2008) and Section 6.1 is based on generalizations of my master thesis published in (Flöry, 2009).

Over the years, my work has been gratefully supported by a Neumaier fellowship, and by the Austrian Science Fund FWF under grants P16002-N05 and P18865-N13. The implementations of the herein presented algorithms rely on various open source and/or free software projects (GCC, GPLK, LAPACK, CVXOPT, OOQP, ANN). The presentation of this thesis wouldn't be possible without LaTeX, povray, gnuplot and octave.

In the past, I had the great opportunity to learn from and work together with numerous outstanding people. First and foremost, I want to express my deep gratitude to my advisor Helmut Pottmann. His immense knowledge of geometry and his creativity have been and will be example for my work. I would like to thank Mario Botsch for taking the job as external reviewer and Michael Hofer for managing one of the research projects and saving me from all the administrative stuff.

I want to acknowledge the support of all my collaborators for publications and collegues at the research group for Geometric Modeling and Industrial Geometry at the Institute of Discrete Mathematics and Geometry at Vienna University of Technology. In particular, I want to thank Niloy Mitra for computing the data that makes up for Figures 4.2, 4.2, 4.8 and 4.9 and Heinz Schmiedhofer for doing the renderings of Figures 6.9 and 6.11. I appreciate the stimulating working environment at my second employer Evolute — a great place for writing a thesis and doing fascinating work for architecture and production technologies at the same time.

It is a neverending quest for me to reconcile work and life. Life minus work have been such exciting times in the past and this is due to extraordinary people around me. I cheer a big *Danke!* to my parents and family, friends, and in particular you, Evelyn.

Contents

1 Introduction **7**
- 1.1 Problem Statement . . . 8
- 1.2 Solving the Minimization Problems . . . 10
- 1.3 Examples . . . 12
  - 1.3.1 Local Registration . . . 13
  - 1.3.2 B–spline Surface Reconstruction . . . 13
- 1.4 Overview . . . 14

I Constrained Registration **15**

2 Registration without Penetration **16**
- 2.1 Related Work . . . 16
- 2.2 Unconstrained Pairwise Registration . . . 17
- 2.3 Constrained Pairwise Registration . . . 20
- 2.4 Simultaneous Constrained Registration . . . 21
- 2.5 Examples . . . 23
  - 2.5.1 Reassembling Fractured Objects . . . 23
  - 2.5.2 Reassembling Ancient Monuments . . . 24

3 Registration in the l_1-norm **27**
- 3.1 Related Work . . . 27
- 3.2 Error Terms for Registration in the l_1-norm . . . 29
- 3.3 Optimization . . . 30
  - 3.3.1 Proximal Bundle Method . . . 30
  - 3.3.2 Non-Smooth vs. Smooth Optimization . . . 32
- 3.4 Results . . . 34

4 Dynamic Geometry Registration **37**
- 4.1 Related Work . . . 38
- 4.2 Motion Parameter Computation in $\mathbb{R}^{3+1}$ . . . 39
- 4.3 Further Discussion . . . 41
  - 4.3.1 Time Scaling . . . 41
  - 4.3.2 Normal Estimation . . . 42
  - 4.3.3 Sensitivity to Noise . . . 44
  - 4.3.4 Relation to ICP . . . 45
  - 4.3.5 Extensions . . . 46
- 4.4 Examples . . . 47

II Constrained Surface Fitting **49**

5 Ruled Surface Approximation **50**
5.1 Related Work . . . 51
5.2 Ruled Surface Approximation . . . 55
5.3 Initial Fitting Shape . . . 58
5.3.1 Ruling Estimation . . . 59
5.3.2 Ruling Interpolation . . . 60
5.4 Initial Smoothing Weight . . . 61
5.5 Knot Adaption . . . 65

6 Constrained Ruled Surface Approximation for Applications **67**
6.1 Avoiding Undercut Errors . . . 67
6.2 Ruled Strip Models for Production Technologies . . . 69
6.3 Leading Tool Top . . . 71
6.3.1 A Kinematic Constraint . . . 72
6.3.2 Reducing the Number of Constraints . . . 73
6.3.3 Discretizing and Linearizing . . . 74
6.4 Ruled Surfaces in Architecture . . . 75
6.5 Smoothing Parameter Lines . . . 76
6.6 G^1 continuous Strip Models . . . 78
6.7 G^2 continuous Strip Models . . . 81

1 Introduction

Matching problems surround us any time, any place. Let it be the key that unlocks a door, the cloth we put on in the morning or the PIN we enter to withdraw some money. As common matching problems are in everyday life, as popular they are in mathematics. Many fields of mathematics such as graph theory or statistics study matching problems. In this work, we consider the matching of geometric shapes. In particular, we develop methods to transform or modify a given shape such that it matches a target shape in an optimal sense.

Shape matching is closely related to shape representation. Typically, the intended application or method of matching determines the employed shape model. Point clouds, polygonal meshes or freeform surface (Campbell and Flynn, 2001), implicit surfaces (Sethian, 1999; Osher and Fedkiw, 2003), objects from constructive solid geometry (McWherter et al., 2001) or spatial data structures such as voxels or binary space partitioning trees (Kazhdan and Funkhouser, 2002) are some of many representation techniques for shape matching. We will consider the matching of point clouds to points clouds and that of B-spline surfaces to point clouds. Our approach to shape representation will be to interpret both discrete point sets and B-spline surfaces as zero level sets of implicit functions. We will define our level set shape representation in detail in the following section.

The last years have seen a large body of research in the area of shape matching. The recent survey by (Tangelder and Veltkamp, 2008) and similar previous summaries (Loncaric, 1998; Campbell and Flynn, 2001; Mamic and Bennamoun, 2002) are excellent sources for a detailed and complete overview of existing methods. Instead of reproducing an extensive summary, we remain with a rough classification of shape matching techniques and cite representative work only. The remaining chapters of this work will discuss specific related literature much more precise.

Many shape matching strategies are *feature based*. These techniques compute features on the shape's geometry, that may be brought into correspondence subsequently. Alternatively, the relative position of features (the distribution of features) is investigated and compared without establishing correspondences of surface points directly. Local signatures such as spin images (Johnson, 1997), shape context (Belongie et al., 2002), multi-scale approaches (Li and Guskov, 2005; Pottmann et al., 2009), heat kernel (Sun et al., 2009) and global histogram techniques (Ankerst et al., 1999) may be attributed as feature based approaches. A controlled search of the underlying transformation space is popular for the matching of deformed shapes and is typically based on a combination of local and global features (Zhang et al., 2008; Lipman and Funkhouser, 2009). Contrary to feature based techniques, *graph based* methods combine information about a shape's geometry and topology. Model graphs (McWherter et al., 2001) and skeleton and Reeb graphs (Biasotti et al., 2003) are the most prominent shape graph models examined for

combinatorial similarity. All of these contributions to shape matching are *global* in a sense that there are no restrictions on the initial pose of the shapes. Our work considers *local* shape matching problems, assuming that the shapes are in rough alignment to each other. For the vast majority of this work, we are going to establish correspondences by nearest neighbor computations on point sets or B-spline surfaces. This may be seen as a greedy, low level feature based approach.

The intention of some shape matching literature and literature on shape retrieval in particular is a measure for shape dissimilarity. The main focus is on the Boolean result, whether two or more shapes match or not. We will not stop at this point but are going to modify the position or geometry of shapes such that dissimilarity is minimized. This will happen with regard to application specific *constraints*, that draw their motivation from a broad spectrum of applications such as physics, production technologies and architecture. In this context, we are going to consider the alignment or registration of point clouds and the B-spline surface reconstruction from point clouds as matching problems. The following section will formalize our problem statement.

1.1 Problem Statement

Let shape A be a subset of Euclidean three space $\mathbb{R}^3$. We define an embedding function $\phi : \mathbb{R}^3 \to \mathbb{R}$, $\phi(\mathbf{x}) = 0$ for $\mathbf{x} \in A$, that describes A implicitly by its zero level set. Given that A is the boundary $\partial\mathcal{A}$ of an open set $\mathcal{A} \subset \mathbb{R}^3$, we call any $\mathbf{x} \in \mathcal{A}$ *inside* and any $\mathbf{x} \in \mathcal{A}^C \setminus \partial\mathcal{A}$ *outside* of shape A.

The definition of ϕ is far from being unique with the values of $\phi(\mathbf{x})$, $\mathbf{x} \notin A$, arbitrary. The set of suitable embedding functions includes the signed distance function to A.

Definition. *Let* $\|\mathbf{x}\| : \mathbb{R}^3 \to \mathbb{R}$ *denote the Euclidean norm in* $\mathbb{R}^3$*. Then, functional* $d(A, \mathbf{x}) : \mathbb{R}^3 \to \mathbb{R}$*, with*

- $d(A, \mathbf{x}) = 0$ *for* $\mathbf{x} \in A$
- $d(A, \mathbf{x}) = -\min_{\mathbf{a} \in A} \|\mathbf{a} - \mathbf{x}\|$ *for* $\mathbf{x}$ *inside of* A
- $d(A, \mathbf{x}) = \min_{\mathbf{a} \in A} \|\mathbf{a} - \mathbf{x}\|$ *for* $\mathbf{x}$ *outside of* A

is called the signed distance function *to shape* A*.*

The level sets of ϕ, $\phi(\mathbf{x}) = t$, may be interpreted as an evolution of the zero level set over time. The signed distance function d describes an evolution of the zero level set at constant speed. It is the viscosity solution of a non-linear partial differential equation, the so-called *Eikonal* equation (Sethian, 1999),

$$\|\nabla d\| = 1, \qquad d(\mathbf{x}) = 0,\ \mathbf{x} \in A.$$

The Eikonal equation is a special case of the Hamilton-Jacobi equations, well-known in mechanics and widely investigated in mathematics (Lions, 1982).

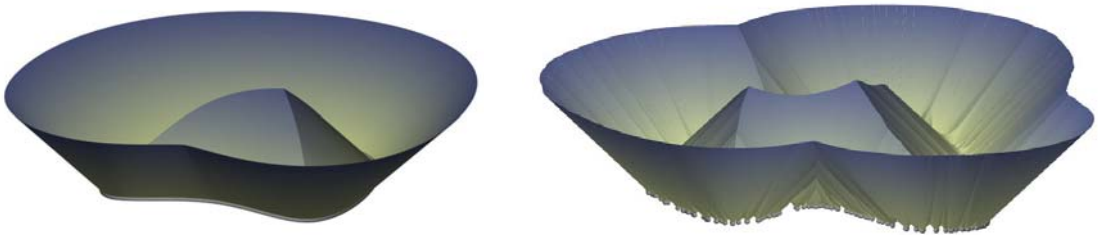

Figure 1.1: Graph of the unsigned distance function to a planar curve (left) and to a discrete point set (right).

Definition. *We call a point* $\mathbf{f}_x \in A$ *the* foot point *of the shortest distance from* $\mathbf{x} \in \mathbb{R}^3$ *to* A*, if*

$$\mathbf{f}_x = \operatorname{argmin}_{\mathbf{a}\in A} \|\mathbf{a} - \mathbf{x}\|.$$

If not stated otherwise, the abbreviated expression "foot point on A" will refer to the foot point of the shortest distance to A. The set of points in $\mathbb{R}^3$, for which the foot point is not unique, is called the *medial axis* of A. In points of the medial axis, the signed distance function is not differentiable.

For the purpose of describing proximity to a shape the sign of $d(A, \mathbf{x})$ becomes redundant. Loosely speaking, it shall not be relevant if we are inside or outside of A if we aim at getting close to A. For this reason, it turns out useful to either employ the absolute value $|d(A, \mathbf{x})|$ or square the signed distance function $d^2(A, \mathbf{x})$ for computations. We summarize both cases notation-wise by writing $d^*(A, \mathbf{x}) = \|\mathbf{f}_x - \mathbf{x}\|^*$, $* = 1, 2$. In other words, we measure the length of the residual vector $\mathbf{f}_x - \mathbf{x}$ with respect to different norms.

The unsigned and squared distance function lead straight forward to a notion of distance between two shapes A and B. We call

$$d^*(A, B) = \int_B d^*(A, \mathbf{b}) \, \mathrm{d}B$$

the unsigned or squared distance from B to A. In general, $d^*(A, B)$ will not equal $d^*(B, A)$ as the foot point relation "$\mathbf{a}$ is the foot point of $\mathbf{b}$" is not symmetric. In order to emphasize this asymmetry, we will call A the *base* and B the *query* shape.

Now, let $T(A) \in \mathcal{T}$ denote a manipulation of shape A by means of a transformation or shape modification. We will get more specific about sets of manipulators $\mathcal{T}$ later. In the meantime, we are able to formally state the matching problems considered in this work. Depending on whether we apply T to the base or to the query shape, we obtain two scenarios: Given two shapes A and B, find that manipulator T such that the distance from B to A is minimized,

$$\min_{T\in\mathcal{T}} d^*(A, T(B)) \qquad \text{or} \qquad \min_{T\in\mathcal{T}} d^*(T(A), B). \tag{1.1}$$

1.2 Solving the Minimization Problems

Without giving any further details about above matching problems we see that minimizations of Equ. (1.1) are of a non-linear nature. In this section, we sketch a solution to these optimizations problems that will be outlined in much more detail in the remaining chapters of this work. Our aim here is to show that most of what follows fits into a general unique framework. We will consider the first minimization problem of Equ. (1.1) only. Results for the second problem $\min_{T\in\mathcal{T}} d^*(T(A), B)$ are obtained in a similar manner.

The non-linear dependancy on T in

$$d^*(A, T(\mathbf{x})) = \|\mathbf{f}_{T(\mathbf{x})} - T(\mathbf{x})\|^*,$$

is tracked down to two main sources. Once, the foot point depends non-linearly on T. Second, at a higher level, d^* is non-linear, apart from that an explicit representation of it won't be available for general shapes. We will tackle these issues with an *iterative* and *alternating* approach to minimization. We first fix manipulator T and compute the foot points of the query shape on the base shape. Then, we fix the foot points and find optimal T by minimizing approximations of d^*. These two steps are repeated over several iterations. The foot point computations for fixed T relate to the exact represenation of the base shape but will pose no further difficulties (cf. Sec. 1.3). For the second part of an iteration, we will derive local approximations of both $|d|$ and d^2 in the following.

The first approximation regards the query shape B as mathematical model that shall be fit to an observation set, in our case the foot points of B on A. Similar setups are very well-known in mathematics and we minimize either the l_1-norm or the squared l_2-norm of the residual

$$d(A, \mathbf{x}) \approx \begin{cases} \|\mathbf{f}_x - \mathbf{x}\| & \mathbf{x} \text{ not inside } A \\ -\|\mathbf{f}_x - \mathbf{x}\| & \mathbf{x} \text{ inside } A \end{cases}$$

with respect to unknown T,

$$|d(A, T(\mathbf{x}))| \approx \|\mathbf{f}_x - T(\mathbf{x})\| \qquad \text{and} \qquad d^2(A, T(\mathbf{x})) \approx \|\mathbf{f}_x - T(\mathbf{x})\|^2.$$

Please note that the residual might well be interpreted as Taylor approximation of order 0 of d. Following the notation of (Wang et al., 2006) we will call these approximations of the signed distance function *point to point* distance terms.

Given that d is differentiable in a foot point $\mathbf{f}_x$, we may compute higher-order approximations of the signed distance function. Let us consider a linearization of the signed distance function,

$$d(A, \mathbf{x}) \approx d(A, \mathbf{f}_x) + \nabla d(A, \mathbf{f}_x)^T \cdot (\mathbf{x} - \mathbf{f}_x).$$

As $\mathbf{f}_x \in A$, the first term vanishes. Recalling that d is an evolution of the zero level set at constant speed, $\nabla d(A, \mathbf{f}_x)$ coincides with the outward oriented normal vector of shape A in $\mathbf{f}_x$. This observation leads straight forward to a geometric interpretation of the first-order Taylor approximation. Linearization of the signed distance function in a

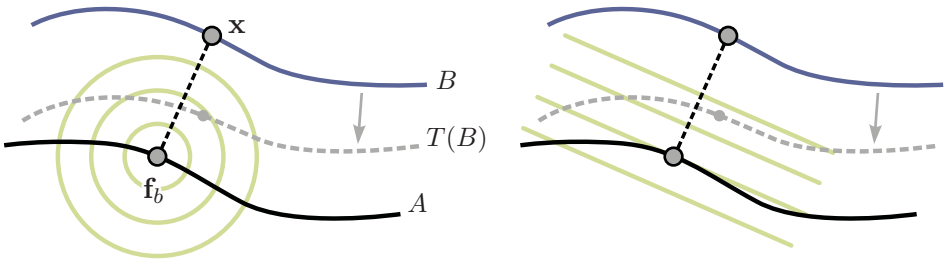

Figure 1.2: Level sets of the point to point (left) and point to tangent plane (right) error terms for local registration.

foot point $\mathbf{f}_x$ yields the signed distance to the tangent plane in that foot point. Taking the residue's l_1-norm completes the second approximation for $|d|$,

$$|d(A, T(\mathbf{x}))| \approx \left|\nabla d(A, \mathbf{f}_x)^T \cdot (\mathbf{f}_x - T(\mathbf{x}))\right|.$$

For the squared distance function, we obtain,

$$d^2(A, T(\mathbf{x})) \approx \left[\nabla d(A, \mathbf{f}_x)^T \cdot (\mathbf{f}_x - T(\mathbf{x}))\right]^2.$$

Given that $d^2(A, \mathbf{x})$ is twice differentiable in a foot point, second-order Taylor approximation of d^2 yields,

$$d^2(A, \mathbf{x}) \approx d^2(A, \mathbf{f}_x) + \nabla d^2(A, \mathbf{f}_x)^T \cdot (\mathbf{x} - \mathbf{f}_x) + \frac{1}{2}(\mathbf{x} - \mathbf{f}_x)^T \nabla^2 d^2(A, \mathbf{f}_x)(\mathbf{x} - \mathbf{f}_x).$$

Recalling that $d(A, \mathbf{f}_x) = 0$, the gradient $\nabla d^2 = 2\nabla d \cdot d = 0$ vanishes and we rewrite the Hessian,

$$\nabla^2 d^2(A, \mathbf{f}_x) = 2(d \cdot \nabla^2 d + \nabla d \cdot \nabla d^T) = 2\nabla d \cdot \nabla d^T.$$

We see that above squared l_2-norm of the linearized signed distance function is indeed a second-order Taylor approximation of d^2. We summarize both higher-order approximations as *point to tangent plane* distance approximations.

Curvature based approximations of the squared distance function (Pottmann and Hofer, 2003) and similar considerations regarding the signed distance function (Flöry and Hofer, 2010) show theoretic improvements over above approximations. We do not further discuss any higher-order approximations but point out that they suit well into the presented algorithmic framework.

In summary, a solution to above minimization problems can be found iteratively by computing the foot points at the beginning of an iteration, approximating d^* in these foot points and minimizing these approximations. The following algorithm summarizes these steps.

Algorithm. *A solution for the non-linear minimization problems of Equ. (1.1) comprises the following steps.*

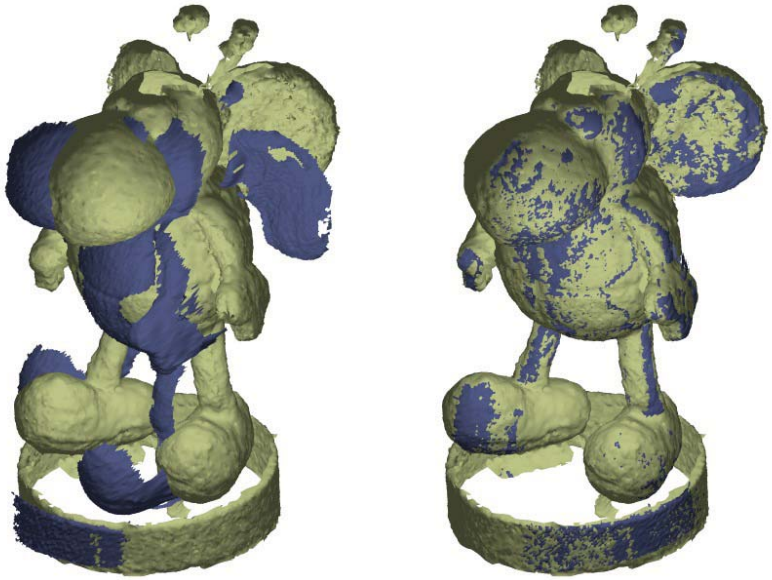

Figure 1.3: Local registration of two point clouds as an example for geometric shape matching. (Left) The two input point clouds are in rough alignment to each other. (Right) Final alignment. The point clouds have been triangulated for better visualization

1. *Initialize the problem. For non-linear optimization problems, a proper initialization is crucial for convergence (Kelley, 1999).*
2. *Compute the foot points for the elements of the query shape on the base shape.*
3. *Approximate d^* in these foot points according to one of the methods above.*
4. *Minimize the approximated distance between query shape and base shape with respect to the unknown shape manipulator.*
5. *Apply the minimizing manipulator. If the distance between the shapes is below a user-defined threshold, stop. Otherwise continue at step 2.*

1.3 Examples

We will conclude the introduction by giving two examples for shape matching problems. These will be the basic problems considered in the remaining two parts of this work. A thorough investigation of the single steps of above algorithm, any extensions and constraints as well as convergence properties will be detailed in the following chapters.

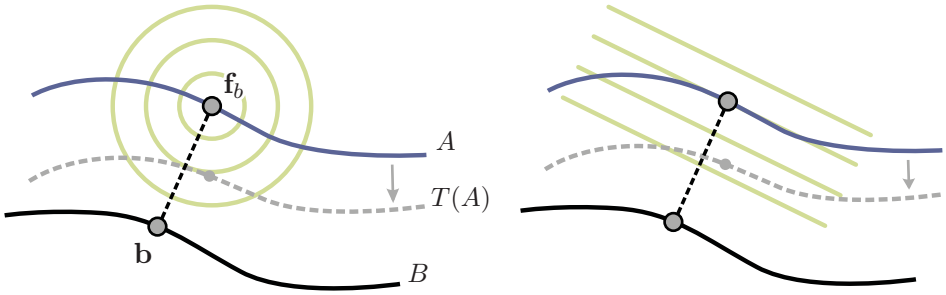

Figure 1.4: Level sets of the point to point (left) and point to tangent plane (right) error terms for curve/surface fitting.

1.3.1 Local Registration

The first matching problem we discuss is the local registration of two point clouds $A = \{\mathbf{a}_i \in \mathbb{R}^3 : \ i = 1, \dots, n_A\}$ and $B = \{\mathbf{b}_i \in \mathbb{R}^3 : \ i = 1, \dots, n_B\}$. Given that the two shapes A and B are in rough alignment to each other, our goal is to find that rigid body motion $T \in \mathcal{T}$ minimizing the distance between the two shapes. Thus, the set of feasible manipulators $\mathcal{T}$ will be the Euclidean group of rigid body transformations E_3^+ on $\mathbb{R}^3$. We fix point cloud A, the *base* or *target system*, and minimize the distance of $T(B)$ (the *query* or *moving system*) to A. The objective of the non-linear minimization problem is given by

$$\min_{T \in E_3^+} \sum_{i=1}^{n_B} d^*(A, T(\mathbf{b}_i)). \tag{1.2}$$

The definition of d^* is based on the foot points of B on A. As both A and B are discrete point sets, the foot point $\mathbf{f}_b$ is identified with the nearest neighbor of $\mathbf{b}$ in A. For the point to tangent plane approximations of d^* we require the signed distance function to be differentiable in the foot points. This requirement is not met for point clouds. Nevertheless, it is possible to estimate normals in the elements of the target point cloud (cf. Sec. 4.3.2) and consider A to behave locally like the estimated tangent plane in $\mathbf{f}_b$.

1.3.2 B–spline Surface Reconstruction

The second problem we consider within above matching framework is the surface reconstruction from point cloud data. For a given point cloud $B = \{\mathbf{b}_i \in \mathbb{R}^3 : \ i = 1, \dots, n_B\}$, our goal will be to compute that tensor product B–spline surface $A(u, v) = \sum_{i=1}^{n_A} N_i(u, v)\mathbf{d}_i$ approximating B best. B-spline surface A will be the deforming shape as well as the base shape for foot point computations. We modify the shape of A by displacing its control points $\mathbf{d}_i$. Accordingly, $\mathcal{T} = \mathbb{R}^{3n_A}$ comprises all $3 \cdot n_A$ dimensional translation vectors.

Putting things together, the objective of the non-linear surface fitting optimization

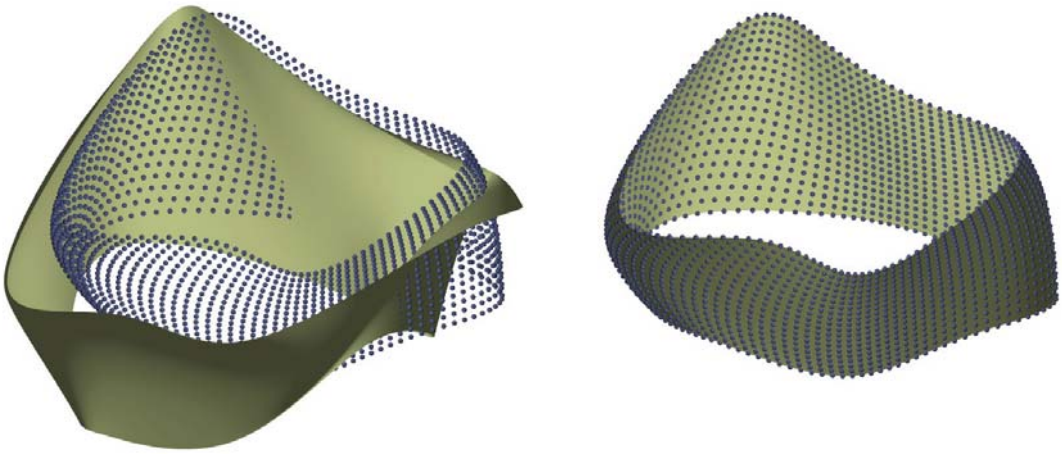

Figure 1.5: Fitting a B–spline surface to a point cloud as an example of geometric shape matching. (Left) Initial point cloud and approximating surface. (Right) Final approximation.

problem is of the form,

$$\min_{T\in\mathbb{R}^{3n_A}} \sum_{i=1}^{n_B} d^*(T(A), \mathbf{b}_i).$$

The foot point of $\mathbf{b}_i$ on a parametric surface $A(u, v)$ is obtained in a Newton iteration minimizing $\min_{(u,v)} \|A(u, v)-\mathbf{b}_i\|^2$. As opposed to the registration problem, $\nabla d(A(u, v))$ is given as surface normal in $A(u, v)$.

1.4 Overview

In this section we defined our understanding of shapes and developed a general framework for the subsequent chapters. How different registration and surface fitting problems might occur at first sight, it is worth to keep in mind how similar they are at the core.

The remainder of this thesis is split into two parts. The first part considers the constrained registration of point clouds. The constraints will be of very diverse nature. Chapter 2 formulates side conditions preventing mutual penetration of aligning shapes. The third chapter investigates the constrained alignment of point clouds with respect to the l_1-norm of residues. The fourth chapter discusses the registration of point clouds, acquired at high temporal density. In a space-time model, this will constrain the alignment locally to a kinematic surface.

Part II shifts the focus to surface fitting problems and the B-spline surface reconstruction from discrete point sets. In Chapter 5, the main restriction will be that we employ ruled surface patches as fitting entities only. Finally, Chapter 6 will show that many application specific requirements from production technologies and architecture are effectively stated as constrained minimization problems.

Part I

Constrained Registration

2 Registration without Penetration

Registration is one of the most prominent matching problems in computer vision and geometry processing. In its most basic form, the goal is to register (or align or match) a *moving* shape by a rigid body transformation to a *target* shape. The rapid development of shape acquisition devices in recent years lets registration enjoy an ever increasing number of applications. Alignment algorithms are used to combine 3D information obtained from different view points into a common coordinate system. In industrial inspection, machine parts are scanned and registered onto their CAD models to control abrasion. More abstract applications rely on registration as well, for example (Botsch et al., 2006) and (Kilian et al., 2008), who employ registration to glue and match soups of prisms/polygons.

Among this wealth of applications, alignment algorithms are used to solve 3D puzzles. Imagine a solid object broken into several fragments. Its reassembly may be considered a 3D puzzle. In classic registration, we tolerate or even require a final alignment to distribute matching errors equally across the model, typically in some least-squares sense. However, physical facts make it impossible to let matching fragment surfaces of a 3D puzzle penetrate each other. In this chapter, we will develop an algorithm to align two or more matching shapes in a penetration free way.

Our work is based on the well established ICP class of registration algorithms, introduced by (Besl and McKay, 1992) and (Chen and Medioni, 1992). ICP chooses an iterative approach to compute an optimal aligning transformation between the input shapes. Hence, it suits well into the framework of the previous chapter and solves the non-linear matching problem in a series of iterations with quadratic objective. We will formulate the penetration free constraint as side condition to these optimization problems thus arriving at a *constrained optimization problem*.

2.1 Related Work

It is common to organize the vast body of literature on rigid registration in two groups. Depending on whether the shapes to be matched are in general or roughly aligned position to each other we speak of a *global* or *local* registration problem. Beyond this section on related work we are going to consider the local registration problem and we will skip the attribute "local" most times.

Observing that the dimension of the rigid alignment problem is rather small (the six degrees of freedom of a rigid body motion in space), a direct search of the solution space for the optimal solution seems suitable. In computer vision, several methods have been proposed building upon this observation. Generalizations of the Hough transform to determine boundaries (Ballard, 1981; Hecker and Bolle, 1994), counting the support of solution candidates with geometric hashing (Hecker and Bolle, 1994; Wolfson and

Rigoutsos, 1997) or randomly sampling the solution space with RANSAC (Fischler and Bolles, 1981) are prominent examples. Recent work based on the geometric hashing paradigm comprises (Gal and Cohen-Or, 2006); the RANSAC principle in combination with 4 point bases is used in (Aiger et al., 2008).

One way of bypassing the need to search most of the solution space are features. Instead of matching the input shapes directly, the problem is reduced to match sets of features. The latter task is often referred to as the *correspondence* problem, describing the need to identify a counterpart for each feature. If the correspondences are known a priori, an explicit least-squares solution was proposed by (Horn, 1987). For unknown correspondences, surface descriptors are used as features. In Chapter 1 we have already mentioned several approaches to global registration. These include spin images (Johnson, 1997), shape contexts (Frome et al., 2004), multi-scale integral invariants (Gelfand et al., 2005) or multi-scale normal differences (Li and Guskov, 2005).

For a comprehensive solution, a global matching technique is often combined with a local method as final refinement step (e.g. Huang et al., 2006). For local registration, the proximity of shapes makes the use of features obsolete. Instead, correspondences are established iteratively by computing closest points between the shapes. This is the basic idea of the classic *Iterative Closest Point* (ICP) algorithms proposed originally by Besl and McKay (1992) and (Chen and Medioni, 1992) and used in many variants nowadays, see (Rusinkiewicz and Levoy, 2001) for a survey. In their initial formulation, these techniques align two shapes. Multiple shapes may be matched pair by pair (Levoy et al., 2000; Bernardini and Rushmeier, 2002), the major drawback being accumulative errors. *Simultaneous* or *multi-view* registration leaves this shortcoming behind by considering all systems at the same time and distributing alignment errors equally (Bergevin et al., 1996; Pulli, 1999).

