Interactive Spacetime Control of Deformable Objects and Modal Shape Analysis beyond Laplacian

A dissertation submitted to the Fachbereich Mathematik und Informatik, Freie Universität Berlin in partial fulfillment of the requirements for the degree of doctor rerum naturalium

Author: Christian Schulz

Defended 6 June 2013

Acknowledgments

I thank Prof.Dr. Polthier for giving me the opportunity to work in this exciting research area. I feel very grateful to Klaus Hildebrandt and Christoph von Tycowicz for their invaluable advise and constant support. I thank Prof.Dr. Olga Sorkine-Hornung for refereeing. And, finally, I want to thank Mimi Tsuruga, Anna Wawrzinek and Ulrich Reitebuch for their incredible patience and helpful suggestions.

Contents

1 Introduction **7**
1.1 Preliminaries and notation 10

2 Interactive Spacetime Control **13**
2.1 Background 14
2.2 Setup 15
2.3 Examples 22
2.4 Spacetime constraint problem 29
2.5 Solving the linearized problem 34
2.6 One-point dynamics 42
2.7 Reduction 47
2.8 Multi-point dynamics 51
2.9 Summary 56

3 Geometry-aware Operators **59**
3.1 Background 60
3.2 Laplace modes and spectrum 61
3.3 Energy based spectrum 64
3.4 Extended Dirichlet energy 72
3.5 Stability of discrete cmc-surfaces 73
3.6 Modal signatures 75
3.7 Applications 76
3.8 Summary 80

4 Patch Layout from Feature Graph **81**
4.1 Background 82
4.2 Setup 83
4.3 Feature graph 84
4.4 Patch layout 89
4.5 Results 93
4.6 Summary 94

5 Outlook **97**

Appendix **99**
5.1 Planar rod 99
5.2 Luxo 100

Zusammenfassung **103**

Verfassererklärung **105**

Chapter 1

Introduction

Geometry processing is a rapidly growing research field combining concepts from mathematics, scientific computing and engineering. Out of the many challenging problems to focus on in this field, this thesis deals with three particular problems: one problem related to computer generated animation and two problems related to shape analysis, specifically differential operator design and surface segmentation.

We present the corresponding results in three independent chapters. Our shape animation approach that generalizes the common spline keyframe interpolation to a physically based keyframe interpolation is discussed in Chapter 2. The results in differential operator design that are based on the definition of a certain feature sensitive surface energy are discussed in Chapter 3. Our surface segmentation approach that aims for a decomposition that is aware of surface features is discussed in Chapter 4. And finally in Chapter 5, we close with a list of open problems.

In computer animation, the spacetime-constraint paradigm was introduced in [Witkin and Kass, 1988] to generate physically plausible motion. In general, this approach involves the solution of a high-dimensional constraint optimization problem. Usually the problem's dimension depends on the degrees of freedom of the object to be animated.

Our method allows the user to create and edit physically plausible motion at interactive rates for objects with a large number of degrees of freedom, e.g., elastic soft bodies or thin shells. To achieve this speed, we linearize and reduce the underlying (possibly) high-dimensional variational problem. By decoupling the resulting system of ordinary differential equations we can then reformulate the spacetime problem as a small set of independent one-dimensional boundary value problems. The solutions to these boundary value problems are found in the space spanned by the wiggly splines introduced in [Kass and Anderson, 2008]. In addition to our reduction strategy, the achieved fast response times also result from our closed-form representation of the wiggly splines. Generating motion for deformable objects using a reduced formulation of the full nonlinear problem is also considered in [Barbič et al., 2009], but their solution curve is found by a numerical optimization procedure. To avoid effects resulting from the use of a linearization, we propose two alternatives that are based on multi-point linearization. We demonstrate the versatility of our approach on a broad class of shapes, including one-, two-, and three-dimensional geometries.

In shape analysis, the spectrum and eigenfunctions of the discrete Laplace operator are excessively explored to identify application relevant isometry invariant surface properties as discussed in [Clements and Zhang, 2006] and [Lévy and Zhang, 2009].

We introduce a family of operators that can serve as an alternative to the discrete Laplace operator for applications in modal shape analysis. In contrast to the extrinsic-feature insensitive discrete Laplace operator, we introduce operators that are sensitive to extrinsic features. The feature sensitivity of an operator, that is, of its spectrum and eigenfunctions, is controlled by a single scalar. The operators are constructed from Hessians of quadratic surface energies that act on surface functions, so they have a concise matrix representation. The Laplace operator is a member of the introduced operator family. It corresponds to the feature insensitive operator of this family.

We further show that the spectrum of a certain operator of this family can be used to estimate the unconstrained stability index of constant-mean-curvature (cmc) surfaces [Polthier, 2002]. Our estimates for three minimal surfaces coincide with the results given in [Polthier and Rossmann, 2002].

As an example of a possible shape analysis application, we derive a feature sensitive point signature and associated feature metric that is based on our extrinsic-feature sensitive spectra and eigenmodes. We compare the results of identified surface points by our signature to results given by the Laplace related heat kernel signature introduced in [Sun et al., 2009].

The generation of a meaningful decomposition of a surface mesh is a challenging task in shape analysis [Shamir, 2006]. For example, in reverse engineering, a scanned triangle mesh must be divided into a set of patches that allows the conversion of this triangle mesh into a low-order spline surface representation [Várady et al., 2007]. We provide an algorithm for the generation of a patch layout for CAD surfaces that consists of smoothly connected weakly curved parts. The construction is based on the generation of a net of smooth surface curves that run along features, that is, the feature graph. The final patch layout is then generated by a thickening of the feature graph curves.

The following summarizes our contributions to the three problems we considered.

Animation

- creating and editing of physically based animation for deformable objects at interactive rates
- closed form solution of the wiggly spline basis functions
- reduced formulation, that allows interpolation and least square approximating of keyframes
- two procedures to overcome linearization artifacts

Differential operators

- introduction of an operator derived from a modified Dirichlet energy possessing a feature sensitive spectrum and eigenfunctions
- introduction of a family of operators derived from weighted sum of Dirichlet energy and the modified Dirichlet energy
- a simple operator whose spectrum can be used to estimate the unconstrained stability index of a cmc-surface
- a multi-scale point signature and an associated multi-scale feature distance to identify surface points with respect to extrinsic surface features

Segmentation

- a method to generate a net of curves, running along geometric surface features
- a curve thickening-offsetting procedure

The results presented in this work have been published in the following refereed articles:

"Interactive Spacetime Control of Deformable Objects"
Klaus Hildebrandt, Christian Schulz, Christoph v. Tycowicz, and Konrad Polthier
Transactions on Graphics, 31, 2012 (presented at Siggraph2012)

"Modal Shape Analysis beyond Laplacian"
Klaus Hildebrandt, Christian Schulz, Christoph v. Tycowicz, and Konrad Polthier
Computer-Aided Geometric Design, 29, 2012 (presented at GMP2010)

"Patch Layout from Feature Graph"
Matthias Nieser, Christian Schulz, and Konrad Polthier
Computer-Aided Design, 42, 2010

1.1 Preliminaries and notation

In this work we restrict ourselves to certain types of discrete geometry. They are described by compact sets $\Omega \subset \mathbb{R}^l$, where we assume $l \leq 3$. Our definition of these discrete geometries is based on the definition of a simplicial complex given in [Munkres, 1984] and [Polthier, 2002]. An n-simplex is defined by:

Definition 1 (n-simplex) *Let $\{p_1, \ldots, p_{n+1}\}$ be a finite set of geometrically independent points $p_i \in \mathbb{R}^l$. Then we define the n-simplex σ to be the convex set*

$$\sigma\left(p_1, \ldots, p_{n+1}\right) = \left\{ x \in \mathbb{R}^l \,\middle|\, x = \sum_{i=1}^{n+1} t_i p_i, \sum_{i=1}^{n+1} t_i = 1, t_i \geq 0 \right\}.$$

The dimension d of an n-simplex σ is $d = n$. We refer to any simplex spanned by a subset of $\{p_1, \ldots, p_{n+1}\}$ as a face of σ. In this work, we will only deal with low-dimensional n-simplices, that is, $n \leq 4$. By σ^0, σ^1, σ^2, σ^3, we refer to a zero-, one-, two-, or three-dimensional simplex, respectively. They correspond to vertices, edges, triangles, or tetrahedra.

Definition 2 (Simplicial Complex) *A simplicial complex K in $\mathbb{R}^l$ is a finite collection of simplicies in $\mathbb{R}^l$ such that:*

1. *every face of a simplex contained in K is in K; and*
2. *the intersection of any two simplices of K is a face of each of them.*

A subset $L \subset K$ that is itself a simplicial complex is called a subcomplex. An example of a subcomplex is the complex containing all simplices of dimension at most d, denoted by $K^{(d)}$. The simplices of $K^{(0)}$ are called vertices.

For a simplicial complex K, the compact set $|K| \in \mathbb{R}^l$ is defined by

$$|K| = \bigcup_k \sigma_k\left(p_{k_1}, \ldots, p_{k_{n+1}}\right), \text{ with } \sigma_k \in K. \tag{1.1}$$

The topology of the set $|K|$ is the induced topology of $\mathbb{R}^l$. For $p_i \in K^{(0)}$ we define the set $A_*(p_i) \subset \mathbb{R}^l$ as the union of all simplices $\sigma \in K$ containing p_i. The set $A_*(p_i)$ corresponds to the union of a finite number of closed sets. Therefore $A_*(p_i)$ is closed.

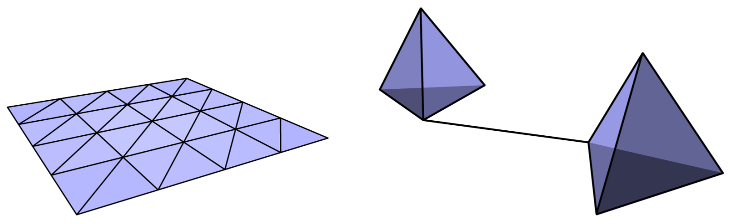

Figure 1.1: Two examples of geometric realizations: a triangulated surface, which is a simplicial geometry on the left and two tetrahedra connected by a line, which is not a simplicial geometry on the right.

Definition 3 (Abstract Simplicial Complex) *Let $V = \{v_1, \ldots, v_m\}$ be a finite set of abstract points. A set $S(V) \subset \mathcal{P}(V)$, where $\mathcal{P}(V)$ denotes the power set of V, is called an abstract simplicial complex if for all elements $\sigma \in S(V)$, every subset $\tau \subset \sigma$ is also contained in $S(V)$.*

The set V is called the vertex set of the abstract simplicial complex $S(V)$. The elements $\sigma_k \in S(V)$ are called simplices or, simply, cells.

Given a map $P : V \to \mathbb{R}^l$, where for brevity $P(v_i) = p_i$, we define a map $\sigma_k \mapsto |\sigma_k| \subset \mathbb{R}^l$ for each n-simplex $\sigma_k \in S(V)$ where $|\sigma_k| = \sigma\left(p_{k_1}, \ldots, p_{k_{n+1}}\right)$ with $p_{k_i} = P(v_{k_i})$ and $v_{k_i} \in \sigma_k$.

We call the result of a general mapping P a realization of $S(V)$. If the resulting set of n-simplices forms a simplicial complex K (see Definition 2) then K is called a geometric realization of $S(V)$. Two examples of geometric realizations are shown in Figure 1.1.

The canonical realization of an abstract simplicial complex $S(V)$ is a certain mapping $P_{ca} : V \to \mathbb{R}^n$ with $P_{ca}(v_j) = [\delta_{1,j}, \ldots, \delta_{i,j}, \ldots, \delta_{n,j}]$, where $\delta_{i,j}$ denotes the Kronecker delta.

Using the canonical realization, we can define a certain abstract simplicial complex which we call:

Definition 4 (Simplicial Geometry) *A simplicial geometry $S_d(V)$ is an abstract simplicial complex $S(V)$ whose canonical realization is a simplicial complex K with the following properties:*

1. *$S_d(V)$ only contains n-dimensional cells, with $n \leq d$*
2. *for each $p_i \in K^{(0)}$, the closed set $A_*(p_i) \subset \mathbb{R}^n$ is homeomorphic to the closed unit d-ball.*

The realization of $S_d(V)$ given by a map $P : V \to \mathbb{R}^l$ is denoted by $K_{d,l}$. In particular, we call $K_{2,3}$ a simplicial surface and $K_{3,3}$ a simplicial volume. For

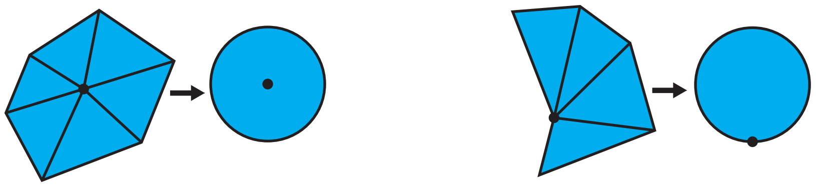

Figure 1.2: Illustration of the possible homeomorphic mappings of $A_*(p_i)$ to the closed unit disc for points of a simplicial geometry $S_2(V)$

a simplicial surface, we refer to $S_2(V)$ and $K_{2,3}$ by $\mathcal{M}$ and $\mathcal{M}_h$, respectively. The two possible homeomorphic mappings of $A_*(p_i)$ with $v_i \in \mathcal{M}$ to the unit disc are shown in Figure 1.2. For a simplicial volume, we refer to $S_3(V)$ and $K_{3,3}$ by $\mathcal{V}$ and $\mathcal{V}_h$, respectively.

Chapter 2

Interactive Spacetime Control

In traditional computer animation, the motions of objects or characters are typically generated from keyframes that specify values for all of the object's or character's degrees of freedom at a sparse set of points in time. The continuous motion is obtained by fitting splines through the keyframes. This technique is attractive since it offers an adequate amount of control over the motion at a low computational cost. One drawback for this technique, however, is that it offers little help to an animator who wants to create physically plausible motion. Moreover, splines are designed to produce functions with high fairness, e.g., functions with few extrema and inflection points, whereas the motion of objects or characters is often oscillatory.

Physical simulation can produce realistic motion, but it is a delicate task to explicitly determine forces and physical quantities that produce the motion that matches an animator's intentions. This is aggravated by the fact that physical simulations are integrated forward in time, which means that small changes at some point in time can have a large impact on the state of the system at a later time. Control over a simulation can be achieved by computing optimal physical trajectories that are solutions of a variational spacetime problem. Such techniques calculate acting forces that minimize an objective functional while guaranteeing that the resulting motion satisfies prescribed spacetime constraints, e.g., that it interpolates a set of keyframes. Resulting forces are optimally distributed over the whole animation and show effects like squash-and-stretch, timing, or anticipation that are desired in animation. However, the computational cost for obtaining these results is that of solving a spacetime optimization problem. To date, recent methods — even those that use dimension reduction techniques — still require at least several minutes to solve the optimization problem for an interesting motion of an object or character.

We describe the elementary assumptions on which our approach is based in Section 2.2. This includes a definition of a class of objects and a description of the expected dynamical properties and their implications. Then we show one-, two-, and three-dimensional example shapes and dynamic models that fit into this setting in Section 2.3. A description of the spacetime problem and a derivation of its linearization is given in Section 2.4. We then present two strategies to combine multiple linearized dynamic models to capture nonlinear behavior in Section 2.8. How the method can be applied to objects possessing a large number of degrees of freedom is shown in Section 2.7. Finally, in Section 2.8.3 we discuss our results.

2.1 Background

Variational problems with spacetime constraints were introduced to computer animation by [Witkin and Kass, 1988]. Their example of the jumping Luxo lamp nicely demonstrates the benefits of this approach. Since then the spacetime constraints paradigm has stimulated much research (see [Fang and Pollard, 2003, Safonova et al., 2004] for a detailed summary).

Based on spacetime constraints, techniques for generating physically plausible motions for various types of physical systems including ropes and strings [Barzel, 1997], rigid body motions [Popović et al., 2003], fluids [Treuille et al., 2003, McNamara et al., 2004], particle systems [Wojtan et al., 2006], and elastic solids [Barbič et al., 2009] have been proposed. The generation of human motions was also attempted using a spacetime constraint formulation, see [Gleicher, 1997, Fang and Pollard, 2003, Safonova et al., 2004, Chai and Hodgins, 2007].

The associated constrained optimization problems are typically high dimensional and are usually solved with gradient-based approaches or Newton methods. Local-to-global strategies, called spacetime windowing, were developed to speed up the solvers by [Cohen, 1992] and by [Treuille et al., 2003]. A particular problem when determining the optimal trajectory is the calculation of the derivatives of the objective functional that captures the physical plausibility of the motion. Automatic and symbolic differentiation has been used by many researchers such as [Witkin and Kass, 1988, Fang and Pollard, 2003, Safonova et al., 2004] and spacetime constraints serve as one of the main target applications for the development of algorithms for automatic differentiation in graphics, see [Guenter, 2007].

Model reduction is an established technique in solid mechanics that can be used to accelerate simulations of elastic solids [Nickell, 1976, Idelsohn and Cardona, 1985, Krysl et al., 2001]. In graphics [Pentland and Williams, 1989] pioneered work in this area by using modal analysis to automatically generate reduced spaces. Based on an efficient representation of the forces in the reduced space, [Barbič and James, 2005] obtained real-time rates for forward simulation of elastic solids. In addition, they extended the automatic generation of subspaces to include modal derivatives, which helps to improve the approximation of large deformations. [Treuille et al., 2006] and [Wicke et al., 2009] used reduced spaces constructed from eigenmodes for fluid dynamics.

In geometry processing, reduced spaces were used for interactive modeling of triangular meshes by [Huang et al., 2006] and [Hildebrandt et al., 2011]. [Kim and James, 2009] used model reduction to speed up the calculation of large simulations for animation by skipping full steps if the reduced steps satisfy an accuracy condition. The reduced model is not built in a preprocess but online as the simulation progresses. Reduced spaces have also been used for spacetime constraints. For human motions, [Safonova et al., 2004] and [Sulejmanpašić and Popović, 2005] constructed reduced spaces from motion capture data. [Barbič et al., 2009] constructed reduced spaces for deformable objects automatically from the keyframes by using vibration modes and tangents of a deformation curve that is fitted to the keyframes. This technique for constructing reduced spaces is similar to the way we construct reduced spaces. However, our method allows us to work in larger reduced spaces than theirs.

Geometric interpolation between two or more shapes is also related to spacetime constraints. The variational problem for geodesics in shape spaces has a comparable complexity and solvers need to deal with varying curves in shape spaces as well. Efficient multiresolution solvers for computing geodesics in shape spaces were proposed by [Kilian et al., 2007] and by [Wirth et al., 2009].

2.2 Setup

We start with the definition of the shape space in Section 2.2.1. It will be used to describe all possible configurations of a discrete geometry. In Section 2.2.2 we describe a general procedure to define the dynamic behavior of a discrete object with respect to its configuration space representation. This includes a description of the linearized dynamic model around a certain state. We show that the dynamics are based on two mappings: a scalar valued function and a metric. A certain type of this scalar valued function is described in more detail in Section 2.2.3.

2.2.1 Shape space

In general, discretizing a smooth object by a finite set of samples results in a point set equipped with a mesh structure. This mesh usually describes the smooth object's extension in space. Common mesh types are triangulations, quadrangulations, tetrahedral grids, or cubical grids. We will restrict ourselves to meshes describing simplicial geometries, see Section 1.1. Since the method can be easily extended to more general mesh types, we refer to the simplicial geometry as a discrete object with a grid structure given by $G = S(V)$. We will also assume that the discrete object's grid structure G is fixed.

We follow [Kilian et al., 2007] for the definition of a shape space. Given a simplicial geometry $S_d(V)$, a configuration is a mapping $P : V \to \mathbb{R}^l$, where V denotes the vertex set of $S_d(V)$. Then we can define the set of all configuration by:

$$\mathbf{X}^{all} = \left\{ P \mid P : V \to \mathbb{R}^l \right\}. \tag{2.1}$$

A mapping $P \in \mathbf{X}^{all}$ corresponds to $P(V) = \{p_1, \ldots, p_n\}$ with $p_i \in \mathbb{R}^l$. It can be identified with a point $p \in \mathbb{R}^{nl}$. In coordinates this identification reads $p = [p_{1_i}, \ldots, p_{1_l}, \ldots, p_{n_i}, \ldots, p_{n_l}]$, so we have $\mathbf{X}^{all} \cong \mathbb{R}^{nl}$.

We refer to the number of degrees of freedom of a discrete object by d. In the constraint free case, each p_i can vary in l directions, so we have $d = nl$. We will also consider constraints of the form:

$$c([p_1, \ldots, p_n]) = 0. \tag{2.2}$$

This results in a reduced number of degrees of freedom, that is, $d = nl - n_c$, where n_c is the number of constraints, each given by Equation 2.2. We denote by $\mathbf{X} \subseteq \mathbf{X}^{all}$ the corresponding set of configurations meeting the constraints, i.e.,

$$\mathbf{X} = \left\{ p \in \mathbf{X}^{all} \mid c_1(p) = 0, \ldots, c_{n_c}(p) = 0 \right\}. \tag{2.3}$$

In accordance with [Kilian et al., 2007], we refer to the set $\mathbf{X} \subseteq \mathbb{R}^{nl}$ as the shape space. In the constraint free case we have $\mathbf{X} = \mathbb{R}^{nl}$.

2.2.2 Dynamics

To derive the dynamics for a discrete object, we follow the Lagrange formalism. In classical mechanics this formalism is usually used to derive the equations of motion for constrained and unconstrained physical systems. For a good introduction, we refer to [Josef Honerkamp, 1993] and [Scheck, 2007].