Above contributions all focus on computing *rigid body motions* to align the input shapes. *Non-rigid* registration methods build on extending the rigid ICP algorithms (Hähnel et al., 2003), learning correspondences in a pose deformation model (Anguelov et al., 2005) or canceling out low frequency acquisition errors (Brown and Rusinkiewicz, 2007).

Our work extends the ICP approach and classifies as local and rigid registration algorithm. We will begin by shortly recapitulating the basic ICP algorithm. Then, we will introduce constraints to achieve penetration free alignments which we generalize to multi-view setups. We conclude this chapter with two extensive examples.

2.2 Unconstrained Pairwise Registration

For a start, let us consider the pairwise registration problem in $\mathbb{R}^3$ and its well-known solutions. Given a target point cloud $A = \{\mathbf{a}_i : \ i = 1, \dots, n_A\}$ and a moving point cloud $B = \{\mathbf{b}_i : \ i = 1, \dots, n_B\}$, we want to compute that rigid body motion,

$$\mathbf{m}(\mathbf{b}) = M \cdot \mathbf{b} + \mathbf{t}, \qquad M \in SO_3, \ \mathbf{t} \in \mathbb{R}^3,$$

minimizing the squared distance $d^2(A, \mathbf{m}(B))$, with $\mathbf{m}(B) = \{\mathbf{m}(\mathbf{b}) : \mathbf{b} \in B\}$. In this chapter, we will not consider the absolute value of the signed distance function. This

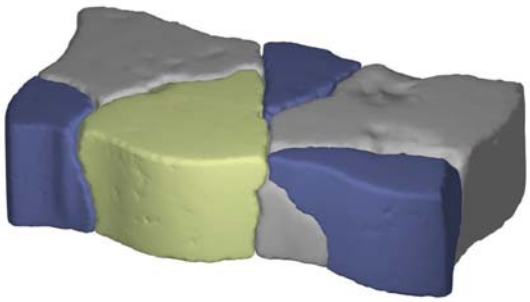

Figure 2.1: Penetration free reassembly of a brick model.

will be the topic of the next chapter.

The unknown of the optimization problem is shape manipulator $\mathbf{m}$. Classic work on registration either computes $\mathbf{m}$ explicitly from pairs of corresponding points (Besl and McKay, 1992) or — if an explicit solution is not available — includes the entries of M and $\mathbf{t}$ in a constrained way as unknowns in the minimization (Chen and Medioni, 1992). In this work, we will choose a different approach discussed in (Pottmann et al., 2002), linearizing the unknown rigid body motion with instantaneous kinematics. We will review some basic facts on instantaneous kinematics shortly, for details we refer the reader to (Pottmann and Wallner, 2001).

Let $\mathbf{m}(\mathbf{b})$ be an instance of a smooth one-parameter family of Euclidean motions,

$$\mathbf{m}(t) = M(t) \cdot \mathbf{b} + \mathbf{t}(t).$$

The first derivative of $\mathbf{m}(t)$, the velocity vector field, is known to be linear and of the form,

$$\mathbf{v}(\mathbf{m}(t)) = \dot{\mathbf{m}}(t) = \bar{\mathbf{c}}(t) + \mathbf{c}(t) \times \mathbf{m}(t), \qquad \bar{\mathbf{c}}(t), \mathbf{c}(t) \in \mathbb{R}^3.$$

A first-order Taylor approximation of $\mathbf{m}(t)$ in $t = 0$ reads in this notation as,

$$\mathbf{m}(\mathbf{b}) \approx \mathbf{b} + \bar{\mathbf{c}} + \mathbf{c} \times \mathbf{b}.$$

We will employ this linearization of the shape manipulator $\mathbf{m}(b)$ in the minimization. Consequently, $(\bar{\mathbf{c}}, \mathbf{c})$ will be the unknowns of the registration problem's iterations. Once we obtained a minimum, we observe that in general the mapping $\mathbf{b} \to \mathbf{b} + \mathbf{v}(\mathbf{b})$ is not a Euclidean motion, but an affine transformation. Hence, we describe two ways for reconstructing a rigid body motion from $\bar{\mathbf{c}}$ and $\mathbf{c}$ in the following.

Consider a one-parameter motion with constant velocity vector field $(\bar{\mathbf{c}}, \mathbf{c})$. Such a motion is called *uniform* and is either the trivial motion ($\bar{\mathbf{c}} = \mathbf{c} = \mathbf{0}$), a translation ($\mathbf{c} = 0$), a rotation about a fixed axis ($\bar{\mathbf{c}} = \bar{\mathbf{c}}^T\mathbf{c} = 0$) or a superimposition of a rotation and translation along the fixed rotation axis (uniform helical motions for $\bar{\mathbf{c}}^T\mathbf{c} \neq 0$). (Pottmann and Wallner, 2001) show how to compute a general uniform helical motion from given $(\bar{\mathbf{c}}, \mathbf{c})$ in a computationally cheap way. The second method, proposed by

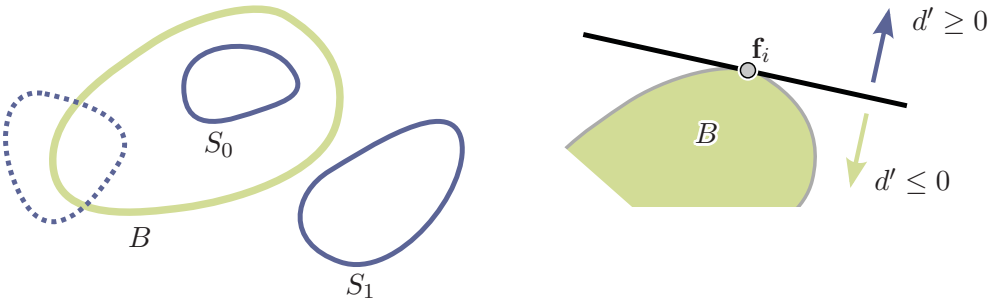

Figure 2.2: (Left) The two blue shapes with solid boundary S_0 and S_1 are in penetration free alignment with the yellow base shape B. (Right) Deriving linear constraints for a penetration free alignment. d' denotes a first-order approximation of the signed distance function in $\mathbf{f}_i$.

(Botsch et al., 2006), considers the registration problem with known correspondences of the initial configurations $\{\mathbf{b}_i\}$ to their affine images $\{\mathbf{b}_i + \mathbf{v}(\mathbf{b}_i)\}$. Using Horn's quaternion approach (Horn, 1987), which comprises eigenvalue computations for a 4×4 matrix, the rigid body motion describing the affine mapping best in a least-squares sense is computed. As we expect only small displacements for local registrations, we use the first, computationally cheap technique.

Now that we have described how the shape manipulator enters the optimization problem, we are able to cast the classic ICP algorithms in the minimization framework presented in the introductory chapter. First, the closest points for the elements of B are computed in A. Let $\mathbf{f}_i \in A$ denote the foot point of $\mathbf{b}_i$ in A. Then, the squared distance function to A is approximated in these closest points. The first approximation of d^2, the point to point error term, was introduced by (Besl and McKay, 1992) to registration. In our notation with instantaneous kinematics, the objective reads,

$$\min_{(\bar{\mathbf{c}},\mathbf{c})} \sum_{i=1}^{n_B} \|\mathbf{b}_i + \mathbf{v}(\mathbf{b}_i) - \mathbf{f}_i\|^2 .$$

The higher-order approximation of d^2, minimizing the squared distance to the tangent plane in the foot point, was proposed by (Chen and Medioni, 1992). Given that $\mathbf{n}_i$ is a normal in the closest point $\mathbf{f}_i \in A$ of $\mathbf{b}_i$, the optimization problem is of the form,

$$\min_{(\bar{\mathbf{c}},\mathbf{c})} \sum_{i=1}^{n_B} \left[\mathbf{n}_i^T \cdot (\mathbf{b}_i + \mathbf{v}(\mathbf{b}_i) - \mathbf{f}_i)\right]^2 .$$

Both objectives are quadratic in the unknowns $\bar{\mathbf{c}}$ and $\mathbf{c}$ and minimization amounts to the solution of a six-dimensional system of linear equations. The convergence characteristics of both objectives have been investigated by (Pottmann et al., 2006) who derive that the point to point error terms exhibits linear convergence. The point to tangent plane approximation is shown to be a Gauss-Newton method and is thus of quadratic local

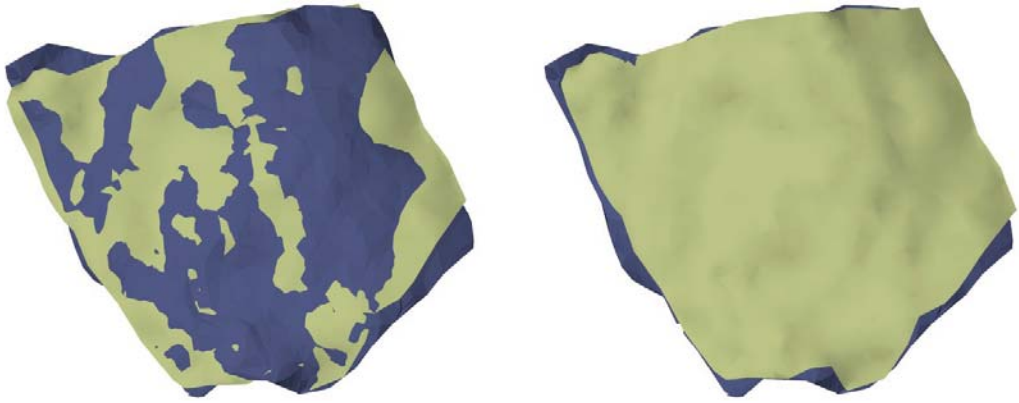

Figure 2.3: Unconstrained (left) and penetration free alignment (right) of two point sets (triangulated for better visualization).

convergence for zero residual problems. As a consequence, a minimizer $(\bar{\mathbf{c}}, \mathbf{c})$ does not yield the optimal update but the descent direction and requires proper step size control (Kelley, 1999).

2.3 Constrained Pairwise Registration

In various real world applications of registration, the input shapes are supposed to align without any mutual penetration. This requirement is translated straight forward to a mathematical context (see also Fig. 2.2).

Definition. *Two shapes A and B (with A the boundary of an open set in $\mathbb{R}^3$) are said to be in* penetration free *or* one-sided *alignment, if $d(A, \mathbf{b})$ is either positive for all $\mathbf{b} \in B$ or negative.*

This definition along with the general registration objective leads directly to a constrained minimization problem for penetration free registration,

$$\begin{array}{lcll} \min_T d^*(A, T(B)) & \text{subject to} & d(A, T(\mathbf{b})) > 0, & \forall \mathbf{b} \in B \\ & \text{or} & d(A, T(\mathbf{b})) < 0, & \forall \mathbf{b} \in B. \end{array}$$

In the following we will consider the first case $d(A, T(\mathbf{b})) > 0$ only. Our results are migrated to the opposite case easily by replacing the greater signs with less signs. To make above definition apply to non-closed shapes, we close them artificially. Practically, an orientated normal field of A suffices, even for A a discrete point cloud, as is shown below.

Some of the most popular methods to solve constrained non-linear optimization problems are summarized as *Sequential Quadratic Programming* in optimization literature (Nocedal and Wright, 1999). They solve the non-linear problem in a series of approximated quadratic programs (with linearly constrained quadratic objectives). We recall

that,

$$d(A, T(\mathbf{b}_i)) \approx \mathbf{n}_i^T \cdot (T(\mathbf{b}_i) - \mathbf{f}_i),$$

with $\mathbf{n}_i$ the outward oriented normal vector in $\mathbf{f}_i$ and obtain the following constrained optimization problems per iteration,

$$\begin{aligned} &\min_{(\bar{\mathbf{c}},\mathbf{c})} \sum_{i=1}^{n_B} \|\mathbf{b}_i + \mathbf{v}(\mathbf{b}_i) - \mathbf{f}_i\|^2 \\ &\qquad\qquad \text{s. t.} \quad \mathbf{n}_i^T \cdot (\mathbf{b}_i + \mathbf{v}(\mathbf{b}_i) - \mathbf{f}_i) > 0, \quad \forall i. \\ &\min_{(\bar{\mathbf{c}},\mathbf{c})} \sum_{i=1}^{n_B} \left[\mathbf{n}_i^T \cdot (\mathbf{b}_i + \mathbf{v}(\mathbf{b}_i) - \mathbf{f}_i)\right]^2 \end{aligned}$$

There are mainly two groups of algorithms solving this kind of optimization problems (Nocedal and Wright, 1999). *Active set* methods constitute the first group of solutions. They remain in the feasible solution space by continuously estimating and updating sets of active constraints describing the constraints locally. In their emphasis on the boundary of the feasible set they resemble the well-known Simplex method for solving linear programs. *Interior point* methods, on the other hand, have been generalized from linear programs to quadratic and general convex programs. They traverse, as their name suggests, strictly the interior of the feasible set.

Active set methods apply well to small scaled problems, whereas Interior point methods perform better for large scale problems. For constrained registration, the dimension of the solution space is six. However, the number of linear constraints equals the possibly large number of points in the moving point cloud. In its dual formulation, the dimension of the optimization problem equals basically the number of constraints of the primal problem. Hence, we may regard the constrained registration problem as medium to large scaled and we are using an Interior point method described in (Gertz and Wright, 2003) to solve the quadratic programs.

We may summarize as follows. We achieve a penetration free and one-sided alignment of two shapes A and B by iteratively minimizing an approximation of the squared distance function from A to B. This minimization is constrained in such a way that every element $\mathbf{b}_i \in B$ is displaced in the half space defined by its foot point's tangent plane and outward oriented normal vector.

2.4 Simultaneous Constrained Registration

Our motivating example for penetration free registration — the reassembly of broken objects — will not comprise pairs of point clouds only but multiple input shapes. Performing a constrained registration for every pair of discrete point sets might depict a sufficient solution at first sight. However, the high error propagation renders this option useless. In Chapter 4 we will illustrate the severe accumulative errors of subsequent pairwise registrations. Constrained *simultaneous* registration of all point clouds overcomes this problem. Instead of repeatedly registering pairs of point sets, the whole scene is brought into better alignment at once. In this section, we will generalize our hitherto existing results to a multi-view constrained registration of multiple shapes. Therefore, we will employ the same approach as in (Pottmann et al., 2002).

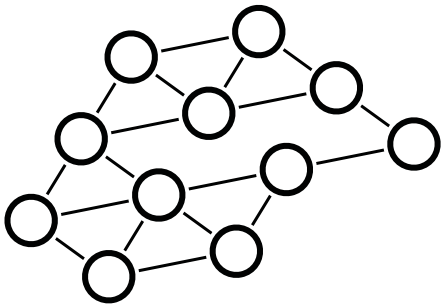

Figure 2.4: The adjacency graph for the cake example in Fig. 2.6.

Let A_i, $i = 0, \ldots, n$, be the $n + 1$ input point sets of simultaneous registration. In general, a single system will not be adjacent to all the other point clouds. Two point clouds are called *neighboring* if their distance is below a user-defined threshold. We use an adjacency relation,

$$a_{jk} = \begin{cases} 1 & A_j \text{ neighboring } A_k \\ 0 & \text{else} \end{cases}$$

to describe the combinatorics of neighborhoods. Let $\mathcal{A} := \{(j,k) : j = 0, \ldots, n,\ k = j+1, \ldots, n,\ a_{jk} = 1\}$ denote the set of adjacent point cloud pairs (see Fig. 2.4).

Without loss of generality we choose A_0 as fixed system. We linearize the unknown motions as before. For any point cloud A_i ($i > 0$), let

$$\mathbf{v}_{i0}(\mathbf{x}) = \bar{\mathbf{c}}_i + \mathbf{c}_i \times \mathbf{x},$$

denote the velocity vector field of its unknown motion towards fixed A_0. The relative velocity of system A_j towards system A_k is described with respect to the fixed system (Pottmann and Wallner, 2001),

$$\mathbf{v}_{jk}(x) = \mathbf{v}_{j0}(x) - \mathbf{v}_{k0}(x), \qquad \forall (j,k) \in \mathcal{A}.$$

The unknowns of the simultaneous registration problem are the $2 \cdot n$ vectors $\bar{\mathbf{c}}_i$ and $\mathbf{c}_i$.

The matching error between two adjacent systems A_j and A_k is approximated with the point to point or the point to tangent plane error as for the pairwise case. The objective of the whole setup is given by summing up all pairwise error terms. As constraints, above linearization is employed. The velocity vectors v_{jk} constitute the set of unknowns,

$$\min_{(\bar{\mathbf{c}}_1, \mathbf{c}_1, \ldots, \bar{\mathbf{c}}_n, \mathbf{c}_n)} \sum_{(j,k)\in\mathcal{A}} \sum_{\mathbf{b}\in A_j} \left[\left(\mathbf{b} + \mathbf{v}_{jk}(\mathbf{b}) - \mathbf{f}_b^k \right)^T \cdot \mathbf{n}_b^k \right]^2$$

$$\text{subject to} \quad \left(\mathbf{b} + \mathbf{v}_{jk}(\mathbf{b}) - \mathbf{f}_b^k \right)^T \cdot \mathbf{n}_b^k > 0 \qquad \forall (j,k) \in \mathcal{A},\ \mathbf{b} \in A_j.$$

Here, $\mathbf{f}_b^k$ denotes the closest point of $\mathbf{b} \in A_j$ in A_k and $\mathbf{n}_\mathbf{b}^k$ a normal of A_k in $\mathbf{f}_b^k$. The formulation of the multi-view point to point based registration problem is obtained in a similar construction.

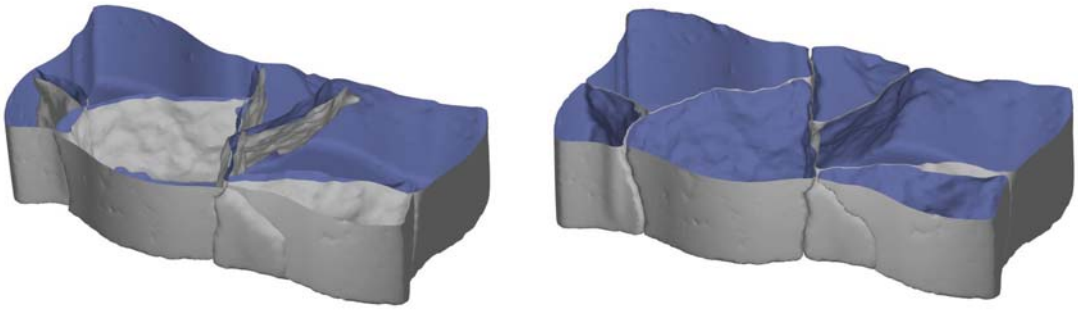

Figure 2.5: The initial poses of a broken brick's fragments (left) are aligned in a penetration free way (right).

In summary, we obtain a constrained optimization problem with quadratic objective and linear side conditions that is solved with an Interior point method. The resulting velocity vector fields $\mathbf{v}_{i0}$ are translated to rigid body motions with any method discussed before.

2.5 Examples

As stated before, the geometric matching algorithms in the present work are of a local nature. Local in this sense means that the algorithms are designed with the assumption in mind that the initial poses of the shapes are in rough alignment to each other. For local registration, the ICP methods by (Besl and McKay, 1992) and (Chen and Medioni, 1992) and their variants constitute the de facto standard and state of the art. In contrast, global registration motivated a wealth of solutions, many specific to a certain class of applications.

2.5.1 Reassembling Fractured Objects

In this first example, we consider the 3D puzzle problem that served as motivation for penetration free alignments. Given the fragments of a recently broken solid, we ask for a reconstruction of the original object. Such a reassembly involves identification of matching fragments (the global matching problem) and alignment of the corresponding pieces (the local constrained registration problem). The input data consists of digital point cloud models of the fragments, obtained for example with a 3D laser scanner. Please note that the generation of the digital input data for a fragment most likely involves unconstrained simultaneous registration to merge the multiple captured views into a single coordinate system.

In (Huang et al., 2006), the presented constrained registration algorithm serves as final local alignment stage in a reassembling pipeline. We want to sketch the associated global matching solution shortly to give a complete picture of the general matching problem. In the first step of the pipeline, the fragments' surfaces are segmented into

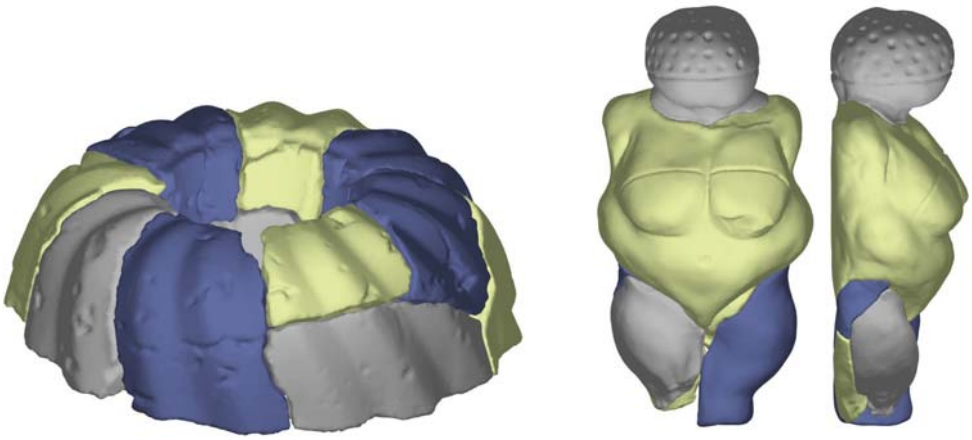

Figure 2.6: Penetration free reassembly of a cake and a venus model.

faces along their automatically extracted edges. On these faces, feature descriptors are computed. In every point of a face, mean curvature and difference of principal curvatures are estimated from integral invariants on local neighborhoods. The curvature estimates are aggregated into multi-scale surface roughness and sharpness descriptors. Subsequently and based on this information, faces are matched pairwise in a statistical robust way. The quality of pairwise matchings initializes a greedy multi-piece matching phase. As output, adjacency relation a_{jk} is obtained that serves as input to the final constrained simultaneous registration. Reassembled broken objects are shown in Figures 2.1, 2.3, 2.5 and 2.6.

2.5.2 Reassembling Ancient Monuments

Above method for the reassembly of fractured objects relies on the fact that corresponding fracture faces carry enough geometric information to let matching and registration succeed in a stable way. This is the case for a wide range of materials, recently broken objects and well preserved ancient excavations. However, a significant percentage of findings has been subject to erosion jeopardizing identification of matching faces as well as an accurate registration. Including information *exterior* to corresponding faces improves the results, enlarging and stabilizing the matching domain. In the following, we derive shortly how a more global view improves the constrained registration algorithm in the presence of strong erosion. The results are an excerpt from (Thuswaldner et al., 2009), where digital data from the Octagon building in Ephesus is used as illustrative example.

Let A and B be two matching faces. We write $N(A) = \{A_i : i = 1, \ldots, n_A\}$ for faces on the same fragment as A that are adjacent to A, and define $N(B) = \{B_i : i = 1, \ldots, n_B\}$ in a similar fashion. Then, faces in $N(A)$ will be in *exterior correspondence* with faces in $N(B)$. For the Octagon data set, more or less all fragments are bounded by planar

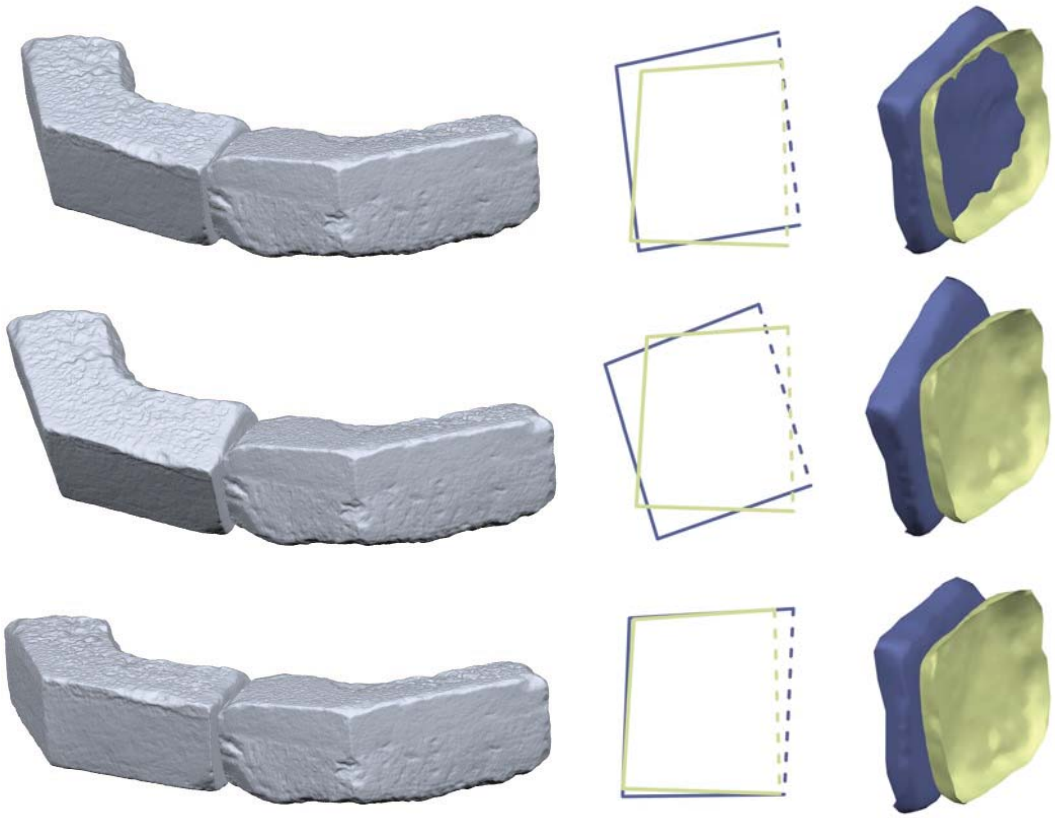

Figure 2.7: Alignment of two building blocks of the Octagon data set (Ephesus). From top to bottom we see an unconstrained alignment, a penetration free alignment and a penetration free alignment with additional enforced coplanarity of exterior corresponding faces. The center column depicts cross sections of the building blocks, the right column illustrates close-ups of the contact areas.

Figure 2.8: Penetration free and coplanar reassembly of six building blocks of the Octagon data set.

faces. As a consequence, exterior correspondence of two faces $\bar{A} \in N(A)$ and $\bar{B} \in N(B)$ will constrain the reassembly process to align $\bar{A}$ and $\bar{B}$ such that they are coplanar. We achieve this by adding a weighted penalty term for any pair $(\bar{A}, \bar{B})$,

$$f_{\text{coplanar}}(\bar{\mathbf{c}}, \mathbf{c}) = \sum_{\mathbf{b} \in \bar{B}} \left[\mathbf{n}_A^T \cdot (\mathbf{b} + \mathbf{v}(\mathbf{b})) + d_A \right]^2$$

to the objective of the constrained registration algorithm. Here, $\mathbf{n}_A^T \cdot \mathbf{x} + d_A = 0$ denotes the signed distance to the fitting plane of locally fixed face $\bar{A}$. If the fragments' faces were exactly planar, it would be more appropriate to state coplanarity as linear equality constraint. However, inevitable noise in our application would yield an infeasible set of constraints. Figures 2.7 and 2.8 illustrate aligned building blocks of the Octagon data set.

3 Registration in the l_1-norm

In the previous chapter we have extended the widely popular solutions to the unconstrained local registration problem by (Besl and McKay, 1992) and (Chen and Medioni, 1992). Their approaches center around approximations of the squared distance function between the shapes. This makes these algorithms two of many in geometry processing minimizing the l_2-norm $\|\mathbf{r}\|_2 := \sqrt{\sum_i r_i^2}$ of some residue $\mathbf{r} \in \mathbb{R}^p$. The wide spread of least-squares approaches is probably due to the simplicity and effectiveness in the linear case, as formalized by the Gauss-Markov theorem (Björck, 1996): Given that the input data has zero mean, same variance and no correlation, the least-squares approach yields the best (by means of minimal variance) linear unbiased estimator.

Unfortunately, the input data does not always meet these requirements and severely challenges the least-squares approach. One particular problem are outliers, which naturally occur in various ways in physical measurement processes used to acquire the input data. Several techniques have been proposed to improve the robustness of l_2-norm optimizations and applications of these methods have found their way into geometry processing.

We choose a different approach and go beyond least-squares. Instead of turning to robust variants of l_2-approximations we choose a norm known to be more robust by itself, the l_1-norm $\|\mathbf{r}\|_1 = \sum_i |r_i|$ (cf. Fig. 3.1). l_1-techniques are far less popular in CAGD which may be due to the challenges of non-smooth optimization resulting from the absolute values in the definition of $\|.\|_1$. Geometric insights allow us to explore the l_1-approach in an elegant way. In many cases, the least-squares techniques are equivalent to a minimization of the squared distances in the setup. In the l_1-norm, this results in working with (the absolute value of) the signed distance function. In the introduction, we have already outlined how to derive approximation algorithms from this premise. Please note that distances will still be taken in the Euclidean norm, but instead of squaring the distance values (l_2-norm) we employ the l_1-norm.

3.1 Related Work

For previous work on registration, we want to refer to the surveys in Chapters 2 and 4. Efforts to increase the robustness of least-squares methods are well summarized in (Pighin and Lewis, 2007). Weighted least-squares basically break with the equal variance assumption of the Gauss-Markov theorem. Intuitively, measurements with large errors are considered outliers and weighted weaker. Iteratively Reweighted Least Squares (Holland and Welsch, 1977) for example suggest to weight the summands of the current error term with reciprocal powers of a previous iteration's residuals. (Sharf et al., 2008) rely partially on this method in a surface reconstruction framework based on a space-time model. Apart from weighting the residuals, least-squares have been combined

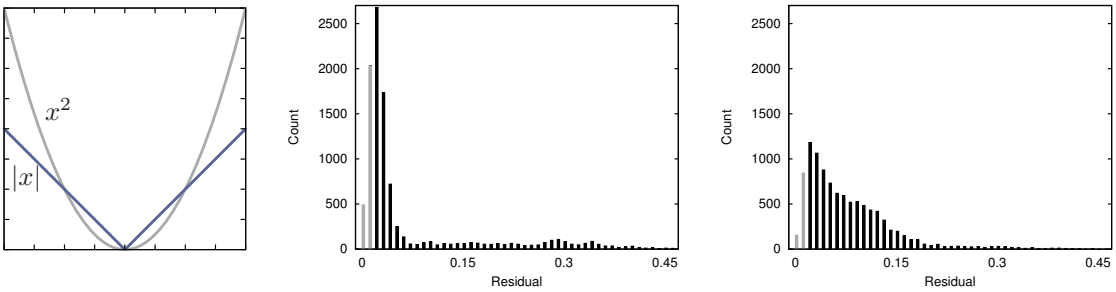

Figure 3.1: The l_1-norm weights outliers much less than the l_2-norm. Moreover, it puts more emphasis on small residues (cf. function graph of $f(x) = |x|$ and $f(x) = x^2$, left). Both properties can be seen in typical histograms of residues for l_1- (center) and l_2-approximations (right). These plots were computed on discrete input data. The histograms' two left most bins (marked in gray) are below the average sampling density and bias the analysis. We will discuss the residual distribution in more detail in Sec. 3.4.

with RANSAC (Fischler and Bolles, 1981). This so-called *least median squares* method (Rousseeuw and Leroy, 1987) has been used among others for multi-view registration of range scan images (Masuda and Yokoya, 1995).

The l_1-norm appears early in geometric optimization problems. (Weber, 1909) states the problem of finding the optimal location of a new industrial site with minimal sum of distances to a set of existing sites. The problem was soon traced back to Fermat in the 17^{th} century and is known as the *Fermat-Weber* problem since then. (Weiszfeld, 1937) was first to propose an iterative approach to solve the problem that was thoroughly investigated by (Kuhn, 1973). The solution of the Fermat-Weber problem is usually called the l_1-*median* or *geometric median*. Recently, (Lipman et al., 2007) presented a projection operator for surface reconstruction that resembles locally the l_1-median. In this context, the terminology of *signed* or *unsigned distance function* is frequently used. If a point set's normal field is required to have unique orientation, the distance function is called signed. Otherwise, it is denoted as unsigned. See the discussion in (Alliez et al., 2007) for pointers to recent literature.

The l_1-norm is used infrequently to solve matching problems. (Zhu et al., 2004) formulate an l_1-minimization of the signed distance function for profile error evaluation but do not seem to further examine this option. A prominent exception from rare l_1-norm appearances are contributions in image processing. For translational image registration, (Barnea and Silverman, 1972) discretize the solution space and use the l_1-norm of the difference images as similarity measure. Consequently, this approach is well known as *sum of absolute differences* technique. It is still widely used nowadays in modern video compression software (Richardson, 2003). (Rudin et al., 1992) propose a variational formulation to the problem of image noise removal that employs the l_1-norm of the minimizer's derivatives as regularization term. (Chan and Esedoglu, 2004; Zach et al., 2007) discuss extensions of this approach to image reconstruction measuring the

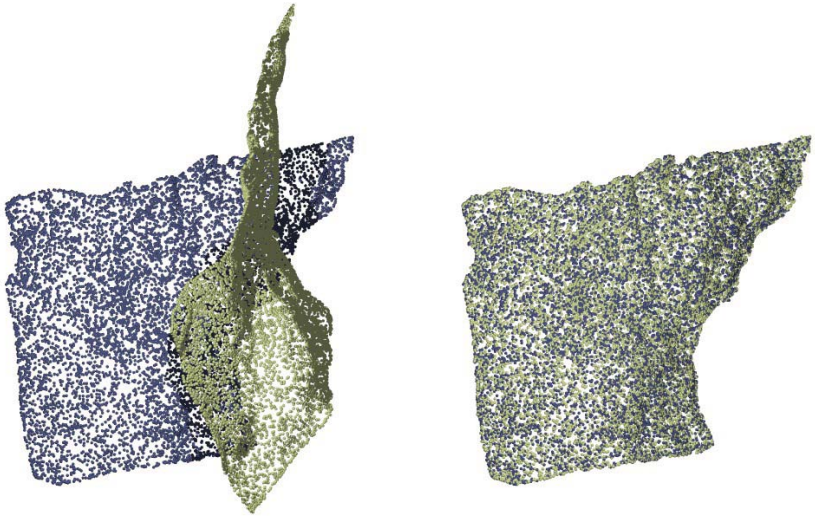

Figure 3.2: Local, rigid registration of two point clouds: starting from the initial setup (left), the final alignment (right) is obtained as solution of a non-smooth minimization problem.

fidelity (approximation) error as l_1-norm of residuals to observation data. References to optimization literature covering non-smooth minimization (among others, sums of absolute values) are given in Sec. 3.3.2.

The remaining parts of this chapter are organized as follows. First, we derive new error terms for l_1-based local registration. For this, we will greatly benefit from the results of the previous chapters. Subsequently, we will put some effort in showing ways to solve the emerging non-smooth optimization problems. The presentation of a handful of examples along with a discussion of the results will conclude this chapter.

3.2 Error Terms for Registration in the l_1-norm

Let us continue in the terminology and notation of Sec. 2.2. Given a target point cloud A and moving point cloud B, our goal is to determine that rigid body motion $\mathbf{m}(\mathbf{b})$ minimizing the unsigned distance function from A to $\mathbf{m}(B)$, $|d(A, \mathbf{m}(B))|$. We solve this non-linear and non-smooth optimization problem within the general framework of Chapter 1. Approximating unknown $\mathbf{m}(\mathbf{b})$ by its linear velocity vector field $\mathbf{v}(\mathbf{b}) = \bar{\mathbf{c}} + \mathbf{c} \times \mathbf{b}$, we can write $T(\mathbf{b}_i) \approx \mathbf{b}_i + \mathbf{v}(\mathbf{b}_i)$ for the shape manipulator of $\mathbf{b}_i \in B$. We denote the closest point of $\mathbf{b}_i$ in A by $\mathbf{f}_i$ and arrive at the point to point error term for l_1-registration,

$$|d(A, T(\mathbf{b}_i))| \approx \|\mathbf{b}_i + \mathbf{v}(\mathbf{b}_i) - \mathbf{f}_i\|.$$

Let $\mathbf{n}_i$ describe a normal of A in $\mathbf{f}_i$. Linearizing the signed distance function and taking the absolute value yields the corresponding point to tangent plane error term,

$$|d(A, T(\mathbf{b}_i))| \approx \left|\mathbf{n}_i^T \cdot (\mathbf{b}_i + \mathbf{v}(\mathbf{b}_i) - \mathbf{f}_i)\right|.$$

Summing up the individual error terms reveals the objectives per iteration of the registration problem in the l_1-norm,

$$\min_{(\bar{\mathbf{c}},\mathbf{c})} \sum_{i=1}^{n_B} \|\mathbf{b}_i + \mathbf{v}(\mathbf{b}_i) - \mathbf{f}_i\| \qquad \text{and} \qquad \min_{(\bar{\mathbf{c}},\mathbf{c})} \sum_{i=1}^{n_B} \left|\mathbf{n}_i^T \cdot (\mathbf{b}_i + \mathbf{v}(\mathbf{b}_i) - \mathbf{f}_i)\right|.$$

Let us have a closer look at the two objectives for a moment. The point to point objective is only C^0 in the zeros of its summands. A term $\mathbf{b}_i + \mathbf{v}(\mathbf{b}_i) - \mathbf{f}_i$ will vanish if the moving point coincides with its corresponding target point. This is certainly a situation we want to achieve and will arise naturally in zero residual problems (e.g. when the moving point cloud is a congruent copy of the target point cloud). For this reason, there is no other way than to consider a non-smooth optimization problem in the first place to minimize above objective. Apart from being not differentiable everywhere, the objective is convex and the arguments of the norm are linear in $\bar{\mathbf{c}}$ and $\mathbf{c}$. Similar facts hold for the point to tangent plane objective. The absolute value function is not differentiable in the zero of its argument, which itself is linear in $\bar{\mathbf{c}}$ and $\mathbf{c}$. The error term however is convex.

3.3 Optimization

Above, we have derived two error terms for local registration based on the unsigned distance function between the input shapes. In this section, we consider the solution of the resulting non-smooth but convex optimization problems. First, we briefly sketch a class of general non-smooth solvers, the so-called *Proximal Bundle* methods. Second, we show that both objectives can be turned into smooth, though constrained minimization problems.

To ease the following discussion, we change to a different notation for the objective functions and leave the previous meanings of variables behind. For $\mathbf{x} \in \mathbb{R}^n$ as unknown, the point to point distance objective shall read $\sum_i \|A_i\mathbf{x} + \mathbf{b}_i\|$. Consequently, the point to tangent objective will be considered as $\sum_i |\mathbf{b}_i^T\mathbf{x} + c_i|$. As we have already stated in the previous section, both functionals are convex.

3.3.1 Proximal Bundle Method

In smooth optimization theory, the objective function is differentiable at least once and information about the first or higher-order derivatives enters the optimization process. Given a non-smooth optimization problem, the objective function is not differentiable everywhere and an adapted idea of gradients is introduced. Non-smooth optimization algorithms such as the *Proximal Bundle* method (Kiwiel, 1990) build upon this concept.

Let f denote the convex non-smooth objective function of a minimization problem $\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x})$. We call $\mathbf{s} \in \mathbb{R}^n$ a *sub-gradient* of f in $\mathbf{x}$, if

$$f(\mathbf{z}) \geq f(\mathbf{x}) + \mathbf{s}^T(\mathbf{z} - \mathbf{x}), \quad \forall \mathbf{z} \in \mathbb{R}^n.$$

The set of sub-gradients in a point $\mathbf{x}$ is called the *sub-differential* $\partial f(\mathbf{x})$ of f in $\mathbf{x}$. In any point $\mathbf{x}$ where f is differentiable, $\partial f(\mathbf{x})$ comprises only a single vector, namely the

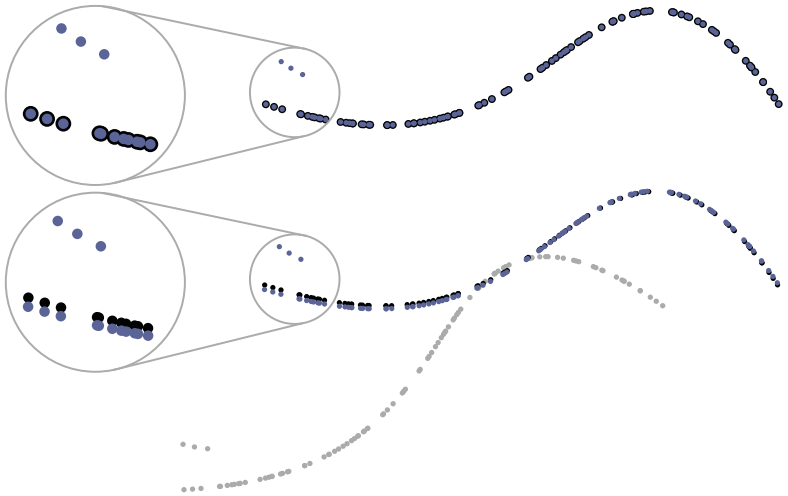

Figure 3.3: The moving point cloud (blue) includes three significant outliers. The l_1-registration (top) is more robust than the l_2-registration (bottom). The initial position of the moving point cloud is shown in gray at the bottom.

gradient $\nabla f(\mathbf{x})$ of f. It is common to derive a general theory of convex derivatives based on $\partial f(\mathbf{x})$ (cf. Rockafellar, 1972).

Example. *From above definition, we immediately obtain the sub-differential of the point to point error term,*

$$\partial \|A\mathbf{x}+\mathbf{b}\| = \begin{cases} \{\mathbf{s}\in\mathbb{R}^n : \|A\mathbf{z}+\mathbf{b}\| \geq \mathbf{s}^T A^{-1}(A\mathbf{z}+\mathbf{b}),\ \forall \mathbf{z}\in\mathbb{R}^n\} & \text{for} \quad A\mathbf{x}+\mathbf{b}=\mathbf{0} \\ A^T(A\mathbf{x}+\mathbf{b})/\|A\mathbf{x}+\mathbf{b}\| & \text{for} \quad A\mathbf{x}+\mathbf{b}\neq\mathbf{0} \end{cases}$$

The sub-differential of the point to tangent plane error term reads

$$\partial\,|\mathbf{b}^T\mathbf{x}+c| = \begin{cases} \alpha\mathbf{b}+(1-\alpha)\mathbf{b} & \text{for} \quad \mathbf{b}^T\mathbf{x}+c=0, \quad \alpha>0 \\ -\mathbf{b} & \text{for} \quad \mathbf{b}^T\mathbf{x}+c<0 \\ \mathbf{b} & \text{for} \quad \mathbf{b}^T\mathbf{x}+c>0 \end{cases}$$

In a point $\mathbf{x}$, each sub-gradient $\mathbf{s}$ supports a hyperplane $h = \{\mathbf{z}\in\mathbb{R}^n : \ f(\mathbf{x}) + \mathbf{s}^T(\mathbf{z}-\mathbf{x})\}$, a linear and lower bound approximation of f in $\mathbf{x}$. Proximal Bundle methods approximate the non-smooth objective by combining several of these linearizations. Assuming that $\mathbf{x}^0$ is an arbitrary starting point, typical iterations for these algorithms are of the form,

$$\mathbf{x}^{k+1} = \operatorname{argmin}\ \hat{f}(\mathbf{x}) + \frac{1}{2t_k}\|\mathbf{x}-\mathbf{x}^k\|^2. \tag{3.1}$$

Here, $\hat{f}$ denotes a linear approximation of f (the *cutting plane model*) that aggregates knowledge from a bundle of previous sub-gradients and corresponding hyperplanes, respectively. The second term controls the maximal allowed step width (proximity) in an iteration, as the minimum of $\hat{f}$ may be unbounded. The approximate objective function

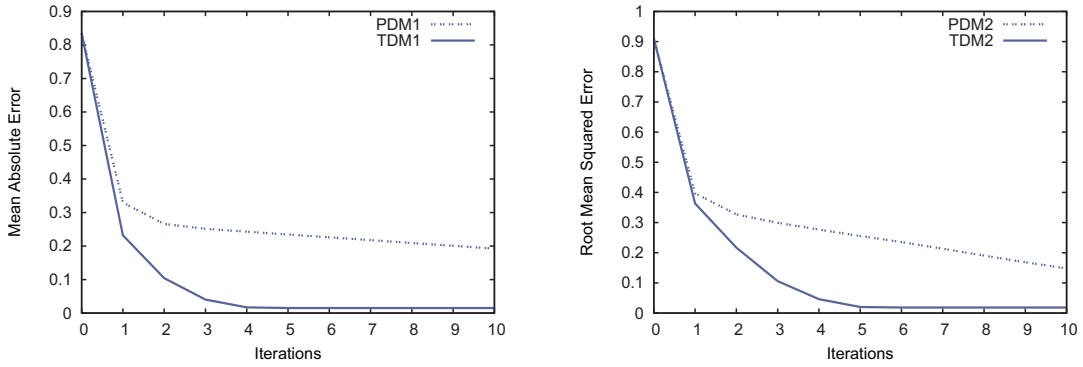

Figure 3.4: For registration in the l_1-norm (left) and the l_2-norm (right), the point to point error term performs worse than the point to tangent plane approach.

of Equ. (3.1) yields a quadratic optimization problem with linear constraints that (or its dual) can be solved with methods of smooth optimization theory. In Chapter 2 we have already encountered this type of minimization problem and we want to refer one more time to (Nocedal and Wright, 1999) for solutions.

The challenges in realizing Proximal Bundle methods are found to be rules to maintain and update the bundle of sub-gradients defining $\hat{f}$ and the choice of the proximity parameter t_k. It can be shown that these methods give an ε-optimal global minimizer $\mathbf{x}^*$ for convex f, that is $f(\mathbf{x}^*) \leq f(\mathbf{z}) + \varepsilon, \forall \mathbf{z} \in \mathbb{R}^n$ for user-defined ε (Kiwiel, 1990). Proximal Bundle methods are general approximate solvers for non-smooth optimization problems. The next section will show that by exploiting certain properties of the objectives, above minimization problems can be solved in an exact way.

3.3.2 Non-Smooth vs. Smooth Optimization

Instead of applying a general but approximate solver such as the Proximal Bundle method, our goal will be to turn the non-smooth problems into smooth optimization problems. This will happen at cost of increasing the dimension and constraining the solution space.

Let us consider the tangent distance minimization term first. In order to avoid minimization of the absolute value of $\mathbf{b}_i\mathbf{x} + c_i$, we require the latter term to evaluate in an interval with variable bounds $[-y_i, y_i]$. If we minimize y_i along with $\mathbf{x}$ now, we imitate the effect of the absolute value function. Applying this idea to the point to tangent plane objective, we obtain

$$\min_{\mathbf{x}} \sum_{i=1}^{m} |\mathbf{b}_i^T \cdot \mathbf{x} + c_i| \quad \Longleftrightarrow \quad \min_{\mathbf{z}=(\mathbf{x},\mathbf{y})} \sum_{i=1}^{m} y_i$$
$$\text{subject to} \quad -y_i \leq \mathbf{b}_i^T \cdot \mathbf{x} + c_i \leq y_i, \qquad i = 1, \ldots, m$$

where $\mathbf{y} = (y_1, \ldots, y_m) \in \mathbb{R}^m$ are m auxiliary variables (Boyd and Vandenberghe, 2004). It is obvious that we face an increase of dimension from 6 to $6 + m$ for the optimization

problem. What we get in return is a *linear program*, that can be solved with a variety of algorithms, e.g. the well-known Simplex method or Interior point methods (Nocedal and Wright, 1999).

The $2 \cdot m$ constraints are part of the price we pay for obtaining a smooth optimization problem. This effect is somewhat reduced if we combine the point to tangent plane registration term in above formulation with the penetration free constraints of the previous chapter. We obtain a constrained optimization problem with slightly simpler constraints,

$$\min_{\mathbf{x}} \sum_{i=1}^{m} |\mathbf{b}_i^T \cdot \mathbf{x} + c_i| \quad \Longleftrightarrow \quad \min_{\mathbf{z}=(\mathbf{x},\mathbf{y})} \sum_{i=1}^{m} y_i$$
$$\text{subject to} \quad 0 \leq \mathbf{b}_i^T \cdot \mathbf{x} + c_i \leq y_i, \qquad i = 1, \ldots, m$$

It will be interesting to evaluate experimentally in Sec. 3.4 whether the one-sided registration problem can be solved more efficiently than its unconstrained counterpart.

It is obvious to apply above conversion to the point to point objective as well. The sum of norms of vector-valued linear functions is turned into,

$$\min_{\mathbf{x}} \sum_{i=1}^{m} \|A_i \cdot \mathbf{x} + \mathbf{b}_i\| \quad \Longleftrightarrow \quad \min_{\mathbf{z}=(\mathbf{x},\mathbf{y})} \sum_{i=1}^{m} y_i \tag{3.2}$$
$$\text{subject to} \quad \|A_i \cdot \mathbf{x} + \mathbf{b}_i\| \leq y_i, \qquad i = 1, \ldots, m.$$

Please note, that we omit the lower bound constraints as norms are not negative. We see, that we still have a linear objective function. However, the constraints are not linear but quadratic and the theory of linear programs does not apply.

Indeed, this optimization problem belongs to the class of *Second-order cone programming* (short *SOCP*) problems (Nesterov and Nemirovskii, 1994; Alizadeh and Goldfarb, 2003). In their most general form, cone programs minimize a linear function under second-order constraints, an intersection of an affine set with a Cartesian product of cone constraints. For our needs, the unit second-order cone,

$$C_0 = \{(\mathbf{x}, t) : \mathbf{x} \in \mathbb{R}^p, t \in \mathbb{R}, \|\mathbf{x}\| \leq t\},$$

is well suitable. C_0 plays an important role in wide fields of second-order cone programming. By applying an affine mapping to C_0 we obtain the general SOCP problem

$$\min_{\mathbf{z}} \mathbf{g}^T \cdot \mathbf{z}$$
$$\text{subject to } \|A_i \cdot \mathbf{z} + \mathbf{b}_i\| \leq \mathbf{c}_i^T \cdot \mathbf{z} + \mathbf{d}_i, \qquad i = 1, \ldots, m.$$

SOCP programs are generalizations of linear and quadratic programs (Lobo et al., 1998). With $\mathbb{R}_+^p$ as cone, we can express any standard linear program as cone programming problem. Moreover, by rewriting the objective of a quadratic program as quadratic constraints in a similar fashion to Equ. (3.2), quadratic programs are seen to be special cases of SOCP as well.

Returning to the point to point objective, Equ. (3.2) matches the structure of a SOCP problem without any further modifications. For solving SOCP problems, Interior point

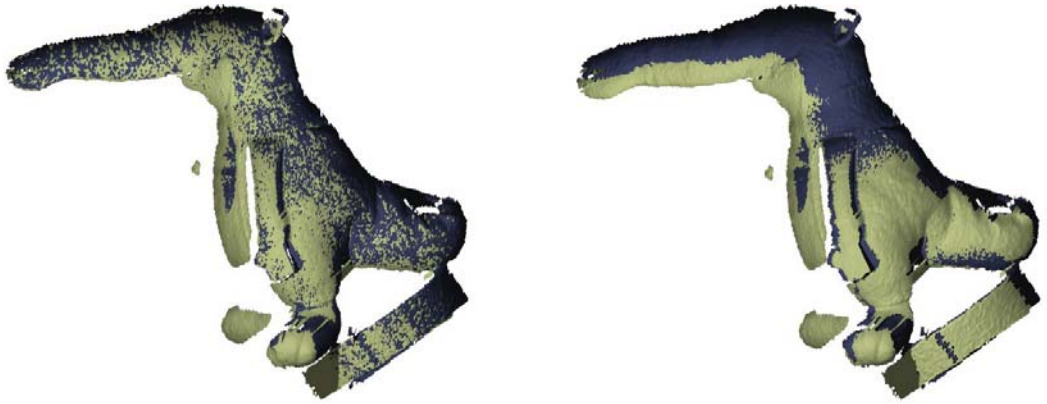

Figure 3.5: The l_1-registration aligns the two input point clouds well (left). In contrast, the least-squares registration result is tilted (right). Both target and moving point cloud are noisy and include numerous outliers.

methods are widely employed. A detailed description of these algorithms is out of scope of this work and we remain with stressing once again that it is straight forward to include further linear constraints to SOCP problems, as we will do to achieve penetration free alignments.

Above conversion from an unconstrained non-smooth to a constrained smooth optimization problem may be adapted to solve the l_∞-minimization problem,

$$\min_{\mathbf{x}} \max_{i=1,\dots,m} |\mathbf{b}_i^T \cdot \mathbf{x} + c_i| \quad \Longleftrightarrow \quad \min_{\mathbf{z}=(\mathbf{x},y)} y$$
$$\text{subject to} \quad -y \leq \mathbf{b}_i^T \cdot \mathbf{x} + c_i \leq y, \qquad i = 1, \dots, m.$$

Min-max problems are of a certain interest to industrial quality management as industrial standards typically norm the maximal errors of a system. (Zhu et al., 2004) investigate min-max registration problems in the context of industrial inspection and we do not further consider l_∞-registration here.

3.4 Results

Let us begin by comparing the two error terms for unsigned distance registration to their l_2-norm counterparts. We refrain from presenting the whole two times two matrix of results but give a representative final alignment for an unsigned point to tangent plane optimization (cf. Fig. 3.2). As expected, the convergence plots of Fig. 3.4 indicate that the point to point minimizations perform worse than the tangential terms. Regarding the computational cost, unconstrained l_2-registration requires a 6×6 system of linear equations to be solved. Hence, we regard the computational effort negligible. This does not hold for the non-smooth objectives which require some effort to be solved with a Proximal Bundle method.

In Fig. 3.3, we compare the performance of a planar l_1-registration to that of a least-squares alignment. The moving point cloud comprises three significant outliers. Apart

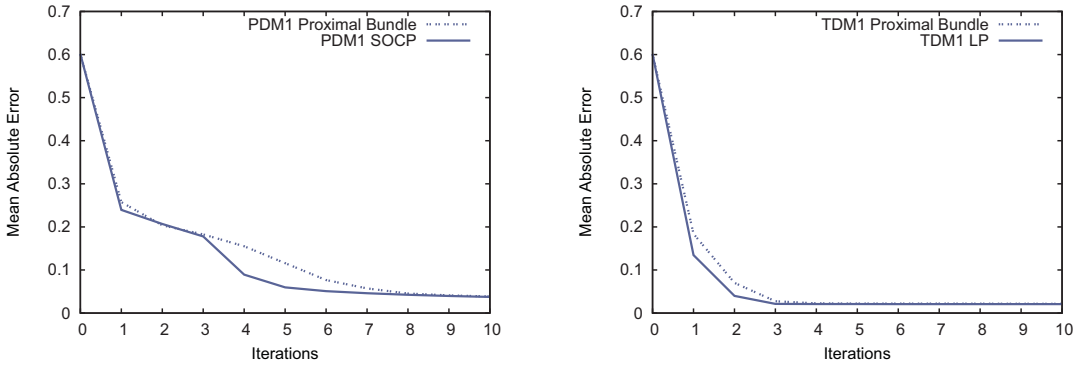

Figure 3.6: An iteration's point to point (left) or point to tangent plane (right) minimization may be turned into a smooth though constrained optimization problem, that can be solved exactly. An exact solution yields better convergence at global scope but is computationally more expensive.

from those, the problem would be a zero residual problem. In Fig. 3.5, we show another robust non-smooth 3D registration and a least-squares alignment. This data set was obtained with a stereo and active illumination based 3D scanner, capable of acquiring 17 frames per second (Weise et al., 2007), see also the next chapter. The high frame rate reduces the quality of the coordinate samples and outliers are very common. We triangulated the data for better visualization. Both tests confirm the robustness of the l_1-methods.

The robustness of l_1-optimization is due to a lesser weighting of samples with large residues. This is best seen in Fig. 3.1 (right) in a comparison of the absolute value function $f(x) = c_0|x|$ and a parabola $f(x) = c_1x^2$. From the graphs we can immediately deduce another property. The l_1-norm considers small residues stronger than the l_2-norm. These two properties are confirmed by histograms of a l_1- and a l_2-registration in Fig. 3.1 (center and right). The discrete nature of the aligned shapes renders the histogram unreliable for residues smaller than half the average sampling density. For this reason, the corresponding bins are grayed out.

In Sec. 3.3.2 we have seen that both the point to point and the point to tangent plane objectives can be converted to constrained smooth minimization problems. Opposed to the approximate general solver for non-smooth systems, the constrained smooth programs of an iteration may be minimized exactly (and thus are expected to converge faster at global scope). Fig. 3.6 confirms our expectations. The improved convergence comes at the price of longer per iteration running times. While the increase in computational cost for solving the linear program is moderate (by a factor of approximately 1.5), the point to point error term's second-order cone program is expensive to minimize.

General registration yields final alignments with target and moving point cloud penetrating each other. This is to be expected as the distance between the two shapes is minimized. For several applications however, it is of interest to achieve penetration free alignments (see Chapter 2). We have seen that the corresponding linear constraints may

Figure 3.7: Unconstrained registration (left) yields a final alignment with mutual penetration of target and moving point cloud. The inclusion of linear constraints — eventually simplifying the smooth variant of the point to tangent plane l_1-registration — achieves a penetration free alignment (right).

be added to any optimization solver discussed above. In the case of the point to tangent plane error term, integration of these constraints even simplifies the optimization problem. Typically, a penetration free registration takes half the time of an unconstrained registration in the l_1-norm. Please note that the corresponding least-squares registration is an optimization problem with quadratic objective and linear constraints that requires some computational effort as well. In Fig. 3.7, we show results of an unconstrained and a penetration free alignment. Again, the point clouds have been converted to triangular meshes after registration for better visualization.

4 Dynamic Geometry Registration

Under certain aspects, the registration of Chapters 2 and 3 may be called *static*. The input data to these algorithms is usually acquired in a process that takes several seconds per view point. During acquisition, both the captured object and the entire measurement setup are required not to move. Though the resulting data exhibits some noise, just as the output of any physical measurement process, it is of considerably high quality.

In contrast, numerous real-time shape acquisition techniques have emerged in recent years. Methods such as active space-time stereo (Zhang et al., 2003; Davis et al., 2005), motion compensated structured light (Fong and Buron, 2005; König and Gumhold, 2008) or combinations of both (Weise et al., 2007) are capable of capturing 3D data at speeds of up to video frame rates. Typically, the object is moved in the scanning device's field of view in order to expose all of its surface to the scanner. In the meanwhile, the scanner takes up to 20 measurements per second. The output data comprises hundreds and thousands of measured coordinate sets; every set uniquely tagged by its frame number. The short exposure time lets even moderate deforming objects appear rigid for the duration of a single frame, with all the motion concentrating between subsequent stills.

A reconstruction of an object scanned in real-time faces the same challenges as the registration of the previous chapters. The point clouds are unstructured and no inter-frame correspondences of data points are available. Any frame covers only those parts of the object that were visible at measurement time. Moreover, a frame's 3D data is stored in the scanner's local coordinate system. However, there is a fundamental difference between both types of input data. While the number of frames in a static acquisition process is typically small with the motion between the frames being large, the real-time 3D data features many frames with small inter-frame motion. For this reason, we refrain from using classic multi-view registration as it scales badly with increasing number of input systems. Instead, our goal is to derive a registration algorithm exploiting the dense spatial and temporal coherence in the input data.

The close temporal coherence of the samples motivates an approach investigating instantaneous kinematic properties of the input data. To give a short in-depth preview of the contents of this chapter, consider a planar curve $\mathbf{c}_0$ and its images $\mathbf{c}_t$ under a smooth one-parameter Euclidean motion $\mathbf{m}(t)$. Embedding $\mathbf{c}_t$ in $\mathbb{R}^3$ with time as third coordinate axis, we obtain a smooth space-time surface S such as that in Fig. 4.1. Any plane $t = \text{const}$ holds an instance of the rigidly transformed initial curve $\mathbf{c}_0$. We consider the input point data P^i to be discrete measurements of such planar intersections of S. In the following we will show, how inter-frame motions can be reconstructed from kinematic properties of space-time surface S. To emphasize the difference to classic static registration approaches, we call our method *dynamic*.

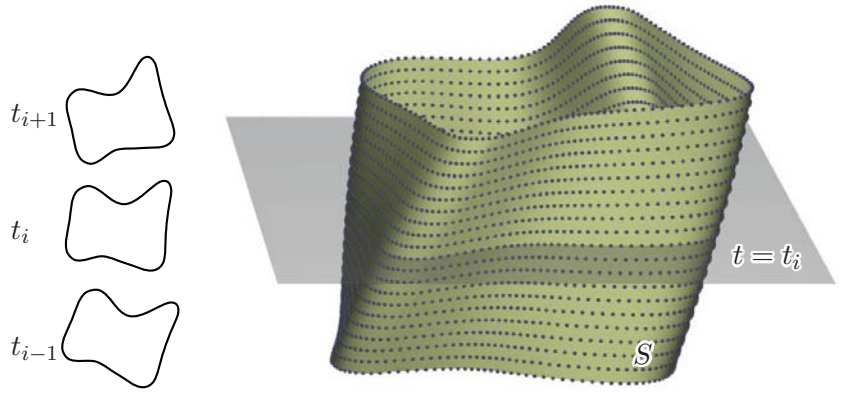

Figure 4.1: The d-dimensional input point cloud data is converted to samples of a smooth space-time surface S in $\mathbb{R}^{d+1}$. The additional coordinate axis t is given by acquisition time t_i.

4.1 Related Work

Registration literature hasn't paid any special attention to dense temporal input data until 2007, when (Wand et al., 2007) and the work presented here (Mitra et al., 2007) have been published. Up to that point of time, static registration algorithms were widely applied to reconstruct the scanned object. For a survey of this classic registration literature we want to refer to the corresponding sections of the previous two chapters.

While the geometry processing community turned its attention to the estimation of inter-frame motions in dense temporal input data only recently, computer vision has considered what is known as object tracking for some time. Techniques such as optical flow (Beauchemin and Barron, 1995) aim at identifying an object over subsequent frames of a video. For this purpose, optical flow methods reconstruct the two-dimensional image motion from which the three-dimensional motions may be derived. Variants of this method have even been translated to the 3D registration context (Rusinkiewicz et al., 2002).