We will assume that we have n_c regular constraints given by Equation 2.2 that define a d-dimensional manifold of permissible configurations $\mathbf{X}$. Furthermore, we assume a proper choice of d local coordinates x_i that parametrrize an open set $\mathbf{U} \subseteq \mathbf{X}$. That is, for an open set $\mathbf{V} \subseteq \mathbb{R}^d$ we have a mapping $A : \mathbf{V} \to \mathbf{U} \subseteq \mathbb{R}^{nl}$ or, in coordinates, $p = A(x)$ with $x = (x_1, \ldots, x_d)$ for each configuration $p \in \mathbf{U}$. According to the Lagrange formalism, these d coordinates are referred to as generalized coordinates.

With this parametrization we will describe the motion of a discrete object by curves in $\mathbb{R}^d$. We refer by $C^k[a,b]$ to k-times differentiable functions $f : \mathbb{R} \to \mathbb{R}$. Further by $C_0^k[a,b] \subset C^k[a,b]$ we denote the subset of compactly supported functions. Then we define the set of k-differentiable curves as:

$$\mathcal{C}_{a,b}^k\left(\mathbb{R}^d\right) = \big\{x : [a,b] \to \mathbb{R}^d \,|\, a,b \in \mathbb{R}, x(t) = [x_1(t), \ldots, x_d(t)] \\ \text{with } x_i(t) \in C^k[a,b]\big\}.$$

The final motion in the shape space is then given by $A(x(t)) \in \mathbf{U}$.

Now we briefly review the quantities that are needed for a finite dimensional space $\mathbb{R}^d$ to set up the dynamics for a discrete object with respect to its generalized coordinates. For a thorough discussion of these quantities we refer to [Jost, 2008]. The tangent space and dual space at $x \in \mathbb{R}^d$ are denoted by T_x and T_x^*, respectively. In addition to the subscript $\cdot_i$ to denote the i-th coordinate of a point, vector, or dual vector, we use $[\cdot]_i$. Elements $v \in T_x$ correspond to vector fields on the discrete object. The space of symmetric and symmetric positive definite $d \times d$ matrices is denoted by SYM_d and SYM_d^+, respectively. Then a metric on $\mathbf{V}$ is defined by a smooth mapping $M : \mathbf{V} \to \mathrm{SYM}_d^+$. We denote the associated inner product and norm by $\langle v, w \rangle_{M(x)}$ and $\|v\|_{M(x)}$, respectively, with $v, w \in T_x$. For a differentiable function $f : \mathbf{V} \to \mathbb{R}$ the dual vector df and the symmetric bilinear form $d^2 f$ at x are given by:

$$df = \left[\ \tfrac{\partial f}{\partial x_1} \cdots \tfrac{\partial f}{\partial x_d}\ \right]^T$$

and

$$d^2 f = \begin{bmatrix} d^2 f_{1,1} & & d^2 f_{1,d} \\ & \ddots & \\ d^2 f_{d,1} & & d^2 f_{d,d} \end{bmatrix} \tag{2.4}$$

where the entries of $d^2 f$ are

$$d^2 f_{i,j} = \frac{\partial^2 f}{\partial x_j \partial x_i} - \sum_{k=1}^{d} \frac{\partial f}{\partial x_k} \Gamma_{i,j}^k.$$

The $\Gamma^d_{i,j}$ are the Christoffel symbols of the metric $M(x)$. Then for a differentiable function $f : \mathbf{V} \to \mathbb{R}$, its gradient ∇f is defined by:

$$\langle \nabla f, v \rangle_{M(x)} = df(v) \quad \forall v \in T_x$$
$$\nabla f = M^{-1}(x) df$$

and its Hessian $\nabla^2 f$ is:

$$\left\langle \nabla^2 f v, w \right\rangle_{M(x)} = d^2 f(v,w) \quad \forall v, w \in T_x$$
$$\nabla^2 f = M^{-1}(x) d^2 f$$

with respect to the inner product $\langle \cdot, \cdot \rangle_{M(x)}$.

In terms of classical mechanics, our constraints given by Equation 2.2 are of holonomic and scleronomic type. Therefore the resulting equation of motion for an unconstrained system and for a constrained system have the same form, see [Scheck, 2007].

Following the Lagrange formalism, we have to express the kinetic energy and the acting forces in generalized coordinates x_i. The kinetic energy is a quadratic functional and is given by:

$$E_{kin}(x,v) = \frac{1}{2} v^T M(x) v \quad \text{with } v \in T_x, M(x) \in \mathrm{SYM}^d_+.$$

Then a trajectory $x \in \mathcal{C}^2_{a,b}(\mathbb{R}^d)$ solves the following system of d differential equations for $t \in [a,b]$:

$$\frac{d}{dt} \frac{\partial E_{kin}(x,\dot{x})}{\partial \dot{x}_i} - \frac{\partial E_{kin}(x,\dot{x})}{\partial x_i} = F_i, \tag{2.5}$$

where F denotes the generalized force given by a dual vector field on $\mathbb{R}^d$. We split the generalized force into two components $F = F_c + F_{nc}$. In contrast to the nonconservative force F_{nc}, the conservative force $F_c(x)$ can be derived from a potential. That is $F_c(x) = -dE_{pot}(x)$ with the potential $E_{pot} : \mathbf{V} \to \mathbb{R}$. Usually the potential is a sum of potentials $E_{pot} = V_I + V_E$. Inner conservative forces, i.e., forces resulting from shape changes, are modeled by V_I. External conservative forces, e.g., gravitation or center force fields, are described by the potential V_E. Then the acting force can be rewritten as

$$F = -dE_{pot} + F_{nc}.$$

Using this split force, the trajectory defining Equation 2.5 yields

$$\frac{d}{dt} \frac{\partial L(x,\dot{x})}{\partial \dot{x}_i} - \frac{\partial L(x,\dot{x})}{\partial x_i} = [F_{nc}]_i, \tag{2.6}$$

where the Lagrangian $L(x, \dot{x})$ is defined by

$$L(x, \dot{x}) = E_{kin}(x, \dot{x}) - E_{pot}(x).$$

By also considering dissipation effects, we add a damping term $D(x, \dot{x})$ to the Langrangian. Then the equations of motion derived from Equation 2.6 become

$$M(x)\ddot{x} + D(x, \dot{x}) + dE_{pot}(x) = F_{nc}. \tag{2.7}$$

Equation 2.7 covers discretizations of mechanical systems with finite elements, finite differences, simple spring systems, and geometrically motivated discrete systems, see [Terzopoulos et al., 1987, Pentland and Williams, 1989, Shabana, 1997, Baraff and Witkin, 1998, Barbič and James, 2005, Chao et al., 2010]. In simulation, a trajectory $x \in \mathcal{C}^2_{a,b}(\mathbb{R})$ is then given as a solution to the initial value problem given by the d second-order nonlinear ordinary differential equations of Equation 2.7.

A common damping assumption is to use Rayleigh damping. Then $D(x, \dot{x})$ has the form:

$$D(x, \dot{x}) = (\alpha\, M(x) + \beta\, d^2E_{pot}(x))\dot{x} \quad \text{with } \alpha, \beta \in \mathbb{R}. \tag{2.8}$$

We set $F_{nc} = 0$ to guarantee the decoupling of the linearized equations of motion. To linearize Equation 2.7 around a state $\mathbf{x} \in \mathbb{R}^d$, we set $x(t) = \mathbf{x} + u(t)$ where $u : \mathbb{R} \to T_{\mathbf{x}}$. By Taylor expansion, a first-order approximation of the conservative forces in a direction $v \in T_{\mathbf{x}}$ is given by

$$dE_{pot}(\mathbf{x} + tv) \approx dE_{pot}(\mathbf{x}) + td^2E_{pot}(\mathbf{x})v. \tag{2.9}$$

Replacing $\mathbf{G} = dE_{pot}(\mathbf{x})$, $\mathbf{K} = d^2E_{pot}(\mathbf{x})$ and $\mathbf{M} = M(\mathbf{x})$ in Equation 2.7 the linearized equations of motion around the state $\mathbf{x}$ are then given by

$$\mathbf{M}\ddot{u} + (\alpha\mathbf{M} + \beta\mathbf{K})\dot{u} + \mathbf{K}u + \mathbf{G} = 0. \tag{2.10}$$

The set of equations defined by Equation 2.10 is a system of second order coupled linear ordinary differential equations.

By construction, we have $\mathbf{M} \in \mathrm{SYM}^+_d$ and $\mathbf{K} \in \mathrm{SYM}_d$. Following [Pentland and Williams, 1989], the solution to the generalized eigenvalue problem

$$\lambda_i \mathbf{M}\phi_i = \mathbf{K}\phi_i \tag{2.11}$$

can be used to decouple Equation 2.10. We assemble the eigenvectors $\phi_i \in T_{\mathbf{x}}$ and eigenvalues λ_i into the matrices Φ and Λ, where

$$\Phi = \begin{bmatrix} | & & | \\ \phi_1 & \dots & \phi_d \\ | & & | \end{bmatrix}_{d\times d} \quad \Lambda = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_d \end{bmatrix}_{d\times d}$$

We refer to the coordinates with respect to the decoupling basis Φ as ω-coordinates. The expressions for $\mathbf{M}$, $\mathbf{K}$ and $\mathbf{G}$ in ω-coordinates are

$$\begin{aligned}\mathbb{1} &= \Phi^T \mathbf{M} \Phi \\ \Lambda &= \Phi^T \mathbf{K} \Phi \\ g &= \Phi^T \mathbf{G}.\end{aligned}$$

Then the system of d decoupled equations of motion in ω-coordinates now takes the form

$$\mathbb{1}\ddot{\omega}(t) + (\alpha\,\mathbb{1} + \beta\,\Lambda)\dot{\omega}(t) + \Lambda\,\omega(t) + g = 0, \quad \omega(t) \in \mathbb{R}^d. \tag{2.12}$$

The linearized solution to the full system in Equation 2.7 with respect to ω-coordinates is mapped to a trajectory $x(t)$ in generalized coordinates by

$$x(t) = \mathbf{x} + \Phi\omega(t).$$

In mechanics, the decoupling is usually performed around an equilibrium state $\mathbf{x} \in \mathbb{R}^d$, i.e., where $dE_{pot}(\mathbf{x}) = 0$. The eigenvectors ϕ_i are then referred to as vibration modes, see [Pentland and Williams, 1989]. We can see from the expansion in Equation 2.9 that at an equilibrium state $\mathbf{x}$, an eigenvalue λ_i denotes a change in the potential when deformed along the corresponding eigenvector $\phi_i \in \mathbf{T}_\mathbf{x}$ up to the third-order, that is, $\lambda_i \approx (E_{pot}(x) + \phi_i) - E_{pot}(x)$.

We refer to the set of decoupled equations in Equation 2.12 as the linear dynamic model around $\mathbf{x}$. We see that to determine a linear dynamic model, two mappings are required: a metric $M(x)$ and a potential $V(x)$.

2.2.3 Inner potentials

An inner potential $V(x)$ is used to measure shape changes of a discrete object. We assume a d-dimensional set of generalized coordinates so that the inner potential or, simply, a potential can be expressed by a mapping $V_I : \mathbf{V} \to \mathbb{R}$, where $\mathbf{V} \subseteq \mathbb{R}^d$ as defined in Section 2.2.2. A certain type of potential can be derived from a deformation energy, that is, from a differentiable function $E_D : \mathbf{V} \times \mathbf{V} \to \mathbb{R}^+$ with $E_D(x, x) = 0$. For a discrete objects equipped with a grid structure G, we define a cell based energy $E_D(x_1, x_2)$ by:

$$E_D(x_1, x_2) = \frac{1}{2} \sum_{\sigma_i \in G_E} \omega_{\sigma_i}(x_1) \left(f^s_{\sigma_i}(x_2) - f^s_{\sigma_i}(x_1) \right)^2, \quad \omega_{\sigma_i} \in \mathbb{R}^+. \tag{2.13}$$

The sum in Equation 2.13 runs over a subset of mesh cells given by $G_E \subseteq G$. The functions $f^s_{\sigma_i} : \mathbb{R}^d \to \mathbb{R}$ are evaluated for each mesh cell $\sigma_i \in G_E$, where

s denotes the number of points p_i on which it depends. We define these functions as a composition of two mappings, that is, $f^s_\sigma(x) = h^s_\sigma \circ \pi^s_\sigma(A(x))$. The mapping $h^s_\sigma : \mathbb{R}^{sl} \to \mathbb{R}$ is associated to the cell σ. The function $\pi^s_\sigma : \mathbb{R}^{nl} \to \mathbb{R}^{sl}$ is the projection given by

$$\pi^s_{\sigma_i}\left(\begin{bmatrix} p_1 \\ \vdots \\ p_n \end{bmatrix}\right) = \begin{bmatrix} p_{i_1} \\ \vdots \\ p_{i_s} \end{bmatrix}.$$

The energies defined by Equation 2.13 measure the deviation of the state x_2 from state x_1 based on weighted differences of f^s_σ on each mesh cell $\sigma_i \in G_E$.

Fixing a configuration $\mathbf{x} \in \mathbb{R}^d$, the inner potential derived from a deformation energy becomes $V_I(x) = E_D(\mathbf{x}, x)$. We refer to the set $\mathbf{R} = \left\{x \in \mathbb{R}^d \,\middle|\, V(x) = 0\right\}$ as the set of rest states of the potential function. The derived inner potential measures how much the given state x deviates from the set of rest states $\mathbf{R}$. For $x \in \mathbf{R}$, the potential $V(x)$ reaches its global minimum, that is, we have $dV(x) = 0$ and the matrix $d^2V(x)$ is at least positive semi-definite. In this case, the eigenvectors ϕ_i corresponding to smaller eigenvalues point into the directions with the least increase in energy. For the potential $V(x)$, the partials needed for the computation of $dV(x)$ and $d^2V(x)$ are given by:

$$\begin{aligned}
\frac{\partial V(x)}{\partial x_j} &= \sum_{\sigma_i \in G_E} \omega_{\sigma_i} \left(f^s_{\sigma_i}(x) - f^s_{\sigma_i}(\mathbf{x})\right) \frac{\partial f^s_{\sigma_i}(x)}{\partial x_j} \\
\frac{\partial^2 V(x)}{\partial x_k \partial x_j} &= \sum_{\sigma_i \in G_E} \omega_{\sigma_i} \left(\frac{\partial f^s_{\sigma_i}(x)}{\partial x_k} \frac{\partial f^s_{\sigma_i}(x)}{\partial x_j} + \left(f^s_{\sigma_i}(x) - f^s_{\sigma_i}(\mathbf{x})\right) \frac{\partial^2 f^s_{\sigma_i}(x)}{\partial x_k \partial x_j} \right).
\end{aligned}$$

For $x \in \mathbf{R}$ the computation of the entries of $d^2V(x)$ simplifies because it only involves the first partials of the $f^s_{\sigma_i}$'s. Furthermore since $dV(x) = 0$, we see from Equation 2.4 that the entries depending on the Christoffel symbols vanish for the computation of $d^2V(x)$.

For the deformation energy in Equation 2.13, we further state

Lemma 1 (Kernel) *Let $V(x)$ be a potential and $\Phi(t, x) : \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}^d$ a mapping for which $f^s_{\sigma_i}(x) = f^s_{\sigma_i}(\Phi(t, x))$, $\forall \sigma_i \in G_E$. Then for $x \in \mathbf{R}$ the linearized variation $v \in T_x$ defined by $v = \left.\frac{d\Phi(t,x)}{dt}\right|_{t=0}$ is contained in the kernel of $d^2V(x)$.*

Proof. Since $\Phi(t, x)$ is an invariant of the component function $f^s_{\sigma_i}(x)$, we have

$$\begin{aligned}\left.\frac{df^s_{\sigma_i}(\Phi(t,x))}{dt}\right|_{t=0} &= \left.\frac{\partial f^s_{\sigma_i}(x)}{\partial x_1}\,\frac{d\Phi(t,x)_1}{dt}\right|_{t=0} + \ldots + \left.\frac{\partial f^s_{\sigma_i}(x)}{\partial x_d}\,\frac{d\Phi(t,x)_d}{dt}\right|_{t=0}\\ &= \sum_j \frac{\partial f^s_{\sigma_i}(x)}{\partial x_j} v_j\\ &= 0.\end{aligned}$$

The Taylor expansion of $V(x)$ with $x \in \mathbf{R}$ up to second order is given by

$$\begin{aligned}V(x+tw) &= \sum_{\sigma_i \in G_E} t^2 \frac{\omega_{\sigma_i}}{2} w^T \begin{bmatrix} \frac{\partial f^s_{\sigma_i}(x)}{\partial x_1}\frac{\partial f^s_{\sigma_i}(x)}{\partial x_1} & \cdots & \frac{\partial f^s_{\sigma_i}(x)}{\partial x_1}\frac{\partial f^s_{\sigma_i}(x)}{\partial x_d} \\ & \ddots & \\ \frac{\partial f^s_{\sigma_i}(x)}{\partial x_d}\frac{\partial f^s_{\sigma_i}(x)}{\partial x_1} & \cdots & \frac{\partial f^s_{\sigma_i}(x)}{\partial x_d}\frac{\partial f^s_{\sigma_i}(x)}{\partial x_d} \end{bmatrix} w\\ &= \sum_{\sigma_i \in G_E} t^2 \frac{\omega_{\sigma_i}}{2} \left(\sum_j \frac{\partial f^s_{\sigma_i}(x)}{\partial x_j} w_j\right)\left(\sum_k \frac{\partial f^s_{\sigma_i}(x)}{\partial x_k} w_k\right).\end{aligned}$$

Set $w = v$ and we have $V(x+tv) = 0$, as stated. ■

The dimension of the kernel of $d^2V(x)$ equals the multiplicity of the eigenvalue $\lambda = 0$. We can deduce that if we have n invariants $\Phi_i(t,x)$, the multiplicity of $\lambda = 0$ for $d^2V(x)$ with $x \in \mathbf{R}$ is at least of the dimension of $\operatorname{span}\{d\Phi_1(t,x)/dt, \ldots, d\Phi_n(t,x)/dt\}$.

This concludes our description of a system with a finite number of degrees of freedom and the necessary quantities to define a linearized dynamic model. We will now show some examples that meet these requirements.

2.3 Examples

In this section we give examples of shape spaces $\mathbf{X}$ for one-, two-, and three-dimensional discrete objects and a possible choice of generalized coordinates. For each type, we present functions $V(x)$ and $M(x)$ to set up the dynamics as described in Section 2.2.2.

In Section 2.3.1, we give two examples of discrete objects that can be modeled by polygonal structures. These two examples have a small number of degrees of freedom and use a kinetic energy that corresponds to a metric given by a dense matrix.

Discrete objects modeling deformable objects having a large number of degrees of freedom are presented in Section 2.3.2 and Section 2.3.3. We consider the dynamics for a geometric based thin shell energy for triangulations

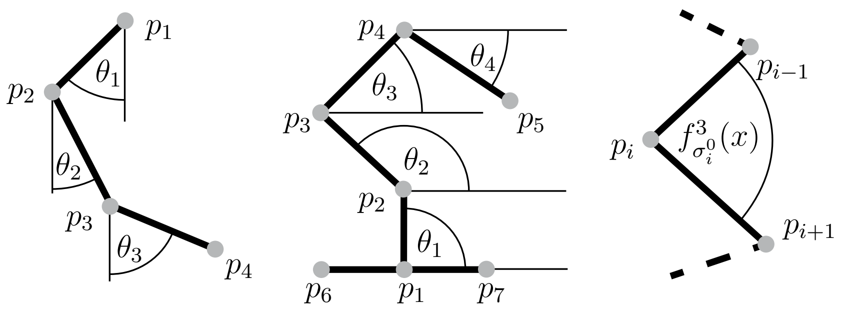

Figure 2.1: Two examples of n-chain models: planar rod model of consisting of four point masses using a 3-chain (left) and the Luxo lamp from [Witkin and Kass, 1988] modeled by a 6-chain (middle). On the right is an illustration of the angle measured by the component function $f^3_{\sigma^0_i}(x)$ which is used in the definition of the potentials for n-chains.

and a model for a three dimensional linear elastic material for tetrahedral sets. For these two object types we use a metric that is given by a sparse, in particular a diagonal, matrix. Thus with the corresponding kinetic energy these objects are treated as a set of point masses.

2.3.1 n-Chain

By an n-chain, we refer to a discrete object equipped with a mesh structure consisting of n edges. This type of discrete object can be used to model a connected system of joints. Two examples are shown in Figure 2.1. We assume that the polygonal mesh corresponds to a connected circle-free graph and that $p_i \in \mathbb{R}^2$, i.e., $l = 2$. In the constraint-free case we have $d = 2(n+1)$ degrees of freedom. We will fix the edge lengths $l_{i,j}$. That is, we impose n constraints of the form in Equation 2.2 by

$$c_i(p_1, \ldots, p_n) = \|p_{i_1} - p_{i_2}\| - l_{i_1,i_2} \quad \forall \sigma^1_i \ \{v_{i_1}, v_{i_2}\} \in G.$$

These n constraints reduce the number of degrees of freedom to $d = n + 2$. We choose as generalized coordinates the position of one vertex and the edge orientations, i.e., a configuration is given by $x = A(x_1, y_1, \theta_1, \ldots, \theta_n)$, see Figure 2.1.