Parallel to this work, (Wand et al., 2007) proposed another method for the reconstruction of temporal coherent data. Based on a Bayesian statistical model, the authors establish correspondences between the data points of adjacent time frames. This is in contrast to our work which does not estimate correspondences in any way. (Wand et al., 2007) formulate various priors for shape reconstruction, noise removal and as-rigid-as-possible inter-frame motions that are maximized in interleaved smooth and discrete optimization steps. The employed surfel based representation turns out to be computational expensive and imposes certain limitations on the usability of the method for data set comprising more than 30 frames (as reported in a follow-up publication by the authors, (Wand et al., 2009)).

(Süßmuth et al., 2008) build upon the approaches of (Wand et al., 2007) and (Mitra et al., 2007) and start from a similar four-dimensional space-time setup derived from the

Figure 4.2: A reconstruction of the Stanford bunny model from more than 300 simulated noisy scans. The original model is included as ground truth (blue) in the center figure. The point clouds have been triangulated for better visualization.

input data. In a next step, an implicit surface is fitted to the four-dimensional point cloud. From its zero level set an initial guess of the original shape is computed that is refined by moving this template over the remaining time instances in an as-rigid-as-possible manner. (Sharf et al., 2008) abandon the rigidity assumption for inter-frame motions in favor of a volume preserving objective. Their approach centers around a mass flow model, defined on the input space-time point cloud projected onto a four-dimensional grid.

(Wand et al., 2009) extend their 2007 work and head for better scalability. Instead of estimating the deformation per surfel and per time slice, they introduce a coarse set of deformation basis functions on the shape. This basis set is defined in a topology aware manner and is subject to optimization favoring rigidity, volume preservation and temporal smoothness. Techniques for performance and garment capture such as (de Aguiar et al., 2008; Bradley et al., 2008) consider the reconstruction of 3D models from multi-view video data in a marker-less way and relate both to object tracking and the registration of point cloud data acquired at high frame rates.

In the following, we present a registration framework based on estimating kinematic properties of a space-time surface. We discuss various theoretical and numerical issues of our optimization approach and conclude with several examples illustrating the effectiveness of the proposed algorithm.

4.2 Motion Parameter Computation in $\mathbb{R}^{3+1}$

In this section we are going to transform the input data into a higher-dimensional space-time framework. With the help of this model, we reconstruct the scanned object.

Assume an object is being moved according to some smooth one-parameter Euclidean motion $\mathbf{p}(t) = R(t) \cdot \mathbf{p}^0 + r(t)$. At several points of time t_i, a scanning device acquires coordinate samples P^i of the currently visible parts of the object. This data is in the

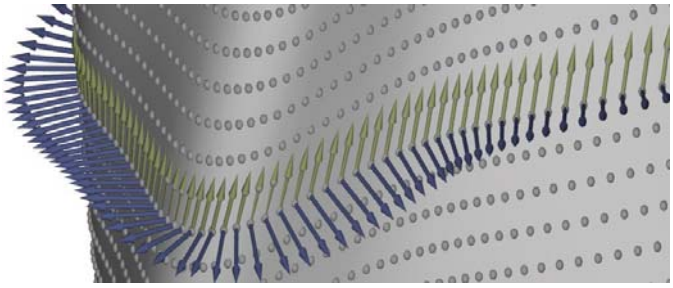

Figure 4.3: The velocity vector (yellow) of a sample on space-time surface S is tangential to the sample's trajectory. Hence, the velocity vector is tangential to S and perpendicular to the surface normal (blue). The registration computes a time frame's unknown velocity vector field in a least-squares setup, minimizing the velocity vectors' deviation from the tangent planes.

scanner's local coordinate system and bears no correspondences between any frames P^i and P^j. We combine the coordinates of a time frame's elements with scanning time t_i and obtain a four-dimensional space-time point cloud $Q^i = \{(\mathbf{p}^i, t_i) : \mathbf{p}^i \in P^i\}$. We summarize all transformed input data in a single point cloud $\mathcal{Q} = \cup_{i=0,\ldots,N} Q^i$. The elements of $\mathcal{Q}$ are the probably noisy samples of an unknown smooth three-dimensional space-time surface S. We can think of S as being generated by a smooth one-parameter Euclidean motion of a profile shape S^0. A trajectory $\mathbf{q}(t) = (R(t) \cdot \mathbf{p}^0 + r(t), t)$ of a point $\mathbf{p}^0 \in S^0$ lies entirely in S. Consequently, a tangent to $\mathbf{q}(t)$ is tangent to S for any t. In other words, velocity vector and the space-time surface normal in a point $\mathbf{q} = (\mathbf{p}, t) \in S$ are perpendicular.

In the space-time model, the velocity vectors of $\mathbf{q}(t)$ are of special form,

$$\mathbf{v}(\mathbf{q}) = (\bar{\mathbf{c}} + \mathbf{c} \times \mathbf{p}, 1).$$

Given that the scanner operated at constant frame rate, the scaling factor of time is completely arbitrary. We will further discuss our choice of unit time scale in Sec. 4.3.1. For the reconstruction of the velocity vector field $\mathbf{v}(\mathbf{q}) = (\bar{\mathbf{c}}_i + \mathbf{c}_i \times \mathbf{p}, 1)$ of time slice Q^i, we minimize the velocity vectors' deviation from the surface's tangent planes in points $\mathbf{q}^i \in Q^i$,

$$\min_{(\bar{\mathbf{c}}_i, \mathbf{c}_i)} \sum_{\mathbf{q}^i \in Q^i} \left[\mathbf{v}(\mathbf{q}^i)^T \cdot \mathbf{n}^i\right]^2. \tag{4.1}$$

Here, $\mathbf{n}^i$ denotes a surface normal of S in $\mathbf{q}_i$, that we estimate from $\mathcal{Q}$ (cf. Sec. 4.3.2). If the motion between two time slices P^i and P^{i+1} is sufficiently small, it is approximated well by the spatial component of the velocity vector field of Q^i. We reconstruct a rigid body transformation from $(\bar{\mathbf{c}}_i, \mathbf{c}_i)$ with one of the methods in Sec. 2.2. Collecting all the data in a single time slice by applying the inter-frame motions yields a reconstruction of the scanned object.

Above optimization constrains a time slice locally to a kinematic surface, a surface invariant with respect to a one-parameter group of Euclidean motions (Pottmann and Wallner, 2001). Such a one-parameter group has constant velocity vector field, which is estimated by above objective. We will arrive at the same optimization problem if we consider the image of a time slice under an unknown, linearized inter-frame motion. This image is required to lie as close as possible to a space-time surface S, given by $\mathcal{Q}$. In order to measure the residuals of the displaced points to S, the latter is approximated by the tangent planes in the samples. Minimizing the squared distance to the tangent planes yields the objective of Equ. (4.1). Optimization involves the solution of a six-dimensional system of linear equations.

In the course of above discussion, we gave several hints at what the final registration algorithm will look like. In the following, we shortly summarize the single steps.

Algorithm. *Given point samples $P^i = \{\mathbf{p}^i \in \mathbb{R}^3\}$, $i = 0, \ldots, N$ of a moving object, acquired at times $t_i = i \cdot \Delta t$, a registration algorithm constraining the measurements in a least-squares sense locally to a kinematic surface comprises the following steps.*

1. *Transform the input data to space-time models $Q^i = \{(\mathbf{p}^i, t_i) \in (\mathbb{R}^3 \times \mathbb{R})\}$.*
2. *For each time slice, compute the unknown parameters $(\bar{\mathbf{c}}_i, \mathbf{c}_i)$ of the inter-frame motion's velocity vector field as solution of Equ. (4.1).*
3. *Reconstruct the inter-frame motions α_i from $(\bar{\mathbf{c}}_i, \mathbf{c}_i)$ with one of the methods in Sec. 2.2.*
4. *Summarize the data at time t_N,*

$$\alpha_{N-1} \circ \alpha_{N-2} \circ \ldots \alpha_i(P^i),$$

for a maximal reconstruction of the scanned object.

4.3 Further Discussion

This section undertakes a detailed discussion of the optimization problem in Equ. (4.1). We consider time scaling issues related to the input data conversion, the essential computation of normals to the space-time surface, numerical characteristics of the objective and links to classic work on registration.

4.3.1 Time Scaling

Above, we have already noted that the scale of the time axis — basically the ratio of spatial vs. temporal unit length — is arbitrary. Hence, we need to consider the influence of a scaling of the time axis by a factor of λ,

$$t_i \mapsto \lambda t_i,$$

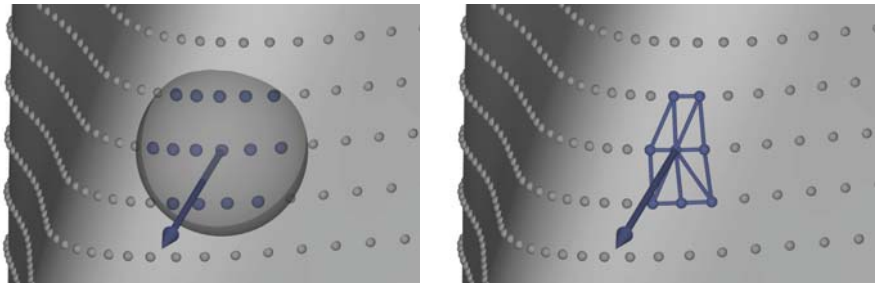

Figure 4.4: Normals of high quality are essential for the proposed algorithm. We propose a PCA based method (left) and a method based on triangulation (tetrahedralization) of the input data (right).

on the estimation of motion parameters $(\bar{\mathbf{c}}_i, \mathbf{c}_i)$. Scaling by λ changes the fourth coordinate of the inter-frame motion's velocity vector,

$$(\bar{\mathbf{c}}_i + \mathbf{c}_i \times \mathbf{p}, 1) \mapsto (\bar{\mathbf{c}}_i + \mathbf{c}_i \times \mathbf{p}, \lambda).$$

Moreover, the space-time surface's unit normals are modified,

$$\mathbf{n} = (\hat{\mathbf{n}}, m) \mapsto \frac{1}{(\lambda^2 \hat{\mathbf{n}}^T \hat{\mathbf{n}} + m^2)^{\frac{1}{2}}} (\lambda \hat{\mathbf{n}}, m).$$

Updating Equ. (4.1) with these new expressions yields a factor of $\rho = \frac{\lambda^2}{\lambda^2 \hat{\mathbf{n}}^T \hat{\mathbf{n}} + m^2}$ per summand. At first glance this seems to bias the optimization severely, as individual scaling of the objective's summands will give different minimization results in general. However, for small, smooth inter-frame motions, the first three coordinates $\hat{\mathbf{n}}$ will dominate the fourth coordinate m significantly. $\|\hat{\mathbf{n}}\| \gg |m|$ and $\hat{\mathbf{n}}^T \hat{\mathbf{n}} \approx 1$ imply $\rho \approx 1$ and thus minimize any effects of scaling the time axis.

4.3.2 Normal Estimation

So far we have not given any details on how to compute normals in the elements of the discrete space-time surface $\mathcal{Q}$. In the following, we will discuss two methods, that differ in how well they adapt to a scaling in time (cf. Fig. 4.4). Above considerations showed, that our method is only insensitive to time scaling, if the normal estimation captures the change in coordinates correctly.

PCA-based method

A widely used method for normal computation fits a hyperplane to a local neighborhood $N_r(\mathbf{q}) = \{\hat{\mathbf{q}} \in \mathcal{Q} : \|\mathbf{q} - \hat{\mathbf{q}}\| < r\}$ of a point $\mathbf{q}$, e.g. by a principal component analysis (PCA) of $N_r(\mathbf{q})$. The normal estimate $\mathbf{n}$ will then be the normal of the estimated fitting hyperplane. The characteristics of $N_r(\mathbf{q})$ greatly affect resulting $\mathbf{n}$. We define two criteria regarding the neighborhood $N_r(\mathbf{q})$ for normals of good quality.

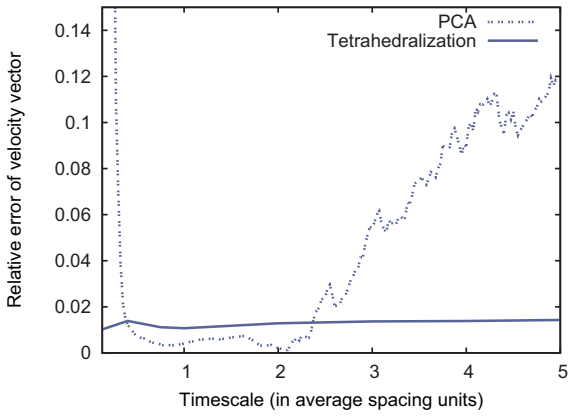

Figure 4.5: The two proposed methods for normal estimation differ in how well they adapt to a scaling of the time axis. While the PCA based method gives good results for timescales from 0.5 to 2 times the average spacing of input data, the connectivity based method is effected only little by time scaling.

First, we make use of the degree of freedom due to the arbitrary scale of time. We require the average temporal distance of frames to equal the average spatial spacing of the point samples. As a second requirement, we determine the radius of $N_r(\mathbf{q})$ from a quality measure for normal estimations with PCA, proposed by (Pauly et al., 2002). With the estimated normal basically being the eigenvector with respect to the smallest eigenvalue of the neighboring data's covariance matrix, the ratio of this smallest eigenvalue to the sum of all eigenvalues may be used as measure for the quality of the estimate. We start with small radius r and increase it until the ratio stops improving.

Returning to a discussion of time scaling effects, we observe that PCA adapts correctly to a scale of time, given that $N_r(\mathbf{q})$ does not change. Scaling of the point's fourth coordinate inversely scales the fourth coordinate of the covariance matrix's eigenvectors. Fig. 4.5 visualizes how violations of the first of the above two criteria influences the normal estimation quality. For the ratio of temporal vs. spatial sampling being between 0.5 and 2, PCA performs well. However, outside this range, the normals' and the result's quality decrease rapidly.

Local tetrahedralization

In low noise conditions, we can reliably estimate normals using a local surface meshing approach. We illustrate this method in $\mathbb{R}^{2+1}$ first, before sketching a generalization of this technique to one dimension higher. For a space-time surface traced by a curve, we perform a local surface triangulation in 3D around each vertex $\mathbf{q} = (\mathbf{p}, t_i)$. A normal estimation in $\mathbf{q}$ can then be achieved by averaging the one-ring's face normals. We observe that the one-ring contains only neighbors of $\mathbf{q}$ in time slices t_{i-1}, t_i and t_{i+1}. Hence, we can limit the local triangulation efficiently to a subset of the input data. The

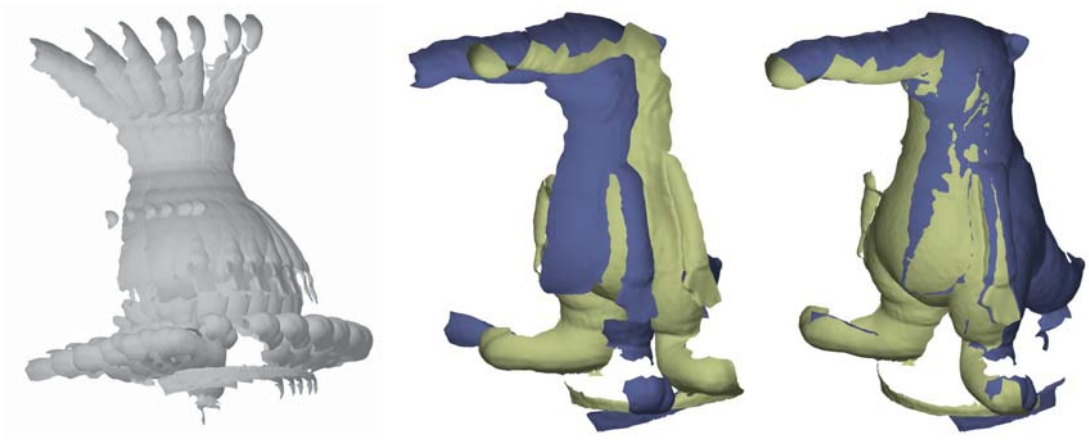

Figure 4.6: Pairwise classic registration of the initial input data (left) introduces high accumulation errors over 300 time frames (center). The dynamic geometry registration yields a better alignment of the first to the last frame (right).

triangulation is very local in its nature, as we may neglect global issues like intersecting triangles or unique normal orientation.

For surfaces, we generalize this approach to four dimensions and generate a local tetrahedral mesh. Again, computing a local tetrahedralization around $\mathbf{q} = (\mathbf{p}, t_i)$ involves the previous, current and next time slice only. With respect to the current time slice, three types of tetrahedra occur. Those with a face entirely in Q^i, those with only one edge in Q^i and finally those with only one vertex in the current time slice. The tetrahedra define the connectivity of the local neighborhood of $\mathbf{q}$. With the four-dimensional variant of the cross product, we compute a normal for each tetrahedra. Averaging all adjacent tetrahedra' normals yields a surface normal in $\mathbf{q}$. Again, we may ignore global issues such as local intersections or orientation issues.

In low noise conditions, this second technique for space-time normal computation yields better results for large time scaling (see Fig. 4.5). For more robustness at cost of increasing complexity, the tetrahedralization may be enhanced to span more than just adjacent time slices, thus averaging k-ring face normals.

4.3.3 Sensitivity to Noise

In order to examine stability issues of our registration algorithm, we consider how noise in the input data propagates to the final motion estimates $\mathbf{x} := (\bar{\mathbf{c}}_i, \mathbf{c}_i)$. Let the noise in the sample points be bounded by ε_P, that in normals by ε_N. For the PCA based method, the relation between ε_P and ε_N has been studied by (Mitra and Nguyen, 2003). Let $A \cdot \mathbf{x} = \mathbf{b}$ denote the linear system to be solved in a minimization of Equ. (4.1). Then, for matrix A and right hand side vector $\mathbf{b}$ we obtain for the Frobenius norm $\|.\|_F$

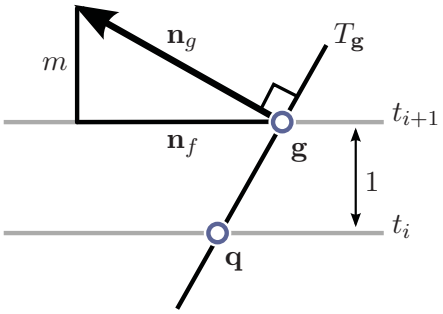

Figure 4.7: Consider sample $\mathbf{q}$ and its closest point $\mathbf{g}$ in the next time frame. By choosing the normal in $\mathbf{g}$ such that $\mathbf{q}$ is in the tangent space $T_\mathbf{g}$ of $\mathbf{g}$, the objective of the proposed algorithm turns out to be the same as that of (Chen and Medioni, 1992) for classic registration.

of distorted $\tilde{A}$ and Euclidean norm of $\tilde{\mathbf{b}}$,

$$\|\tilde{A}\|_F = \|A\|_F + O(\varepsilon_P) + O(\varepsilon_N)$$

and

$$\|\tilde{\mathbf{b}}\| = \|\mathbf{b}\| + O(\varepsilon_P) + O(\varepsilon_N).$$

Well-known results for the numerical stability of methods to solve systems of linear equations let us examine the effects of noise on the solution. Given that $\varepsilon_P + \varepsilon_N < \frac{1}{\|A^{-1}\|_F}$, an a priori error estimate for direct methods such as Gaussian elimination gives (Isaacson and Keller, 1994),

$$\frac{\|\tilde{\mathbf{x}} - \mathbf{x}\|}{\|\mathbf{x}\|} \leq \frac{k(A)}{1 - k(A)\frac{\|\tilde{A}-A\|_F}{\|A\|_F}} \left(\frac{\|\tilde{A} - A\|_F}{\|A\|_F} + \frac{\|\tilde{\mathbf{b}} - \mathbf{b}\|}{\|\mathbf{b}\|} \right).$$

We see that the relative error in $\mathbf{x}$ is linear both in ε_P and in ε_N. However, the constant may be large for bad condition number $k(A)$, e.g. when we solve for nearly slippable motions.

4.3.4 Relation to ICP

The estimation of normals $\mathbf{n}_i$ to the space-time model $\mathcal{Q}$ leaves some degree of freedom, as we have seen above. In this section, we are going to use this freedom one more time. We will show, that for specific choice of $\mathbf{n}_i$, our registration method is equivalent to the ICP algorithm in the variant of (Chen and Medioni, 1992).

For a given point $\mathbf{p} \in \mathbb{R}^3$ acquired at time t_i, let $\mathbf{f} \in \mathbb{R}^3$ be the closest (corresponding) point in adjacent time frame $t_{i+1} = t_i + 1$. Accordingly, we may write, $\mathbf{q} = (\mathbf{p}, t_i)$ and $\mathbf{g} = (\mathbf{f}, t_{i+1})$ in the space-time model (see Fig. 4.7). Let $\mathbf{n}_f$, $\|\mathbf{n}_f\| = 1$, denote the normal in $\mathbf{f}$ to the $(i+1)$-th time slice. Then we define the space-time normal in $\mathbf{g}$ as,

$$\mathbf{n}_g := (\mathbf{n}_f, m).$$

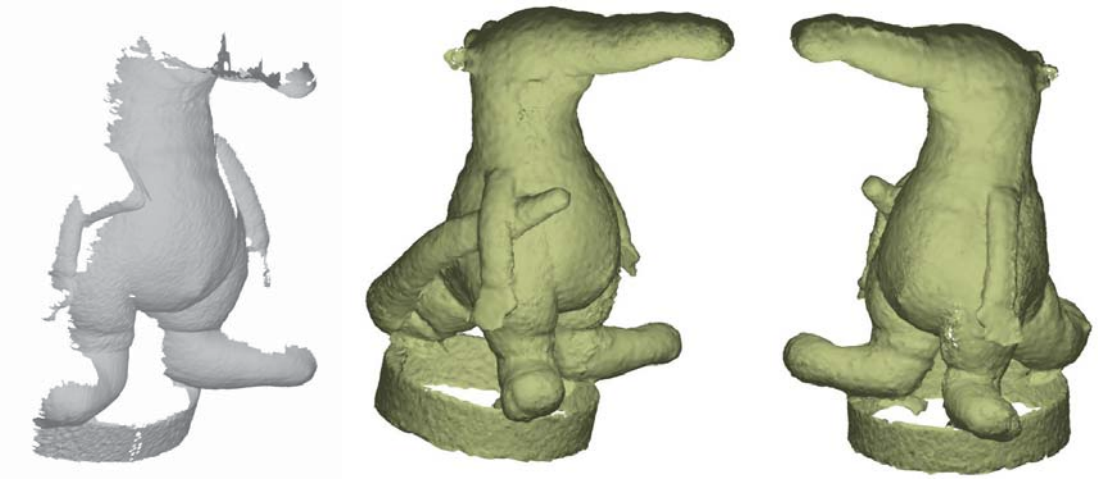

Figure 4.8: A reconstruction of a coati model from more than 2000 point clouds, acquired at 17 frames per second, without noise removal or general error distribution. The point clouds have been triangulated for better visualization.

We choose m such that $\mathbf{q}$ is element of the tangent space in $\mathbf{g}$ (cf. Fig. 4.7).

This special normal choice yields for an unsquared summand of Equ. (4.1),

$$\begin{aligned}\mathbf{v}(\mathbf{q})^T \cdot \mathbf{n}_q &= \mathbf{v}(\mathbf{q})^T \cdot \mathbf{n}_q + (\mathbf{q}-\mathbf{g})^T \cdot \mathbf{n}_q = (\mathbf{q}+\mathbf{v}(\mathbf{q})-\mathbf{g})^T \cdot \mathbf{n}_q \\ &= \begin{pmatrix} \mathbf{p}+\mathbf{v}(\mathbf{p})-\mathbf{f} \\ t_i+1-t_{i+1} \end{pmatrix}^T \cdot \begin{pmatrix} \mathbf{n}_f \\ m \end{pmatrix} = (\mathbf{p}+\mathbf{v}(\mathbf{p})-\mathbf{f})^T \cdot \mathbf{n}_f.\end{aligned}$$

Squaring the latter expression gives the squared distance of displaced point $\mathbf{p}$ to the tangent space in foot point $\mathbf{f}$. We have already encountered this distance measure before in Sec. 2.2 as error term of the ICP registration algorithm proposed by (Chen and Medioni, 1992).

This comparison emphasizes some of the space-time registration algorithm's advantages. Whereas classic ICP employs approximations of the squared distance function, the proposed algorithm is locally exact in terms of instantaneous kinematics. The quality of a solution of Equ. (4.1) depends only on the quality of estimated normals. We also note that the ICP method is based on correspondences between points of matching data sets. The space-time algorithm is correspondence-free and incorporates information from multiple time slices at once via the normal estimates (cf. Fig. 4.6).

4.3.5 Extensions

The presented algorithm can be extended in various ways. The normal estimation may be performed iteratively by updating the neighborhood for the PCA method with the inter-frame motions of the last iteration. The major assumption in above derivation has been that the unknown motion is rigid. By performing the optimization of Equ. (4.1) not

over the entire time slice but only over small subsets, the algorithm may be applied to the reconstruction of articulated or even deformable objects. We do not further discuss these extensions here and refer to the original publication (Mitra et al., 2007) instead.

4.4 Examples

The Stanford bunny (Fig. 4.2), the coati model (Fig. 4.8) and the bee model (Fig. 4.9) illustrate results of the proposed algorithm. The Stanford bunny model was reconstructed from more than 300 point clouds counting 33k elements each. The input data was obtained in a simulated scanning process on the GPU's z-buffer, that added artificial Gaussian noise to the data. The alignment took 13 minutes on a 2.4GHz Athlon Dual Core computer, more than 80% of the computation time was spent on neighborhood queries for the PCA based normal estimation. The coati and the bee input data was acquired with a scanner developed by (Weise et al., 2007) at 17 frames per second. The alignment of the 2200 frames per model with more than 20k (coati) and 28k (bee) samples each took 51 and 71 minutes respectively. For these as for all other of our experiments neither noise removal nor any global error distribution was carried out.

The numerical experiment on the scalability of the two proposed normal estimation techniques with respect to a scaling of the time axis (cf. Fig. 4.5) was done for the Stanford bunny data set. The unit length of the time axis was given by the average spatial sampling density of the input data. In addition, we compared the point to tangent plane ICP algorithm to the proposed algorithm at hand of 300 time frames from the coati data set. For each time frame, the velocity vector field was estimated with either approach. Finally, the first frame was aligned to the last frame. Fig. 4.6 illustrates that the registration algorithm estimating kinematic properties of a space-time surface shows significantly less accumulation error than the classic ICP approach.

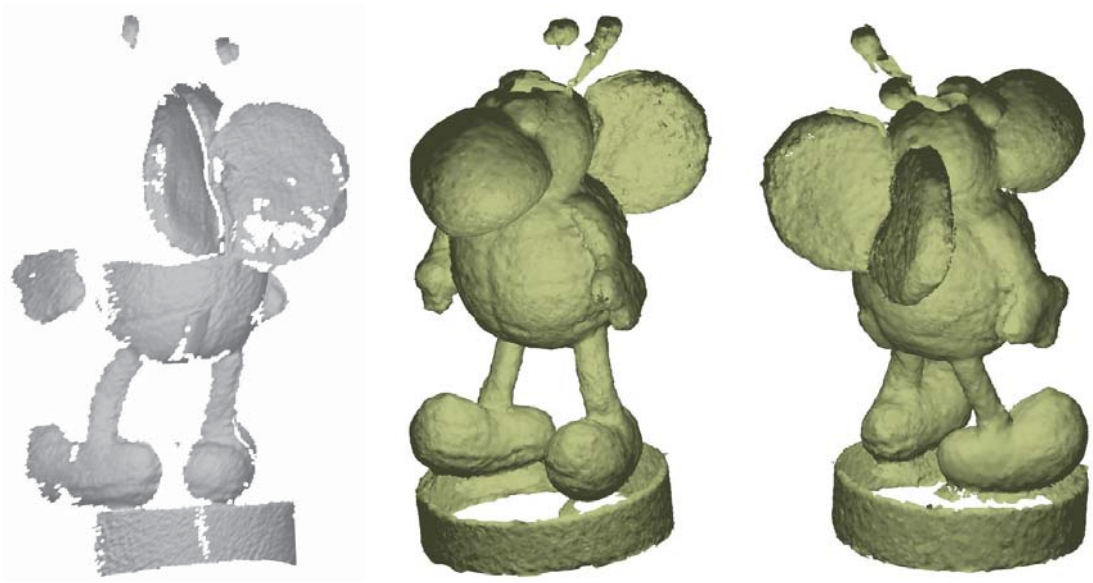

Figure 4.9: A reconstruction of a bee model from more than 2000 point clouds, acquired at 17 frames per second, without noise removal or general error distribution. The point clouds have been triangulated for better visualization.

Part II

Constrained Surface Fitting

5 Ruled Surface Approximation

The second part of this work moves the focus away from registration, without giving up on it totally. We will return in a short note to the local alignment of shapes. In this part, we turn our attention towards a class of surfaces, both simple and versatile in applications. Consider a straight line, moved through space according to a smooth one-parameter Euclidean motion. The line generates a surface, a so-called *ruled surface*. In mathematical terms, such a ruled surface may be described in parametric form,

$$\mathbf{x}(u, v) = \mathbf{g}(v) + u \cdot \mathbf{e}(v) \qquad (u, v) \in U \subseteq \mathbb{R}^2.$$

$\mathbf{e}(v) : \mathbb{R} \to \mathbb{R}^3$ denotes the direction vector of the *ruling* or *generator* at parameter value v. $\mathbf{g}(v) : \mathbb{R} \to \mathbb{R}^3$ is often called *base curve* or *directrix*.

Well-known objects of geometry are ruled surfaces: planes, right circular and general cylinders or one-sheeted hyperboloids, for example. The simple origin of ruled surfaces let them appear in various handcraft and production processes. In particular, production technologies such as milling and mold creation process ruled surfaces at their core. At larger scale, the simple generation and elegant shape of ruled surfaces attract architecture. In Chapter 6 we will discuss applications and uses of ruled surfaces in much more detail. For now, we sketch two examples from CNC milling and shape manufacturing to outline our motivation for *approximations* with ruled surfaces.

Let us consider the geometric properties of CNC milling with cylindrical cutting tools, so-called *side* or *flank* milling (cf. Fig. 5.1). At any time of the milling process, the cutting tool's position is defined by the locus of its axis. Given that the cylinder axis is a straight line segment, we observe that the cutting tool's motion is described by a ruled surface. The cutter itself is ideally in contact with the base material along its entire lateral surface. Consequently, the tool axis is required to be in fixed offset distance (namely the tool's radius) to the target shape. We obtain two requirements for describing the tool path for flank milling. Once the tool's axis must lie on a ruled surface and second, this ruled surface is supposed to be in fixed offset to the target shape. As in general the offset of a surface is not a ruled surface, both criteria may not be fulfilled exactly and we face an approximation problem.

Another example for the wide use of ruled surfaces in production technologies is heated-wire cutting. This technique originates from the model making community, where model parts are cut by moving a heated wire under tension through a low melt point material such as expanded polystyrene foam (Mayer and Moaveni, 2008). Identifying the heated wire with a straight line segment we see that the cutting surfaces are ruled. The simplicity of heated-wire cutting makes it a fast and inexpensive method and it found many applications from rapid prototyping (Broek et al., 2002) to the production of molds for architectural panels (Veltkamp, 2007). The question arises, how well general non-ruled shapes may be processed with this technology. Given that the target shape

meets certain geometric criteria, that we will detail below, an approximation with one or more ruled surface patches will be feasible. From an abstract point of view we see that we arrive at a very similar approximation problem as before: given a shape, the task is to remodel it as a ruled surface. The second part of this thesis aims at developing a comprehensive framework for such ruled surface approximations.