An example of an n-chain is a planar rod shown in Figure 2.1, left. It consists of $n + 1$ connected points p_i with mass m_i. Choosing a certain configuration $\mathbf{x}$, we additionally fix the position of p_1 to the length constraints. Then a possible set of generalized coordinates to parametrize an open set $\mathbf{U} \subset \mathbf{X}$ are the edge orientations, i.e., $p = A(x)$., with $x = (\theta_1, \ldots, \theta_n)$ (see Figure 2.1, left). The quadratic form $M(x)$ defining the kinetic energy of this system is described in the Appendix.

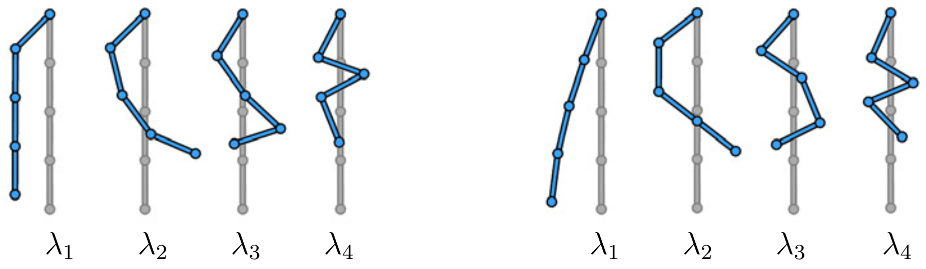

Figure 2.2: Vibration modes for a planar rod having point masses set to one. The rest state is shown in grey. Four vibration modes with non-zero gravitation, i.e., $g = 9.81$ with $0.024 < \lambda_i < 78.0$ (left) and four vibration modes with zero gravitation, i.e., $g = 0.0$ with $0.01 < \lambda_i < 1.25$ (right).

The deformation energy is defined for the vertices $G_E^{Rod} = \left\{\sigma_1^0, \ldots, \sigma_{n-1}^0\right\}$. In the general form of Equation 2.13, it is described by:

$$\omega_{\sigma_i^0} = \frac{1}{2}, \quad f^s_{\sigma_i^0}(x) = \begin{cases} f^2_{\sigma_1^0}(x) = \theta_1 & \sigma_i^0 = \sigma_1^0 \\ f^3_{\sigma_i^0}(x) = \theta_i - \theta_{i-1} & \text{otherwise} \end{cases} \quad \sigma_i^0 \in G_E^{Rod}$$

The associated potential $V(x)$ measures the deviation of the actual configuration x to $\mathbf{x}$ based on differences in θ_1 and relative edge orientations $f^3_{\sigma_i^0}$, see Figure 2.1 right. Examples of all four vibration modes for a 4-chain for a planar rod are shown in Figure 2.2.

The Luxo lamp model was introduced in [Witkin and Kass, 1988]. It can be described by the 6-chain shown in Figure 2.1 middle. In addition to fixing the edge lengths, we also fix the angle between the edges $\{v_1, v_2\}$ and $\{v_1, v_6\}$ and the angle between the edges $\{v_1, v_2\}$ and $\{v_1, v_7\}$. Then a possible set of generalized coordinates to parametrize an open set $\mathbf{U} \subset \mathbf{X}$ is $x = (x_1, y_1, \theta_1, \theta_2, \theta_3, \theta_4)$, that is, the position of one vertex and four edge orientations. In [Witkin and Kass, 1988] the edges σ_i^1 are assumed to model planar rigid bodies. In this case, the quadratic form $M(x)$ describing the kinetic energy corresponds to the sum of the kinetic energy of the connected six planar rigid bodies. Its entries are given in the Appendix. The deformation energy is defined for the vertices $G_E^{Luxo} = \{\sigma_2^0, \sigma_3^0, \sigma_4^0\}$. In the general form of Equation 2.13, it is described by:

$$\omega_{\sigma_i^0} = \frac{1}{2}, \quad f^3_{\sigma_i^0}(x) = \theta_i - \theta_{i-1} \qquad \sigma_i^0 \in G_E^{Luxo}$$

The component functions $f^3_{\sigma_i^0}$ (see Figure 2.1 right) are invariant with respect to planar rigid motions. Hence the kernel of $d^2V(x)$ is two-dimensional for $x \in \mathbf{R}$. The vibration modes corresponding to the four remaining nonzero eigenvalues, i.e., $\lambda_i > 0$, for the Luxo model are shown in Figure 2.3.

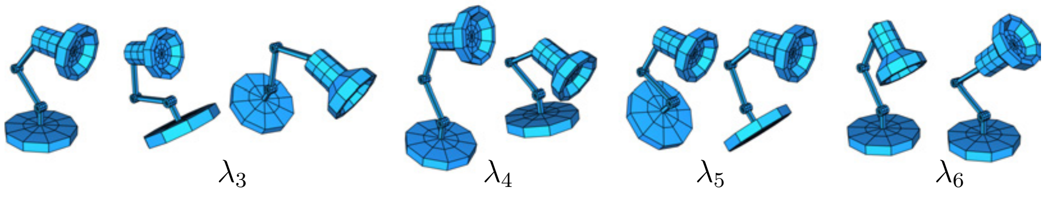

Figure 2.3: Vibration modes for the Luxo model with equal mass distribution. Vibrations for $\lambda_1 = \lambda_2 = 0$ correspond to planar translations. From left to right: rest state and the Luxo model deformed along the four vibrations modes corresponding to the nonzero eigenvalues $\{\lambda_3, \lambda_4, \lambda_5, \lambda_6\}$.

2.3.2 Discrete shells

Thin shells are a class of three-dimensional objects that can be modeled by a two-dimensional surface with thickness $h > 0$, where h is small compared to its diameter. The middle surface of a thin shell is used to describe its dynamic behavior. Two possible ways to derive a dynamic model are by asymptotic analysis or by using geometric considerations.

By using asymptotic analysis, a thin shell energy can be derived by a limit process from an energy defined for a three-dimensional object. For a thorough discussion of asymptotic analysis for thin shells, see [Ciarlet, 2004, Ciarlet, 1997].

Energies based on geometric assumptions use quantities from differential geometry evaluated on the middle surface to define an energy for thin shells. Usually the energies are designed in such a way that they share certain properties with general energies for shape changes used in classical mechanics, e.g., invariance with respect to rigid motions.

Let f be a smooth surface given by a mapping $f : \Omega \subset \mathbb{R}^2 \to \mathbb{R}^3$. Its metric tensor is denoted by $g_f : \Omega \to \mathrm{SYM}_2^+$ and its curvature tensor by $b_f : \Omega \to \mathrm{SYM}_2$. Given the undeformed and deformed state of a surface by mappings f_0 and f_d, respectively, a geometric energy that is invariant to rigid motions to measure the deviation is defined by:

$$E(f, h) = \int_\Omega (\alpha \left\| g_{f_0} - g_{f_d} \right\|^2 + \beta \left\| b_{f_0} - b_{f_d} \right\|^2) dA, \tag{2.14}$$

see [Terzopoulos et al., 1987]. This equation measures the change between f_0 and f_d with respect to differences in metric and curvature properties. The constants $\alpha, \beta \in \mathbb{R}^+$ in Equation 2.14 determine the energy's sensitivity to intrinsic and extrinsic surface changes, respectively. Alternatively geometric energies can be derived from distances between the differential of a deformation and the rotation group, see [Sorkine and Alexa, 2007].

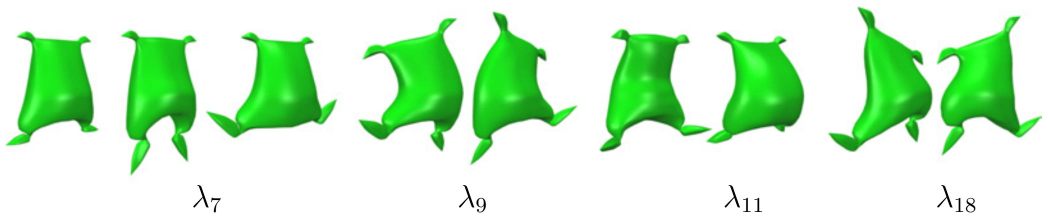

Figure 2.4: Lower vibration modes for the flour sac model. The rest state is shown on the left. Deformations along the vibration modes corresponding to the eigenvalues λ_7, λ_9, λ_{11} and λ_{18} (from left to right). The vibrations for $\lambda_1 = \cdots = \lambda_6 = 0$ correspond to linearized rigid motions.

We will assume that the discrete object describing a smooth surface is given by a simplicial surface $\mathcal{M}$ with $\mathcal{M}_h \subset \mathbb{R}^3$ (see Section 1.1). In the constraint-free case we can choose the set of generalized coordinates by $x = p$, i.e., $x \in \mathbb{R}^{3n}$. To derive a potential, we consider the discrete shells energy introduced in [Grinspun et al., 2003]. This energy is a weighted sum of two energies given by

$$E_{DS}(x_1, x_2) = \alpha E_F(x_1, x_2) + \beta E_M(x_1, x_2). \tag{2.15}$$

The components are referred to as the flexural energy $E_F(\cdot,\cdot)$ and the membrane energy $E_M(\cdot,\cdot)$. The flexural term measures differences in mesh properties related to curvature, whereas the membrane term is sensitive to metric changes.

This separation into an intrinsic and extrinsic term resembles the splitting used in the smooth case given in Equation 2.14. The weights α and β reflect properties of the material to be simulated, for example, in cloth simulation the membrane energy is usually given a high weight due to the stretch resistance of cloth.

The energies used for the parts $E_F(\cdot,\cdot)$ and $E_M(\cdot,\cdot)$ fit into the class of cell based deformation energy defined in Equation 2.13. The flexural term is given as a summation over the edges $\sigma_i^1 \in \mathcal{M}$ of the triangulation. In the general form of Equation 2.13, it is described by:

$$f^4_{\sigma_i^1}(x) = \theta_i(x) \quad \text{and} \quad \omega_{\sigma_i^1}(x) = \frac{3 l_{\sigma_i^1}(x)^2}{A_{\sigma_i^1}(x)}.$$

Here $\theta_i(x)$ is the dihedral angle at the edge σ_i^1. The combined area of the two triangles incident to edge σ_i^1 is denoted by $A_{\sigma_i^1}(x)$ and $l_{\sigma_i^1}(x)$ is the length of the edge.

The membrane energy $E_M(\cdot,\cdot)$ is a sum of two terms: $E_M(x_1, x_2) = E_L(x_1, x_2) + E_A(x_1, x_2)$. The first term $E_L(x_1, x_2)$ is a sum over edges measuring the change in length. In the general form of Equation 2.13, it is described by:

$$f^2_{\sigma^1_i}(x) = l_{\sigma^1_i}(x) \quad \text{and} \quad \omega_{\sigma^1_i}(x) = \frac{1}{l_{\sigma^1_i}(x)}.$$

$E_A(x_1, x_2)$ is a sum over triangles measuring changes in area. In the general form of Equation 2.13, it is described by:

$$f^3_{\sigma^2_i}(x) = A_{\sigma^2_i}(x) \quad \text{and} \quad \omega_{\sigma^2_i}(x) = \frac{1}{A_{\sigma^2_i}(x)},$$

where $A_{\sigma^2_i}(x)$ denotes the area of the triangle σ^2_i. The discrete shell energy $E_{DS}(\cdot,\cdot)$ is only computable on a set of appropriate triangulations $\Omega_{DS} \subset \mathbf{X}$. This set contains only triangulations having triangles of nonzero area, i.e.,

$$\Omega_{DS} = \left\{ x \in \mathbb{R}^d \mid A_{\sigma^2_i}(x) > 0,\ \forall \sigma^2_i \in \mathcal{M} \right\}.$$

Its complement $\Omega^{\complement}_{DS} = \mathbf{X} \backslash \Omega_{DS}$ can be written as the union of a finite number of closed sets:

$$\begin{aligned} \Omega^{\complement}_{DS} &= \bigcup A^0_{\sigma^2_i} \quad \forall \sigma^2_i \in \mathcal{M} \\ A^0_{\sigma^2_i} &= \{x \in \mathbb{R}^d \mid A_{\sigma^2_i}(x) = 0\}. \end{aligned}$$

Hence Ω_{DS} is open. We will assume that a given triangulation corresponds to a configuration $x \in \Omega_{DS}$. This guarantees that the associated discrete shell potential $V_{DS}(x)$ is differentiable.

For the matrix $M_{DS}(x)$ that defines the kinetic energy we use mass lumping. This means $M_{DS}(x)$ is diagonal with entries:

$$m_{3i,3i} = m_{3i-1,3i-1} = m_{3i-2,3i-2} = \frac{1}{3} \sum_{v_i \in \sigma^2_j} A_{\sigma^2_j}(x) \qquad 1 \le i \le n.$$

Restricting to configurations $x \in \Omega_{DS}$ ensures that $M_{DS}(x) \in \mathrm{SYM}^+_{3n}$. Thus for $x \in \Omega_{DS}$ we can determine the linearized decoupled dynamical model by Equation 2.11.

For a given triangulation, the dihedral angles, edge lengths, and triangle areas are invariant with respect to rigid motions. This gives six invariants for the component functions $f^4_{\sigma^1_i}$, $f^2_{\sigma^1_i}$, and $f^3_{\sigma^2_i}$ used in the definition of the flexural and membrane part of $E_{DS}(\cdot,\cdot)$. Thus the kernel of $d^2V_{DS}(x)$ with $x \in \mathbf{R}$ is at least six-dimensional containing the linearized directions of translations and rotations. Deformation captured by vibration modes for the flour sac model are shown in Figure 2.4.

2.3.3 Solids

A configuration of a three-dimensional body is given by a set $\Omega \subset \mathbb{R}^3$, see [Ciarlet, 2005]. In elasticity, changes in Ω are measured with respect to a certain rest state configuration $\Omega_0 \subset \mathbb{R}^3$. We denote the set of k differentiable vector fields on Ω by $V^k(\Omega)$, that is,

$$V^k(\Omega) = \left\{u = (u_1, u_2, u_3) \,|u_i : \Omega \to \mathbb{R} \text{ and } u_i \in C^k \text{ for } i = 1, 2, 3\right\}.$$

For hyperelastic materials, the possible deformations away from the rest state are described by vector fields $u \in V^k(\Omega_0)$. Furthermore, the forces resulting from a deformation can be derived from a potential $V : V^k(\Omega_0) \to \mathbb{R}$. Under the assumption of a St.Venant-Kirchhoff material, i.e., a material that exhibits hyperelastic, isotropic, and homogenous behavior, the potential $V(\cdot)$ is given by:

$$V(u) = \int_{\Omega_0} W(u(x))dx \qquad \text{for} \qquad u \in V^1(\Omega_0)$$

where

$$W(u) = \frac{\lambda}{2}\left(\text{tr}E(u)\right)^2 + 2\mu\text{tr}E(u)^2$$

with $\lambda, \mu \in \mathbb{R}$ and $E \in \text{SYM}_3$ with entries e_{ij} defined by

$$e_{ij} = \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} + \sum_{k=1}^{3} \frac{\partial u_k}{\partial x_i}\frac{\partial u_k}{\partial x_j}.$$

The integrand $W(\cdot)$ defining the potential is called the stored energy function. The matrix $E(u)$ is the Green-St.Venant strain tensor. By definition it removes translational and linearized rotational components from the vector field u. The two scalars λ and μ defining the material properties are referred to as Lamé parameters. If $\mu > 0$ and $3\lambda + 2\mu > 0$, then $W(u) \geq 0 \ \forall u \in V^1(\Omega_0)$. Thus this choice of Lamé parameters ensures that $d^2V(0)$ is positive semi-definite.

Discrete objects describing three-dimensional solids are equipped with meshes containing volumetric cells. We assume that a solid is described by a tetrahedral mesh $\mathcal{V}$ describing a volume $\mathcal{V}_h \subset \mathbb{R}^3$ (see Section 1.1). In [Barbič and James, 2005, Capell et al., 2002a, Capell et al., 2002b] a St.Venant-Kirchhoff material is modeled by tetrahedral sets. In the constraint-free case we have $p = x$, i.e., $x \in \mathbb{R}^{3n}$. We fix a rest state $\mathbf{x} \in \mathbb{R}^{3n}$. A pointwise vector field $v \in T_{\mathbf{x}}\mathbb{R}^{3n}$ is mapped to a piecewise linear vector field v_{PL} on $\mathcal{V}_h$

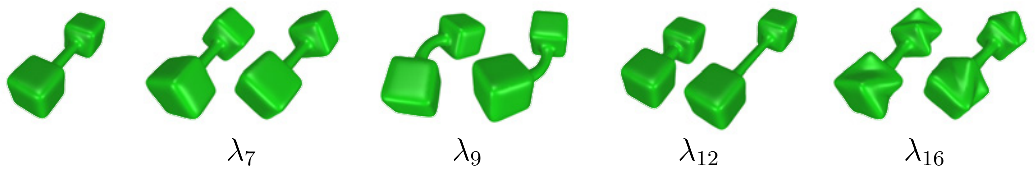

Figure 2.5: Lower vibration modes for an elastic dumbell model. The rest state is shown on the left. Deformations along the vibration modes are shown corresponding to the eigenvalues λ_7, λ_9, λ_{12} and λ_{16} (from left to right). Vibrations for λ_1 to λ_6 correspond to linearized rigid motions.

using the Lagrange basis functions. A vector field v_{PL} is continuous along boundaries of the tetrahedral cells and linear within each tetrahedra $|\sigma_i^3|$. Then a discrete St.Venant-Kirchhoff potential is defined by

$$V_{StVK}(v) = \sum_{\sigma_i^3 \in \mathcal{V}} \int\limits_{|\sigma_i^3|} W(v_{PL}(x))dx \text{ with } v \in T_{\mathbf{x}}\mathbb{R}^{3n}.$$

As shown in [Capell et al., 2002b] this discrete St.Venant-Kirchhoff potential becomes a fourth order polynomial in the components of v. Using the Green-St.Venant strain measure, the kernel of $d^2V_{StVK}(0)$ is at least six-dimensional and contains the linearized rigid motions.

For the matrix $M_{StVK}(x)$ that defines the kinetic energy we use mass lumping. Using $V_{\sigma_j^3}(x)$ for the volume of the tetrahedron σ_j^3 this means $M_{StVK}(x)$ is diagonal with entries:

$$m_{3i,3i} = m_{3i-1,3i-1} = m_{3i-2,3i-2} = \frac{1}{4} \sum_{v_i \in \sigma_j^3} V_{\sigma_j^3}(x) \qquad 1 \le i \le n$$

To ensures the positive definiteness of the mass matrix $M_{StVK}(x)$, we only consider configurations $x \in \mathbb{R}^{3n}$ for which all volumes are nonzero. Restricting to these configurations we can then determine the linearized decoupled dynamical model by Equation 2.11. Deformations captured by the lower vibration modes for an elastic dumbell model are shown in Figure 2.5.

2.4 Spacetime constraint problem

In the beginning of this section, we will give a brief general description of the spacetime constraint problem. In Section 2.4.1 the nonlinear spacetime constraint problem is described. We then derive its linearization in Section 2.4.2. For the linearized model, we will derive its corresponding Euler–Lagrange equations.

The spacetime constraint problem is used to generate physically plausible trajectories $x \in \mathcal{C}^k_{a,b}(\mathbb{R}^d)$ where a and b denote the start and end time, respectively. The physical plausibility of the motion x is measured by a functional $E_{\text{sp}} : \mathcal{C}^k_{a,b}(\mathbb{R}^d) \to \mathbb{R}$. The spacetime constraint problem is then formulated as a constraint minimization problem with the objective function set to $E_{\text{sp}}(\cdot)$. Usually, the set of constraints includes interpolation constraints. We expect this set of interpolation constraints to be given by a set of poses $\mathbf{P}$ with respect to generalized coordinates, i.e., $\mathbf{P} = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$ with $\mathbf{x}_i \in \mathbb{R}^d$ and corresponding interpolation times $\mathbf{T} = \{t_1, \ldots, t_m\}$ with $t_i \in \mathbb{R}$ and $t_{i-1} < t_i$. In particular, the solution to the spacetime constraint problem is then the curve $x \in \mathcal{C}^k_{t_1,t_m}(\mathbb{R}^d)$ that minimizes $E_{\text{sp}}(\cdot)$ meeting the constraints, i.e., $x(t_i) = \mathbf{x}_i$. We can see from the formulation of the problem that the desired behavior of the resulting trajectory is strongly affected by the choice of the energy. Hence the physical aspects being considered have to be encoded in the energy. We will use an energy that depends on velocities and accelerations of a curve $x \in \mathcal{C}^k_{a,b}(\mathbb{R}^d)$. The energy measures the deviation of the given interpolating trajectory to a certain constraint free trajectory.