We will assume that the initial shape is given as discrete point set P. Point sets are a very general representation for shapes and we ensure this way that our algorithms apply to a wide range of input data. The approximating ruled surface patch will be modeled as B-spline surface patch linear in one parameter direction. Building upon the work of (Pottmann and Leopoldseder, 2003), we will set up an optimization framework, deforming the B-spline surface patch until it approximates P in an optimal sense. Recalling the terminology of the introductory chapter, the deformation of the approximating surface coincides with our notion of a shape manipulator. We will match the approximating shape with the target shape by solving for an unknown shape manipulator.

The contribution of the following chapters will be various extensions to this basic approximation algorithm. In particular, we are going to address the initialization of the optimization (initial approximating surface and choice of parameters), constraints to the generator's motion in space, applications to milling and architecture and certain smoothness aspects.

5.1 Related Work

We organize the following review of related literature in three parts. First, we consider contributions specific to the flank milling of ruled surfaces. Though these methods are not explicitly related to our work, they give a good picture of the wide use of ruled surfaces in the past. Second, we turn our attention to literature on the approximation of surfaces with ruled surfaces. Finally, we survey related work that motivates our specific approach of surface approximation.

Positioning a cylindrical cutter for the side milling of a ruled surface has been addressed since years (Stute et al., 1979). As outlined above, this problem requires an approximate solution. (Redonnet et al., 1998) for example propose to locate the tool's axis such that it touches both boundary directrices and a ruling. Undercutting errors (e.g. material that gets accidentally removed by the cutter, see Fig. 6.1) have been specifically addressed by (Tsay and Her, 2001) who give an analytic expression for the errors that are minimized at each time step. (Bedi et al., 2003) do not consider the loci of tool axes directly but examine the tool's envelope surface as the cutter is slided along the boundary curves. (Gong et al., 2005) build upon this method and suggest a least-squares optimization to approximate the tool path as offset of the designed surface. (Lartigue et al., 2003) have used B-spline surfaces to describe the loci of axes before. They obtain the final tool path by directly minimizing the distance from the cutting cylinder's envelope to a given surface. (Senatore et al., 2008) combine the theoretical results of (Gong et al., 2005) with the method of (Redonnet et al., 1998) and analyze the approximation error of their envelope surface based technique. A different approach is taken by (Sprott and Ravani, 2008) who state the tool path generation problem for cylindrical milling in terms of line

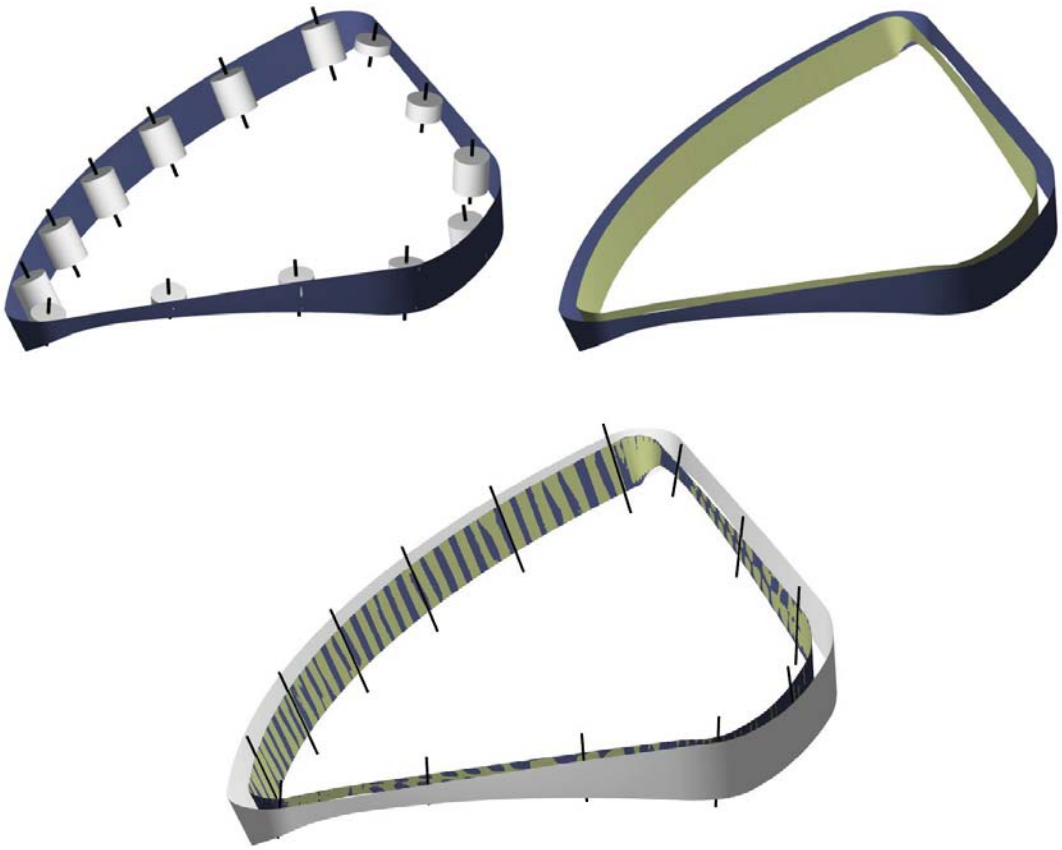

Figure 5.1: The tool path computation for cylindrical flank milling faces two constraints. Once, the tool's axis, being a straight line segment, moves on a ruled surface (top, left). Second, the tool's axis is required too move in constant offset distance to the target shape (top, right). As both requirements may not be met at the same time in general, a ruled surface approximation of the offset surface is necessary (bottom).

geometry. The authors build upon the observation that the surface normals along a non-torsal ruling lie on a hyperbolic paraboloid. The cutting tool axis is then determined from that family of rulings the initial generator originated from. More general milling techniques with conical (Li et al., 2006) or toroidal (Roth et al., 2001) cutters have not found such a comprehensive treatment as the cylindrical case.

Outside the milling community, ruled surfaces have been of interest in computer aided geometric design (often with milling applications in mind). Given a surface, various techniques of approximating it with ruled surfaces have been proposed. (Hoschek and Schwanecke, 1998) consider ruled surfaces as envelopes of their tangent planes. The authors employ dual tensor product B-spline surfaces linear in one parameter direction to model the unknown ruled surface patch. Besides the interpolation of given rulings or data points, the authors approximate scattered data, triangulated in a preprocessing step, with a ruled surface. Interpolation and approximation with ruled surfaces with line geometry is the topic of (Chen and Pottmann, 1999). The authors describe two affine spaces in which the interpolation or approximation of a line set turns down to a curve interpolation or approximation problem. Once, the end points of a line segment are aggregated into a point in Euclidean 6-space. Second, the set of lines intersecting two parallel planes is mapped from the Klein quadric into an affine 4-space by a stereographic projection.

The reconstruction of and the approximation with torsal ruled surfaces have found certain attention as well. (Hoschek and Schneider, 1997; Pottmann and Wallner, 2001) consider developable surfaces as envelope of their one-parameter family of tangent planes, (Peternell, 2004) tackles the problem from a Laguerre geometry point of view and (Kilian et al., 2008) design and reconstruct developable surfaces in a discrete setup.

When it comes to approximation of arbitrary freeform surfaces, a single ruled surface patch may not suffice to obtain an approximation of satisfactory quality. (Elber and Fish, 1997) recursively approximate a given parametric surface with C^0 joined patches of ruled surfaces, until a global error estimate is satisfied. The authors construct a ruled surface patch from two boundary curves, by linearly interpolating points of equal parameter value. The obtained piecewise ruled surface approximation is offset such that the cutting tool is tangent to the target surface locally. (Han et al., 2001) propose a more general approach and segment a freeform surface into patches with approximately similar normal vector. Respecting the different possible topologies of the patches, the authors obtain ruled surfaces by linearly interpolating between the regions' boundaries. With rapid prototyping applications in mind, (Koc and Lee, 2002) slice a discrete point model with parallel planes in adaptively varying distance. From the intersection contours, a C^0 continuous ruled surface approximation is obtained by connecting points on the contours. However, as opposed to the previous two contributions, the authors minimize the squared distance of corresponding boundary points to determine the ruled surface patch's parametrization.

In (Pottmann and Leopoldseder, 2003), the authors propose an active contour model for surface approximation. An initial parametric surface (for example a B-spline surface) is deformed by minimizing approximations of the squared distance function from the approximating shape to the target shape until it fits the original model. Ruled surface

approximation is a special case of this contribution by choosing a ruled fitting surface. This is our method of choice for ruled surface approximation and we will extend and constrain it in the following chapters.

The remaining part of this literature review is devoted to related work specific to this method of B-spline surface approximation. The general problem of reconstructing spline curves or surfaces from scattered point data has been of interest since the early days of computer aided design (Cox, 1971; Hayes and Halliday, 1974). Typically, each data point gets a unique point on the approximating parametric shape assigned. In this point on the fitting shape, the distance function to the data point is described approximately. Subsequently, an improved fitting is obtained from minimizing the approximated distance term. Given that the point cloud is ordered, several rules for computing the parameter values have been proposed such as the chord length or the centripetal parametrization method, see (Farin, 1988) for an overview.

The choice of parameters has a strong effect on the final result and parameter correction strategies have been proposed to improve the initial selection. (Hoschek, 1988) describes a linear approximation to exact foot point computation to update the parameter value such that the residual vector is nearly orthogonal to the approximating curve. With the updated parameter values, the distance minimization is entered again. This procedure is repeated until a satisfying approximation quality is achieved. Recalling that shape matching problems are highly non-linear, it comes as no surprise that the iterative nature of this approach yields good results. Higher-order approximations of the foot point relation have been studied by (Saux and Daniel, 2003; Hu and Wallner, 2005). We are going to rely on exact foot point computations at each iteration (Hoschek and Lasser, 1993).

Once a foot point was assigned to each data point, the distance from the fitting shape to the target point cloud is approximated. Summation over the single distance approximations yields the objective of a minimization problem bearing an updated position of the fitting shape as solution. The order of distance approximation distinguishes the different methods and we employ the terminology of (Wang et al., 2006) in the following overview. *Point to point* methods minimize the length of the residual vector from data points to foot point. Probably due to its simplicity, this method is the one widest spread and has found many applications in curve (Plass and Stone, 1983; Hoschek, 1988; Saux and Daniel, 2003) and surface fitting (Eck and Hoppe, 1996; Weiss et al., 2002).

In computer vision, *active contours* or *snakes* are widely employed for the reconstruction of contours in images and beyond. Initially proposed by (Kass et al., 1988), the iterative minimization of an external and an internal energy has been the motivation for many iterative methods for B-spline curve and surface fitting (Pottmann and Leopoldseder, 2003; Wang et al., 2006). The internal energy resembles the approximations of the squared distance function we are about to discuss. The external energy is typically a regularization or smoothing term, we are going to detail below as well. Returning to our overview of distance approximation methods, (Blake and Isard, 1998) propose to minimize the distance from the data point to the tangent plane in the foot point in an active contour setup. Considering that the tangent plane encodes local information about the foot point's neighborhood, this *point to tangent plane* technique gives

faster convergence than point to point methods. Further improvements with respect to convergence yields the curvature based method by (Wang et al., 2006). In particular, this work undertakes a thorough theoretic investigation of the convergence characteristics of above three techniques.

All these methods have in common that they minimize approximations of the *squared* distance function between fitting shape and point cloud. The reason for this lies mainly in the easy to solve quadratic objectives that are typical for linear least-squares approaches. Drawbacks immanent to least-squares, such as high sensitivity to outliers, are addressed in (Flöry and Hofer, 2010). The authors consider non-smooth l_1-optimization problems arising from approximations of the *unsigned* distance function, as has been done for registration in Chapter 3.

The remaining parts of this chapter are organized as follows. First, we shortly review the active contour model of (Pottmann and Leopoldseder, 2003) for ruled surface approximation. We proceed by discussing the initialization of the approximation algorithm, both in terms of initial fitting shape and choice of parameters. We will conclude this chapter by sketching a method of knot adaption for surface fitting.

5.2 Ruled Surface Approximation

In this section, we define the ruled surface approximation problem and briefly summarize the algorithm in (Pottmann and Leopoldseder, 2003) for its solution. We show how it integrates well into the general shape matching framework of the introductory chapter and obtain this way the foundation for the remaining chapters of this work.

Let $P = \{\mathbf{p}_i \in \mathbb{R}^3 : i = 1, \dots, n\}$ denote a set of points in $\mathbb{R}^3$, called the *point cloud* henceforth. For tool path computations for cylindrical flank milling, P will comprise samples in offset distance to the base shape. For general ruled surface approximation, P will comprise samples on the input shape. Moreover, let $\mathbf{x}(u, v) = \sum_{j=1}^{m_u} N_j^1(u) \sum_{k=1}^{m_v} N_k^d(v)\mathbf{d}_{jk}$ be the approximating tensor product B-spline surface. $\mathcal{D} = \{\mathbf{d}_{jk}\}$ denotes the set of control points and N_k^d the k-th B-spline basis function of degree d. For ruled surface approximation, we set the degree in the first parameter direction u to 1. Hence, parameter u parametrizes the ruling at parameter v. If not stated otherwise, we assume that the second parameter direction v will be of cubic degree, $d = 3$. For the sake of simplicity, we summarize $N_i(u, v) = N_j^1(u)N_k^3(v)$ and $\mathbf{d}_i = \mathbf{d}_{jk}$ for $i = (j-1)m_v + k$ and write

$$\mathbf{x}(u, v) = \sum_{i=1}^{m=m_u m_v} N_i(u, v)\mathbf{d}_i. \tag{5.1}$$

If $m_u = 2$ we speak of a single *ruled surface patch*, or simply a ruled surface. Given that $m_u > 2$, we call $\mathbf{x}(u, v)$ a *ruled surface strip model*.

We are going to modify the shape of a ruled B-spline surface in an optimization framework. Therefore, we write

$$\mathbf{x}^\delta(u, v) = \sum_{i=1}^{m} N_i(u, v)(\mathbf{d}_i + \delta_i)$$

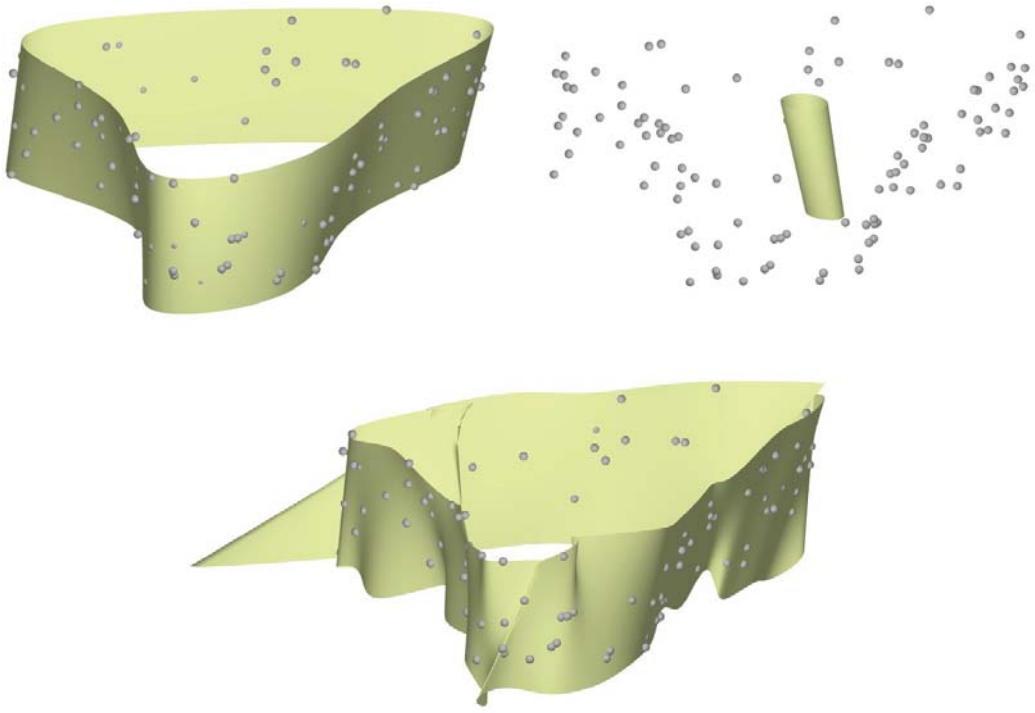

Figure 5.2: (Top, left) A ruled surface approximation. (Top, right) If the initial smoothing weight is chosen too large, the fitting surface shrinks. (Bottom) If the initial smoothing weight is chosen too small, the fitting surface's visual appearance is not satisfying.

with $\delta_i \in \mathbb{R}^3$ for a deformed or updated approximating surface. Please note that we will not modify the knot vectors of $\mathbf{x}(u, v)$ or the number of control points in $\mathcal{D}$ until Sec. 5.5. We call $\delta = (\delta_1^T, \ldots, \delta_m^T)$ the control points' displacement vector. In the terminology of Chapter 1, the linear map

$$T : \mathbb{R}^{3m} \to \mathbb{R}^{3m}, \quad \mathbf{d} \mapsto \mathbf{d} + \delta$$

denotes the shape manipulator for the geometric shape matching problem of ruled surface approximation.

In this setup, we say that $\mathbf{x}^\delta$ approximates point cloud P best, if for given $\mathbf{x}^0$ and P,

$$\delta = \operatorname{argmin} d^2(\mathbf{x}^\delta, P) = \operatorname{argmin} \sum_{k=1}^{n} d^2(\mathbf{x}^\delta, \mathbf{p}_k)$$

We see that this optimization problem is a special instance of the general shape matching problem in Chapter 1. For a solution, we alternately optimize for best foot points and control point displacements. Parametrization involves computation of the foot point

$\mathbf{x}^\delta(u_k, v_k)$ of the shortest distance from $\mathbf{x}$ to $\mathbf{p}_k \in P$. In the foot points, the squared distance from $\mathbf{x}^\delta$ to P is approximated and minimized. These steps are iterated until the approximation error falls below a user-defined threshold.

We will restate the two approximations of the squared distance function d^2 given in Chapter 1 in the notation of the surface fitting problem. The point to point error term reads as,

$$f_P(\delta) = \sum_{k=1}^{n} \|\mathbf{x}^\delta(u_k, v_k) - \mathbf{p}_k\|^2.$$

For the second-order Taylor approximation of d^2, we observe that the gradient ∇d of the distance function to $\mathbf{x}$ in a point $\mathbf{x}(u_k, v_k)$ coincides with the surface normal $\mathbf{n}_k$. Consequently, the point to tangent plane error term is of the form,

$$f_T(\delta) = \sum_{k=1}^{n} \left[\mathbf{n}_k^T \cdot \Big(\mathbf{x}^\delta(u_k, v_k) - \mathbf{p}_k\Big)\right]^2.$$

As the displacement map T of control points is linear and any point on $\mathbf{x}(u, v)$ is a linear combination of elements in $\mathcal{D}$, the objectives f_P and f_T are quadratic in the unknown displacements δ and minimization amounts to the solution of a linear system of equations.

Minimizing either approximation of the squared distance function yields mathematical correct solutions. However, the emerging minimal shape might not be visually pleasing, e.g. oscillations are frequently observed (cf. Fig. 5.2). To tackle this problem, a weighted regularization or smoothing term is added to $f_P(\delta)$ or $f_T(\delta)$. A prominent example originating from the literature on thin plate splines (Duchon, 1977) is the integral over the squared second partial derivatives of $\mathbf{x}^\delta(u, v)$,

$$f_s(\delta) = \iint (\mathbf{x}^\delta_{uu})^2 + 2(\mathbf{x}^\delta_{uv})^2 + (\mathbf{x}^\delta_{vv})^2 \,\mathrm{d}u\,\mathrm{d}v.$$

Numerical approximation of this smoothness measure with, for example, the trapezoidal rule yields another term quadratic in δ that is added as weighted penalty term to the distance term,

$$\min_\delta f_d(\delta) + \lambda \cdot f_s(\delta). \tag{5.2}$$

Here, f_d denotes any distance approximation f_P or f_T. Below, we will refer to λ as the *smoothing weight*.

With above distance approximation, enhanced by a regularization term, we have the main objective for the general shape matching algorithm of Chapter 1. For given approximating surface $\mathbf{x}$ we compute the foot points of the data points on $\mathbf{x}$, obtain displacements of the control points by minimizing Equ. (5.2) and modify $\mathbf{x}$ accordingly. Given that the approximation error is above a user-defined threshold, we repeat these steps. Two questions remain open with regard to this algorithm. How do we obtain the initial fitting shape $\mathbf{x}^0$ and how do we select the initial smoothing weight λ? The remaining sections of this chapter address these and related initialization topics.

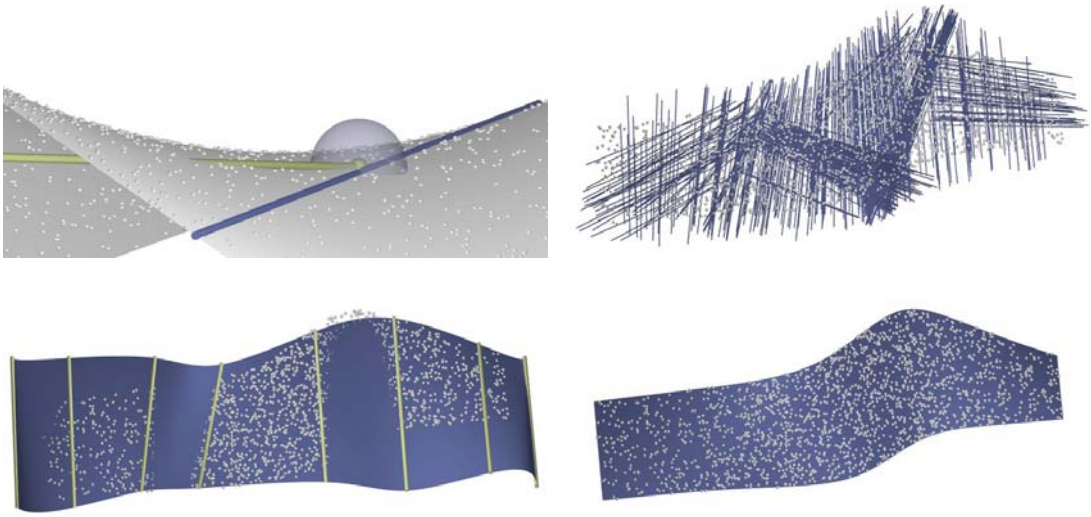

Figure 5.3: (Top, left) In an element of the target point cloud, a candidate ruling (blue) is computed by local constrained registration of an initial pose (yellow). Computation of candidate rulings in many samples of the point cloud yields a dense line set (top, right), that is pruned, ordered and interpolated (bottom, left). (Bottom, right) shows the final ruled surface approximation of the point cloud.

5.3 Initial Fitting Shape

The ruled surface approximation algorithm solves a non-linear optimization problem in an iterative way. In fact, as shown in (Wang et al., 2006), the point to point method is a variant of a steepest descent method, whereas the point to tangent technique resembles a Gauss-Newton minimization. As it is for non-linear optimization problems, good initialization is of significant importance for convergence at all and convergence speed in particular. For surface fitting, the initial solution $\mathbf{x}^0$ is given by the start shape of the approximating surface. Automatic computation of $\mathbf{x}^0$ is not only convenient for applications but lets us ensure certain important properties of the initial fitting shape. For ruled surface fitting, such a property would be that the rulings of $\mathbf{x}^0$ are well aligned with the asymptotic directions (tangential directions of vanishing normal curvature) in points of the target shape.

In (Kilian et al., 2008), a discrete model for approximation with developable surfaces is automatically initialized from a dense polyhedral mesh M. We shortly summarize this method that estimates rulings in vertices of the input mesh. Consider a vertex $\mathbf{p} \in M$ and the set of points on M in constant geodesic distance to $\mathbf{p}$. A pair of opposite points on this geodesic circle with minimal deviation in Euclidean and geodesic distance and minimal turning normal vectors along its connecting geodesic is considered a candidate

ruling. Without the additional check on normal twist this approach would be suitable as first step of the following proposed initialization pipeline. However, an adapted version handling point cloud inputs will comprise computational expensive neighborhood queries. For this reason, we propose another technique based on registration, given that we have already been concerned with registration in the first part of this work.

Given a point cloud P as above, our goal is to compute an initial ruled B-spline surface patch $\mathbf{x}^0$. We will first determine possible ruling directions in the elements of P. From this probably large set of candidate rulings we select a subset of high quality rulings in a pruning step. After ordering the line segments in a robust way, we obtain the initial fitting surface $\mathbf{x}^0$ by interpolating start and end points of the ordered rulings. We are going to detail these steps in the following, Fig. 5.3 illustrates the different stages.

5.3.1 Ruling Estimation

Let $\mathbf{p}_i$ be an element of the input point cloud P and l_r a user-defined initial ruling length. Typically, we choose l_r five times the average point spacing of P. The computation of a candidate ruling of length l_r, passing though an ε-neighborhood of $\mathbf{p}_i$ (to account for possible noise), may be regarded as shape matching problem. Indeed, it can be seen as registration problem aligning a straight line segment in an optimal sense locally to P.

Before describing this local shape matching in more detail we consider its initialization, in particular the choice of initial ruling directions. We will base this initialization on estimates of principal curvatures κ_1, κ_2 and corresponding principal curvature directions of P in $\mathbf{p}_i$, that we obtain for example by fitting a polynomial to a neighborhood of $\mathbf{p}_i$ (Yang and Lee, 1999). Locally, tangential directions of vanishing normal curvature — so-called *asymptotic directions* — resemble straight lines on P best (do Carmo, 1976). The number of asymptotic directions in a point $\mathbf{p}_i$ depends on the Gaussian curvature $K = \kappa_1 \cdot \kappa_2$ and may be derived for example from Dupin's Indicatrix.

- In hyperbolic points ($K < 0$) two asymptotic directions exist. Accordingly, we will compute two candidate rulings, each initialized from one of the asymptotic directions. The asymptotic directions may be computed from the principal curvature directions with Euler's curvature formula.
- In parabolic points (exactly one of κ_1 and κ_2 is zero) a single asymptotic direction exists (and is given by the principal curvature direction to zero principal curvature).
- In elliptic points ($K > 0$) there are no asymptotic directions. If at all, we may initialize with the principal curvature direction to principal curvature of minimal absolute value.
- In umbilical points ($\kappa_1 = \kappa_2$) any tangential direction is a principal curvature direction. We ignore such points.

From this overview we see that only points with Gaussian curvature $K \leq 0$ promise good local approximation. This observation generalizes to a global level, as ruled surfaces are surfaces with negative or zero Gaussian curvature everywhere (do Carmo, 1976). An

analysis of Gaussian curvature in points on the input shape is a helpful tool to decide whether a ruled surface approximation of P is feasible.

Let us return to the shape matching problem for the computation of an initial ruling in $\mathbf{p}_i$ (see Fig. 5.3). Assume we dispose of an initial ruling direction and let $\mathbf{r}$ be a set of equidistant samples on an initial ruling of length l_r centered in $\mathbf{p}_i$. In the notation of Chapter 1, registration of $\mathbf{r}$ to P amounts to the solution of

$$\min d^*(P, \mathbf{m}(\mathbf{r})) \tag{5.3}$$

for unknown rigid body motion $\mathbf{m}$. We refer to Chapters 2 and 3 for iterative minimization approaches and remain with two observations. Once, the point to tangent plane error terms will outperform the point to point methods significantly, as the displacements will happen mostly in directions tangential to P. Second, minimization of Equ. (5.3) may transform $\mathbf{r}$ out of an ε-neighborhood of $\mathbf{p}_i$. For this reason, we add a weighted (and squared for least-squares optimizations) penalty term to Equ. (5.3) favoring proximity of the ruling's mid point $\mathbf{r}_m$ to $\mathbf{p}_i$,

$$\|\mathbf{m}(\mathbf{r}_m) - \mathbf{p}_i\|.$$

Algorithm. *In summary, an algorithm for computing a candidate ruling in* $\mathbf{p}_i \in P$ *comprises the following steps.*

- *Initialize* $\mathbf{r}$ *according to the estimated principal curvatures in* $\mathbf{p}_i$*.*
- *Align* $\mathbf{r}$ *to* P *with above constrained registration.*
- *Extend* $\mathbf{r}$ *alternately beyond its start and end point and repeat the registration as long as the fitting error is below user-defined* δ*. Otherwise, stop.*

After computing ruling candidates in a subset of elements of P, we rate each ruling according to its length. We prune the set of rulings by selecting the ruling with highest rating and by removing any rulings close to it. Please note that for example in samples of a double ruled surface patch we will possibly face two candidate rulings with equal rating in a point. We do not consider this special case any further, as the regularization term will smooth any inconsistencies in the subsequent ruled surface approximation. We repeat this selection of best candidates until all rulings have either been chosen or removed. We remain with a sparse set $\mathcal{L}$ of high-quality ruling candidates.

5.3.2 Ruling Interpolation

As a next step, we construct a B-spline surface from $\mathcal{L}$. Therefore, we order the rulings and interpolate them with a ruled B-spline surface. Defining a partial order in a meaningful way as a binary relation on $\mathcal{L}$ fails for our purposes. However, the assumption that the rulings are samples of a sufficiently smooth surface allows the following definition (see also Fig. 5.4).

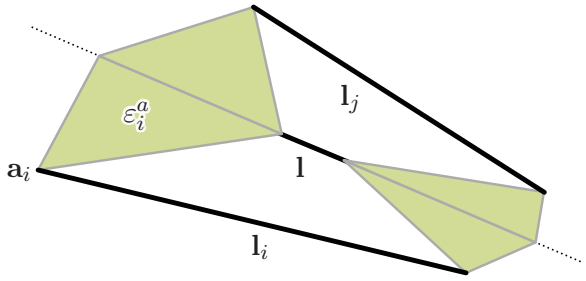

Figure 5.4: Three candidate rulings are said to form an ordered triplet if the angles of planes spanned by the end points of $\mathbf{l}_i$ and $\mathbf{l}_j$ with $\mathbf{l}$ are below a user-defined threshold α_T (typically $\alpha_T \in [\frac{\pi}{6}, \frac{\pi}{4}]$).

Definition (Ordered ruling triplet)**.** *Consider three rulings* $\mathbf{l}$, $\mathbf{l}_i = (\mathbf{a}_i\mathbf{b}_i)$, $\mathbf{l}_j = (\mathbf{a}_j\mathbf{b}_j) \in \mathcal{L}$. *Furthermore, let* $\mathbf{n}_i^a$ *denote the normal of plane* ε_i^a *spanned by* $\mathbf{l}$ *and* $\mathbf{a}_i$. $\mathbf{n}_j^a$, $\mathbf{n}_i^b$ *and* $\mathbf{n}_j^b$ *shall be defined accordingly. Then, we call a triplet of rulings* $(\mathbf{l}_i, \mathbf{l}, \mathbf{l}_j)$ ordered*, if the angles between normals* $\angle(\mathbf{n}_i^a, \mathbf{n}_j^a)$ *and* $\angle(\mathbf{n}_i^b, \mathbf{n}_j^b)$ *are smaller than user-defined* α_T.