2.4.1 Nonlinear case

As an energy for trajectories, we choose the energy $E_{f^2}(x) : \mathcal{C}^2_{a,b}(\mathbb{R}^d) \to \mathbb{R}^+$ proposed in [Kass and Anderson, 2008]. Given a metric $M(x)$, a potential $V(x)$, and a trajectory x, this energy is based on the pointwise defined artificial force given by:

$$F_{art}(x, \dot{x}, \ddot{x}) = M(x)\ddot{x} + D(x, \dot{x}) + dV(x), \tag{2.16}$$

where $D(x, \dot{x})$ is the Rayleigh damping term from Equation 2.8. Then the nonlinear energy $E^{full}_{f^2}(x)$ is defined as the integral:

$$E^{full}_{f^2}(x) = \int_{t_1}^{t_m} \left\| M^{-1}(x) F_{art}(x, \dot{x}, \ddot{x}) \right\|^2_M dt. \tag{2.17}$$

We see that if $E^{full}_{f^2}(x) = 0$, the considered curve x solves the initial value problem given in Equation 2.7. Hence the energy measures the deviation of a curve $x(t)$ from a real trajectory determined by a chosen kinetic and potential energy. Then the associated spacetime constraint problem can be stated by:

Problem 1 (Nonlinear Spacetime Problem) *In the space of admissible* *motions $x \in \mathcal{C}^2_{t_1,t_m}(\mathbb{R}^d)$ find*

$$\arg\min_x \; E^{full}_{f^2}(x) \tag{2.18}$$

meeting the interpolation constraints

$$x(t_k) = \mathbf{x}_k \quad \forall \;\; k \in \{1, \ldots, m\}.$$

We can see from this formulation that the solution to Problem 1 will be the solution to a nonlinear constraint optimization problem with the objective function set to Equation 2.17.

2.4.2 Linearization

The artificial force $F_{art}(x, \dot{x}, \ddot{x})$ can be approximated by the linearized dynamical model given in Equation 2.10. Replacing the nonlinear force in Equation 2.17 by the linearized force, the corresponding energy becomes

$$E^{\text{lin}}_{f^2}(u) = \int_{t_1}^{t_m} \left\| \mathbf{M}^{-1} \left(\mathbf{M}\ddot{u} + (\alpha\mathbf{M} + \beta\mathbf{K})\dot{u} + \mathbf{K}u + \mathbf{G} \right) \right\|^2_{\mathbf{M}} dt. \tag{2.19}$$

Similar to the nonlinear energy case (see Equation 2.17), the curves u for which $E^{lin}_{f^2}(u) = 0$ solve a certain system of differential equations. In the case of the linearized energy in Equation 2.19, solutions to the linearized dynamics in Equation 2.10 have zero energy. Expressing the linearized artificial force in ω-coordinates, the energy in Equation 2.19 is given by the sum:

$$E^{\text{lin}}_{f^2}(\omega) = \sum_{i=1}^{d} E_{f^2}(\omega_i), \tag{2.20}$$

where $\omega(t) = [\omega_1(t), \ldots, \omega_d(t)]$ and

$$\begin{aligned} E_{f^2}(\omega_i) &= \int_a^b f^2(\omega_i(t))dt \\ &= \int_a^b \left(\ddot{\omega}_i(t) + (\alpha + \beta\,\lambda_i)\dot{\omega}_i(t) + \lambda_i\,\omega_i(t) + g_i \right)^2 dt. \end{aligned}$$

In ω-coordinates, the energy in Equation 2.19 is split into d independent one-dimensional integrals $E_{f^2}(\omega_i)$. We use $\boldsymbol{\kappa}_k \in T_{\hat{\mathbf{x}}}$ to denote the k-th pose expressed in ω-coordinates, i.e., $\boldsymbol{\kappa}_k = \Phi^T \mathbf{M}(\mathbf{x}_k - \hat{\mathbf{x}})$. Thus using the decoupled expression from Equation 2.20, the linearized spacetime constraint problem can be formulated by d independent one-dimensional problems. That means we have to solve d problems, where the i-th problem with $1 \leq i \leq d$ is of the form

Problem 2 (Decoupled Spacetime Problem) *In the space of admissible curves $\omega_i \in \mathcal{C}^2_{t_1,t_m}(\mathbb{R})$ find*

$$\arg\min_{\omega_i} \; E_{f^2}(\omega_i), \tag{2.21}$$

where ω_i meets the interpolation constraints

$$\omega_i(t_k) = [\boldsymbol{\kappa}_k]_i \quad \forall\, k \in \{1, ..., m\}.$$

An extremizer of Problem 2 is characterized by the corresponding Euler–Lagrange equation, which is given by

Lemma 2 *The Euler–Lagrange equation of the one-dimensional Problem 2 is the fourth-order ODE*

$$\begin{aligned} &\ddot{\ddot{\omega}}_i(t) + 2(\lambda_i - 2\delta_i^2)\,\ddot{\omega}_i(t) + \lambda_i^2\,\omega_i(t) + \lambda_i\, g_i = 0 \\ &\text{with } \delta_i = \frac{1}{2}(\alpha + \beta\,\lambda_i), \end{aligned} \tag{2.22}$$

where the admissible variations away from a minimizer ω_i are given by $\omega_h = \omega_i + hu$ with $u \in C^2_0\,[t_1, t_m]$, $u(t_k) = 0, h \in \mathbb{R}$.

Proof.

The energy of the variation ω_h is given by

$$E_{f^2}\,(\omega_h) = \int_{t_1}^{t_m} (\ddot{\omega}_i + h\ddot{u} + 2\delta_i\,(\dot{\omega}_i + h\dot{u}) + \lambda_i\,(\omega_i + hu) + g_i)^2\, dt.$$

Split the integral into a sum over $m-1$ segments, and denote by $\omega_{i,k}$ the restriction of ω_i to $[t_k, t_{k+1}]$. We have

$$E_{f^2}\,(\omega_h) = \sum_{k=1}^{m-1} \int_{t_k}^{t_{k+1}} (\, \ddot{\omega}_{i,k} + h\ddot{u} + 2\delta_i\,(\dot{\omega}_{i,k} + h\dot{u}) + \lambda_i\,(\omega_{i,k} + hu) + g_i\,)^2\, dt.$$

Now differentiate.

$$\left.\frac{\delta_i E_{f^2}(\omega_h)}{\delta_i h}\right|_{h=0} = \sum_{k=1}^{m-1} \int_{t_k}^{t_{k+1}} (\ \ddot{\omega}_{i,k} + 2\delta_i \dot{\omega}_{i,k} + \lambda_i \omega_{i,k} + g_i)\,(\ddot{u} + 2\delta_i \dot{u} + \lambda_i u)\, dt$$

Rearranging terms gives

$$\begin{aligned} = \sum_{k=1}^{m-1} \int_{t_k}^{t_{k+1}} (\ & \underbrace{(\ddot{\omega}_{i,k} + 2\delta_i \dot{\omega}_{i,k} + \lambda_i \omega_{i,k} + g_i)}_{I_k} \ddot{u} \\ & \underbrace{+ \left(2\delta_i \ddot{\omega}_{i,k} + 4\delta_i^2 \dot{\omega}_{i,k} + 2\delta_i \lambda_i \omega_{i,k} + 2\delta_i g_i\right)}_{II_k} \dot{u} \\ & \underbrace{+ \left(\lambda_i \ddot{\omega}_{i,k} + 2\delta_i \lambda_i \dot{\omega}_{i,k} + \lambda_i^2 \omega_{i,k} + \lambda_i g_i\right)}_{III_k} u\ )\, dt. \end{aligned}$$

Integrating by parts twice yields

$$= \sum_{k=1}^{m-1} \int_{t_k}^{t_{k+1}} \left(\ddot{I}_k - \dot{II}_k + III_k\right) u\ dt + \left.\underbrace{(I_k \dot{u} - \dot{I}_k u + II_k u)}_{B_k}\right|_{t_k}^{t_{k+1}}.$$

Using the fact that successive boundary terms for t_k with $1 < k < m$ cancel each other, we get

$$= \sum_{k=1}^{m-1} \left[\int_{t_k}^{t_{k+1}} \left(\ddot{I}_k - \dot{II}_k + III_k\right) u\ dt \right] + B_m(t_m) - B_1(t_1).$$

As required, u has compact support. Hence the boundary terms $B_m(t_m)$ and $B_1(t_1)$ vanish. Since ω_i is a minmizer, that is,

$$\left.\frac{\delta_i E_{f^2}(\omega_h)}{\delta_i h}\right|_{h=0} = 0 \quad \forall u \in C_0^2\,[t_1, t_m]\,,$$

we conclude that

$$\left(\ddot{I}_k - \dot{II}_k + III_k\right) u = 0.$$

Expanding I_k, II_k, and III_k, we get

$$\ddddot{\omega}_{i,k}(t) + 2(\lambda_i - 2\delta_i^2)\,\ddot{\omega}_{i,k}(t) + \lambda_i^2\,\omega_{i,k}(t) + \lambda_i\, g_i = 0$$

as desired. ■

We can further characterize the extremizer $\omega_i(t)$ by:

Lemma 3 *The energy $E_{f^2}(\omega_i)$ is positive definite with respect to variations of type $\omega_h = \omega_i + hu$ with $u \in C_0^2\,[t_1, t_m]$.*

Proof.

The second variation of $E_{f^2}(\omega_h)$ is given by

$$\frac{\delta^2 E_{f^2}(\omega_h)}{\delta h^2} = \sum_{k=1}^{m-1} \int_{t_k}^{t_{k+1}} \left(\ddot{u} + 2\delta_i \dot{u} + \lambda_i u\right)^2 dt.$$

Therefore for $u \neq 0$ we get $\dfrac{\delta^2 E_{f^2}(\omega_h)}{\delta h^2} > 0$ as stated. ■

Since the energy is positive definite, a solution to the Euler–Lagrange equation must minimize the linearized spacetime constraint Problem 2.

From Equation 2.22, we conclude that a minimizer u of the linearized spacetime Problem 2.19 is a function that is four times continuously differentiable within all intervals (t_k, t_{k+1}) and is twice continuously differentiable at $t_k \in \mathbf{T}$.

The fourth-order Euler–Lagrange Equation 2.22 can be rewritten as two coupled second-order equations $h_1(x)$ and $h_2(x)$ by:

$$\begin{aligned} 0 &= \ddot{h}_1(x) - (\alpha + \beta\lambda)\dot{h}_1(x) + \lambda h_1(x) \\ h_1(x) &= \ddot{h}_2(x) + (\alpha + \beta\lambda)\dot{h}_2(x) + \lambda h_2(x) + g \end{aligned}$$

A solution to Equation 2.22 also solves this system of second-order differential equations. We see that a solution $h_2(t)$ describes a forced vibration. The force driving the motion is given by $h_1(t)$, which is a solution to a second order differential equation.

2.5 Solving the linearized problem

We showed in Section 2.4 that the minimizer of the linearized problem is characterized by a fourth-order linear ordinary differential equation. We start this section with the computation of the explicit solutions to the Euler–Lagrange equation derived in Section 2.5.1. Based on these explicit solutions, we demonstrate two strategies to compute a trajectory. In Section 2.5.2 we show how the trajectory can be defined as a minimizer of a quadratic energy. As an alternative, we show in Section 2.5.3 that the computation of the curve by boundary constraints amounts to solving a low-dimensional banded structure system.

2.5.1 Wiggly base

The solutions to the Euler–Lagrange Equation 2.22 solve a fourth-order linear ordinary differential equation. The solutions are contained in a four-dimensional (affine) vector space. Therefore the restriction of the minimizer to any interval $[t_k, t_{k+1}]$ is a combination of four basis functions $b_i^l(t)$ with $l \in \{1, 2, 3, 4\}$. Then, for every $k \in \{1, 2, ..., m-1\}$ there are four coefficients $w_{i,k}^1, w_{i,k}^2, w_{i,k}^3, w_{i,k}^4$ such that

$$\omega_i(t)\,|_{[t_k,t_{k+1})} = \omega_{i,k}(t) = \sum_{l=1}^{4} w_{i,k}^l b_i^l(t) - c_i. \tag{2.23}$$

In the generic case where $\delta_i \neq 0$ and $\delta_i^2 - \lambda_i \neq 0$ the space of complex solutions of the Euler–Lagrange Equation 2.22 is spanned by the four complex functions

$$b_i^l(t) = e^{\left(\pm\delta_i \pm \sqrt{\delta_i^2 - \lambda_i}\right)t}, \quad l \in \{1, 2, 3, 4\}\,. \tag{2.24}$$

We will only use solutions of the one-dimensional spacetime constraint problem over the real numbers. However, the complex solutions are interesting as well; [Kass and Anderson, 2008] show great examples of how they can be used for designing and augmenting motions of characters.

The type of functions that are in the space of real solutions depends on the values of λ_i and δ_i. We classify these different types into six cases and explicitly list the basis functions $\{b_i^1(t), b_i^2(t), b_i^3(t), b_i^4(t)\}$ that span the spaces of solutions. For brevity, we set $\eta_i = \sqrt{|\delta_i^2 - \lambda_i|}$, which is the frequency in the case of oscillation.

We split the six fundamental solutions into two categories. We distinguish between two main cases and four special cases. The two main cases are:

I. When $\delta_i^2 - \lambda_i < 0$, the mesh oscillates in the direction of the mode ϕ_i (the square root in Equation 2.24 is imaginary) and the basis functions are:

$$\begin{aligned} b_i^1(t) &= e^{-\delta_i t} \cos(\eta_i t) & b_i^2(t) &= e^{-\delta_i t} \sin(\eta_i t) \\ b_i^3(t) &= e^{\delta_i t} \cos(\eta_i t) & b_i^4(t) &= e^{\delta_i t} \sin(\eta_i t)\,. \end{aligned}$$

II. When $\delta_i^2 - \lambda_i > 0$, the mesh exponentially decays or grows in the direction of ϕ_i and the basis functions are:

$$\begin{aligned} b_i^1(t) &= e^{(-\delta_i + \eta_i)t} & b_i^2(t) &= e^{(-\delta_i - \eta_i)t} \\ b_i^3(t) &= e^{(\delta_i + \eta_i)t} & b_i^4(t) &= e^{(\delta_i - \eta_i)t}. \end{aligned}$$

The four special cases are

1. $\delta_i = 0$ and $\lambda_i > 0$ with basis functions:

$$\begin{aligned} b_i^1(t) &= \cos(\eta_i t) & b_i^2(t) &= \sin(\eta_i t) \\ b_i^3(t) &= t\,\cos(\eta_i t) & b_i^4(t) &= t\,\sin(\eta_i t)\,. \end{aligned}$$

2. $\delta_i = 0$ and $\lambda_i < 0$ or $\eta_i = 0$ and $\lambda_i > 0$ with basis functions:

$$\begin{aligned} b_i^1(t) &= e^{-\sqrt{|\lambda_i|}t} & b_i^2(t) &= e^{\sqrt{|\lambda_i|}t} \\ b_i^3(t) &= t\,e^{-\sqrt{|\lambda_i|}t} & b_i^4(t) &= t\,e^{\sqrt{|\lambda_i|}t}. \end{aligned}$$

3. $\delta_i \neq 0$ and $\lambda_i = 0$ with basis functions:

$$\begin{aligned} b_i^1(t) &= 1 & b_i^2(t) &= t \\ b_i^3(t) &= \frac{e^{-2\delta_i t}}{4\delta_i^2} & b_i^4(t) &= \frac{e^{2\delta_i t}}{4\delta_i^2}. \end{aligned}$$

4. $\delta_i = \lambda_i = 0$. In this case $\omega_i(t)$ is a B-spline with basis functions:

$$\begin{aligned} b_i^1(t) &= 1 & b_i^2(t) &= t \\ b_i^3(t) &= t^2 & b_i^4(t) &= t^3. \end{aligned}$$

The constant c_i in Equation 2.23 is given by

$$c_i = \begin{cases} 0 & \lambda_i = 0 \\ g_i / |\lambda_i| & \lambda_i \neq 0. \end{cases}$$

Now that we have the analytic expressions of the basis functions we can compute the following type dependent quantities for a wiggly spline $\omega_i(t)$:

$$\begin{aligned} b_i^{l,\alpha}(t) &= \frac{d^\alpha b_i^l(t)}{dt^\alpha} \\ B_{i,k}^{l,\alpha} &= \int_{t_k}^{t_{k+1}} b_i^{l,\alpha}(t)dt \\ B_{i,k}^{l_1,l_2,\alpha_1,\alpha_2} &= \int_{t_k}^{t_{k+1}} b_i^{l_1,\alpha_1}(t) b_i^{l_2,\alpha_2}(t)dt \end{aligned}$$

with $l \in \{1,2,3,4\}$, $a \in \mathbb{N}_0$ and $k \in [1,\ldots,m-1]$.

With these quantities we are able to define the vectors $\mathbf{b}_i^\alpha(t)$, $\mathbf{B}_{i,k}^\alpha$, and the matrix $\mathbf{B}_{i,k}^{\alpha_1,\alpha_2}$ by:

$$\begin{aligned}\mathbf{b}_i^\alpha(t) &= [b_i^{1,\alpha}(t), b_i^{2,\alpha}(t), b_i^{3,\alpha}(t), b_i^{4,\alpha}(t)]\\ \mathbf{B}_{i,k}^\alpha &= \left[B_{i,k}^{1,\alpha}, \dots B_{i,k}^{4,\alpha}\right]\\ \mathbf{B}_{i,k}^{\alpha_1,\alpha_2} &= \begin{bmatrix} B_{i,k}^{1,1,\alpha_1,\alpha_2} & \dots & B_{i,k}^{1,4,\alpha_1,\alpha_2}\\ \vdots & \ddots & \vdots\\ B_{i,k}^{4,1,\alpha_1,\alpha_2} & \dots & B_{i,k}^{4,4,\alpha_1,\alpha_2}\end{bmatrix}.\end{aligned} \tag{2.25}$$

Furthermore, we collect the wiggly coefficients $\omega_{i,k}^l$ of the k-th segment in

$$\boldsymbol{\omega}_{i,k} = \left[\omega_{i,k}^1, \omega_{i,k}^2, \omega_{i,k}^3, \omega_{i,k}^4\right]^T. \tag{2.26}$$

Using the previously defined quantities, the evaluation of the wiggly spline and its first and second derivative can be written as;

$$\begin{aligned}\omega_i(t)\,|_{[t_k,t_{k+1})} &= \mathbf{b}_i^0(t)\boldsymbol{\omega}_{i,k} - c_i\\ \dot\omega_i(t)\,|_{[t_k,t_{k+1})} &= \mathbf{b}_i^1(t)\boldsymbol{\omega}_{i,k}\\ \ddot\omega_i(t)\,|_{[t_k,t_{k+1})} &= \mathbf{b}_i^2(t)\boldsymbol{\omega}_{i,k}.\end{aligned} \tag{2.27}$$

Since we want curves $x(t) \in \mathcal{C}^2_{t_1,t_m}(\mathbb{R}^d)$, corresponding C^2 regularity conditions have to be enforced for $\omega(t) = [\omega_1(t), \dots, \omega_d(t)]$. To guarantee these regularity requirements for $\omega(t)$, we require that two successive segments $\omega_{i,k}(t)$ and $\omega_{i,k+1}(t)$ meet smoothly at t_{k+1}. The continuity assumption is given by the interpolation constraint $\omega(t_k) = \boldsymbol{\kappa}_k$. It is expressed for each segment by the two conditions:

$$\omega_{i,k}(t_k) = [\boldsymbol{\kappa}_k]_i \quad \text{and} \quad \omega_{i,k}(t_{k+1}) = [\boldsymbol{\kappa}_{k+1}]_i\,. \tag{2.28}$$

The C^2 transition between two successive segments $\omega_{i,k}(t_k)$ and $\omega_{i,k+1}(t_k)$ at t_{k+1} is enforced by:

$$\dot\omega_{i,k}(t_k) = \dot\omega_{i,k+1}(t_k) \quad \text{and} \quad \ddot\omega_{i,k}(t_k) = \ddot\omega_{i,k+1}(t_k). \tag{2.29}$$

This amounts to $4m-6$ constraints for $4m-4$ unknown wiggly coefficients $\omega_{i,k}^l$. In the following sections, we present two approaches to uniquely determine the wiggly spline $\omega_i(t)$: by using a quadratic energy or by the prescription of boundary conditions.

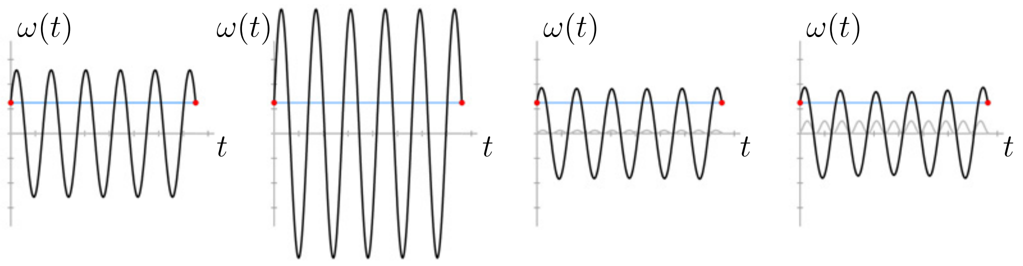

Figure 2.6: Comparison of curves (black) minimizing $E_{f^2}(\omega_i(t))$ with regularized minimizers $E_{reg}(\omega_i(t))$ with $\alpha = 1$ and $\beta = \gamma = 0$. Also plotted is the squared artificial force $f^2(\omega_i(t))$ (light grey). The two curves on the left are computed for a slightly pertubed boundary condition. Both have $E_{f^2}(\omega_i(t)) = 0$. The two curves on the right are computed for the same boundary conditions using a regularizing term resulting in $E_{f^2}(\omega_i(t)) > 0$.

2.5.2 Energy minimizer

As shown in Section 2.2.2, the metric expressed in ω-coordinates is given by the identity. Using the definitions of $\mathbf{B}^{\alpha}_{i,k}$ and $\mathbf{B}^{\alpha_1,\alpha_2}_{i,k}$ (see Equation 2.25), we can compute the energy $E^{lin}_{f^2}(\omega(t))$ (see Equation 2.20) explicitly. The corresponding d independent energy terms are given by:

$$\begin{aligned} E_{f^2}(\omega_i(t)) = \sum_{k=1}^{m-1} \boldsymbol{\omega}^T_{i,k} \big(\mathbf{B}^{2,2}_{i,k} & + 4\delta_i \mathbf{B}^{1,2}_{i,k} + 2\lambda_i \mathbf{B}^{0,2}_{i,k} \\ & + 4\delta_i^2 \mathbf{B}^{1,1}_{i,k} + 4\delta_i\lambda_i \mathbf{B}^{0,1}_{i,k} + \lambda_i^2 \mathbf{B}^{0,0}_{i,k}\big)\, \boldsymbol{\omega}_{i,k}. \end{aligned} \tag{2.30}$$

The computation of $E_{f^2}(\omega_i(t))$ involves only quadratic terms, the constant term and the linear term vanish. The set of minimizers, i.e., curves for which $E_{f^2}(\omega_i(t)) = 0$, solve the initial value problem given in Equation 2.10. These minimizing solutions can also be characterized as solutions to a boundary value problem associated to the initial value problem from Equation 2.10. Depending on the constraints, this boundary value problem can have a unique solution, but possibly also multiple or no solution. Thus the set of our energy minimizers (see Equation 2.19) also contains the corresponding ill-conditioned solutions. These motions typically oscillate with a large amplitude as shown in Figure 2.6 left. Therefore, we add a regularizing term to $E_{f^2}(\omega_i(t))$. Using $\mathbf{B}^{\alpha}_{i,k}$ and $\mathbf{B}^{\alpha_1,\alpha_2}_{i,k}$ (see Equation 2.25), we can compute the following energies explicitly:

$$E_{L^2}(\omega_i(t)) = \int_{t_1}^{t_m} \omega_i(t)^2 dt = \sum_{k=1}^{m-1} \boldsymbol{\omega}_{i,k}^T \mathbf{B}_{i,k}^{0,0} \boldsymbol{\omega}_{i,k} - 2c_i(\mathbf{B}_{i,k}^{0})\boldsymbol{\omega}_{i,k} + c_i^2$$

$$E_v(\omega_i(t)) = \int_{t_1}^{t_m} \dot{\omega}_i(t)^2 dt = \sum_{k=1}^{m-1} \boldsymbol{\omega}_{i,k}^T \mathbf{B}_{i,k}^{1,1} \boldsymbol{\omega}_{i,k}$$

$$E_a(\omega_i(t)) = \int_{t_1}^{t_m} \ddot{\omega}_i(t)^2 dt = \sum_{k=1}^{m-1} \boldsymbol{\omega}_{i,k}^T \mathbf{B}_{i,k}^{2,2} \boldsymbol{\omega}_{i,k}.$$

Similar to the computation of $E_{f^2}(\omega_i(t))$, these energies are defined as sums of quadratic terms for each segment. So for $\alpha, \beta, \gamma \in \mathbb{R}^+$ we can define a general type of regularized energy by:

$$E_{reg}(\omega_i(t)) = E_{f^2}(\omega_i(t)) + \alpha E_{L^2}(\omega_i(t)) + \beta E_v(\omega_i(t)) + \gamma E_a(\omega_i(t)). \quad (2.31)$$

In Figure 2.6 an ill-conditioned minimizer of E_{f^2} is compared to minimizer of E_{reg} using the same boundary conditions.

The wiggly spline shape can also be controlled by prescribing soft interpolating constraints, i.e., an interpolation that is optimal in the least square sense. Denoting the pose that has to be soft interpolated at time $\bar{t}$ by $\bar{\boldsymbol{\kappa}}$, the corresponding least square energy is given by:

$$\|\omega(\bar{t}) - \bar{\boldsymbol{\kappa}}\|^2 = \sum_{k=1}^{m-1} \sum_{i=1}^{d} E_{k,soft}\left(\omega_i(\bar{t}), [\bar{\boldsymbol{\kappa}}]_i\right)$$

where

$$E_{k,soft}\left(\omega_i(\bar{t}), [\bar{\boldsymbol{\kappa}}]_i\right) = \begin{cases} E_{k,ls}\left(\omega_i(\bar{t}), [\bar{\boldsymbol{\kappa}}]_i\right) & \bar{t} \in [t_k\ ,\ t_{k+1}) \\ 0 & otherwise \end{cases}$$

with

$$E_{k,ls}\left(\omega_i(\bar{t}), [\bar{\boldsymbol{\kappa}}]_i\right) = \boldsymbol{\omega}_{i,k}^T \mathbf{b}_i^0(\bar{t})^T \mathbf{b}_i^0(\bar{t}) \boldsymbol{\omega}_{i,k} - 2([\bar{\boldsymbol{\kappa}}]_i + c_i)\mathbf{b}_i^0(\bar{t})\boldsymbol{\omega}_{i,k} + ([\bar{\boldsymbol{\kappa}}]_i + c_i)^2$$

Using the relations from Equation 2.27, soft interpolation constraints for velocities and accelerations can be formulated in a similar manner. We can now define a C^2 interpolating wiggly spline $\omega_i(t)$ as the extremizer of a constrained optimization problem. For the objective function, we choose an energy described in Equation 2.31. The C^2-regularity assumption enters as

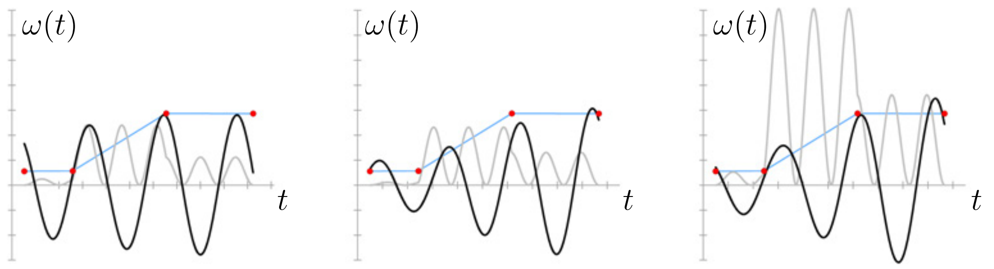

Figure 2.7: An example of a wiggly splines (black curve) with soft interpolation constraints (blue polygon) Also shown is the resulting squared artificial force $f^2(\omega_i(t))$ (light grey). The least square weights are varied, that is, from left to right they are set to: $\{1, 10, 10, 1\}$, $\{10, 1, 1, 10\}$ and $\{10, 10, 10, 10\}$.

equality constraints. Using Langrange multipliers l_k^i and assuming n_{ls} soft interpolation constraints with weights $\eta_j \in \mathbb{R}$, we define the wiggly spline coefficient to be the extremizer of:

$$\begin{aligned}
E(\omega_i(t)) = & E_{reg}(\omega_i(t)) + \sum_{j=1}^{n_{ls}} \eta_j \sum_{k=1}^{m-1} E_{k,soft}(\omega_i(\bar{t}_j), [\bar{\boldsymbol{\kappa}}_j]_i) \\
& \overbrace{- \sum_{k=1}^{m-1} l_k^1 \left([\boldsymbol{\kappa}_k]_i - \mathbf{b}_i^0(t_k)\boldsymbol{\omega}_{i,k}(t_k) + c_i \right)}^{\text{interpolation at } t_k\colon\, \omega_{i,k}(t_k)=[\boldsymbol{\kappa}_k]_i} \\
& \overbrace{- \sum_{k=1}^{m-1} l_k^2 \left([\boldsymbol{\kappa}_{k+1}]_i - \mathbf{b}_i^0(t_{k+1})\boldsymbol{\omega}_{i,k}(t_{k+1}) + c_i \right)}^{\text{interpolation at } t_{k+1}\colon\, \omega_{i,k}(t_{k+1})=[\boldsymbol{\kappa}_{k+1}]_i} . \\
& \overbrace{- \sum_{k=2}^{m-1} l_k^3 \left(\mathbf{b}_i^1(t_k)\boldsymbol{\omega}_{i,k-1} - \mathbf{b}_i^1(t_k)\boldsymbol{\omega}_{i,k} \right)}^{C^1 \text{ at } t_k\colon\, \dot{\omega}_{i,k-1}(t_k)-\dot{\omega}_{i,k}(t_k)=0} \\
& \overbrace{- \sum_{k=2}^{m-1} l_k^4 \left(\mathbf{b}_i^2(t_k)\boldsymbol{\omega}_{i,k-1} - \mathbf{b}_i^2(t_k)\boldsymbol{\omega}_{i,k} \right)}^{C^2 \text{ at } t_k\colon\, \ddot{\omega}_{i,k-1}(t_k)-\ddot{\omega}_{i,k}(t_k)=0}
\end{aligned} \tag{2.32}$$

Since this is a quadratic energy the computation of the extremizer involves solving a linear system of equations of dimension $8m - 10$. By the construction of E_{reg} and E_{soft} the found wiggly spline coefficients also minimize Equation 2.32. An example of an energy minimizing spline with soft interpolation constraints is shown in Figure 2.7.

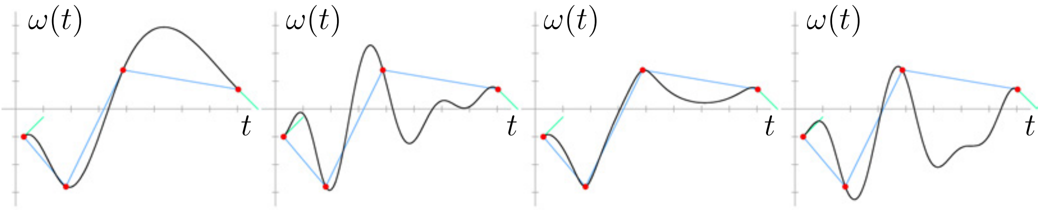

Figure 2.8: Wiggly splines (black) with varying stiffness and damping parameters, interpolating polygon (blue), and boundary conditions (turquoise). From left to right the parameters are: $(\lambda = 0, \delta = 0, g = 0)$, $(\lambda = 5, \delta = 0, g = 0)$, $(\lambda = 5, \delta = 5, g = 0)$, $(\lambda = 5, \delta = 0, g = 1)$.

2.5.3 Boundary conditions

At the end of Section 2.5.1, we showed that there are two degrees of freedom left to uniquely determine the wiggly spline ω_i, i.e., its $4m - 4$ wiggly coefficients $\omega_{i,k}^l$. Here we will to prescribe boundary conditions. In particular, we fix a start velocity $\mathbf{v}_{start} \in T_\mathbf{x}$ and an end velocity $\mathbf{v}_{end} \in T_\mathbf{x}$. Expressing these vectors in ω-coordinates, we have $\Psi_{start} = \Phi^T \mathbf{M} \mathbf{v}_{start}$ and $\Psi_{end} = \Phi^T \mathbf{M} \mathbf{v}_{end}$. Then the coressponding boundary conditions become:

$$\dot{\omega}_{i,1}(t_1) = [\Psi_{start}]_i \quad \text{and} \quad \dot{\omega}_{i,m-1}(t_m) = [\Psi_{end}]_i \,. \tag{2.33}$$

With these two additional conditions, the $4m - 4$ wiggly spline coefficients of $\omega_i(t)$ are uniquely determined. As an alternative to these boundary conditions, we could also have set start and end acceleration.

By exploiting the structure of the interpolation and regularity constraints, they can be arranged to form a band matrix of bandwidth 8 by

$$\begin{bmatrix}
\mathbf{b}_i^1(t_1) & & & & & & \\
 & \ddots & & & & & \\
 & & \mathbf{b}_i^1(t_k) & \text{-}\mathbf{b}_i^1(t_k) & & & \\
 & & \mathbf{b}_i^2(t_k) & \text{-}\mathbf{b}_i^2(t_k) & & & \\
 & & & \mathbf{b}_i^0(t_k) & & & \\
 & & & \mathbf{b}_i^0(t_{k+1}) & & & \\
 & & & \mathbf{b}_i^1(t_{k+1}) & \text{-}\mathbf{b}_i^1(t_{k+1}) & & \\
 & & & \mathbf{b}_i^2(t_{k+1}) & \text{-}\mathbf{b}_i^2(t_{k+1}) & & \\
 & & & & & \ddots & \\
 & & & & & & \mathbf{b}_i^1(t_m)
\end{bmatrix}
\begin{bmatrix}
\boldsymbol{\omega}_{i,1} \\ \vdots \\ \vdots \\ \boldsymbol{\omega}_{i,k-1} \\ \boldsymbol{\omega}_{i,k} \\ \boldsymbol{\omega}_{i,k+1} \\ \vdots \\ \vdots \\ \boldsymbol{\omega}_{i,m-1}
\end{bmatrix}
=
\begin{bmatrix}
[\Psi_{start}]_i \\ \vdots \\ 0 \\ 0 \\ [\boldsymbol{\kappa}_k]_i + c_i \\ [\boldsymbol{\kappa}_{k+1}]_i + c_i \\ 0 \\ 0 \\ \vdots \\ [\Psi_{end}]_i
\end{bmatrix} .$$

Since the number of poses m is typically small, solving such a system requires only fractions of a millisecond; even on a custom laptop and without parallelization, one can compute 10K wiggly splines within a second. An

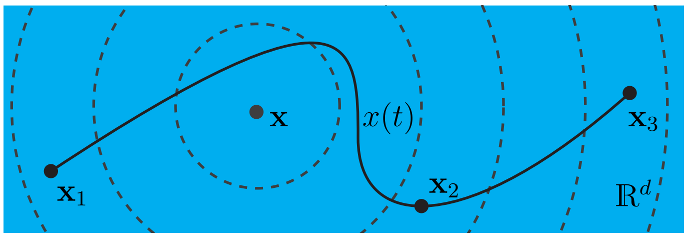

Figure 2.9: An illustration of the one-point dynamic approach. The interpolation poses are $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$. The linarized dynamic model is determined at $\mathbf{x}$. It controls the dynamics in $\mathbb{R}^d$ (concentric circles). The resulting trajectory is $x(t) \in \mathcal{C}^2_{t_1,t_3}(\mathbb{R}^d)$ (black curve).

example of a wiggly spline with fixed boundary velocities and varying λ, δ, and g is shown in Figure 2.8.

2.5.4 Remarks

The proposed computation of the wiggly splines has three major advantages over a finite difference scheme. The first is that the result is a wiggly spline in closed form, not just an approximation. The second is that it has a significantly lower computational cost. Thirdly, we do not need to deal with stability issues caused by discretization. For a thorough discussion of the stability problem caused by a finite difference approximation of derivatives, we refer to [Kass and Anderson, 2008].

Our algorithm directly generalizes to the calculation of complex wiggly splines. Then, to determine a complex wiggly spline, the interpolation Equation 2.28, continuity Equation 2.29, and boundary constraints Equation 2.33 need to be treated as complex equations.

2.6 One-point dynamics

Using the one-point dynamic approach means that the resulting curve $x(t) \in \mathcal{C}^2_{t_1,t_m}(\mathbb{R}^d)$ is based on a single linearized model around a state $\mathbf{x}$. As described in Section 2.2.2, the linearized dynamic model is derived from a metric $M(x)$ and potential $V(x)$. An illustration of the one-point dynamic approach is shown in Figure 2.9. The necessary steps to generate a motion using the one-point dynamic approach are summarized in Algorithm 1. The proposed construction in Algorithm 1 can be split into two parts.

At first, the decoupled linearized model is determined, that is, $\mathbf{\Phi}$ and $\mathbf{\Lambda}$ are computed. This part involves the computation of first- and second-order derivatives of $V(x)$ followed by the solution to the associated d-dimensional generalized eigenvalue problem (see Equation 2.11).

In the second part, the poses and boundary constraints are expressed in ω-coordinates, that is, $m+2$ projections have to be performed, followed by the actual computation of the wiggly coefficients of the d curves $\omega_i(t)$.

The first part contains the computationally expensive operations. Therefore by restricting to operations that leave the linearized dynamic model unaffected, we can shift the expensive computation of the decoupled linearized model to a preprocess.

Algorithm 1: one point dynamics

```
Data: base for dynamic model $\mathbf{x}$, metric $M(x)$, potential $V(x)$,
      poses $\mathbf{P} = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$, times $\mathbf{T} = \{t_1, \ldots, t_m\}$
Result: trajectory $x(t) \in \mathcal{C}^2_{t_1,t_2}(\mathbb{R}^d)$ with $x(t_i) = \mathbf{x}_i$
Preprocess
compute $\mathbf{G}$, $\mathbf{K}$ and $\mathbf{M}$
solve $\lambda_i \mathbf{M}\Phi_i = \mathbf{K}\Phi_i$
Interaction Phase
foreach $\mathbf{x}_i$ do
    map $\mathbf{x}_i$ to $\omega$-coordinates
end
map boundary constraints to $\omega$-coordinates
for $i=1$ to $d$ do
    solve for wiggly spline coefficients $[\boldsymbol{\omega}_{i,1}, \ldots, \boldsymbol{\omega}_{i,m-1}]$
end
```

We call the second step the interaction phase. Within this step we allow all operations that do not involve a recomputation of the decoupling base. The set of allowed operations include:

- change of damping parameters
- change of boundary conditions
- choice of interpolation times (including changing the order of poses)
- deformation/addition/removal of poses $\mathbf{x}_i$
- stiffness control through the use of a scaled potential $s \cdot V(x)$.

In particular, changing the stiffness involves only the uniform rescaling of the λ_i's and g_i's by s. From Section 2.5.1 we see that if $\lambda_i = 0$ the corresponding

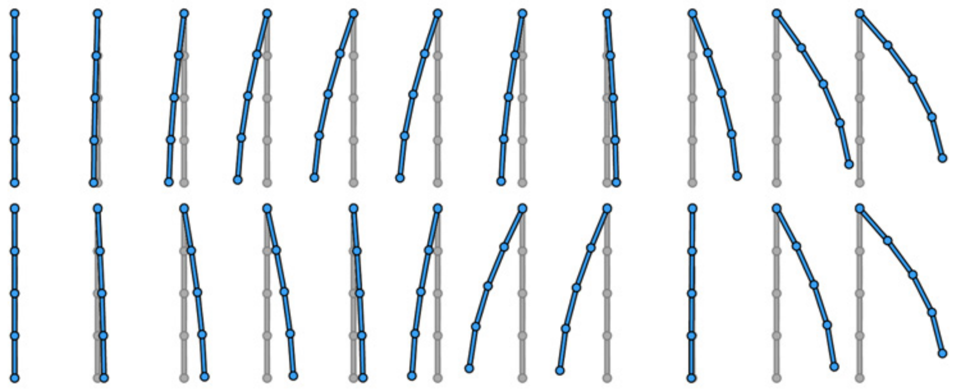

Figure 2.10: Snapshots of two one-point dynamic animations using the same boundary conditions but different stiffness scalings. Low scaling results in only one swing (top row). High scaling causes two swinging motions (bottom row).

curve $\omega_i(t)$ corresponds to a cubic interpolating spline. In case of boundary constraints the coefficients $\boldsymbol{\omega}_{i,k}$ are determined as the solution to d banded linear systems of dimension $4(m-1)$.

To compute the entries of $\mathbf{K} = dV^2(\mathbf{x})$ with $\mathbf{x} \in \mathbf{R}$ for a potential $V(x)$ derived from a general deformation energy, see Equation 2.13, we only need first partials of the component functions $f^s_{\sigma_i}$'s (see Lemma 1). We use the automatic-differentiation library ADOL-C [Griewank et al., 1996] to determine the entries of $\mathbf{K}$.

2.6.1 Experiments

We present three examples using the one point dynamic approach. Two examples are computed for a low-dimensional shape space, a planar rod consisting of four points, and the Luxo model. The last example shows results computed for a thin shell model of a block that is described in a 1800-dimensional shape space.

The planar rod is given by a 4-chain (see Figure 2.10 left). In addition to the length constraint, we fix the the position of the top point. The motion is computed to move from the straight state (see Figure 2.10, left) to the deformed state (see Figure 2.10, right). The results from using different stiffness scalings are shown in Figure 2.10. Higher stiffness causes two swings to reach the final state in contrast to the single swing used in the less stiff setup.

A jump of the Luxo lamp is modeled by three poses as shown in Figure

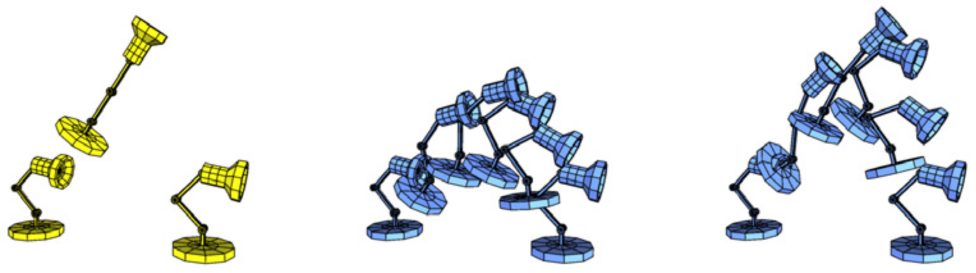

Figure 2.11: The jumping Luxo lamp created from an energy minimizing wiggly spline and soft interpolation. Constant high weights are used for start and end poses and the weight for the middle keyframe is changed. Keyframes in yellow (left). Low least square weight (middle) and increased least square weight (right).

2.11 left. The motion was generated using an energy minimizing wiggly spline, where all poses enter as soft interpolation constraints. Results from varying weights for the middle pose are shown in Figure 2.11 middle and right. Observe that the choice of weight controls the height of the jump.

In the third example, we compute a jumping block animation. The final jump is composed of three wiggly splines resulting in a continous motion. The three curves describe the three parts of the motion: preparing the jump, jumping, and landing.