This ternary relation induces a binary relation on $\mathcal{L}$. We write $\mathbf{l}_i \sim \mathbf{l}_j$ if $\mathbf{l}_i$ and $\mathbf{l}_j$ are element of an ordered triplet but there exists no $\mathbf{l} \in \mathcal{L}$ such that $(\mathbf{l}_i, \mathbf{l}, \mathbf{l}_j)$ is ordered. Please note that $\sim$ is not a partial order. However, we may call $\mathcal{L}' = \{\mathbf{l}'_1, \ldots, \mathbf{l}'_m\}$ ordered if $\mathbf{l}'_i \sim \mathbf{l}'_{i+1}$, $\forall i = 1, \ldots, m-1$. Let $\mathcal{P}$ be the set of not yet processed rulings, initialized with $\mathcal{L}$. Then, the algorithm in listing 5.1 computes an ordered set $\mathcal{S}$ of rulings by recursively prepending, inserting and appending rulings to an initial pair of rulings. It is possible that $|\mathcal{S}| < |\mathcal{L}|$ if a ruling is included in few ordered triplets. This is intended as such a ruling does not meet our assumption of a smooth base surface and is consequently considered an outlier. If $\mathcal{S}$ is significantly smaller than $\mathcal{L}$, the initial pair of rulings included at least one outlier. In this case, we reinitialize with a pair from $\mathcal{S}^C$.

For a sketch of a runtime analysis, let $n = |\mathcal{L}|$. The best-case occurs for example if $\mathbf{l}_2 \sim \mathbf{l}_3 \sim \ldots \sim \mathbf{l}_n \sim \mathbf{l}_1$ with $(\mathbf{l}_1, \mathbf{l}_2)$ as initial pair. It is of linear asymptotic complexity $O(n)$. A worst-case starts with a pair $(\mathbf{l}_1, \mathbf{l}_n)$ and repeatedly inserts the median element after a linear search of $\mathcal{P}$. We arrive at $O(n)$ searches of linear complexity which reveals a total worst-case complexity of $O(n^2)$. We omit an average-case analysis.

Finally, the ordered set of ruling end points is interpolated by a ruled B-spline surface patch. The resulting surface $\mathbf{x}^0$ may be additionally extended beyond its boundaries such that all elements of P have foot points in the interior of $\mathbf{x}^0$.

5.4 Initial Smoothing Weight

Besides the fitting shape, smoothing weight λ of Equ. (5.2) requires proper initialization. This parameter controls the influence of the smoothing term in opposition to that of the fitting term during optimization. Smoothing topics are frequently encountered in many areas of applied mathematics, from image processing to signal processing and statistics.

```
S = orderTriplets(P)
  S := {P1, P2};
  P := P \ {P1, P2};
  orderTripletsInsert(S1, P, prepend);
  orderTripletsInsert(S1, P, insert);
  orderTripletsInsert(S1, P, append);

orderTripletsInsert(l, P, dir)
  if dir = prepend then
    find l* ∈ P such that (l*,l,next(l)) is ordered
    if l* = ∅ then return;
    P := P \ l*;
    insert l* before l;
    orderTripletsInsert(l*, P, prepend);
    orderTripletsInsert(l*, P, insert);
  else if dir = insert then
    /* analog to dir = prepend */
  else if dir = append then
    /* analog to dir = prepend */
```

Listing 5.1: Algorithm to determine an ordered set of rulings based on the definition of ordered ruling triplets. $\mathcal{P}$ and $\mathcal{S}$ are efficiently realized as doubly linked lists.

While an automatic choice of smoothing parameters has especially been addressed in statistics (Simonoff, 1998), smoothing weight selection in geometry processing is usually done manually. In the following, we are going to develop a heuristic to choose λ automatically.

The combination of fitting and smoothing terms compares two quantities of different type, once a sum of squared distances and second a measure for the fitting shape's bending energy. To combine the both, one is added weighted to the other. In its simplest formulation, the objective at the i-th iteration may be written as,

$$f_i(\delta, \lambda) = f_d(\delta) + \lambda \cdot f_s(\delta).$$

Here, $f_d(\delta) = \delta^T A_d \delta + 2\mathbf{b}_d^T \delta + k_d$ denotes the fitting error term, $f_s(\delta) = \delta^T A_s \delta + 2\mathbf{b}_s^T \delta + k_s$ the smoothing term, δ the fitting surface's unknown displacements and finally λ the smoothing weight whose automatic choice is the topic of our considerations.

If λ is not chosen appropriately, the optimization fails to converge. Failed convergence lets the approximating surface degenerate. If λ is too large, the fitting shape will collapse into a single point in the limit (cf. Fig. 5.2, top right). Contrary, if λ is too small, the fitting shape will oscillate strongly (cf. Fig. 5.2, bottom). Either case bears the same effect on the control points' updates from one iteration to another. The displacements will be larger than in the successful setting. This observation motivates the following heuristic to choose an initial smoothing weight. We determine initial smoothing weight λ^* such that the maximal displacement of control points is minimal after the first iteration.

The optimal control points' update δ^* depends on the choice of λ and is given as

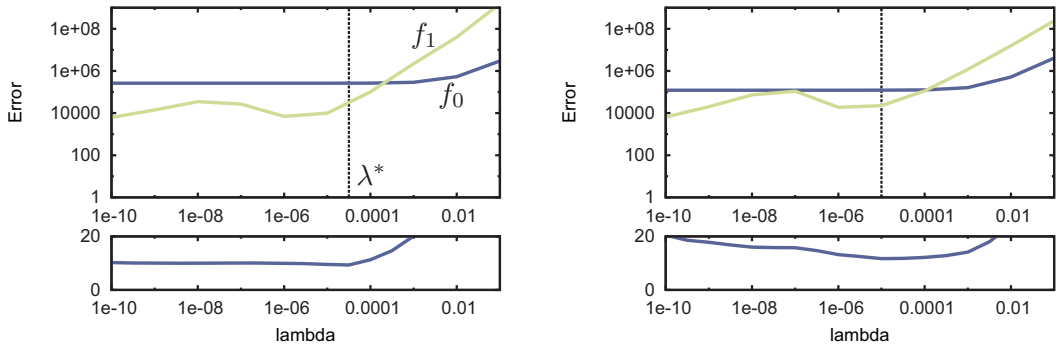

Figure 5.5: Initial smoothing weight λ^* is given for minimal l_∞-norm of the updates to the control points in the first iteration (bottom plots). Comparison of initial error f_0 to the error after the first approximation f_1 indicates that λ^* yields good initial convergence (top plots). Please note that all axes are of logarithmic scale.

minimizer of above functional $f_i(\delta, \lambda)$, quadratic in δ,

$$\delta^*(\lambda) = -(A_d + \lambda A_s)^{-1} \cdot (b_d + \lambda b_s).$$

According to above heuristic, the initial smoothing weight is given by,

$$\lambda^* = \text{argmin}_{\,\lambda \in \mathbb{R}} \|\delta^*(\lambda)\|_\infty$$

Given that the ruled surface approximation algorithm is robust to the choice of λ up to certain extents we do not attempt to solve above min-max problem exactly. Instead, we discretize the solution space $\lambda_k = 10^{-\frac{k}{2}}$ and determine λ^* from $\delta^*(\lambda_k)$.

To investigate the quality of so obtained λ^*, we compare the values of f_0 and f_1 (that is the objective's values before and after the first iteration) for λ fixed. In Fig. 5.5 we observe, that for large λ, the optimization does not converge as $f_1 > f_0$. For decreasing λ, we encounter a short interval of improving convergence before the ratio f_0/f_1 shows only little change. λ^*, that is included as vertical line in Fig. 5.5, lies well in the small interval of improving convergence. Fig. 5.6 shows f_0, f_1 and λ^* for several more data sets.

Above heuristic requires the solution of that many systems of linear equations as there are samples λ_k. Apart from the computational complexity, this method yields good initial smoothing weights as we have confirmed in above empirical tests. For point to tangent plane minimizations, that typically comprise a line search in the computation of δ^* (cf. Wang et al., 2006), running times would be much higher. Moreover, a line search will yield no degenerate updates but rather displacements of vanishing magnitude for bad parameter configurations. As the point to point term is an upper bound to the point to tangent plane term, $f_T(0) \leq f_P(0)$, we may either simply use the point to point's initial smoothing weight or multiply it by a factor of $f_T(0)/f_P(0)$.

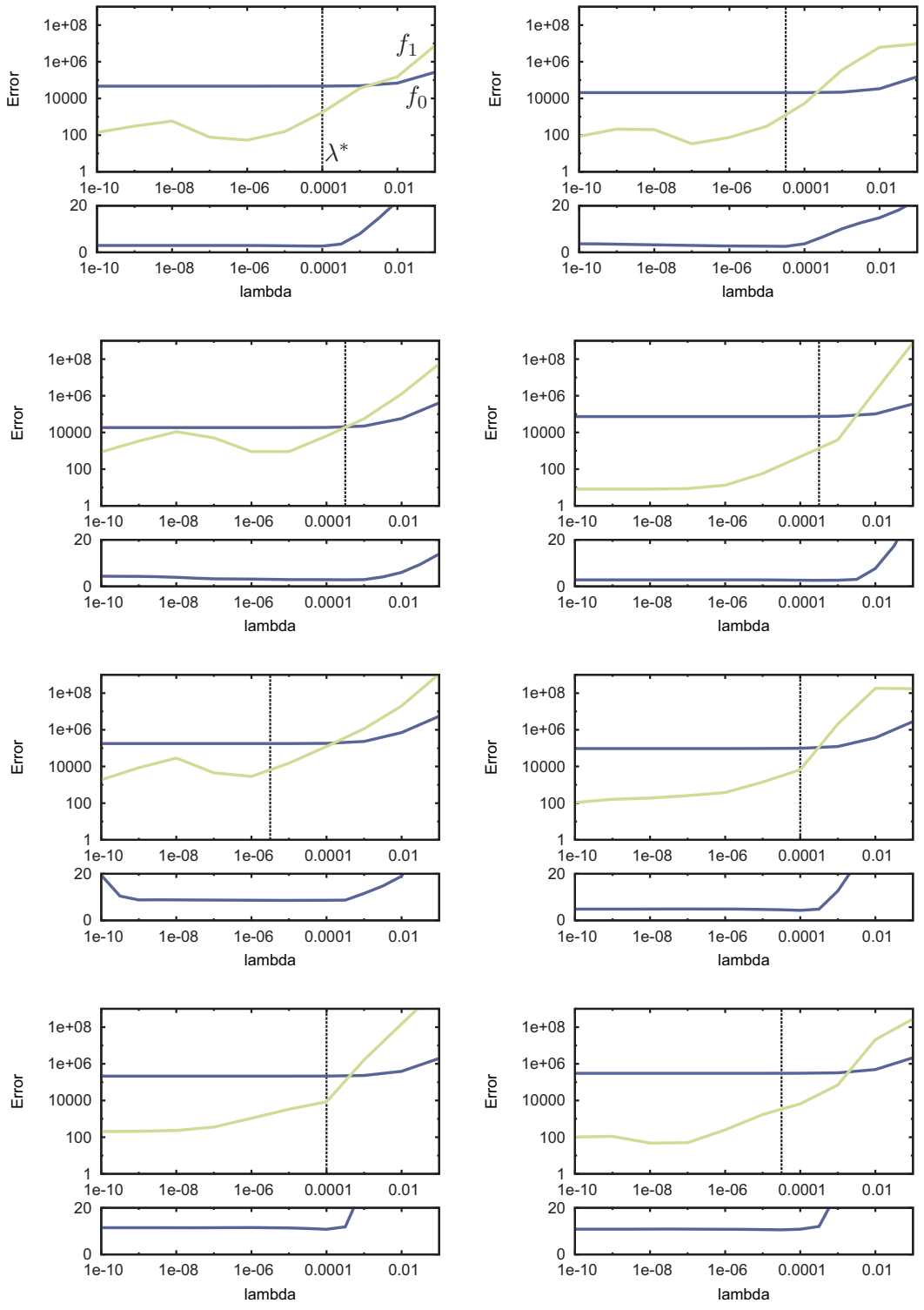

Figure 5.6: Further examples for the automatic initialization of the smoothing weight (see also Fig. 5.5).

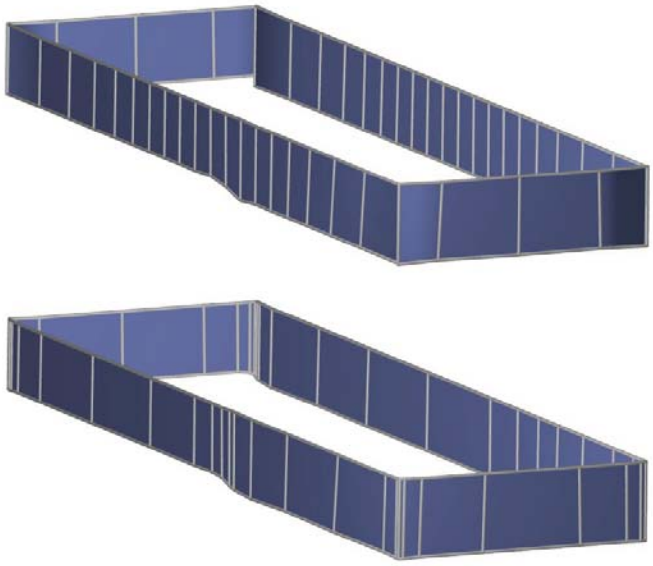

Figure 5.7: (Top) A ruled surface approximation without knot adaption. The highlighted control polygon distributes uniformly over the target shape. (Bottom) shows a ruled surface approximation with knot adaption. In planar regions, there are fewer control points. In regions with significant features such as the kink along the lower boundary or the corners exhibit additional control points.

5.5 Knot Adaption

So far, we have considered the B-spline surface's number of control points and definition of knot vectors fixed. In this section, we want to solve this last remaining initialization issue by adapting the knot vectors during approximation. As in our setup only the knot vector to parameter v is of interest, we will loosely speak of *the* knot vector below. Several methods as those summarized in (Dierckx, 1993) consider the entries of the knot vector as unknowns, along with the control points. We will not follow this approach as we are interested in changing the number of control points (basically the degrees of freedom). Our primary goal will be to remove control points in regions where this is feasible and add control points where this is necessary to achieve an approximation within given tolerance ε.

(Yang et al., 2004) examine knot adaption for B-spline curve fitting in a setting very similar to ours. The authors first approximate a given planar set of data points with a B-spline curve. Then, they undertake a single post-processing step that first inserts additional and then removes redundant control points. In either case, a modified set of control points is *locally* fitted to the target geometry to obtain the new set of control points. We will generalize the approach of (Yang et al., 2004) to the surface case. Instead of including a post-processing step, we will interleave knot *insertion* and knot *removal*

events with the iterations of the optimization, if the fitting's average approximation error is above ε after n initial iterations. Given that the approximating surface is a B-spline surface of degree $(1, d)$ with only 2 control points along the linear parameter, the parameter domain can be seen as a series of rectangular domains. Each domain corresponds to a tensor product Bézier surface segment. We compute the foot points of the shortest distance from the data points to the approximating surface and determine the average approximation error per domain. A knot will be inserted to surface segments with average error above ε. From *pairs of surface segments* with average error below a threshold $\varepsilon/2$, a knot will be removed. Both modifications will happen according to very simple rules. Given that the knot adaption step is embedded in the iterative minimization of the approximation error, introduced additional error will be optimized in the subsequent iteration.

If the average approximation error in an interval $[v_i, v_{i+1}]$ of the knot vector is larger than ε, a control point gets inserted and the knot interval splits into two new intervals $[v_i, v^*]$ and $[v^*, v_{i+1}]$. Knot insertion in combination with curve fitting is studied up to some extent, see for example the comparison in (Cham and Cipolla, 1999). These methods differ in the way v^* is chosen. The new knot is always inserted with Boehm's knot insertion algorithm (Boehm, 1980). We choose the new knot value to be the interval midpoint $v^* = (v_i + v_{i+1})/2$.

While knot insertion splits a knot interval into two, knot removal merges two adjacent knot intervals. Thus, we test the average error of pairs of surface segments for an approximation error less than $\varepsilon/2$. For general knot removal no equivalent to Boehm's knot insertion algorithm exists. From the $(d + 2)$ pairs of control points defining these two surface segments, one pair needs to be discarded. There has been some work considering knot removal and especially knot removal introducing as little additional error as possible, see (Eck and Hadenfeld, 1995) and the references therein. We simply delete the middle pair of the control points (assuming $d = 2k + 1$).

Please note, that by alternating knot adaption with the iterations of the optimization, we change the number of control points at a less local level than (Yang et al., 2004). In the iteration after a knot adaption step, all the control points of the approximating B-spline surface get optimized in any case. Fig. 5.7 visualizes the results of above knot adaption procedure for an ruled surface approximation. The highlighted control polygon shows how the number of control points gets reduced in planar regions while it increases in the corner regions.

6 Constrained Ruled Surface Approximation for Applications

In this chapter, we will extend the basic ruled surface approximation algorithm of the previous chapter to include application specific constraints. These constraints will draw their motivation from mainly two areas: production technologies and architecture. As different these two may appear at first sight, we will see that problems of architecture often resemble those of production technologies, simply at a larger scale.

Regarding production technologies, we will explore flank milling techniques with cylindrical cutting tools. If we recall that the basic ruled surface approximation yields a least-squares fitting of the target shape's offset, we avoid undercut errors (the removal of too much material) by including linear side conditions. Second, we will constrain the angle of the cutting tool with respect to its linearized motion. By requiring the cutter's top to advance its bottom, stress on the tool is minimized.

In contemporary architecture, the popularity of freeform surfaces not only fascinates clients and fans but also challenges and sometimes puzzles engineers. Often, a one-to-one realization of an architectural design is rendered impossible by limits of constructions or simply money. Hence, a simplification of the initial shape is common[1]. With ruled surfaces being a one-parameter family of straight lines, this class of surfaces has proved to be a suitable approximation entity for architecture, bearing favorable structural properties. Hence, we will investigate several applications of (strips of) ruled surfaces in architecture.

Given that we have studied literature related to flank milling and ruled surface approximation extensively in the previous chapter, we will omit a review of related work herein. The following discussion of flank milling and architectural applications will include further specific references if they have not been cited before.

6.1 Avoiding Undercut Errors

In its unconstrained formulation, the ruled surface approximation algorithm for cylindrical flank milling heads for a least-squares fitting of the target shape's offset. With respect to the target shape, we may identify two kinds of approximation errors. If too much material is removed, we speak of *undercut* errors. The opposite case, when too few material is cut away, is called *overcut* or *scallop* error (see Fig. 6.1). In applications, undercut errors are considered more harmful than overcut errors. Hence, this section will derive a method to avoid undercut at all, at the cost of larger overcut errors.

We will achieve undercut free approximations by constraining the ruled B-spline surface fitting optimization problem. Side conditions to curve and surface approximation have been discussed for different purposes in the past and we intend to give a short

[1] In architecture, this approximation is often called *rationalization*, which has nothing to do with the algebraic notion of a rational function.

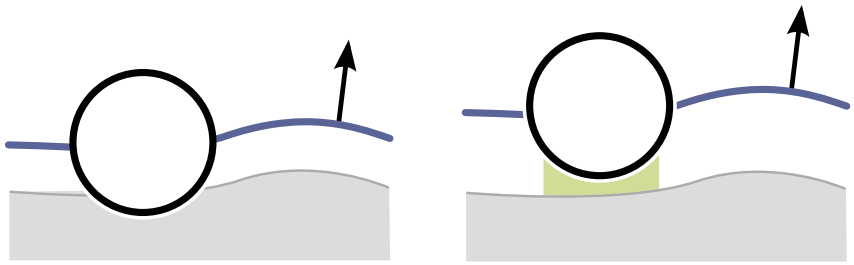

Figure 6.1: (Left) In case of undercut errors, the cutting tool penetrates the target shape and too much material is removed. (Right) The opposite case, when the cutting tool does not touch the intended target shape, causes overcut error. Residual material (depicted in yellow) remains. The trajectory of the tool's axis is included as blue curve.

summary in the following. Constraints may restrict the class of surfaces employed in the approximation, e.g. surface fitting with ruled or developable surfaces (see the literature discussion in the previous chapter). More generally, the constrained approximation of point clouds with Bézier or B-spline curves or surfaces (Rogers, 1989; Bercovier and Jacobi, 1994) may have various aims in minds such as increasing the quality of the final fitting, simplifying it or ensuring certain geometric properties of the solution. Especially the latter goal is addressed in detail by (Hoschek and Kaklis, 1996).

Below, we are going to consider the target point cloud as obstacle to the B-spline surface approximation. Side conditions have been imposed on interpolation methods with cubic splines and rational cubics to handle obstacles (Opfer and Oberle, 1988; Meek et al., 2003). (Myles and Peters, 2005) construct splines of prescribed smoothness obliged to stay in a channel with piecewise linear boundaries. (Lin and Wang, 2002) border planar point clouds by a B-spline curve interpolation of boundary points. The method sketched herein is discussed in more detail in (Flöry, 2009). Please note that in terms of general constrained matching, the following considerations are related closely to the penetration free registration of Chapter 2.

As before, we base our considerations on the signed distance function to a shape. The sign of the distance function was defined with respect to a closed shape's interior in Chapter 1. For an orientable surface patch, we provide the following alternative definition.

Definition. *Let* $\mathbf{x}(u, v)$ *be an orientable (open) surface patch and* $\mathbf{n}(u, v)$ *its oriented normal field. Then, the* signed distance *of a point* $\mathbf{p}$ *to* $\mathbf{x}$ *with foot point* $\mathbf{f}_p = \mathbf{x}(u_p, v_p)$ *and* $\mathbf{n}_p = \mathbf{n}(u_p, v_p)$ *is given by,*

$$d(\mathbf{x}, \mathbf{p}) = \operatorname{sgn}\left(\mathbf{n}_p^T \cdot (\mathbf{p} - \mathbf{f}_p)\right) \cdot \|\mathbf{p} - \mathbf{f}_p\|.$$

For flank milling applications, interior and exterior of the target shape are well defined and we let $\mathbf{n}(u, v)$ point away from the target shape (cf. Fig. 6.1). Undercut errors arise in situations when the approximating surface $\mathbf{x}$ does not exactly interpolate the target

Figure 6.2: (Left) Least-squares approximation (blue) of a target shape (yellow). (Right) undercut free and one-sided approximation of the same setup. The target shape has been triangulated for better visualization.

offset point cloud P. More precisely speaking, undercut errors imply that the signed distance to $\mathbf{x}$ of at least one element in P is positive.

Definition. $\mathbf{x}(u, v)$ *is called an* approximation without undercut error *of* P*, if* $d^*(\mathbf{x}, P)$ *is minimal and*

$$d(\mathbf{x}, \mathbf{p}) \leq 0 \qquad \forall \mathbf{p} \in P.$$

We see that an undercut free approximation yields a one-sided approximation of the target point cloud. Above requirement yields together with the ruled surface approximation objective of Equ. (5.2) a constrained non-linear minimization problem,

$$\begin{array}{rll} \min_\delta & f_d(\delta) + \lambda \cdot f_s(\delta) & \\ \text{subject to} & d(\mathbf{x}, \mathbf{p}) \leq 0 & \forall \mathbf{p} \in P. \end{array}$$

In a similar fashion to Sec. 2.3, we linearize these constraints in the foot points $\mathbf{x}(u_k, v_k)$ of $\mathbf{p}_k \in P$. We solve the constrained optimization problem in a sequence of quadratic programs, with linear constraints of the form

$$\mathbf{n}(u_k, v_k)^T \cdot (\mathbf{p}_k - \mathbf{x}(u_k, v_k)) \leq 0 \qquad \forall \mathbf{p}_k \in P.$$

With regard to the initial non-linear problem this may be seen as a variant of a *Sequential Quadratic Programming* method (Nocedal and Wright, 1999). In (Flöry, 2009), setup and convergence of this minimization are discussed in some detail. Fig. 5.1 and Fig. 6.2 show undercut free or one-sided approximations of point clouds.

6.2 Ruled Strip Models for Production Technologies

Before we continue our discussion of application specific constraints for flank milling, we want to consider the use of ruled strip models for production technologies. In flank

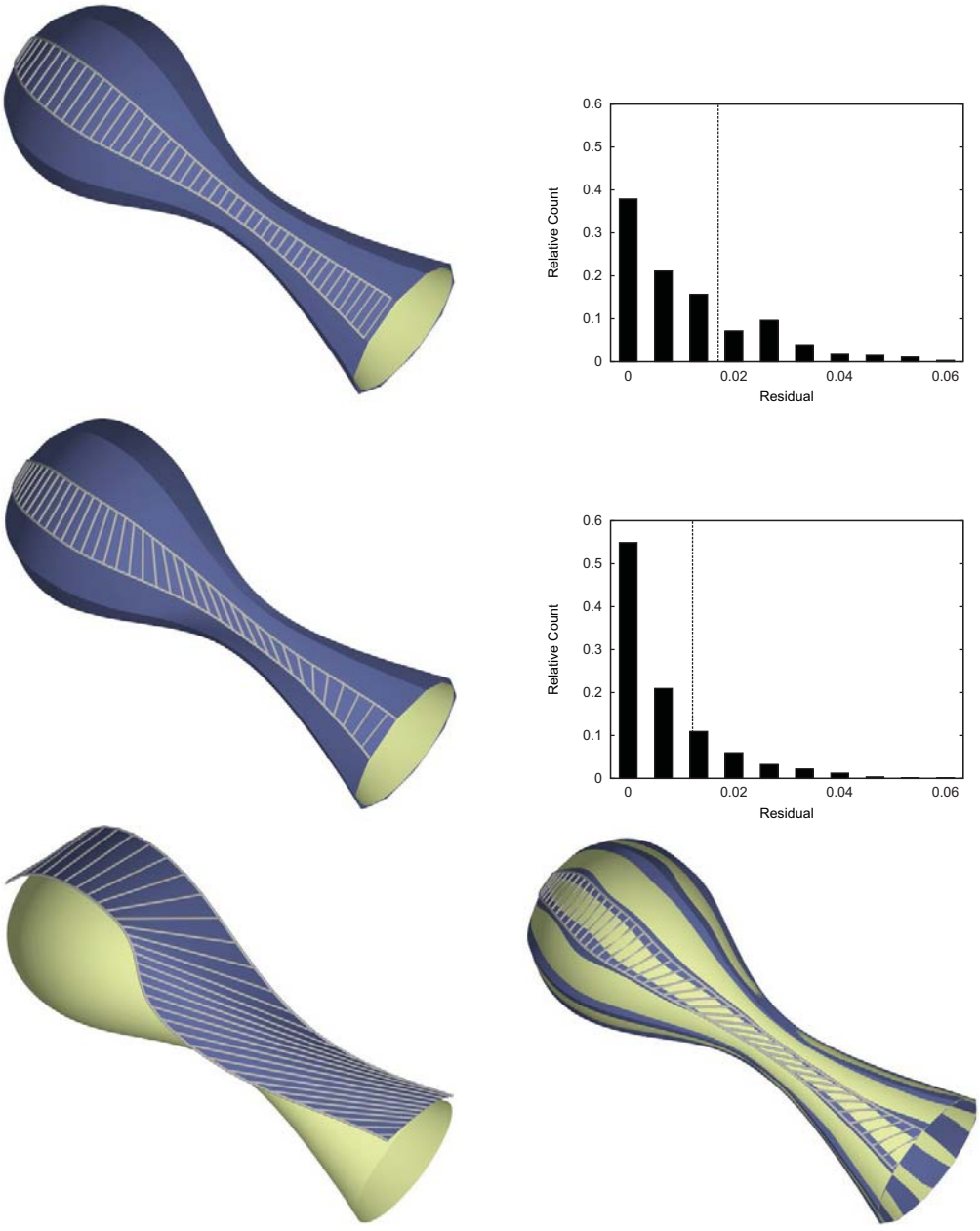

Figure 6.3: Strip models for production technologies: (Top, left) Simple reference approach with horizontal heated wire. (Center, left) A strip model approximation without undercut error. The wire configuration aligns well with the model's asymptotic lines in areas of negative Gaussian curvature. As the residual histograms (top and center, right) confirm the approximation error is improved. The bottom row shows an optimized single strip approximation and a least-squares strip model fitting.

milling, the mechanics and geometry of the milling process let strip models appear less suitable. However, in specific heated-wire foam cutting configurations strip models occur naturally. Recent developments increase the degrees of freedom of heated-wire cutters, for example by adding a rotary device on which the uncut block is placed (Step Four GmbH, 2007). In the following, we want to explore the possibilities of the so far described algorithms for a geometry processing for heated-wire cutting with rotary table.

The use of a rotary device leads directly to strip models. At each fixed angular position, the heated-wire cutter processes a ruled surface patch. The collection of these patches yields a strip model describing an approximation of the initial target shape. In Fig. 6.3, we intend to compare different methods of tool path computation for heated-wire cutting. For all the examples, the same initial model resembling the shape of a bowling pin was used. The spherical top part will not allow a ruled surface approximation of good quality whereas the middle and bottom regions with negative Gaussian curvature will be advantageous for this purpose. The target shape shall be approximated by 10 strips.

Our simple reference solution comprises consecutive horizontal cuts from top to bottom (Fig. 6.3, top). Unless stated otherwise, we require an undercut free approximation. For a first optimized solution, we employ a strip model in the form of Equ. (5.1) with $m_u = 10$, closed in u parameter direction. As expected, one-sided approximation yields a fitting of less residual error (Fig. 6.3, center). The rulings of the approximating surface align well with the asymptotic directions in regions with negative Gaussian curvature. If we leave behind the ruled strip model for a moment and consider a single strip, we may further constrain the approximation in a simple way to include further side conditions (Fig. 6.3, bottom left). The heated wire's end points are mounted and moved in vertical planar frames. The size of these frames limits the solution space of the approximating surface's rulings. Without loss of generality, let the frames' supporting planes ε_0 and ε_1 be parallel to the (x, y)-coordinate plane. If we choose the control points of the approximating ruled surface patch to be in either ε_0 or ε_1, we obtain for $i = 1, \ldots, m$,

$$\begin{aligned} (0,0,1)^T \cdot \delta_i &= 0 \\ x_{min} \leq (1,0,0)^T \cdot \mathbf{d}_i + \delta_i &\leq x_{max} \\ y_{min} \leq (0,1,0)^T \cdot \mathbf{d}_i + \delta_i &\leq y_{max}. \end{aligned}$$

These linear constraints may be added to the ruled surface approximation's objective without any further modification. If undercut errors are irrelevant, a least-squares approximation of the target shape by a ruled strip model yields an optimal result (Fig.6.3, bottom right). We will return to ruled strip models in our discussion of applications in architecture.

6.3 Leading Tool Top

In this section, we will consider the side condition that the cutter's top shall advance its bottom (see Fig. 6.4). We will call this behavior, termed *voreilen* in German, *leading* henceforth. A leading cylinder top induces a pulling motion of the cutting tool. The

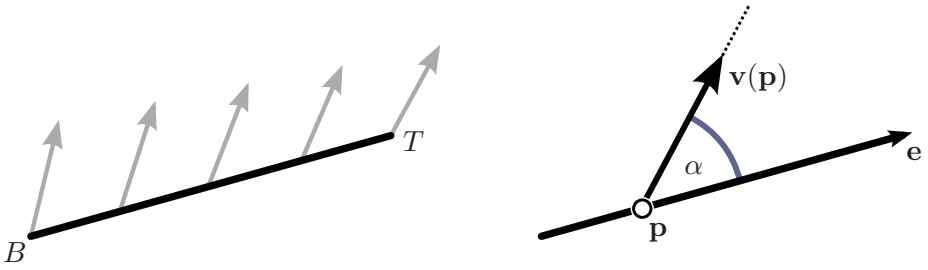

Figure 6.4: (Left) Linearized motion of the tool's axis. The top point T advances the bottom point B. (Right) The leading constraint is stated in terms of angle α spanned by velocity vector $\mathbf{v}(\mathbf{p})$ and ruling direction $\mathbf{e}$.

opposite, a pushing motion, would mean specific stress along the cutter that is imperative to avoid.