During the first and last part, the base of the block is fixed to the ground with equality constraints. Each of the three wiggly splines interpolates two successive keyframes and satisfies boundary conditions which prescribe first derivatives (velocities) at all vertices of both keyframes. The prescribed velocities at the first keyframe vanish and all vertices of the second keyframe (except those at the base) have the same upward pointing velocity.

The first and second wiggly spline meet at the second keyframe. Both splines interpolate the second keyframe and have the same velocities at this point. Therefore, the compound curve is differentiable at this keyframe. As a result, the motion anticipates the jump and the force required to accelerate the block is distributed over the time interval.

All vertices of the third keyframe (including the base) have the same velocity pointing to the ground. This models a collision with the ground. Since the base of the block is fixed in the third wiggly spline, the compound curve is continuous but not differentiable at the third keyframe. The effect is that there is no anticipation of the landing and the block collides with the ground. Since the block cannot bounce upwards, the collision causes a deformation of the block and a wave travels up the block.

Snapshots of the final block jump animation are shown in Figure 2.12. For

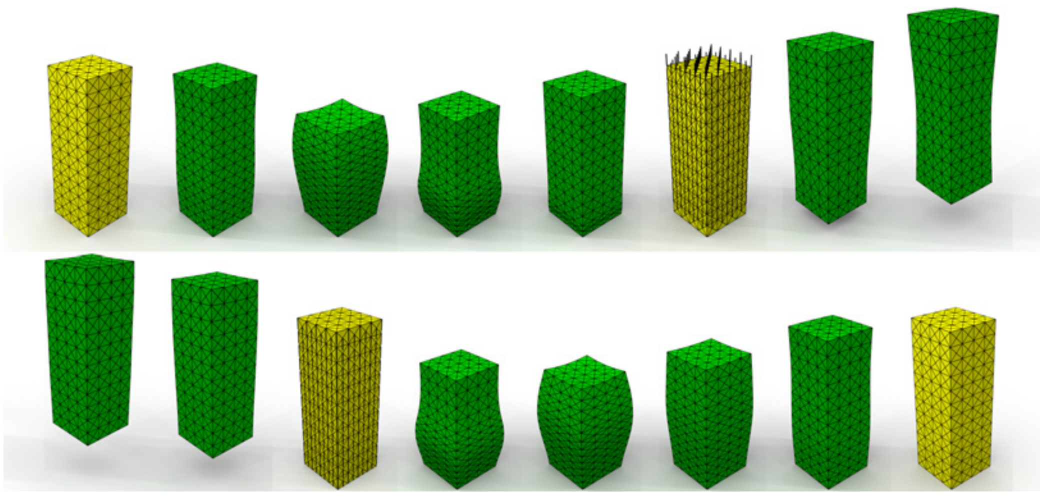

Figure 2.12: Snapshots from an animation of a jumping block that interpolates keyframes (yellow) and velocities at the keyframes (black arrows). Before the jump, the block deforms in order to achieve the prescribed velocity at the second keyframe. During the first part of the animation, the base of the block is fixed to the ground with equality constraints.

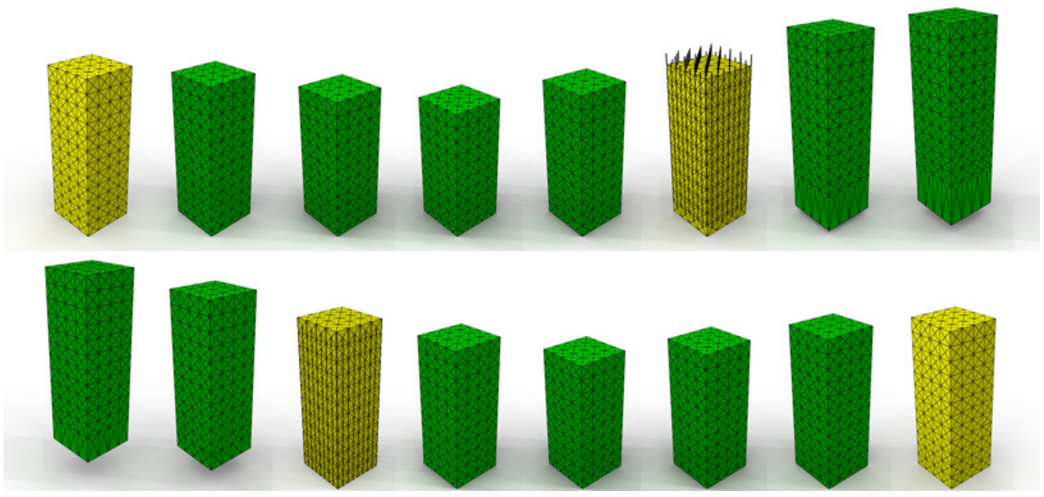

Figure 2.13: Snapshots from an animation of a jumping block using cubic B-splines with the same interpolation and boundary constraints shown in Figure 2.12.

comparison, a motion generated with cubic B-splines that satisfy the same interpolation and boundary conditions is shown in Figure 2.13. Our motions show physical behavior, like waves that travel up and down the block, but the B-spline animation only varies the length of the block. Details for the

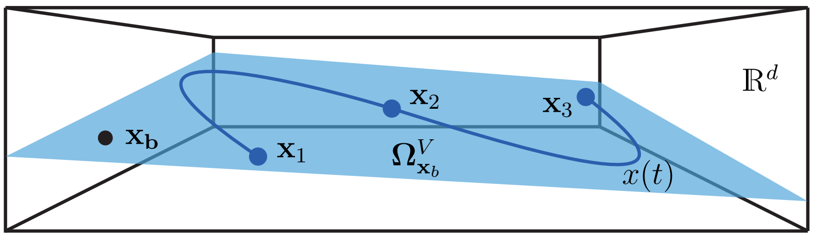

Figure 2.14: Illustration of the reduction approach. The affine subspace $\mathbf{\Omega}^V_{\mathbf{x}_b} \subset \mathbb{R}^d$ (blue plane) is shown. The subspace is given by a base point $\mathbf{x}_b \in \mathbb{R}^d$ and a base V. The motion $x(t)$ (blue curve) is contained in $\mathbf{\Omega}^V_{\mathbf{x}_b}$ and interpolates poses $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3 \in \mathbf{\Omega}^V_{\mathbf{x}_b}$.

block example are summarized in Table 2.1 including timings.

2.7 Reduction

In general, working with discrete objects describing shells or three-dimensional solids involves high dimensional shape spaces, that is, objects with a large number of degrees of freedom as shown in the third example described in Section 2.6.1. We introduce a strategy based on modal analysis to drastically reduce the number of degrees of freedom. Analogous to the full set of d degrees of freedom, we can solve a corresponding reduced linearized space-time problem for the discrete objects with respect to the reduced set of $\bar{d}$ generalized coordinates, with $\bar{d} \ll d$.

Our reasons to use a reduced set of generalized coordinates are threefold. Firstly, the accurate representation of a geometry usually needs high dimensional discrete representation whereas animations often require only a fraction of the degrees of freedom, see [Kim and James, 2009]. Secondly, by using a smaller number of degrees of freedom the computational cost decreases dramatically. Finally, our reduced spaces cut off high frequency modes and thereby lower the stiffness of the optimization problem which in turn increases the numerical robustness of the resulting method.

2.7.1 Reduced dynamic model

We refer to the subset $\mathbf{\Omega}^V_{\mathbf{x}_b} \subseteq \mathbb{R}^d$ as the reduced space, where $\mathbf{x}_b \in \mathbb{R}^d$ and V is a linear subspace of $\mathbb{R}^d$. Then the reduced space is defined by

$$\mathbf{\Omega}^V_{\mathbf{x}_b} = \left\{ \mathbf{x}_b + v \,|\, \mathbf{x}_b \in \mathbb{R}^d,\ v \in V \right\}. \tag{2.34}$$

We refer to $\mathbf{x}_b$ the base point of $\mathbf{\Omega}^V_{\mathbf{x}_b}$. The set of directions spanning $V \subseteq \mathbb{R}^d$ is given by a matrix V_b whose columns form a basis for V, that is,

$$V_b = \begin{bmatrix} | & & | \\ v_1 & \dots & v_{\bar{d}} \\ | & & | \end{bmatrix}_{d \times \bar{d}} \qquad v_i \in \mathbb{R}^d.$$

Hence the corresponding reduced space is $\bar{d}$-dimensional. Coordinates with respect to the basis V_b are referred to as $\bar{x}$-coordinates. To determine the set of reduced linear equations of motion at a point $\mathbf{x} \in \mathbf{\Omega}^V_{\mathbf{x}_b}$, we express $\mathbf{M}$, $\mathbf{K}$ and $\mathbf{G}$ in $\bar{x}$-coordinates by:

$$\begin{aligned} \mathbf{M}_r &= V_b^T \mathbf{M} V_b \\ \mathbf{K}_r &= V_b^T \mathbf{K} V_b \\ \mathbf{G}_r &= V_b^T \mathbf{G}. \end{aligned}$$

Then in $\bar{x}$-coordinates, the equations of motion become:

$$\mathbf{M}_r\,\ddot{q}(t) + (\alpha\,\mathbf{M}_r + \beta\,\mathbf{K}_r)\dot{q}(t) + \mathbf{K}_r\,q(t) + \mathbf{G}_r = 0. \tag{2.35}$$

Because $\mathbf{M}_r \in \mathrm{SYM}^+_{\bar{d}}$ and $\mathbf{K}_r \in \mathrm{SYM}_{\bar{d}}$, we can follow the steps from Section 2.2.2 to compute a basis $\mathbf{\Phi}_r$ to decouple the system from Equation 2.35. Then the $\bar{d}$-dimensional decoupled system is given by

$$\mathbb{1}_r\ddot{\omega}(t) + (\alpha\,\mathbb{1}_r + \beta\,\Lambda_r)\dot{\omega}(t) + \Lambda_r\,\omega(t) + g_r = 0. \tag{2.36}$$

Restricting to poses $\mathbf{x}_i \in \mathbf{\Omega}^V_{\mathbf{x}_b}$, we can now formulate and solve a $\bar{d}$-dimensional linearized spacetime problem as described in Section 2.5.

2.7.2 Reduction based on a deformation energy

We propose a strategy to create the set of directions spanning a reduced space that is based on vibration modes of a deformation energy $E(\cdot,\cdot)$. The necessary steps are summarized in Algorithm 2. The intermediate set of vectors S consists of two vector types: finite differences $\mathbf{x}_i - \mathbf{x}_b$ and eigenvectors $\Phi_{i,j}$. The finite differences ensure that the poses $\mathbf{x}_i$ are contained in the final subspace. By setting $E(\mathbf{x}_i, \mathbf{x}_i)$, we solve the generalized eigenvalue problem for semi positive definite matrices $d^2E(\mathbf{x}_i, \mathbf{x}_i)$. Hence the vectors $\Phi_{i,j}$ considered for the base construction are vibration modes, see Figure 2.4 and Figure 2.5. For each pose $\mathbf{x}_i$, we add the first k lower vibration modes to S. The final orthonormal base V_b is obtained by an orthonormalization using a Singular Value Decomposition of S. Depending on the complexity of the motion, we choose k between 5 and 30. Hence the reduced space V constructed by Algorithm 2 contains a set of $\bar{d} \le (k \cdot m)$ directions of least energy changes for every keyframe.

Algorithm 2: Orthonormal basis for affine subspace in the shape space

```
Data: poses $\mathbf{P} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, base point $\mathbf{x}_b$, metric $M(x)$,
      deformation energy $E(\cdot, \cdot)$ and $k \in \mathbb{N}$
Result: orthonormal base $V_b = [v_1, \ldots, v_{\bar{d}}]$, $v_i \in \mathbb{R}^d$
$S = \emptyset$ ;
foreach $\mathbf{x}_i$ do
    $S \cup (\mathbf{x}_i - \mathbf{x}_b)$ ;
    solve $\lambda_{i,j} M(\mathbf{x}_i) \Phi_{i,j} = d^2 E(\mathbf{x}_i, \mathbf{x}_i) \Phi_{i,j}$ ;
    $S \cup \{\Phi_{i,1}, \ldots, \Phi_{i,k}\}$ ;
end
$V_b =$ orthonormalize $(S)$ ;
```

2.7.3 Experiments

We present two examples using a reduced space. The first example is a thin shell model whose parts are held fixed during the animation. The second example creates a motion of a solid using sliding constraints.

In the first example, a walking cycle of a running bug is enriched with secondary motion. The primary motion interpolates 12 keyframes that form a walking cycle, which we took from the book [Ritchie et al., 2005]. The secondary motion is added onto the motion from the walking cycle, that is, we overlaid the walking cycle motion with a second motion. For this second motion, we used two keyframes and constrained a part of the models (grey area in Figure 2.15) that has been carefully crafted and should not be affected by secondary motion. Then we computed a reduced space and added simple vector fields to the keyframes (black lines). We used the projections of these vector fields into the reduced space as derivatives of the motion and forced the wiggly splines to interpolate them.

The second example shows a sliding motion of a cactus modeled as a three-dimensional solid consisting of 1.7k vertices (see Figure 2.16) computed within a reduced space with $\bar{d} = 15$. The sliding constraints were applied to the vertices on the bottom. The start and end poses coincide. The motion was generated by the prescription of start and end velocities. The start velocity is nonzero at the trunk and the end velocity is set to zero. These boundary conditions result in a motion where the cactus branches follow the trunk in a swinging motion.

Details for these two examples are summarized in Table 2.1 including timings.

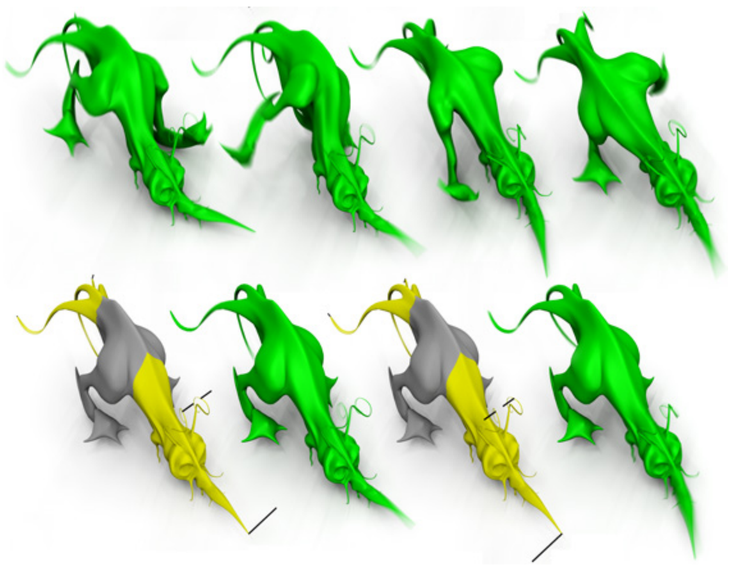

Figure 2.15: Snapshots from an animation of a walking bug enriched with secondary motion is shown in top row. The generation of the secondary motion from a simple vectorfield (black lines) is illustrated in the bottom row. The yellow/grey models describe start and end pose, i.e., grey parts are fixed and the yellow parts are deformable. The green models are snapshots from the resulting head swinging motion.

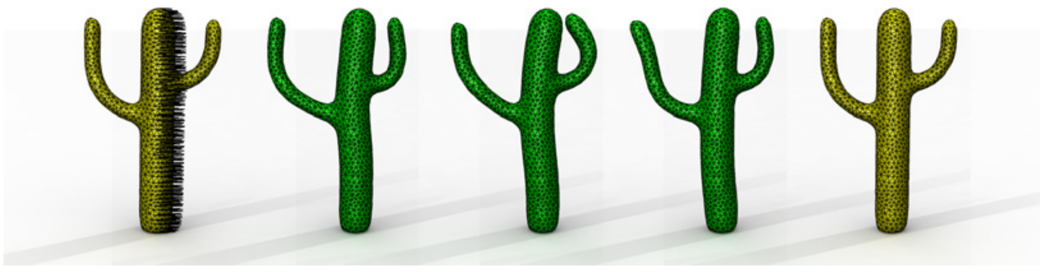

Figure 2.16: Sliding motion for a tedrahedral model of a cactus. The sliding constraints are applied to the vertices at the bottom. Start and end poses conicide and are shown in yellow. Start velocity is nonzero at the trunk pointing sideways (black lines on the left). The end velocity is zero. The resulting deformations in between generated by one-point dynamics in a reduced space are shown in green.

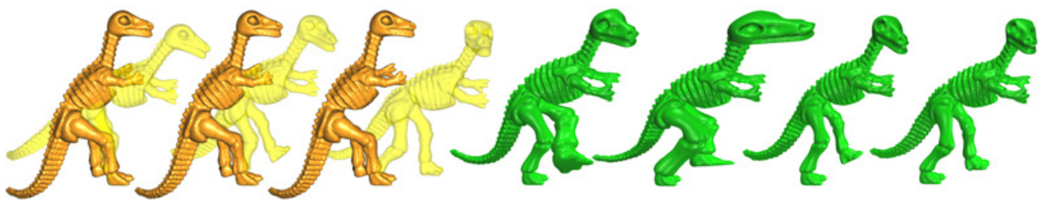

Figure 2.17: Comparison between one-point dynamics and multi-point dynamics for one step of a walking cycle for the dinosaur model. Keyframes (yellow) and reststate (orange) are shown on the left. The two green models in the middle are results from one point dynamics. The last two green models depict the same results from multi-point dynamics using coupling.

2.8 Multi-point dynamics

A problem when using the set of linearized equations of motion is that artifacts may develop for larger deformations away from the base pose $\mathbf{x}$ of the linearized model (see Figure 2.17). To counteract such problems, we introduce the multi-point dynamic approach. This approach generates motions $x(t) \in \mathcal{C}^k_{a,b}(\mathbb{R}^d)$ that are composed of various wiggly spline curves. These wiggly splines are allowed to differ in type, i.e., λ_i and δ_i can vary over time. The resulting motions are generated from multiple linearized dynamic models. An illustration of the multi-point dynamic approach is shown in Figure 2.18. We describe the multi-point strategy with respect to x-coordinates, but the proposed steps are also applicable in the reduced $\bar{x}$-coordinates introduced in Section 2.7. Given a set of points $\mathbf{Z} = \{\mathbf{z}_1, \dots, \mathbf{z}_n\}$ with $\mathbf{z}_j \in \mathbb{R}^d$, a metric $M(x)$, and a potential $V(x)$, we can compute the linearized dynamic model at each $\mathbf{z}_j$. Following the notions given in Section 2.2.2, we denote the quantities describing the j-th linearized dynamic model at a point $\mathbf{z}_j$ by $\mathbf{M}_j = M(\mathbf{z}_j)$, $\mathbf{K}_j = d^2V(\mathbf{z}_j)$, $\mathbf{G}_j = dV(\mathbf{z}_j)$, and the decoupling base by:

$$\mathbf{\Phi}_j = \begin{bmatrix} | & & | \\ \Phi_{j,1} & \dots & \Phi_{j,d} \\ | & & | \end{bmatrix}_{d \times d}.$$

This corresponds to a sampling of the nonlinear dynamics described by $M(x)$ and $V(x)$ at n points $\mathbf{z}_j \in \mathbb{R}^d$.

We propose two strategies to construct an animation that is sensitive to multiple linearized dynamic models. They are called blending and coupling.

In both approaches, an influence interval $I_j = [t_j^-, t_j^+)$ needs to be given for the j-th linearized dynamic model with $t_j^- < t_j^+$. The choice of the interval depends on the strategy used. The final motion $x(t)$ is then constructed as

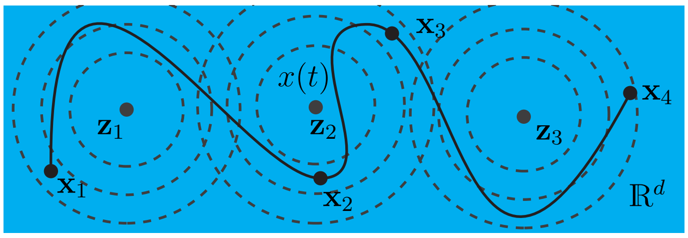

Figure 2.18: An illustration of the multi-point dynamic approach. The inter polation poses are: $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$ and $\mathbf{x}_4$. Linearized dynamic models are computed at $\mathbf{z}_1$, $\mathbf{z}_2$ and $\mathbf{z}_3$, that have a limited influence within regions of $\mathbb{R}^d$(concentric circles). The resulting trajectory $x(t) \in \mathcal{C}^k_{t_1,t_4}(\mathbb{R}^d)$ (black curve).

a weighted sum of n curves $z_j(t)$. Using $h_j : \mathbb{R} \to \mathbb{R}$ for the j-th interval dependent weighting function, the final motion $x(t)$ is given by:

$$x(t) = \sum_{j=1}^{n} h_j(t) z_j(t) \tag{2.37}$$

where

$$z_j(t) = \mathbf{z}_j + \mathbf{\Phi}_j \omega_{z_j}(t)$$

The curves $z_j(t)$ are wiggly splines computed for the linearized dynamic model around the sample point $\mathbf{z}_j$. The two approaches differ in the computation of the wiggly coefficients for the curves $z_j(t)$ and the weighting functions $h_j(t)$. Blending can smoothly change between curves $z_j(t)$. In contrast, coupling performs hard switches between linearized models.

For both approaches, we show how an interpolating curve $x(t) \in \mathcal{C}^2_{t_1,t_m}(\mathbb{R}^d)$ is constructed if $\mathbf{Z} = \mathbf{P}$.

2.8.1 Blending

Motions generated by blending are composed of independently computed curves $z_j(t) \in \mathcal{C}^2_{t_1,t_m}(\mathbb{R}^d)$, where $j \in \{1, \dots n\}$. Each curve $z_j(t)$ is computed by Algorithm 1 using the same set of poses $\mathbf{P}$ and interpolation times $\mathbf{T}$, but with the base of the dynamic model set to $\mathbf{z}_j$. This results in n independent interpolating curves $z_j(t)$. Then the choice of the blending functions h_j controls the smoothness of the final motion and determines how the n solutions $z_j(t)$ affect the final motion (see Equation 2.37).

Hence to compute a motion $x(t)$ during the interactive phase using blending, we need to solve nd independent banded systems of dimension $4(m-1)$.

If the base set $\mathbf{Z}$ and pose set $\mathbf{X}$ coincide, we have $n = m$ with $\mathbf{z}_i = \mathbf{x}_i$. Then we can easily design a blending function that ensures that the resulting motion $x(t)$ interpolates, i.e., $x(t_k) = \mathbf{z_k}$ and corresponds around the interpolated pose $\mathbf{z}_k$ to the associated wiggly spline $z_k(t)$. We choose for the intervals I_j the times $t_j^- = (t_j - t_{j-1})/4$ and $t_j^+ = 3(t_j - t_{j-1})/4$ with $t_j \in \mathbf{T}$. This choice results in overlapping intervals, that is, $I_{j-1} \cap I_j \neq \emptyset$ and $I_j \cap I_{j+1} \neq \emptyset$. Then we define the blending function to be used in Equation 2.37 by:

$$h_j(t) = \begin{cases} 1 & t \in I_j \setminus (I_{j-1} \cup I_{j+1}) \\ s_j^-(t) & t \in I_j \cap I_{j-1} \\ s_j^+(t) & t \in I_j \cap I_{j+1} \\ 0 & \text{otherwise.} \end{cases} \tag{2.38}$$

For the functions $s_j^-(t)$ and $s_j^+(t)$ that control the transition between two successive models, we use cubic splines to fade in and out of a models influence. By this type of blending, we generate smooth motions, i.e., $x(t) \in \mathcal{C}^2_{t_1,t_m}(\mathbb{R}^d)$.

2.8.2 Coupling

Motions generated by coupling are composed of wiggly splines $z_j(t) \in \mathcal{C}^2_{t_j^-,t_j^+}(\mathbb{R}^d)$, where $j \in \{1, \dots n\}$, whose computation depend on each other. For the influence intervals I_j, we assume $\bigcup I_j = [t_1, t_m)$ with $I_j \cap I_i = \emptyset$ if $j \neq i$, and $t_j^+ = t_{j+1}^-$. To assemble the final motion from the curves $z_j(t)$ (see Equation 2.37) by coupling, we use a weighting function h_j defined by

$$h_j(t) = \begin{cases} 1, & t \in I_j \\ 0, & \text{otherwise.} \end{cases} \tag{2.39}$$

A curve $z_j(t)$ is determined by only $4d$ wiggly spline coefficients $\omega^l_{j,k}$. In contrast to Section 2.5.1, in this section we use $\omega^l_{j,k}$ to denote the wiggly spline coefficients at the j-th base along the k-th direction $\Phi_{j,k}$ with the wiggly base index $l \in \{1, 2, 3, 4\}$.

To construct a curve $x(t) \in \mathcal{C}^2_{t_1,t_m}(\mathbb{R}^d)$ by coupling, we formulate transition conditions between curves $z_j(t) \in \mathcal{C}^2_{t_j^-,t_j^+}(\mathbb{R}^d)$ and $z_{j+1}(t) \in \mathcal{C}^2_{t_{j+1}^-,t_{j+1}^+}(\mathbb{R}^d)$, where we assume that $t_j^+ = t_{j+1}^-$. Hence we have to ensure $z_j(t_j^+) = z_{j+1}(t_j^+)$, $\dot{z}_j(t_j^+) = \dot{z}_{j+1}(t_j^+)$, and $\ddot{z}_j(t_j^+) = \ddot{z}_{j+1}(t_j^+)$. To formulate these constraints with respect to the unknown $4dn$ wiggly coefficients, we define:

$$\begin{aligned}
\mathbf{\Phi}_{j,k}^{X,\alpha}(t) &= \begin{bmatrix} | & | & | & | \\ b_k^{1,\alpha}(t)\Phi_{j,k} & b_k^{2,\alpha}(t)\Phi_{j,k} & b_k^{3,\alpha}(t)\Phi_{j,k} & b_k^{4,\alpha}(t)\Phi_{j,k} \\ | & | & | & | \end{bmatrix}_{d\times 4} \\
\mathbf{B}_j^{X,\alpha}(t) &= \begin{bmatrix} | & & | \\ \mathbf{\Phi}_{j,1}^{X,\alpha}(t) & \dots & \mathbf{\Phi}_{j,d}^{X,\alpha}(t) \\ | & & | \end{bmatrix}_{d\times 4d} \\
\boldsymbol{\omega}_j^X &= \left[\omega_{j,1}^1 \dots \omega_{j,1}^4 \dots \omega_{j,d}^1 \dots \omega_{j,d}^4\right]_{4d\times 1} \\
\mathbf{c}_j^X &= \left[c_{j,1} \ \dots \ c_{j,d}\right]_{d\times 1}.
\end{aligned}$$

The constraints ensuring twice differentiable transitions between successive curves $z_j(t)$ and $z_{j+1}(t)$ at time t_j^+ are then given by:

$$\begin{aligned}
&\mathbf{B}_i^{X,0}(t_j^+)\boldsymbol{\omega}_j^X - \mathbf{B}_i^{X,0}(t_j^+)\boldsymbol{\omega}_{j+1}^X = (\mathbf{z}_{j+1} - \mathbf{z}_j) + (\mathbf{c}_j^X - \mathbf{c}_{j+1}^X) \\
&\mathbf{B}_i^{X,1}(t_j^+)\boldsymbol{\omega}_j^X - \mathbf{B}_i^{X,1}(t_j^+)\boldsymbol{\omega}_{j+1}^X = 0 \\
&\mathbf{B}_i^{X,2}(t_j^+)\boldsymbol{\omega}_j^X - \mathbf{B}_i^{X,2}(t_j^+)\boldsymbol{\omega}_{j+1}^X = 0.
\end{aligned} \tag{2.40}$$

This amounts to $3d(n-1)$ constraints for the $4dn$ wiggly coefficients that determine the curves $z_j(t)$.

If the base set $\mathbf{Z}$ and pose set $\mathbf{P}$ coincide, we have $n = m$ with $\mathbf{z}_i = \mathbf{x}_i$. We choose for the intervals I_j the times $t_j^- = (t_j - t_{j-1})/2$ and $t_j^+ = (t_j - t_{j-1})/2$ with $t_j \in \mathbf{T}$. The interpolation condition $x(t_j) = \mathbf{z}_j$ with $t_i \in \mathbf{T}$ is expressed by

$$\mathbf{B}_j^{X,0}(t_j)\boldsymbol{\omega}_j^X = \mathbf{c}_j^X. \tag{2.41}$$

This amounts to nd interpolation conditions and $3d(n-1)$ transition conditions for the $4nd$ unknown wiggly spline coefficients. To uniquely determine the wiggly coefficients, we define them as the solution to a constraint optimization problem, that is, the objective function is set to a quadratic energy as defined in Equation 2.31 and the transition conditions (see Equation 2.40) and interpolating conditions (see Equation 2.41) enter as constraints. Hence to compute a motion $x(t)$ during the interactive phase using coupling, we need to solve a $4dn + nd + 3d(n-1) = 8nd - 3d$ dimensional system of linear equations.

2.8.3 Experiments

We present two examples that use the multi-point dynamic approach in a reduced space. In both examples the reduced space was created from the set of keyframes.

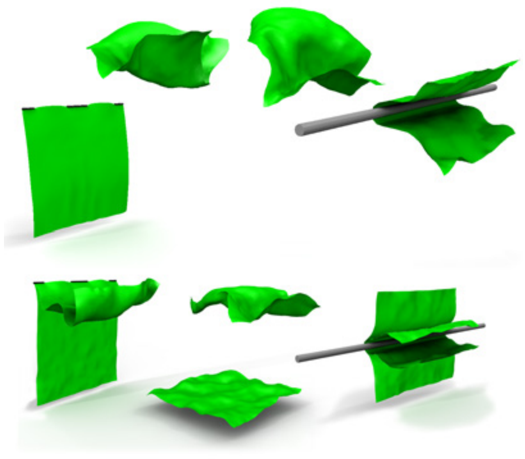

Figure 2.19: Snapshots from an animtion modeling the controlled flight of a cloth piece are shown in the top row. The keyframes to create this motion are taken from three independent simulation runs, shown in the bottom row, i.e. hanging, blown up and colliding with a cylinder.

The first example shows a piece of cloth that first hangs on a line, then is blown into the air, and finally gets stuck on a cylinder (see Figure 2.19, top row). The motion is assembled from three wiggly splines by blending. The first and last wiggly spline satisfy equality constraints that fix parts of the piece of cloth to the line and cylinder, respectively. The keyframes were generated by taking snapshots from three independent forward simulations: cloth hanging on a line, cloth blown in the air, and cloth stuck on a cylinder (see Figure 2.19, bottom row). The benefit of our approach is that we can combine these (otherwise unrelated) animations into one animation in which the cloth is flying through the air and hits the cylinder.

The second example in Figure 2.20 shows a walking dinosaur. The motion is composed of five wiggly splines, one for each step. Each step is modeled with three keyframes and the foot on the ground is fixed with equality constraints, see Figure 2.17 left. The boundary conditions for each of the wiggly splines are finite differences of the keyframes. The animation of each step were generated by blending. The results are similar to the non-linear method of [Barbič et al., 2009]. To increase comparability, we used reduced spaces of approximately the same size. A difference between these two methods is that their scheme does not interpolate and only approximates the keyframes. The original keyframes are given as tetrahedral meshes that enclose the triangulated dinosaur mesh. These tetrahedral meshes have 1.8K vertices and thus are coarser than the triangle meshes, which have 28K vertices. For our animation, we used the triangle meshes with higher resolution. This explains

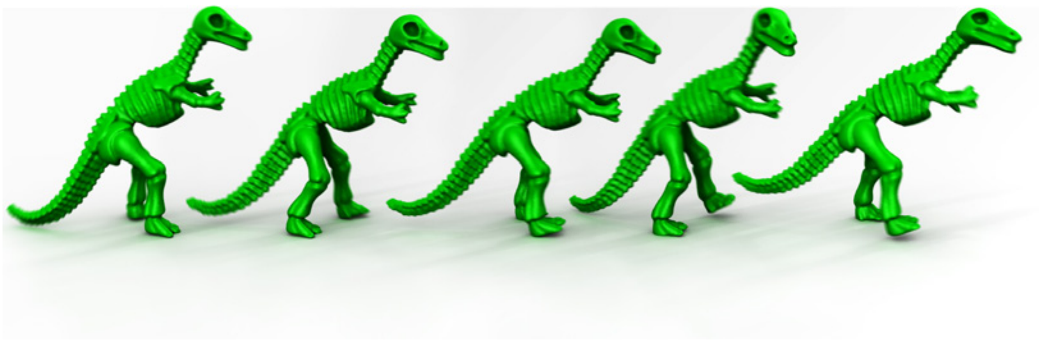

Figure 2.20: Physically based soft-body dinosaur animation that interpolates animator-specified keyframes, that model five walking steps. Results are generated by multi-point dynamics using blending.

why we need 500s to construct the reduced space for the whole animation, while they needed only 25s. After the preprocess, our method needs 0.6s to compute or update the animation, while their optimization takes 15min.

To generate a similar walking dino animation by coupling, we used a reduced space of the same dimension, that was created only from vibration modes computed at the rest state, see Figure 2.17 right.

Details for these two examples are summarized in Table 2.1 including timings.

2.9 Summary

We linearized and decoupled a nonlinear spacetime formulation. We then showed that by using the decoupled formulation, the actual trajectory computation is expressed by a number of independent one-dimensional boundary value problems of the same type. The solutions to these problems are contained in the space of wiggly splines. We then showed that the Euler–Lagrange equation characterizing the solutuions to the one-dimensional boundary value problems is a linear fourth-order ordinary differential equation. By solving this ordinary differential equation we determined a closed form representation of the wiggly spline basis functions.

We proposed the one-point dynamic algorithm to solve the decoupled linearized spacetime problem. Then we applied a modal reduction strategy to enable the motion generation for objects with a large number of degrees of freedom, e.g., deformable objects. Our reduced space is built from vibration modes of an energy that measures shape changes. We further proposed two multi-point dynamic approaches to overcome artifacts caused by the use of a

Model	#Verts	#S	Dim.	#K	t_p	t_u
Block	450	3	1350,1350,1350	6	83	0.54
Bug	4358	1	30	2	9	0.03
Cactus	1691	1	15	2	1.6	0.01
Cloth	1861	3	92,143,186	12	74	0.85
Dinosaur	28098	5	18,18,18,18,18	11	505	0.6

Table 2.1: Performance measured on a desktop PC with an Intel Xeon 3.33 GHz CPU (single thread). From left to right we use #Verts for number of vertices, #S for number of composite splines, Dim. for the dimension, #K for the number of keyframes, t_p and t_u for time in seconds for preprocess and an animation update during the interaction phase, respectively.

single decoupled linearized model used in the one-point dynamic approach.

We presented a variety of animation results including planar rods, thin shells and solids. This included a comparision with the results from common b-spline interpolation. We further demonstrated that our reduced formulation allows for the creation and editing of an animation for deformable objects at interactive rates.

The linear dynamics are computed with respect to a certain rest state so a possible extension is to automatically determine of an appropiate rest state in case no rest state is explicitly given. Using a multi-point dynamic approach allows the use of a varying rest states; how this affects the final motion could also be studied in the future. Furthermore the differences of the two multi-point dynamics approaches, i.e., blending and coupling, need to be better understood.

We further plan to add contact handling including self collisions to the construction of the reduced space.

We used interpolation and soft interpolation of keyframes. Therefore another feature to work on is the use of partial keyframes, i.e., only a subset of vertices of an object enter as boundary constraints.

It would also be interesting to apply our approach to control other physical systems such as fluids or smoke.

Chapter 3

Geometry-aware Operators

In this chapter, we derive operators whose eigenmodes and spectra can serve as an alternative to the spectrum and modes of the Laplacian for applications in geometry processing and shape analysis. The eigenfunctions of these operators share properties with the eigenfunctions of the Laplacian. They form an orthogonal basis of a space of variations of the surface. However there are also fundamental differences. These spectra and eigenfunctions depend not only on intrinsic quantities but also on the extrinsic curvature of the surface.

The spectrum and eigenfunctions of the Laplace–Beltrami operator of a surface have stimulated much research in shape analysis and geometry processing ranging from parametrization, segmentation, and symmetry detection to shape signatures and mesh filtering as described in Section 3.1. Such methods profit from properties of the eigenfunctions of the Laplace–Beltrami operator. For example, on a curved surface, they form an orthogonal basis of the space of L^2-functions on the surface. Furthermore, the Laplacian depends only on the metric of the surface, hence the eigenvalues and eigenfunctions are invariant under isometric deformations of the surface. A consequence of the invariance to isometric deformations is an insensitivity to extrinsic features of the surface, like sharp bends, that are of essential importance for some applications in shape analysis.

In Section 3.2 we recall the definition of the smooth and discrete Laplace–Beltrami operator and its associated spectrum and eigenfunctions. We then consider operators that correspond to the Hessian of energies in Section 3.3. The energies we consider are functionals defined on a space of functions on a surface. We show that the Laplace–Beltrami operator can be derived from the Hessian of the Dirichlet energy that is contained in this energy class. Furthermore we introduce a feature senstive energy called the modified energy and compare the feature senstive eigenfunctions of its associated operator to the eigenfunctions of the Laplace–Beltrami operator.

Based on the new modified Dirichlet energy, we define a family of feature sensitive energies, i.e., a family of feature sensitive operators, in Section 3.4. In Section 3.5 we show that the spectrum of a certain operator from this family of feature sensitive operators can be used to estimate the stability of a cmc surface.

In Section 3.6 we introduce a feature sensitive multi-scale signature based on the eigenmodes of the modified Dirichlet energy. To measure differences with respect to this signature, we further propose a corresponding feature sensitive multi-scale pseudo-metric on the surface. As an application, we then show on a variaty of examples that the resulting feature distance can be used to identify features on a mesh and compare the results to the heat kernel signature in Section 3.7.

3.1 Background

Recently, there has been a large interest in the use the eigenvalues and eigenfunctions of the Laplace–Beltrami operator as an ingredient in algorithms in geometry processing and shape analysis. An overview of this development can be found in the survey by [Clements and Zhang, 2006] and in the course notes of [Lévy and Zhang, 2009]. Here, we briefly outline the work that has been most relevant for our results.

The spectrum of the Laplace–Beltrami operator of a Riemannian manifold contains a significant amount of information about the manifold and the metric. Though it does not fully determine the Riemannian manifold, it can be used as a powerful shape descriptor of a class of isometric Riemannian manifolds. [Reuter et al., 2005,Reuter et al., 2006] used the spectrum of the Laplace–Beltrami operator to construct a fingerprint of surfaces, which they call the Shape-DNA. By construction, this fingerprint is invariant under isometric deformations of a surface. The Shape-DNA can be used for shape matching, copyright protection, and database retrieval, among other applications. [Rustamov, 2007] developed the Global Point Signature (GPS), a signature that can be used to classify shapes up to isometry. Based on the GPS, [Ovsjanikov et al., 2008] developed a method for the detection of global symmetries in shapes.

[Dong et al., 2006] presented an elegant technique that uses the Morse–Smale complex (and the quasi-dual complex) of a carefully chosen Laplace eigenfunction to generate a coarse quadrangulation of a surface mesh. This approach was extended by [Huang et al., 2008], who used a least-squares optimization routine that modifies the selected Laplace eigenfunction and, consequently, its Morse–Smale complex. They also provide the user with control over the shape, size, orientation, and feature alignment of the faces of the resulting quadrangulation.

The computation of the spectrum and eigenfunctions of the Laplacian is a delicate and computationally expensive task, even for medium sized meshes. [Vallet and Lévy, 2008] proposes an efficient shift-and-invert Lanczos method and presents an implementation that is designed to handle even large meshes.

Using the eigenfunctions of the Laplacian, one can compute the heat kernel of the surface, that is, one can simulate diffusion processes on curved domains. [Sun et al., 2009] proposed the heat kernel signature, a surface signature based on the heat kernel, which they use to derive a measure for the geometric similarity of different regions of the surface. By construction, this measure is invariant under isometric deformations of the surface.

Independent from this work, [Gebal et al., 2009] proposed a similar signature, the Auto-Diffusion-Function, and used it for mesh skeletonization and segmentation. In the context of shape matching and retrieval, [Dey et al., 2010] used the persistent extrema of the heat kernel signature to construct a robust and efficient pose-oblivious matching algorithm for three-dimensional shapes. Given a corresponding pair of points on two shapes, [Ovsjanikov et al., 2010] used the heat kernel to construct an isometric map between the shapes which allows them to find intrinsic symmetries and match partial, incomplete, or isometrically deformed shapes.

3.2 Laplace modes and spectrum

In this section, we briefly review the eigenvalue problem of the Laplacian on surfaces. We first state a weak form of the problem for smooth surfaces. This shows that the spectrum is based on two operators. These two operators will be defined for discrete surface meshes which are then used in the computation of the spectrum in the discrete setting.

3.2.1 Smooth setting

Let Σ be a smooth, compact surface in $\mathbb{R}^3$. For clarity clear we assume that Σ has no boundary.

We denote $C^k(\Sigma)$ to be the set of k-times differentiable functions $f : \Sigma \to \mathbb{R}$. Then the Laplace–Beltrami operator corresponds to a mapping $\Delta : C^2(\Sigma) \to C^0(\Sigma)$. We use $\Delta f = -div(\nabla(f))$, but depending on the context it is sometimes also defined by $\Delta f = div(\nabla(f))$. Using our notation we define

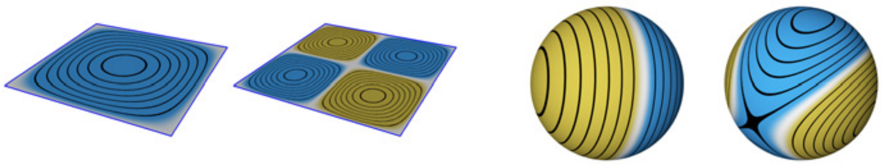

Figure 3.1: Illustration of two lower eigenmodes of the Laplace operator for the plate and the sphere. The eigenmodes for the plate using Dirichlet boundary conditions shown on the left. The eigenmodes on the sphere are shown on the right.

Problem 3 (Laplace Eigenvalue Problem) *On a compact manifold* Σ *find all pairs* $(\lambda, f) \in \mathbb{R} \times C^\infty(\Sigma)$ *such that* $\Delta f = \lambda f$.