6.3.1 A Kinematic Constraint

Let $\hat{\mathbf{x}}(u,v)$, $(u,v) \in \hat{U} \subset \mathbb{R}^2$, describe a ruled surface patch as above with u parametrizing the generator. Moreover, let us assume that the striction curve, the locus of any possible singular points, is outside the considered patch. We define the parameter line for constant $u \equiv 0$, $\hat{\mathbf{x}}(0,v)$, to be the trajectory of the tool's axis bottom point B. We write $\mathbf{e}(v) = \hat{\mathbf{x}}(1,v) - \hat{\mathbf{x}}(0,v)$ for the unnormalized generator of $\hat{\mathbf{x}}$. With this notation in mind, we rewrite $\hat{\mathbf{x}}(u,v)$ as being generated by a one-parameter motion,

$$\mathbf{x}(u,v) = \hat{\mathbf{x}}(0,v) + u \cdot \frac{\mathbf{e}(v)}{\|\mathbf{e}(v)\|} = A(v)\frac{u \cdot \mathbf{e}(0)}{\|\mathbf{e}(0)\|} + \hat{\mathbf{x}}(0,v) \qquad (u,v) \in U \subset \mathbb{R}^2$$

with $A(v) \in SO_3$. Basically, we obtain $\mathbf{x}(u,v)$ from $\hat{\mathbf{x}}(u,v)$ through a reparametrization. If we discuss kinematic properties of points or rulings in the following, we do so with respect to above generating motion and $\mathbf{x}$.

For investigating the local behavior of a one-parameter motion we rely one more time on its first-order approximation (cf. Chapter 2). The velocity vector in a point coincides with the tangent of its trajectory,

$$\mathbf{v}(\mathbf{x}(u,v)) = \bar{\mathbf{c}}(v) + \mathbf{c}(v) \times \mathbf{x}(u,v) = \mathbf{x}_v(u,v).$$

and lets us define the leading constraint mathematically (see also Fig. 6.4).

Definition. *In a point of a ruled surface patch, let $\alpha(u,v) = \angle(\mathbf{v}(\mathbf{x}(u,v)), \mathbf{e}(v))$ denote the angle spanned by the velocity vector and the ruling's direction vector. Then, $\mathbf{x}(u,v)$ is said to be generated by a* leading *motion, if*

$$\alpha(u,v) \in [\frac{\pi}{2} - \tau, \frac{\pi}{2}] \qquad \forall (u,v) \in U$$

for user-defined τ.

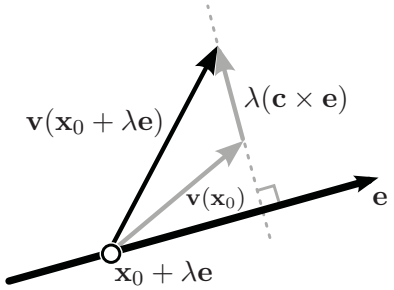

Figure 6.5: In any point $\mathbf{x}_0 + \lambda\mathbf{e}$ on ruling $\mathbf{e}$, the velocity vector is the sum of the velocity vector in $\mathbf{x}_0 \in \mathbf{e}$ and a vector $\lambda(\mathbf{c} \times \mathbf{e})$ orthogonal to $\mathbf{e}$.

The upper bound can be rewritten as,

$$\mathbf{e}(v)^T \cdot \mathbf{v}(\mathbf{x}(u, v)) \geq 0 \qquad \forall (u, v) \in U. \tag{6.1}$$

The lower bound — with τ typically ranging from $10°$ to $20°$ — reads,

$$\cos\alpha = \frac{\mathbf{e}^T \cdot \mathbf{v}(\mathbf{x})}{\|\mathbf{e}\|\|\mathbf{v}(\mathbf{x})\|} \leq \cos(\frac{\pi}{2} - \tau) = \sin(\tau) \tag{6.2}$$

6.3.2 Reducing the Number of Constraints

The inequality constraints of Equ. (6.1) and Equ. (6.2) must be fulfilled in any point of the approximating ruled surface patch. In this section we seek to reduce the set of points in which the side conditions must be checked.

Assume parameter v fixed. For the velocity vector $\mathbf{v}(\mathbf{x}_0 + \lambda\mathbf{e})$ of a point on ruling $\mathbf{e} = \mathbf{e}(v)$ through $\mathbf{x}_0 = \mathbf{x}(0, v)$, we see that (cf. Fig. 6.5),

$$\mathbf{v}(\mathbf{x}_0 + \lambda\mathbf{e}) = \bar{\mathbf{c}} + \mathbf{c} \times \mathbf{x}_0 + \lambda(\mathbf{c} \times \mathbf{e}) = \mathbf{v}(\mathbf{x}_0) + \lambda(\mathbf{c} \times \mathbf{e}).$$

Please note that the parameters of the velocity vector field $(\bar{\mathbf{c}}, \mathbf{c})$ depend only on v and are thus constant along $\mathbf{e}$. Hence, the velocity vector in any point on $\mathbf{e}$ is the sum of the velocity vector of $\mathbf{x}_0$ and a vector orthogonal to $\mathbf{e}$. Moreover, $\alpha(u, v) = 0$ implies $\mathbf{e}$ and $\mathbf{v}(\mathbf{x}_0 + \lambda\mathbf{e})$ linearly dependent, which may only be the case in singular points of the ruled surface. Consequently, $\alpha(u, v)$ changes sign only in points of the striction curve, if at all. These observations lead directly to the following results.

Lemma. *Let $\mathbf{x}(u, v)$, $(u, v) \in U \subset \mathbb{R}^2$, be a ruled surface patch as above. Then, the following two properties hold for the velocity vectors $\mathbf{v}(\mathbf{x}_0 + \lambda\mathbf{e})$ of points on a patch's generator $\mathbf{e}$ through $\mathbf{x}_0$.*

- *The projection of $\mathbf{v}(\mathbf{x}(u, v))$ onto $\mathbf{e}(v)$ is constant, $\mathbf{e}^T \cdot \mathbf{v}(\mathbf{x}_0 + \lambda\mathbf{e}) = \mathbf{e}^T \cdot \mathbf{v}(\mathbf{x}_0)$.*
- *Angle $\alpha(u, v)$ is monotonic for increasing λ.*

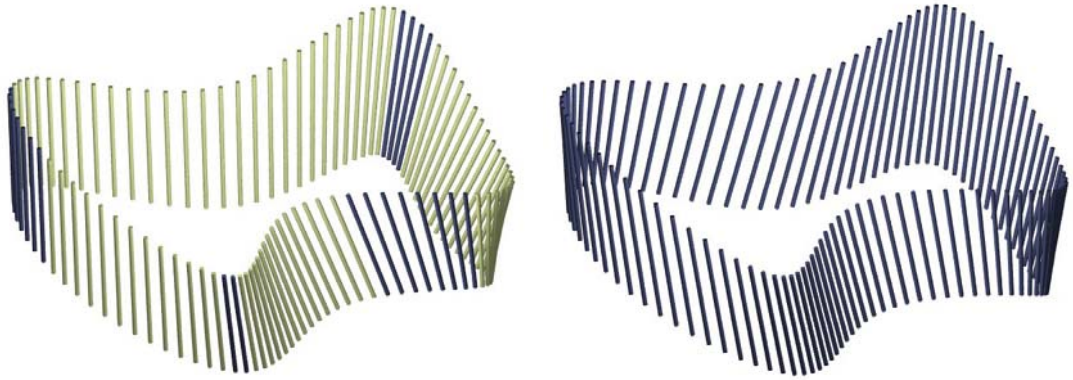

Figure 6.6: As opposed to an unconstrained optimization (left) a tool path respecting leading constraints (right) reduces stress along the cutting tool. Rulings violating the constraints are colored in yellow, feasible rulings are shown in blue.

We conclude for the first constraint of Equ. (6.1), that if it is fulfilled in one point on $\mathbf{e}$, it is fulfilled in all points on $\mathbf{e}$. Thus, we have to include this constraint only once per generator and we will do so for the bottom point $\mathbf{x}_0$. The monotonic behavior of $\alpha(u, v)$ requires the second constraint in Equ. (6.2) to be checked only in the bottom and top point of the cutting tool's axis. In summary we see that we may reduce the set of constraints to three per ruling.

6.3.3 Discretizing and Linearizing

Combining the leading constraints with the ruled surface approximation objective we obtain another non-linear constrained optimization problem. Again, we solve it with a Sequential Quadratic Programming method. We discretize $\mathbf{x}(u, v)$ in N rulings and obtain a finite set of $3 \cdot N$ constraints. Recalling that the unknowns of the optimization problem are the control points' displacements δ, we write

$$L^\delta(u, v) = \mathbf{e}^\delta(v) \cdot \mathbf{v}^\delta(\mathbf{x}^\delta(u, v)) = \mathbf{e}^\delta(v) \cdot \mathbf{x}_v^\delta(0, v)$$

for the left-hand side of Equ. (6.1). According to above lemma, $L^\delta(u, v)$ does not depend on parameter u and we omit it in the following. First-order Taylor approximation of $L^\delta(v)$ with respect to λ yields upper bound constraints of the form,

$$\nabla L^T(v)|_{\delta=0} \cdot \delta \geq -L^0(v),$$

linear in δ.

For the lower bound constraints in Equ. (6.2), we have to deal additionally with the product of norms in the denominator. We will ignore the dependency of these norms on

Figure 6.7: Ruled surfaces in architecture: An electricity pylon by Vladimir Shukhov, the roof of the Escoles Sagrada Família by Antoni Gaudí and the Philips Pavilion by Le Corbusier and Iannis Xenakis (from left to right, photo credits: Igor Kazus, Carlos Martínez, (Jencks, 2000)).

the changing control points and we obtain an incomplete linearization of the constraints only. We arrive at,

$$L^{\delta}(v) \leq \sin(\tau)\|\mathbf{e}(v)\|\|\mathbf{v}(\mathbf{x}(0,v))\|$$
$$L^{\delta}(v) \leq \sin(\tau)\|\mathbf{e}(v)\|\|\mathbf{v}(\mathbf{x}(1,v))\|$$

We see that we only need to include one of the two constraints, namely that with minimal right-hand side. This observation gives the final expression for the second constraint,

$$\nabla L^{T}(v)|_{\delta=0} \cdot \delta \leq -L^{0}(v) + \sin(\tau) \min_{u=0,1} \|\mathbf{v}(\mathbf{x}(u,v))\|.$$

which is linear in δ.

Fig. 6.6 shows both an unconstrained and a leading ruled surface approximation of a target shape. We show the fitting surface's rulings color-coded. Rulings violating the leading constraint are depicted in yellow, feasible rulings are visualized in blue.

6.4 Ruled Surfaces in Architecture

We have seen in Sec. 6.2 that multiple ruled surface patches arranged in a strip pattern are well suitable to approximate complex objects. In the remainder of this chapter, we will further explore these so-called *ruled surface strip models* with a special emphasis on applications in architecture. In particular, we are going to address smoothness issues along the common boundaries of adjacent strips, where such models are only C^0 continuous in general.

From a geometric point of view, ruled strip models are of a hybrid nature. Consider a strip model as a two parametric surface $\mathbf{x}(u,v)$ with parameter u describing the ruled direction (cf. Equ. (5.1)). Parameter curves for constant v are polygonal lines and hence discrete. Contrary, lines for constant u parameter are smooth curves that in the case of

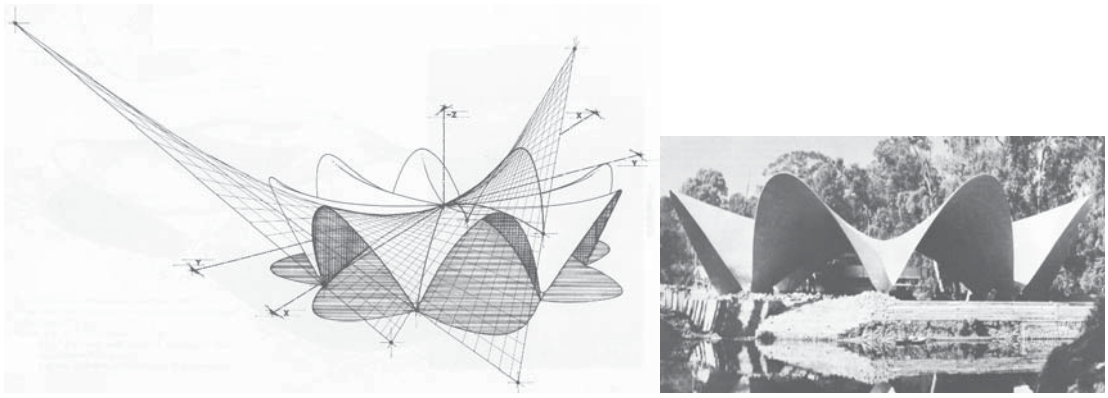

Figure 6.8: Ruled surfaces in architecture: Illustrative drawing (left) and photo (right) of a restaurant in Xochimilco, Mexico, by Félix Candela (from Faber, 1963).

B-spline curves may be seen as result of a subdivision process from a coarse initial discrete control polygon. Consequently, we may regard ruled strip models as *semi-discrete* surface representations that have found some attention in literature recently (Pottmann et al., 2008). Neglecting any smoothness considerations along common boundaries, work previously cited from CAD and production technologies (Elber and Fish, 1997) or literature investigating the making of papercraft models (Mitani and Suzuki, 2004) comprise strip model constructions.

Ruled surfaces have been part of the architectural discourse for more than one hundred years (see Fig. 6.7 and Fig. 6.8). Towards the end of the 19^{th} century, Russian engineer Vladimir Shukhov and Spanish Catalan architect Antoni Gaudí independently discovered the structural advantages of ruled surfaces for their works. Since then, numerous architects employed ruled surfaces, among them popular names such as Le Corbusier, Félix Candela or Frank Gehry. Despite the success of general doubly curved freeform architecture, ruled surfaces remain topic of discussion in contemporary architectural theory (Vollers, 2001; Kolarevic, 2003). In some way, our work bridges the structural unaware point of view of general doubly curved designs with the constructional elegance of ruled surfaces.

In the following, we are going to present three approaches towards a smooth appearance of strip models. First, we smooth the discrete parameter lines of ruled strip models. Second, we optimize for coinciding tangent planes and G^1 continuity along the common boundary of adjacent strips. Finally, we discuss a model for G^2 continuous ruled surfaces strips.

6.5 Smoothing Parameter Lines

The ruled surface approximation's objective of Equ. (5.2) combines terms describing fitting error and fairness of the solution. The latter is necessary for a satisfying visual appearance of the final result. In this section, we want to increase our control on the final

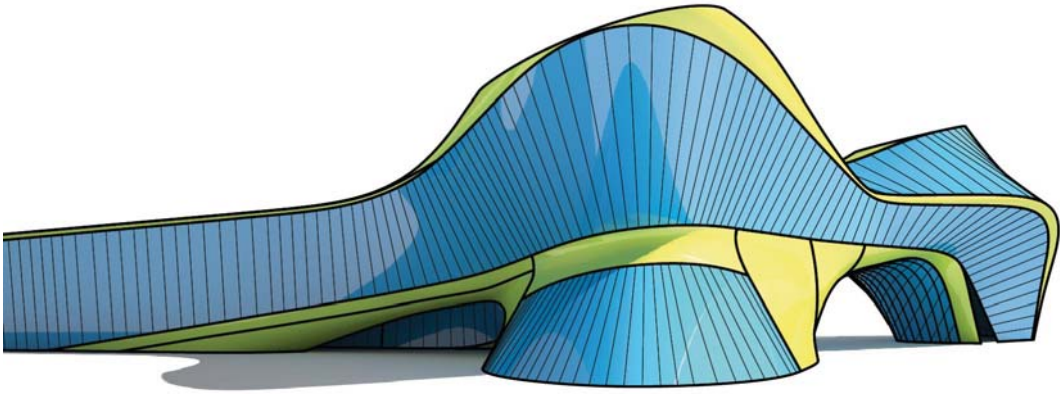

Figure 6.9: Ruled surface approximation of parts of a facade by Zaha Hadid Architects for the Cagliari Contemporary Arts Center in Sardinia. The reconstructed ruled surface patches are depicted in blue, general doubly curved blends are shown in yellow.

shape of the approximating surface, in particular on the appearance of surface curves. Let $w(t) : t \to (u, v)$ denote a mapping from the real numbers to the parameter space $[0, 1] \times [0, 1]$ of a ruled strip model $\mathbf{x}(u, v)$. For w fixed, we will investigate two special types of curves $t \to \mathbf{x}(w(t))$ on $\mathbf{x}$,

- $w(t) : t \to (u(t), \text{const})$, with $\dot{u}(t) \equiv 1$, so-called *u-parameter lines* or *polygonal ruling lines* to constant v parameter.
- $w(t) : t \to (\text{const}, v(t))$, with $\dot{v}(t) \equiv 1$, so-called *v-parameter lines*.

Please note that dotted expressions such as $\dot{v}$ denote derivatives with respect to parameter t.

The topic of curve regularization has been addressed numerous times. Given that the ruled surface approximation draws its motivation from active shape models, we will employ methods from the active contour literature (Blake and Isard, 1998). As part of an active contour's internal energy, fairness terms for length and bending of the curve are minimized. These read in our notation with regularization weights λ_i,

$$\lambda_0 \int_t \|\dot{\mathbf{x}}(w(t))\|^2 \, \mathrm{d}t + \lambda_1 \int_t \|\ddot{\mathbf{x}}(w(t))\|^2 \, \mathrm{d}t. \tag{6.3}$$

Let us consider v-parameter lines first, that is $w(t) = (\text{const}, v(t))$ with $\dot{v}(t) \equiv 1$. Observing that $\dot{\mathbf{x}}(w(t)) = \mathbf{x}_v(w(t))$ and $\ddot{\mathbf{x}}(w(t)) = \mathbf{x}_{vv}(w(t))$ discretization yields,

$$\lambda_0 \sum_{k=1}^{n} \left\| \mathbf{x}_v^\delta(\text{const}, v_k) \right\|^2 + \lambda_1 \sum_{k=1}^{n} \left\| \mathbf{x}_{vv}^\delta(\text{const}, v_k) \right\|^2 .$$

Figure 6.10: Smoothing of polygonal rulings lines (right) yields a visually more pleasing ruled surface strip reconstruction than an unconstrained approximation (left).

Here, δ describes the vector of unknown displacements of the B-spline surface's control points. We see, that this expression is quadratic in the unknowns δ. It is added without further modification as additional penalty function to the approximation's objective.

For a u-parameter line (a polygonal ruling line), the situation is slightly different. In common strip boundary points $\mathbf{s}_k^\delta = \mathbf{x}^\delta(\frac{k-1}{m_u-1}, \text{const})$, $k = 2, \ldots, m_u - 1$ the surface and in particular the u-parameter lines are not differentiable. Instead of minimizing length and bending energies of a smooth curve we target a fairing of the discrete curve $(\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_{m_u})$. Following above argumentation, this is directly achieved by replacing the derivatives in Equ. (6.3) with finite differences and discretizing integrals,

$$\lambda_0 \sum_{k=1}^{m_u-1} \left\| \mathbf{s}_{k+1}^\delta - \mathbf{s}_k^\delta \right\|^2 + \lambda_1 \sum_{k=2}^{m_u-1} \left\| \mathbf{s}_{k+1}^\delta - 2\mathbf{s}_k^\delta + \mathbf{s}_{k-1}^\delta \right\|^2 .$$

Again, we obtain fairness terms quadratic in the unknowns δ that we add to the objective in Equ. (5.2).

For construction, rulings and polygonal ruling lines are often realized as visible structural elements and hence their appearance is crucial. Above regularization terms decouple specific parameter lines from the general smoothing term and let us optimize the shape of specific surface curves more exactly. In Fig. 6.10 and Fig. 6.11, these regularizations have been applied to architectural models.

6.6 G^1 continuous Strip Models

In this section we will enhance the ruled surface approximation by an additional penalty term to obtain a higher degree of smoothness in common strip points. Recall, that the ruled strip model is in general only C^0 continuous in $\mathbf{s}_k(v) = \mathbf{x}(u_k, v)$ for $u_k = \frac{k-1}{m_u-1}$, $k = 2, \ldots, m_u - 1$.

The concept of *parametric* continuity is too restrictive to describe the geometric requirements for smoothness of two joining surfaces, as it depends on the surfaces' parametrizations. Two surfaces are said to meet C^k (parametric) continuous in a common boundary point if the first $i = 0, \ldots, k$ derivatives coincide. For applications in geometry processing, the notion of *geometric* continuity is better suited. Two surfaces

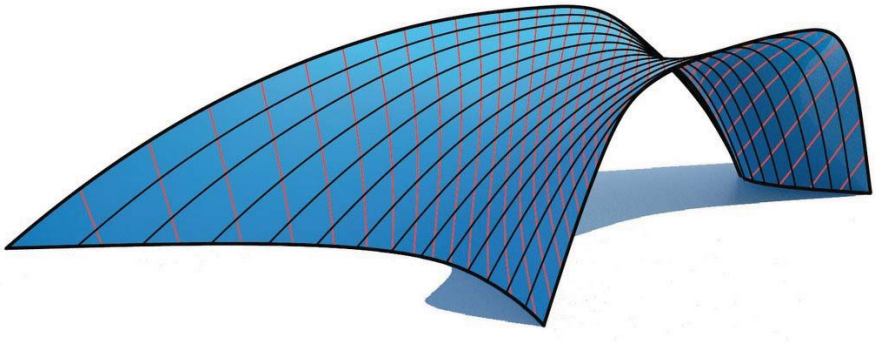

Figure 6.11: Detail of the facade of the Cagliari Contemporary Arts Center in Sardinia by Zaha Hadid Architects. The initial surface was approximated by a strip model comprising 14 strips. The optimization included penalties for G^1 continuity and smooth polygonal ruling lines.

join G^k geometric continuous in a point if there exists a reparametrization such that the surfaces agree C^k parameter continuous. Many specific notions of geometric continuity have found there way into the literature, see for example (Hoschek and Lasser, 1993; Peters, 2002) and the references therein. G^1 continuity is equivalent to matching tangent planes (Veron et al., 1976) whereas coinciding osculating paraboloids imply G^2 continuous surfaces (Veron et al., 1976; Herron, 1987). For ruled surfaces in particular (Schaaf and Ravani, 1998) derived a theory of higher-order contact. In CAD, conditions for G^1 continuous B-spline and NURBS surfaces are still actively studied (Shi et al., 2004; Che et al., 2005). Related to our active shape setup are contributions like that of (Milroy et al., 1995) who minimize the deviation of surface normals along common strip boundaries in a non-linear least-squares optimization.

Consider a common boundary point of two strips, $\mathbf{s}_k = \mathbf{x}(u_k, v_0)$, u_k as above and k and v_0 fixed (cf. Fig. 6.12, left). In above strip model, the partial derivative $\mathbf{x}_v$ in $\mathbf{s}_k$ coincides with the tangent to the common boundary curve $\mathbf{x}(u_k, v)$. Moreover, let $\mathbf{x}_u^-$ and $\mathbf{x}_u^+$ be the left and right-hand side derivatives of $\mathbf{x}$ in $\mathbf{s}_k$ with respect to u. Please note that $\mathbf{x}_u^-$ and $\mathbf{x}(u_k, v_0) - \mathbf{x}(u_{k-1}, v_0)$ are linearly dependent, a similar relation holds for $\mathbf{x}_u^+$. G^1 continuity in $\mathbf{s}_k$ requires the left-hand side tangent plane T_- spanned by $\mathbf{x}_u^-$ and $\mathbf{x}_v$ and the right-hand side tangent plane T_+ spanned by $\mathbf{x}_u^+$ and $\mathbf{x}_v$ to coincide. Hence, the condition that the three partial derivatives $\mathbf{x}_v$, $\mathbf{x}_u^-$ and $\mathbf{x}_u^+$ lie in a common plane and are thus linearly dependent,

$$\det(\mathbf{x}_v, \mathbf{x}_u^-, \mathbf{x}_u^+) = \mathbf{x}_v^T \cdot (\mathbf{x}_u^- \times \mathbf{x}_u^+) = 0$$

is necessary for G^1 continuity. Vanishing of the determinant is not equivalent to G^1 continuity as the tangent planes could flip in $\mathbf{s}_k$. However, the regularization term of the ruled surface approximation will prevent from such cases. Given that all three vectors depend on the displacements δ of the active surface model, we optimize for G^1 continuity

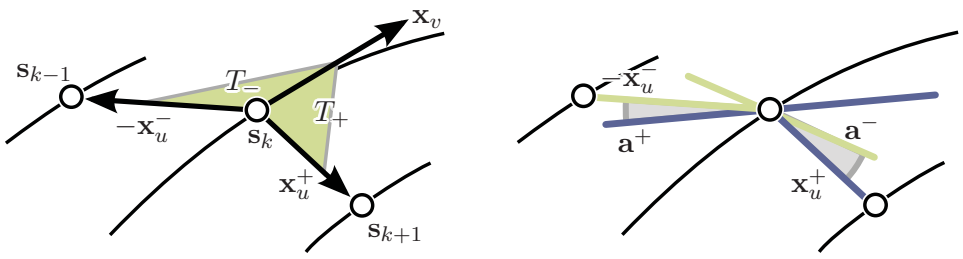

Figure 6.12: For G^1 continuity in $\mathbf{s}_k$, tangent planes T_- and T_+ need to coincide (left). For a G^2 continuous ruled strip model, the asymptotic direction $\mathbf{a}_-$ of the left strip is required to be linearly dependent to the ruling direction of the right strip and vice versa (right).

by adding the weighted penalty

$$f_{G1}(\delta) = \lambda \left[\frac{\mathbf{x}_v^T(\delta)}{\|\mathbf{x}_v(\delta)\|} \cdot \frac{\mathbf{x}_u^-(\delta) \times \mathbf{x}_u^+(\delta)}{\|\mathbf{x}_u^-(\delta) \times \mathbf{x}_u^+(\delta)\|} \right]^2 \tag{6.4}$$

to the approximation's objective. Linearization of the dot product yields along with the point to tangent plane error term a Gauss-Newton iteration. A closer look at the penalty leads to the following two observations.

Consider the crossed quadrilateral Q spanned by the origin O and the points $\mathbf{x}_v$, $\mathbf{x}_u^-$ and $\mathbf{x}_v + \mathbf{x}_u^+$. The diagonals of Q are $O + s \cdot \mathbf{x}_u^-$ and $\mathbf{x}_v + t \cdot \mathbf{x}_u^+$. The distance of the diagonals is given by $\mathbf{x}_v^T \cdot \left(\frac{\mathbf{x}_u^- \times \mathbf{x}_u^+}{\|\mathbf{x}_u^- \times \mathbf{x}_u^+\|} \right)$. We see that minimization of above determinant (basically the volume of the parallelepiped spanned by three tangent vectors) is closely related to minimizing the diagonal distance. The latter measure is used in (Pottmann et al., 2008) to reduce the tangent plane twist along a ruled surface's generator to obtain approximate developability.

In a second note, a simpler iterative approach could fix the right-hand side tangent plane T_+ (with normal $\mathbf{n}_+ = \mathbf{x}_u^+ \times \mathbf{x}_v$) and minimize the deviation of the left-hand side tangent to T_+,

$$\left[\mathbf{n}_+^T \cdot \mathbf{x}_u^-(\delta) \right]^2 .$$

The same is done for T_- and variable right-hand side tangent. Let us consider the first case only and omit norms of vectors for a moment to ease the discussion. Differentiating the dot product in Equ. (6.4) yields,

$$\nabla f_{G1}(\delta) = D(\mathbf{x}_u^+(\delta) \times \mathbf{x}_v(\delta)) \cdot \mathbf{x}_u^-(\delta) + (\mathbf{x}_u^+(\delta) \times \mathbf{x}_v(\delta)) \cdot D\mathbf{x}_u^-(\delta).$$

Here, D denotes the Jacobian of an expression. For a first-order Taylor approximation of f_{G1} we obtain,

$$f_{G1}(\delta) \approx \mathbf{n}_+^T \cdot \mathbf{x}_u^-(\mathbf{0}) + \left(\mathbf{n}_+^T \cdot D\mathbf{x}_u^- \right)^T \cdot \delta + o(\|\delta\|) = \mathbf{n}_+^T \cdot \mathbf{x}_u^-(\delta) + o(\|\delta\|).$$

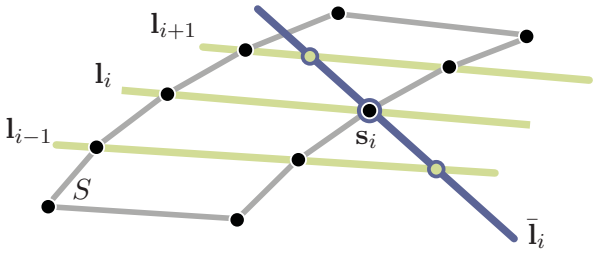

Figure 6.13: Discrete model of G^2 continuous ruled surface strips. The subsequent ruling direction in $\mathbf{s}_i$ is given by that line $\bar{\mathbf{l}}_i$ intersecting pairwise skew $\mathbf{l}_{i-1}$, $\mathbf{l}_i$ and $\mathbf{l}_{i+1}$.

Hence, minimization of the squared determinant comprises the terms of the simpler iterative approach fixing tangent planes.

Fig. 6.11 shows a ruled strip model approximation with minimized tangent plane deviation along common strip boundaries as part of a design by Zaha Hadid Architects.

6.7 G^2 continuous Strip Models

For G^2 continuity in a common boundary point $\mathbf{s}_k$, we require the osculating paraboloids of adjacent strips to coincide. The osculating paraboloid is uniquely determined by three pairwise different tangential directions and the normal curvatures therein. Given that the strip model is G^1 continuous in $\mathbf{s}_k$, the two adjacent strips share common normal curvature in common boundary tangent direction $\mathbf{x}_v$. We obtain another tangent plus normal curvature for each strip: the ruling directions of normal curvature zero.

Let $\mathbf{a}^-$ be the second asymptotic direction of the left-hand strip ($\mathbf{x}_u^- \times \mathbf{a}^- \neq \mathbf{0}$) and let $\mathbf{a}^+$ be defined accordingly (see Fig. 6.12, right). If the asymptotic directions of the left-hand side strip are pairwise linearly dependent to those of the right-side strip, they define along with $\mathbf{x}_v$ in $\mathbf{s}_k$ the common osculating paraboloid of the two joining strips. Note, that if the ruling directions $\mathbf{x}_u^-$ and $\mathbf{x}_u^+$ are linearly dependent, we obtain a single ruled surface patch. As this would undo the benefits of a strip model, we require that the ruling direction of one strip coincides with the general asymptotic direction on the other strip. The rulings will form a zig-zag pattern if we repeat this construction over several strips (cf. Fig. 6.14, left).

The linear dependency of asymptotic directions motivates the weighted penalty,

$$f_{G2}(\delta) = \lambda \left[\left(\frac{\mathbf{x}_u^-(\delta)}{\|\mathbf{x}_u^-(\delta)\|} \times \frac{\mathbf{a}^+(\delta)}{\|\mathbf{a}^+(\delta)\|} \right)^2 + \left(\frac{\mathbf{x}_u^+(\delta)}{\|\mathbf{x}_u^+(\delta)\|} \times \frac{\mathbf{a}^-(\delta)}{\|\mathbf{a}^-(\delta)\|} \right)^2 \right],$$

for a ruled surface approximation with optimized G^2 continuity. Numerical experiments for $\mathbf{a}^+$ and $\mathbf{a}^-$ fixed over an iteration indicate, that the degrees of freedom of G^2 continuous strip models are restricted. In order to evaluate this further, we will explore the design possibilities in a discrete model of G^2 continuous ruled strip models.

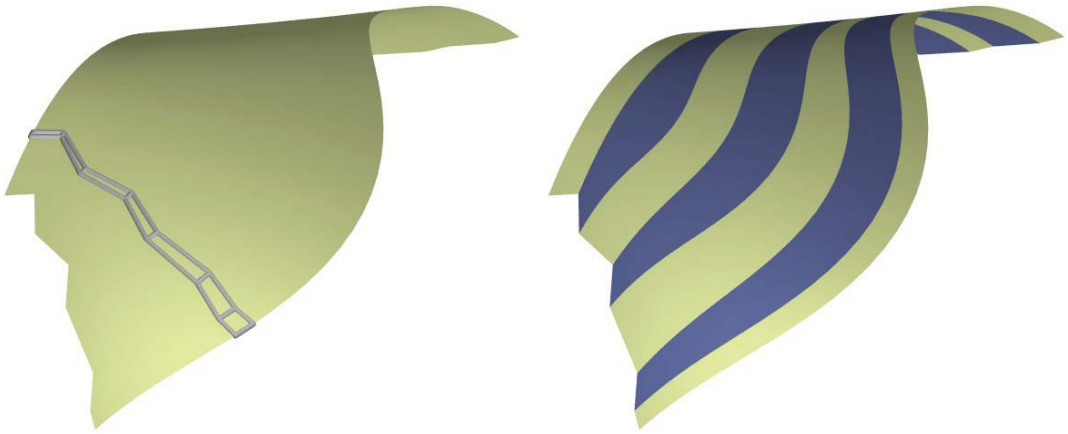

Figure 6.14: Discrete G^2 continuous ruled strip model: (left) the rulings follow a zig-zag pattern, (right) color-coded strips.

Consider the discrete model of a ruled surface strip S in Fig. 6.13. Let $\mathbf{l}_i$ denote the supporting line of a ruling incident to $\mathbf{s}_i$. The ruling direction of subsequent strip T in $\mathbf{s}_i$ shall be a discrete analogue to the asymptotic direction on a smooth ruled surface. For a C^2 continuous space curve $\mathbf{c}$, the osculating plane in a point $\mathbf{c}(u)$ is given as limit of the plane spanned by points $\mathbf{c}(u-h)$, $\mathbf{c}(u)$ and $\mathbf{c}(u+h)$ for $h \to 0$. For three rulings $\mathbf{l}(u-h)$, $\mathbf{l}(u)$, $\mathbf{l}(u+h)$ of a smooth ruled surface $\mathbf{x}(u,v)$, a similar construction yields an osculating quadric (Hoschek, 1971).

The set of lines intersecting three pairwise skew lines $\mathbf{l}(u-h)$, $\mathbf{l}(u)$, $\mathbf{l}(u+h)$ is called a regulus $\mathcal{R}(h)$ (Pottmann and Wallner, 2001). Any regulus lies on a quadric carrying a complementary regulus $\mathcal{R}'(h)$. For $h \to 0$ we obtain a limit regulus $\mathcal{R}$ whose carrier quadric is called *osculating quadric* or *Lie quadric*. The Lie quadric has second-order contact with $\mathbf{x}$ along $\mathbf{l}(u)$. Dupin's Indicatrices and thus the asymptotic directions of both surfaces coincide. As the quadric's asymptotic directions are its generators, the elements of $\mathcal{R}$ are the asymptotic directions of $\mathbf{x}$ along $\mathbf{l}(u)$. Hence, in the discrete model, we choose the line passing though $\mathbf{s}_i \in \mathbf{l}_i$ and intersecting $\mathbf{l}_{i-1}$, $\mathbf{l}_{i+1}$ as subsequent generator $\bar{\mathbf{l}}_i$.

The discrete model allows us to start from a single discrete ruled surface strip and enhance it strip by strip according to above rule. Interactive experiments further indicate that the design possibilities with G^2 continuous ruled strip models are limited. Fig. 6.14 shows a discrete model of osculating ruled surface strips, revealing the expected zig-zag pattern of ruling segments.

Bibliography

Aiger, D., Mitra, N. J., Cohen-Or, D., 2008. 4-points congruent sets for robust pairwise surface registration. ACM Trans. Graph. 27 (3), 85:1–85:10.

Alizadeh, F., Goldfarb, D., 2003. Second-order cone programming. Math. Program. 95 (1), 3–51.

Alliez, P., Cohen-Steiner, D., Tong, Y., Desbrun, M., 2007. Voronoi-based variational reconstruction of unoriented point sets. In: Proc. SGP 2007. pp. 39–48.

Anguelov, D., Srinivasan, P., Koller, D., Thrun, S., Rodgers, J., Davis, J., 2005. Scape: shape completion and animation of people. ACM Trans. Graph. 24 (3), 408–416.

Ankerst, M., Kastenmüller, G., Kriegel, H.-P., Seidl, T., 1999. 3D shape histograms for similarity search and classification in spatial databases. In: Proc. SSD 1999. pp. 207–226.

Ballard, D. H., 1981. Generalizing the Hough transform to detect arbitrary shapes. Pattern Recognition 13 (2), 111–122.

Barnea, E. I., Silverman, H., 1972. A class of algorithms for fast digital image registration. IEEE Trans. on Comp. C-21 (2), 179–186.

Beauchemin, S. S., Barron, J. L., 1995. The computation of optical flow. ACM Comput. Surv. 27 (3), 433–466.

Bedi, S., Mann, S., Menzel, C., 2003. Flank milling with flat end milling cutters. Computer Aided Design 35 (3), 293 – 300.

Belongie, S., Malik, J., Puzicha, J., 2002. Shape matching and object recognition using shape contexts. IEEE Trans. Pattern Anal. Mach. Intell. 24 (4), 509–522.

Bercovier, M., Jacobi, A., 1994. Minimization, constraints and composite Bézier curves. Computer Aided Geometric Design 11 (5), 533–563.

Bergevin, R., Soucy, M., Gagnon, H., Laurendeau, D., 1996. Towards a general multi-view registration technique. IEEE Trans. Pattern Anal. Mach. Intell. 18 (5), 540–547.

Bernardini, F., Rushmeier, H. E., 2002. The 3D model acquisition pipeline. Comp. Graph. Forum 21 (2), 149–172.

Besl, P. J., McKay, N. D., 1992. A method for registration of 3D shapes. IEEE Trans. Pattern Anal. Mach. Intell. 14 (2), 239–256.

Biasotti, S., Marini, S., Mortara, M., Patanè, G., 2003. An overview on properties and efficacy of topological skeletons in shape modelling. In: Shape Modeling International. pp. 245–256.

Björck, Å., 1996. Numerical Methods for Least Squares Problems. SIAM.

Blake, A., Isard, M., 1998. Active Contours. Springer.

Boehm, W., 1980. Inserting new knots into B-spline curves. Computer Aided Design 12 (4), 199–201.

Botsch, M., Pauly, M., Gross, M., Kobbelt, L., 2006. PriMo: coupled prisms for intuitive surface modeling. In: Proc. SGP 2006. pp. 11–20.

Boyd, S., Vandenberghe, L., 2004. Convex Optimization. Cambridge University Press.

Bradley, D., Boubekeur, T., Heidrich, W., 2008. Accurate multi-view reconstruction using robust binocular stereo and surface meshing. In: Proc. CVPR 2008. pp. 1–8.

Broek, J. J., Horváth, I., de Smit, B., Lennings, A. F., Rusák, Z., Vergeest, J. S. M., 2002. Free-form thick layer object manufacturing technology for large-sized physical models. Automation in Construction 11, 335–347.

Brown, B., Rusinkiewicz, S., 2007. Global non-rigid alignment of 3D scans. ACM Trans. Graph. 26 (3).

Campbell, R. J., Flynn, P. J., 2001. A survey of free-form object representation and recognition techniques. Comput. Vis. Image Underst. 81 (2), 166–210.

Cham, T.-J., Cipolla, R., 1999. Automated B-spline curve representation incorporating MDL and error-minimizing control point insertion strategies. IEEE Trans. Pattern Anal. Mach. Intell. 21 (1), 49–53.

Chan, T. F., Esedoglu, S., 2004. Aspects of total variation regularized l_1 function approximation. SIAM Journal on Applied Mathematics 65 (5), 1817–1837.

Che, X., Liang, X., Li, Q., 2005. G1 continuity conditions of adjacent NURBS surfaces. Computer Aided Geometric Design 22 (4), 285–298.

Chen, H.-Y., Pottmann, H., 1999. Approximation by ruled surfaces. J. Comput. Appl. Math. 102 (1), 143–156.

Chen, Y., Medioni, G. G., 1992. Object modelling by registration of multiple range images. Image Vision Comput. 10 (3), 145–155.

Cox, M. G., 1971. Curve fitting with piecewise polynomials. J. Appl. Math. 8 (1), 36–52.

Davis, J., Nehab, D., Ramamoorthi, R., Rusinkiewicz, S., 2005. Spacetime stereo: A unifying framework for depth from triangulation. IEEE Trans. Pattern Anal. Mach. Intell. 27 (2), 296–302.

de Aguiar, E., Stoll, C., Ahmed, N., Seidel, H.-P., 2008. Performance capture from sparse multi-view video. ACM Trans. Graph. 27 (3), 1–10.

Dierckx, P., 1993. Curve and surface fitting with splines. Oxford University Press, Inc.

do Carmo, M. P., 1976. Differential Geometry of Curves and Surfaces. Prentice-Hall.

Duchon, J., 1977. Splines minimizing rotation-invariant semi-norms in Sobolev spaces. In: Schempp, W., Zeller, K. (Eds.), Constructive Theory of Functions of Several Variables. Springer, pp. 85–100.

Eck, M., Hadenfeld, J., 1995. Knot removal for B-spline curves. Computer Aided Geometric Design 12 (3), 259–282.

Eck, M., Hoppe, H., 1996. Automatic reconstruction of B-spline surfaces of arbitrary topological type. In: Proc. SIGGRAPH 1996. pp. 325–334.

Elber, G., Fish, R., 1997. 5-Axis freeform surface milling using piecewise ruled surface approximation. Journal of Manufacturing Science and Engineering 119, 383–387.

Faber, C., 1963. Candela/The Shell Builder. Reinhold Publishing Corporation.

Farin, G., 1988. Curves and Surfaces for Computer-Aided Geometric Design: A Practical Guide. Academic Press Professional, Inc.

Fischler, M. A., Bolles, R. C., 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM 24 (6), 381–395.

Flöry, S., 2009. Fitting curves and surfaces to point clouds in the presence of obstacles. Computer Aided Geometric Design 26 (2), 192–202.

Flöry, S., Hofer, M., 2010. Surface fitting and registration of point clouds using approximations of the unsigned distance function. Computer Aided Geometric Design 27 (1), 60–77.

Fong, P., Buron, F., 2005. Sensing deforming and moving objects with commercial off the shelf hardware. In: Proc. CVPR 2005. p. 101.

Frome, A., Huber, D., Kolluri, R., Bülow, T., Malik, J., 2004. Recognizing objects in range data using regional point descriptors. In: Proc. ECCV 2004. pp. 224–237.

Gal, R., Cohen-Or, D., 2006. Salient geometric features for partial shape matching and similarity. ACM Trans. Graph. 25 (1), 130–150.

Gelfand, N., Mitra, N. J., Guibas, L. J., Pottmann, H., 2005. Robust global registration. In: Proc. SGP 2005. pp. 197–206.

Gertz, E. M., Wright, S. J., 2003. Object-oriented software for quadratic programming. ACM Trans. Math. Softw. 29 (1), 58–81.

Gong, H., Cao, L.-X., Liu, J., 2005. Improved positioning of cylindrical cutter for flank milling ruled surfaces. Computer Aided Design 37 (12), 1205 – 1213.

Hähnel, D., Thrun, S., Burgard, W., 2003. An extension of the ICP algorithm for modeling nonrigid objects with mobile robots. In: Proc. IJCAI 2003. pp. 915–920.

Han, Z., Yang, D. C. H., Chuang, J.-J., 2001. Isophote-based ruled surface approximation of free-form surfaces and its application in NC machining. International Journal of Production Research 39, 1911–1930.

Hayes, J. G., Halliday, J., 1974. The least-squares fitting of cubic spline surfaces to general data sets. J. Appl. Math. 14 (1), 89–103.

Hecker, Y., Bolle, R., 1994. On geometric hashing and the generalized Hough transform. IEEE Trans. on Systems, Man and Cybernetics 24, 1328–1338.

Herron, G., 1987. Techniques for visual continuity. In: Geometric Modeling: Algorithms and New Trends. SIAM, pp. 163–174.

Holland, P., Welsch, R., 1977. Robust regression using iteratively reweighted least squares. Commun. in Stat. A 6, 813–827.

Horn, B. K. P., 1987. Closed-form solution of absolute orientation using unit quaternions. Journal of the Optical Society of America 4 (A 4), 629–642.

Hoschek, J., 1971. Liniengeometrie. Bibliographisches Institut.

Hoschek, J., 1988. Intrinsic parametrization for approximation. Computer Aided Geometric Design 5 (1), 27–31.

Hoschek, J., Kaklis, P., 1996. Advanced Course on FAIRSHAPE. Teuber.

Hoschek, J., Lasser, D., 1993. Fundamentals of Computer Aided Geometric Design. AK Peters.

Hoschek, J., Schneider, M., 1997. Interpolation and approximation with developable surfaces. In: Mehaute, A. L., Schumaker, L. L., Rabut, C. (Eds.), Curves and Surfaces with Applications in CAGD. Vanderbilt University Press, pp. 185–202.

Hoschek, J., Schwanecke, U., 1998. Interpolation and approximation with ruled surfaces. In: The Mathematics of Surfaces VIII. Information Geometers, pp. 213–231.

Hu, S.-M., Wallner, J., 2005. A second-order algorithm for orthogonal projection onto curves and surfaces. Computer Aided Geometric Design 22 (3), 251–260.

Huang, Q.-X., Flöry, S., Gelfand, N., Hofer, M., Pottmann, H., 2006. Reassembling fractured objects by geometric matching. ACM Trans. Graph. 25 (3), 569–578.

Isaacson, E., Keller, H. B., 1994. Analysis of numerical methods. Dover, New York.

Jencks, C., 2000. Le Corbusier and the Continual Revolution in Architecture. Monacelli.

Johnson, A., 1997. Spin-images: A representation for 3D surface matching. Ph.D. thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.

Kass, M., Witkin, A. P., Terzopoulos, D., 1988. Snakes: Active contour models. Int. J. Comp. Vis. 1 (4), 321–331.

Kazhdan, M., Funkhouser, T., 2002. Harmonic 3D shape matching. In: Proc. SIGGRAPH 2002. pp. 191–191.

Kelley, C. T., 1999. Iterative Methods for Optimization. No. 18 in Frontiers in Applied Mathematics. SIAM.

Kilian, M., Flöry, S., Chen, Z., Mitra, N. J., Sheffer, A., Pottmann, H., 2008. Curved folding. ACM Trans. Graph. 27 (3), 75:1–75:9.

Kiwiel, K. C., 1990. Proximity control in bundle methods for convex nondifferentiable minimization. Math. Program. 46, 105–122.

Koc, B., Lee, Y.-S., 2002. Adaptive ruled layers approximation of stl models and multi-axis machining applications for rapid prototyping. Journal of Manufacturing Systems 21, 153–166.

Kolarevic, B. (Ed.), 2003. Architecture in the Digital Age: Design and Manufacturing. Spon Press.

König, S., Gumhold, S., 2008. Image-based motion compensation for structured light scanning of dynamic surfaces. Int. J. Intell. Syst. Technol. Appl. 5 (3/4), 434–441.

Kuhn, H. W., 1973. A note on Fermat's problem. Math. Program. 4, 98–107.

Lartigue, C., Duc, E., Affouard, A., 2003. Tool path deformation in 5-axis flank milling using envelope surface. Computer Aided Design 35 (4), 375 – 382.

Levoy, M., Pulli, K., Curless, B., Rusinkiewicz, S., Koller, D., Pereira, L., Ginzton, M., Anderson, S., Davis, J., Ginsberg, J., Shade, J., Fulk, D., 2000. The Digital Michelangelo project: 3D scanning of large statues. In: Proc. SIGGRAPH 2000. pp. 131–144.

Li, C., Bedi, S., Mann, S., 2006. Flank milling of a ruled surface with conical tools-an optimization approach. Int. J. of Adv. Manufacturing Technology 29, 1115 – 1124.

Li, X., Guskov, I., 2005. Multi-scale features for approximate alignment of point-based surfaces. In: Proc. SGP 2005. pp. 217–226.

Lin, H., Wang, G., 2002. Interval B-spline curve evaluation bounding point cloud. In: Proc. PCCGA 2002. p. 424.

Lions, P.-L., 1982. Generalized Solutions of Hamilton-Jacobi Equations. Pitman.

Lipman, Y., Cohen-Or, D., Levin, D., Tal-Ezer, H., 2007. Parameterization-free projection for geometry reconstruction. ACM Trans. Graph. 26 (3), 22.

Lipman, Y., Funkhouser, T., 2009. Möbius voting for surface correspondence. ACM Trans. Graph. 28 (3), 1–12.

Lobo, M. S., Vandenberghe, L., Boyd, S., Lebret, H., 1998. Applications of second-order cone programming. Linear Algebra and its Applications 284 (1-3), 193–228.

Loncaric, S., 1998. A survey of shape analysis techniques. Pattern Recognition 31 (8), 983–1001.

Mamic, G., Bennamoun, M., 2002. Representation and recognition of 3D free-form objects. Digital Signal Processing 12 (1), 47 – 76.

Masuda, T., Yokoya, N., 1995. A robust method for registration and segmentation of multiple range images. Comput. Vis. Image Underst. 61 (3), 295–307.

Mayer, P., Moaveni, S., 2008. Close-contact melting as a subtractive machining process. Int. J. of Adv. Manufacturing Technology 37, 980–995.

McWherter, D., Peabody, M., Regli, W. C., Shokoufandeh, A., 2001. Transformation invariant shape similarity comparison of solid models. In: Proc. ASME DETC 2001.

Meek, D. S., Ong, B. H., Walton, D. J., 2003. Constrained interpolation with rational cubics. Computer Aided Design 20 (5), 253–275.

Milroy, M. J., Bradley, C., Vickers, G. W., Weir, D. J., 1995. G1 continuity of B-spline surface patches in reverse engineering. Computer Aided Design 27 (6), 471–478.

Mitani, J., Suzuki, H., 2004. Making papercraft toys from meshes using strip-based approximate unfolding. ACM Trans. Graph. 23 (3), 259–263.

Mitra, N. J., Flöry, S., Ovsjanikov, M., Gelfand, N., Guibas, L., Pottmann, H., 2007. Dynamic geometry registration. In: Proc. SGP 2007. pp. 173–182.

Mitra, N. J., Nguyen, A., 2003. Estimating surface normals in noisy point cloud data. In: Proc. SCG 2003. pp. 322–328.

Myles, A., Peters, J., 2005. Threading splines through 3D channels. Computer Aided Design 37 (2), 139–148.

Nesterov, Y., Nemirovskii, A., 1994. Interior-Point Polynomial Algorithms in Convex Programming. Vol. 13 of SIAM Studies in Applied Mathematics. SIAM.

Nocedal, J., Wright, S. J., 1999. Numerical Optimization. Springer.

Opfer, G., Oberle, H. J., 1988. The derivation of cubic splines with obstacles by methods of optimization and optimal control. Numer. Math. 52, 17–31.

Osher, S. J., Fedkiw, R. P., 2003. Level Set Methods and Dynamic Implicit Surfaces. Springer.

Pauly, M., Gross, M., Kobbelt, L. P., 2002. Efficient simplification of point-sampled surfaces. In: Proc. IEEE VIS 2002. pp. 163–170.

Peternell, M., 2004. Developable surface fitting to point clouds. Computer Aided Geometric Design 21 (8), 785–803.

Peters, J., 2002. Geometric continuity. In: Handbook of Computer Aided Geometric Design. Elsevier, pp. 193–229.

Pighin, F., Lewis, J. P., 2007. Practical least-squares for computer graphics. In: SIGGRAPH 2007 Courses. pp. 1–57.

Plass, M., Stone, M., 1983. Curve-fitting with piecewise parametric cubics. SIGGRAPH Comput. Graph. 17 (3), 229–239.

Pottmann, H., Hofer, M., 2003. Geometry of the squared distance function to curves and surfaces. In: Hege, H. C., Polthier, K. (Eds.), Visualization and Mathematics III. pp. 223–244.

Pottmann, H., Huang, Q.-X., Yang, Y.-L., Hu, S.-M., 2006. Geometry and convergence analysis of algorithms for registration of 3D shapes. Int. J. Comp. Vis. 67 (3), 277–296.

Pottmann, H., Leopoldseder, S., 2003. A concept for parametric surface fitting which avoids the parametrization problem. Computer Aided Geometric Design 20 (6), 343–362.

Pottmann, H., Leopoldseder, S., Hofer, M., 2002. Simultaneous registration of multiple views of a 3D object. In: Proc. PCV 2002. p. A: 265.

Pottmann, H., Schiftner, A., Bo, P., Schmiedhofer, H., Wang, W., Baldassini, N., Wallner, J., 2008. Freeform surfaces from single curved panels. ACM Trans. Graph. 27 (3).

Pottmann, H., Wallner, J., 2001. Computational Line Geometry. Springer.

Pottmann, H., Wallner, J., Huang, Q.-X., Yang, Y.-L., 2009. Integral invariants for robust geometry processing. Computer Aided Geometric Design 26 (1), 37–60.

Pulli, K., 1999. Multiview registration for large data sets. In: Proc. 3DIM 1999. pp. 160–168.

Redonnet, J.-M., Rubio, W., Dessein, G., 1998. Side milling of ruled surfaces: Optimum positioning of the milling cutter and calculation of interference. Int. J. of Adv. Manufacturing Technology 14, 459–465.

Richardson, I. E., 2003. H.264 and MPEG-4 Video Compression: Video Coding for Next Generation Multimedia. Wiley.

Rockafellar, R., 1972. Convex Analysis. Princeton University Press.

Rogers, D. F., 1989. Constrained B-spline curve and surface fitting. Computer Aided Design 21 (10), 641–648.

Roth, D., Bedi, S., Ismail, F., Mann, S., 2001. Surface swept by a toroidal cutter during 5-axis machining. Computer Aided Design 33 (1), 57 – 63.

Rousseeuw, P. J., Leroy, A. M., 1987. Robust regression and outlier detection. Wiley.

Rudin, L. I., Osher, S., Fatemi, E., 1992. Nonlinear total variation based noise removal algorithms. Physica D 60 (1-4), 259–268.

Rusinkiewicz, S., Hall-Holt, O., Levoy, M., 2002. Real-time 3D model acquisition. ACM Trans. Graph. 21 (3), 438–446.

Rusinkiewicz, S., Levoy, M., 2001. Efficient variants of the ICP algorithm. In: Proc. 3DIM 2001. pp. 145–152.

Saux, E., Daniel, M., 2003. An improved Hoschek intrinsic parametrization. Computer Aided Geometric Design 20 (8-9), 513 – 521.

Schaaf, J. A., Ravani, B., 1998. Geometric continuity of ruled surfaces. Computer Aided Geometric Design 15 (3), 289 – 310.

Senatore, J., Monies, F., Landon, Y., Rubio, W., 2008. Optimising positioning of the axis of a milling cutter on an offset surface by geometric error minimisation. Int. J. of Adv. Manufacturing Technology 37, 861 – 871.

Sethian, J. A., 1999. Level Set Methods and Fast Marching Methods. Cambridge University Press.

Sharf, A., Alcantara, D. A., Lewiner, T., Greif, C., Sheffer, A., Amenta, N., Cohen-Or, D., 2008. Space-time surface reconstruction using incompressible flow. ACM Trans. Graph. 27 (5), 1–10.

Shi, X., Wang, T., Wu, P., Liu, F., 2004. Reconstruction of convergent G1 smooth B-spline surfaces. Computer Aided Geometric Design 21 (9), 893–913.

Simonoff, J. S., 1998. Smoothing Methods in Statistics. Springer.

Sprott, K., Ravani, B., 2008. Cylindrical milling of ruled surfaces. Int. J. of Adv. Manufacturing Technology 38, 649 – 656.

Step Four GmbH, 2007. Drehteller und Drehachsensteuerung. Data sheet.

Stute, G., Storr, A., Sielaff, W., 1979. NC programming of ruled surfaces for five axis machining. Annals CIRP 28, 267–271.

Sun, J., Ovsjanikov, M., Guibas, L., 2009. A concise and provably informative multi-scale signature based on heat diffusion. In: Proc. SGP 2009. pp. 1383–1392.

Süßmuth, J., Winter, M., Greiner, G., 2008. Reconstructing animated meshes from time-varying point clouds. Comp. Graph. Forum 27 (5), 1469–1476.

Tangelder, J. W., Veltkamp, R. C., 2008. A survey of content based 3D shape retrieval methods. Multimedia Tools Appl. 39 (3), 441–471.

Thuswaldner, B., Flöry, S., Kalasek, R., Hofer, M., Huang, Q.-X., Thür, H., 2009. Digital anastylosis of the octagon in ephesos. ACM JOCCH 2 (1), 1–27.

Tsay, D. M., Her, M. J., 2001. Accurate 5-axis machining of twisted ruled surfaces. Journal of Manufacturing Science and Engineering 123 (4), 731–738.

Veltkamp, M., 2007. Free Form Structural Design. IOS Press.

Veron, M., Ris, G., Musse, J.-P., 1976. Continuity of biparametric surface patches. Computer Aided Design 8 (4), 267–273.

Vollers, K., 2001. Twist & Build: Creating Non-Orthogonal Architecture. 010 Publishers.

Wand, M., Adams, B., Ovsjanikov, M., Berner, A., Bokeloh, M., Jenke, P., Guibas, L., Seidel, H.-P., Schilling, A., 2009. Efficient reconstruction of non-rigid shape and motion from real-time 3D scanner data. ACM Trans. Graph. 28 (2), 1–15.

Wand, M., Jenke, P., Huang, Q.-X., Bokeloh, M., Guibas, L., Schilling, A., 2007. Reconstruction of deforming geometry from time-varying point clouds. In: Proc. SGP 2007. pp. 49–58.

Wang, W., Pottmann, H., Liu, Y., 2006. Fitting B-spline curves to point clouds by curvature-based squared distance minimization. ACM Trans. Graph. 25 (2), 214–238.

Weber, A., 1909. Über den Standort der Industrien, 1. Teil: Reine Theorie des Standortes. J.C.B. Mohr.

Weise, T., Leibe, B., Gool, L. V., 2007. Fast 3D scanning with automatic motion compensation. In: Proc. CVPR 2007. pp. 1–8.

Weiss, V., Andor, L., Renner, G., Várady, T., 2002. Advanced surface fitting techniques. Computer Aided Geometric Design 19 (1), 19–42.

Weiszfeld, E., 1937. Sur le point pour lequel la somme des distances de points donnés est minimum. Tohoku Mathematics Journal 43, 355–386.

Wolfson, H. J., Rigoutsos, I., 1997. Geometric hashing: An overview. IEEE Comput. Sci. Eng. 4 (4), 10–21.

Yang, H., Wang, W., Sun, J.-G., 2004. Control point adjustment for B-spline curve approximation. Computer Aided Design 36 (7), 639–652.

Yang, M., Lee, E., 1999. Segmentation of measured point data using a parametric quadric surface approximation. Computer Aided Design 31 (7), 449–457.

Zach, C., Pock, T., Bischof, H., 2007. A globally optimal algorithm for robust TV-l_1 range image integration. Proc. ICCV 2007 0, 1–8.

Zhang, H., Sheffer, A., Cohen-Or, D., Zhou, Q., van Kaick, O., Tagliasacchi, A., 2008. Deformation-drive shape correspondence. Proc. SGP 2008, 1431–1439.

Zhang, L., Curless, B., Seitz, S. M., 2003. Spacetime stereo: Shape recovery for dynamic scenes. In: Proc. CVPR 2003. pp. 367–374.

Zhu, L., Xiong, Z., Ding, H., Xiong, Y., 2004. A distance function based approach for localization and profile error evaluation of complex surface. Journal of Manufacturing Science and Engineering 126 (3), 542–554.

Curriculum Vitae

Simon Flöry, born October 10, 1980 in Graz, Austria.

Education

09/1990 - 06/1998	Grammar school in Bludenz, Austria. Graduation (Matura) in 1998 (with distinction).
10/1999 - 11/2005	Vienna University of Technology, Austria. Graduation Dipl.-Ing. (M.Sc.) in Technical Mathematics (with distinction).
09/2003 - 05/2004	Helsinki University of Technology, Finland. Exchange year (Erasmus), studies of Mathematics and Informatics.
since 12/2005	Vienna University of Technology, Austria. Ph.D. studies in Applied Geometry, advisor Helmut Pottmann.

Employment, Research and Teaching Experience

since 12/2005	Research assistant at Vienna University of Technology, Austria. Research in Applied Geometry, teaching students of Mathematics and Geometry.
07/2007 - 10/2007	Internship at Siemens Corporate Research, Princeton, NJ, USA. Research and implementation of algorithms for feature detection and shape completion.
since 03/2008	Project manager at Evolute GmbH, Vienna, Austria. Research and consulting in geometry processing.

Scientific Publications

Flöry, S., 2009. Fitting curves and surfaces to point clouds in the presence of obstacles. Computer Aided Geometric Design 26 (2), 192–202.

Flöry, S., Hofer, M., 2008. Constrained curve fitting on manifolds. Computer Aided Design 40 (1), 25–34.

Flöry, S., Hofer, M., 2010. Surface fitting and registration of point clouds using approximations of the unsigned distance function. Computer Aided Geometric Design 27 (1), 60–77.

Huang, Q.-X., Flöry, S., Gelfand, N., Hofer, M., Pottmann, H., 2006. Reassembling fractured objects by geometric matching. ACM Trans. Graph. 25 (3), 569–578.

Kilian, M., Flöry, S., Chen, Z., Mitra, N. J., Sheffer, A., Pottmann, H., 2008a. Curved folding. ACM Trans. Graph. 27 (3), 75:1–75:9.

Kilian, M., Flöry, S., Chen, Z., Mitra, N. J., Sheffer, A., Pottmann, H., 2008b. Developable surfaces with curved creases. In: Proc. AAG 2008. pp. 33–36.

Mitra, N. J., Flöry, S., Ovsjanikov, M., Gelfand, N., Guibas, L., Pottmann, H., 2007. Dynamic geometry registration. In: Proc. SGP 2007. pp. 173–182.

Thuswaldner, B., Flöry, S., Kalasek, R., Hofer, M., Huang, Q.-X., Thür, H., 2009. Digital anastylosis of the octagon in ephesos. ACM JOCCH 2 (1), 1–27.