Let $H^1(\Sigma)$ denote the Sobolev space of weakly differentiable functions with square integrable derivative on Σ. Hence we have $C^\infty(\Sigma) \subset H^1(\Sigma)$. On $H^1(\Sigma)$, we consider the two bilinear forms for $u, v \in H^1(\Sigma)$:

$$\langle u, v\rangle_{L^2} = \int_\Sigma uv\; dA \qquad \text{and} \qquad \langle u, v\rangle_{H^1_0} = \int_\Sigma \langle \nabla u, \nabla v\rangle_\Sigma \; dA. \tag{3.1}$$

Using integration by parts on Problem 3 we can formulate an eigenvalue problem for functions in $H^1(\Sigma)$, that is,

Problem 4 (Weak Laplace Eigenvalue Problem) *On a compact manifold* Σ *find all pairs* $(\lambda, \phi) \in \mathbb{R} \times H^1$ *such that* $\langle \phi, \sigma\rangle_{H^1_0} = \lambda \langle \phi, \sigma\rangle_{L^2}$ *for all* $\sigma \in H^1$.

To extend Definitions 3 and 4 to compact surfaces with boundary, corresponding boundary conditions have to be prescribed. In this case, appropriate subspaces and terms resulting from possible non-vanishing boundary integrals have to be considered.

Two prominent examples of solutions to Problem 3 are shown in Figure 3.1. To visualize the eigenfunctions, we use blue for positive values, white for zero crossings, and orange for negative values. Additionally, we draw isolines in black. In particular, the eigenfunctions on the plane, see Figure 3.1 left, correspond to vibration modes restricted to normal directions of a clamped flat plate. The eigenfunctions on the sphere, see Figure 3.1 right, are also called spherical harmonics.

3.2.2 Discrete setting

We consider simplicial surfaces $\mathcal{M}_h \subset \mathbb{R}^3$ (see Section 1.1) describing smooth surfaces immersed in $\mathbb{R}^3$. We also refer to the associated abstract simplicial geometry $\mathcal{M}$ as the surface mesh. Furthermore we denote its vertices $\sigma_i^0 \in \mathcal{M}$ and triangles $\sigma_i^2 \in \mathcal{M}$ by v_i and T_i, respectively. We refer by $|T_i|$ to the compact set $|\sigma_i^2| \subset \mathbb{R}^3$. Then we define on the surface $\mathcal{M}_h$

Definition 5 (Function Space S_h) *The space of piecewise linear functions S_h on $\mathcal{M}_h$ is given by*

$$S_h = \left\{u : \mathcal{M}_h \to \mathbb{R} \mid u \in C^0(\mathcal{M}_h) \textit{ and } u \textit{ is linear on each} |T_i| \subset \mathcal{M}_h\right\}.$$

The elements of the space S_h form a linear vector space, and a function $u \in S_h$ is determined by its function values $u_i = u(p_i)$ at the vertices $p_i \in \mathcal{M}_h$. Furthermore the bilinear forms $\langle\cdot,\cdot\rangle_{L^2}$ and $\langle\cdot,\cdot\rangle_{H_0^1}$ are well defined on S_h as shown in [Dziuk, 1988] and [Hildebrandt et al., 2006].

We choose the Lagrange basis also called the nodal basis for S_h. It is formed by the linear functions $\varphi_i \in S_h$ defined by

$$\varphi_i(p_j) = \delta_{ij} \text{ with } p_j = P(v_j).$$

In the Lagrange basis, a function $u \in S_h$ is then expressed by

$$u(x) = \sum_i^n u_i \; \varphi_i(x) \qquad x \in \mathcal{M}_h.$$

The space S_h is equipped with a constant metric, that is, a matrix M with entries

$$m_{ij} = \sum_{T_k \in \mathcal{M}} \int_{|T_k|} \varphi_i(x)\varphi_j(x)\; dA.$$

The associated scalar product to this metric is defined by:

$$\langle u, v\rangle_{L^2} = \sum_i^n \sum_j^n u_i v_j m_{ij} \text{ with } u, v \in S_h. \tag{3.2}$$

The gradient $\nabla f(x)$ is constant within a triangle $|T|$. Then the operator $\langle\cdot,\cdot\rangle_{H_0^1}$ can be defined for functions in S_h by splitting the integrals into a sum over the triangles $T \in \mathcal{M}$, that is,

$$\langle u, v\rangle_{H_0^1} = \sum_{T_k \in \mathcal{M}} \int_{|T_k|} \langle \nabla u, \nabla v\rangle \, dA \text{ with } u, v \in S_h. \tag{3.3}$$

The matrix representation of $\langle\cdot,\cdot\rangle_{H_0^1}$ with respect to the Lagrange basis becomes the stiffness matrix S, whose entries are

$$s_{ij} = \langle \varphi_i, \varphi_j\rangle_{H_0^1}. \tag{3.4}$$

Explicit representations of the matrices M and S can be found in [Pinkall and Polthier, 1993], [Wardetzky et al., 2007], and [Vallet and Lévy, 2008]. Then the discretization of the eigenvalue Problem 4 is given by:

Problem 5 (Discrete Laplace Eigenvalue Problem) *On a simplicial surface $\mathcal{M}_h$ without boundary find all pairs $(\lambda, \phi) \in \mathbb{R} \times S_h$ such that $S\phi = \lambda M\phi$.*

This is a generalized eigenvalue problem for (possibly large) sparse matrices. Fast solvers for this problem are discussed in [Saad, 1992, Vallet and Lévy, 2008] and an example of a software package that specializes in such problems is Arpack (see [Lehoucq et al., 1998]). Since S is symmetric and M is positive definite, all eigenvalues of Problem 5 are real and there exists an L^2-orthonormal basis of S_h consisting of eigenvectors.

In the following, we always consider the diagonal lumped mass matrix M as a discrete L^2-scalar product [Wardetzky et al., 2007]. It is given by a diagonal matrix, where the i-th diagonal entry is a third of the combined area of the triangles adjacent to the i-th vertex of the mesh. We refer to the i-th diagonal entry as the mass m_i of the vertex v_i. Because $m_i > 0$, the lumped mass matrix is a symmetric and positive definite matrix. Hence the eigenvalue Problem 5 with the lumped mass matrix has real eigenvalues, and we can always find an orthonormal basis of S_h consisting of eigenvectors of Problem 5, where orthonormality is now measured with respect to the scalar product on S_h given by the lumped mass matrix M.

3.3 Energy based spectrum

In this section, we consider the modes and spectrum of the Hessian of surface energies. In Section 3.3.2 we show that the Laplacian eigenvalue problem appears as the eigenvalue problem of the Hessian of the Dirichlet energy. Based on this we construct a new energy — the modified Dirichlet energy — that, unlike the Dirichlet energy, is sensitive to the extrinsic curvature of the surface in Section 3.3.3. Finally we investigate the modes of this new energy in Section 3.3.4.

3.3.1 Energies

A surface energy is a twice continuously differentiable mapping $E : S_h \to \mathbb{R}$. The Hessian depends on the second derivatives of E and the scalar product on S_h. For $u, v \in S_h$, let $D^2_m E(u, v)$ be the second derivative at m in the direction of u and v. Then we refer to the operator $\nabla^2_m E : S_h \to S_h$ as the Hessian associated to the energy E at $m \in S_h$. It is given by

$$\left\langle \nabla^2_m E(u), v \right\rangle_{L^2} = D^2_m E(u, v) \tag{3.5}$$

for all $u, v \in S_h$.

With respect to the Lagrange basis of S_h and the induced L^2-product (see Equation 3.2), $\nabla^2_m E$ has the following matrix representation:

$$\nabla^2_m E = M^{-1} \partial^2_m E,$$

where $\partial^2_m E$ is the symmetric matrix containing the second partial derivatives at m and M is the mass matrix. The eigenmodes and eigenvalues of the energy E at m are the solutions $(\lambda, \phi) \in \mathbb{R} \times S_h$ to the generalized eigenvalue problem

$$\partial^2_m E \phi = \lambda M \phi \qquad \text{or} \qquad \lambda \phi = M^{-1} \nabla^2_m E \phi. \tag{3.6}$$

Since $\nabla^2_m E$ is self-adjoint with respect to the discrete L^2-product, all eigenvalues of $\nabla^2_m E$, i.e., the solutions to Equation 3.6, are real and there is an L^2-orthonormal basis of S_h that consists of eigenmodes $\nabla^2_m E$. In such a basis, the matrix representation of $D^2_m E$ and of $\langle \cdot, \cdot \rangle_{L^2}$ are diagonal matrices. Assuming that m is the minimum of E, the matrix $\partial^2_m E$ is positive semi-definite which implies that the eigenvalues of Equation 3.6 are non-negative. Hence for a pair (λ, ϕ) that solves Equation 3.6 with $\langle \phi, \phi \rangle_M = 1$, we have

$$E(m + \phi) - E(m) \approx \left\langle \nabla^2_m E(\phi), \phi \right\rangle_{L^2} = \lambda.$$

The eigenmodes associated to the smaller eigenvalues of the Hessian at a minimum of E point in the direction that locally causes the least increase of energy.

3.3.2 Dirichlet energy

An example of an energy defined in Section 3.3.1 is the discrete Dirichlet energy. For a compact smooth surface Σ, the Dirichlet energy is defined for weakly differentiable functions $f \in H^1(\Sigma)$ by

$$E_\Delta(f) = \frac{1}{2} \langle f, f \rangle_{H^1_0}. \tag{3.7}$$

Then, the discrete Dirichlet energy on a surface $\mathcal{M}_h$ is defined for functions $u \in S_h$ by

$$E_D(u) = \frac{1}{2} u^T S u, \tag{3.8}$$

where S is the stiffness matrix whose entries are defined in Equation 3.4. For a rigorous treatment of the discrete Dirichlet energy and a convergence analysis, see [Dziuk, 1988, Hildebrandt et al., 2006]. The discrete Dirichlet energy is a quadratic functional and therefore has a constant Hessian. At any $u \in S_h$, we have $\partial_u^2 E_D = S$. Then the associated generalized eigenvalue problem from Equation 3.6 corresponds to the Laplace eigenvalue Problem 5.

We set the entries of the stiffness matrix s_{ij} to the cotan weights given in [Pinkall and Polthier, 1993]. We further assume that the corresponding stiffness matrix S is semi-positive definite.

3.3.3 Modified Dirichlet energy

Assume that Σ is an orientable smooth surface in $\mathbb{R}^3$ and let $N : \Sigma \to \mathbb{R}^3$ denote the normal vector field of Σ. Then for $N = (N_1, N_2, N_3)$ the three coordinate functions N_k are smooth functions. Hence for $f \in H^1(\Sigma)$, we have $f\,N_k \in H^1(\Sigma)$. We then define an energy that is sensitive to extrinsic surface features by:

$$E_\Delta^N(f) = \sum_{k=1}^{3} E_\Delta(f\,N_k). \tag{3.9}$$

The following lemma shows that this energy can be split into two parts: a feature insensitive part and a feature sensitive part.

Lemma 4 (Modified Dirichlet Energy) *The energy defined in Equation 3.9 satisfies*

$$E_\Delta^N(f) = E_\Delta(f) + \frac{1}{2}\int_\Sigma f^2(\kappa_1^2 + \kappa_2^2)\,dA, \tag{3.10}$$

where κ_1 and κ_2 are the principal curvatures of Σ.

This means that $E_\Delta^N(f)$ is the sum of the Dirichlet energy of f and the f^2-weighted total curvature of Σ.

Proof. Given a function $f \in H^1(\Sigma)$, we define a mapping $e_\triangle^N : \Sigma \to \mathbb{R}$ by

$$e_\triangle^N = \frac{1}{2}\sum_{k=1}^{3} \left\langle \nabla\left(fN_k\right), \nabla\left(fN_k\right)\right\rangle. \tag{3.11}$$

By construction $e_\triangle^N$ is defined almost everywhere on Σ. Expanding the sum from Equation 3.11 into its three parts and using the product rule we get:

$$e_\triangle^N = \frac{1}{2}\sum_{k=1}^{3} N_k^2\left\langle \nabla f, \nabla f\right\rangle + f^2\left\langle \nabla N_k, \nabla N_k\right\rangle + 2fN_k\left\langle \nabla N_k, \nabla f\right\rangle.$$

Note $e^N_\triangle$ from Equation 3.11 is invariant under transformations from the specialized orthogonal group. We assume local Monge coordinates around a point $x \in \Sigma$. Then the normal at x is given by:

$$N(x) = (0, 0, 1).$$

By choosing a Monge chart where the coordinate lines are aligned along principle curvature lines, the extended shape operator [Hildebrandt, 2013] has the form

$$\begin{pmatrix} \kappa_1 & 0 & 0 \\ 0 & \kappa_2 & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

Using $e_1 = [1, 0, 0]$ and $e_2 = [0, 1, 0]$ the normal derivatives in this coordinate chart is given by

$$\nabla N_k(x) = \begin{cases} -\kappa_k e_k & k = 1, 2 \\ 0 & k = 3. \end{cases}$$

Using these relations for the normal and its derivatives, $e^N_\triangle$ (see Equation 3.11) can then be rewritten as:

$$e^N_\triangle = \frac{1}{2} \langle \nabla f, \nabla f \rangle + \frac{1}{2} f^2 \left(\kappa_1^2 + \kappa_2^2 \right). \tag{3.12}$$

Finally integrating over the surface Σ yields

$$E^N_\triangle (f) = \int_\Sigma e^N_\triangle dA$$

and we have our desired result. ■

To define a discrete modified Dirichlet energy, we define the space of piecewise linear vector fields on $\mathcal{M}_h$ by

$$V_h = \left\{ w : \mathcal{M}_h \to \mathbb{R}^3 \,| w(x) = (w_1(x), w_2(x), w_3(x)) \text{ with } w_i \in S_h \right\}.$$

Then we can define a mapping $s : S_h \times V_h \to V_h$ that scales a vector field, that is, $s(u, w) = (u(x)w_1(x), u(x)w_2(x), u(x)w_3(x))$. Fixing a certain vector field $w \in V_h$, we can define $s_w : S_h \to V_h$ by $s_w(u) = s(u, w)$. This map establishes an isomorphism between functions in S_h and vector fields that are parallel to a certain $w \in V_h$. Similar to the smooth setting (see Equation 3.9), we can then define an energy by

Definition 6 (Discrete Modified Dirichlet Energy) *Given a unit normal vector field $N \in V_h$ the discrete modified Dirichlet energy is a mapping $E_D^N : S_h \to \mathbb{R}$ defined by*

$$E_D^N(u) = \sum_{k=1}^{3} E_D(s_N(u)_k) \text{ with } u \in S_h.$$

In particular the modified discrete energy $E_D^N(\cdot)$ has the following matrix representation

Lemma 5 (Discrete Modified Dirichlet Energy) *The discrete modified Dirichlet energy $E_D^N(u)$ is a quadratic form given by*

$$E_D^N(u) = \frac{1}{2} u^T A u$$

where the entries a_{ij} of A are

$$a_{ij} = n_{ij} s_{ij}$$

where

$$n_{ij} = \begin{cases} \langle N(p_i), N(p_j) \rangle & \sigma_k^1 \in \mathcal{M}, \quad v_i \in \sigma_k^1 \text{ and } v_j \in \sigma_k^1 \\ 0 & \text{otherwise}, \end{cases}$$

where $p_i = P(v_i)$, $p_j = P(v_j)$ (see Section 1.1), and s_{ij} correspond to the entries of the stiffness matrix S from Equation 3.4.

Proof. The Dirichlet energy for the k-th component function of the scaled normal is given by:

$$E_D(s_N(u)_k) = \frac{1}{2} u^T \hat{N}_k^T S \hat{N}_k u,$$

where $\hat{N}_k$ denotes a diagonal matrix with diagonal entries $\hat{n}_{k,ii}$ set to the k-th normal component at p_i. Using this notation we expand E_D^N from Definition 6 into its three terms

$$E_D^N(u) = \frac{1}{2} u^T \left(\hat{N}_1^T S \hat{N}_1 + \hat{N}_2^T S \hat{N}_2 + \hat{N}_3^T S \hat{N}_3 \right) u.$$

We then set

$$A = \hat{N}_1^T S \hat{N}_1 + \hat{N}_2^T S \hat{N}_2 + \hat{N}_3^T S \hat{N}_3.$$

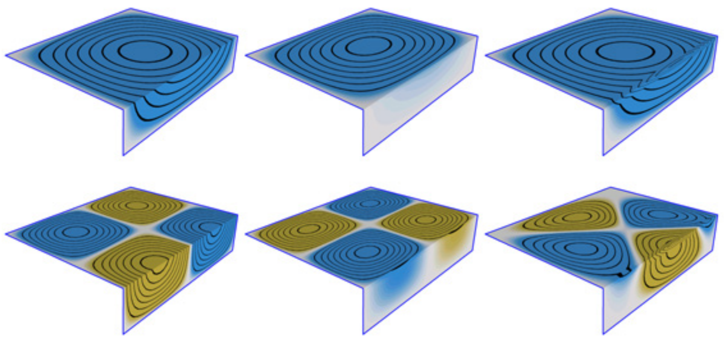

Figure 3.2: Visualization of modes of different energies. The left column shows the Laplacian eigenmodes, the middle column shows the eigenmodes of the modified Dirichlet energy E_D^N, and the third column shows the vibrations modes derived from the thin shell energy restricted to normal variations as described in [Hildebrandt et al., 2012].

Collect terms using the fact that the $\hat{N}_k$ are diagonal so the entries a_{ij} of A are of the form

$$a_{ij} = (\hat{n}_{1,ii}\hat{n}_{1,jj} + \hat{n}_{2,ii}\hat{n}_{2,jj} + \hat{n}_{3,ii}\hat{n}_{3,jj})\, s_{ij}.$$

Then using the correspondence

$$\langle N(p_i), N(p_j)\rangle = (\hat{n}_{1,ii}\hat{n}_{1,jj} + \hat{n}_{2,ii}\hat{n}_{2,jj} + \hat{n}_{3,ii}\hat{n}_{3,jj})$$

we get our result. ■
We split the matrix representation of $E_D^N(u)$ into two parts

$$u^T A u = u^T S u + u^T C u. \tag{3.13}$$

The entries of C are then given by

$$c_{ij} = \begin{cases} 0 & i = j \\ (n_{ij} - 1)\, s_{ij} & \textit{otherwise.} \end{cases}$$

By construction, the positive definiteness of A depends on the values of n_{ij} and the entries s_{ij}. The discrete modified Dirichlet energy is a quadratic functional and therefore has a constant Hessian. At any $u \in S_h$, we have $\partial_u^2 E_D^N = A$.

Given a surface $\mathcal{M}_h$, we assume that the normals satisfy $0 \leq n_{ij} \leq 1$. Together with our assumptions on the entries s_{ij} of the stiffness matrix S at the end of Section 3.3.2, that is, $s_{ij} < 0$ for $i \neq j$, the entries c_{ij} of C satisfy $c_{ij} \geq 0$ (see Equation 3.13). Thus under these conditions A and, in particular, C are semi-positive definite.

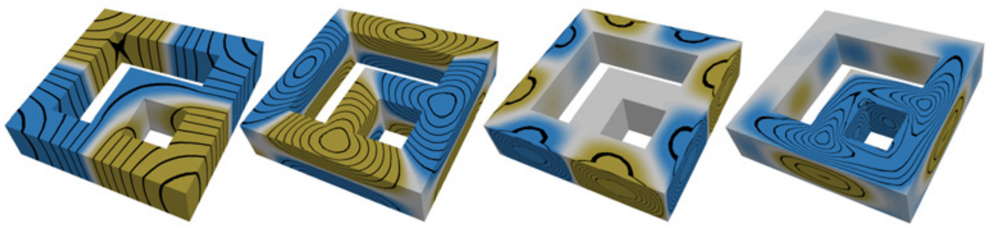

Figure 3.3: Two eigenmodes of the lower spectrum on the double torus with sharp features, left: Laplacian, and right: modified Dirichlet energy.

3.3.4 Eigenmodes

We experimented with the eigenfunctions of $\nabla^2 E_D^N$ and the eigenfunctions of the Laplace operator, that is, the eigenfunctions of $\nabla^2 E_D$.

To compute the eigenmodes of a mesh, we solved the generalized eigenvalue Problem 5. Since M is a diagonal matrix, this problem can be transformed into a standard eigenvalue problem as described in [Vallet and Lévy, 2008]. Then we solve the resulting standard eigenvalue problem with the shift-and-invert Lanczos scheme described in [Vallet and Lévy, 2008]. For most examples and applications, we do not need to compute the full spectrum, but only the lower part of the spectrum.

For the first example, we study how the eigenmodes change when we isometrically deform a flat plate by sharp bend, see Figure 3.2. On the undeformed flat plate, the eigenmodes of E_D^N coincide with the eigenmodes of the Laplacian shown in Figure 3.1. As illustrated in Figure 3.2, there are certain differences between the three types of considered modes when computed on the deformed plate. Due to its intrinsic nature, the Laplacian eigenmodes ignore the newly introduced feature, Figure 3.2 left. In contrast, the eigenmodes of E_D^N are sensitive to the feature, Figure 3.2 middle. The eigenmodes of E_D^N corresponding to lower eigenvalues almost vanish at the feature. For further comparison, eigenfunctions from an operator introduced in [Hildebrandt et al., 2012] are shown in Figure 3.2 right. In contrast to the eigenfunctions E_D^N, these eigenfunctions place critical points along features.

Investigating the differences between the eigenmodes of the Laplacian and E_D^N further, we compute them on the double torus with sharp features shown in Figure 3.3. Each of the Laplacian eigenmodes show a more or less equally distributed set of extrema as well as a certain reflective symmetry, Figure 3.3 left. The corresponding isolines suggest a low influence of the sharp features to the considered Laplacian eigenmodes. Similar to the Laplacian modes, the two eigenmodes for E_D^N also have a reflection symmetry, Figure 3.3 right. But here we find that the eigenmodes of the lower part of the spectrum correspond

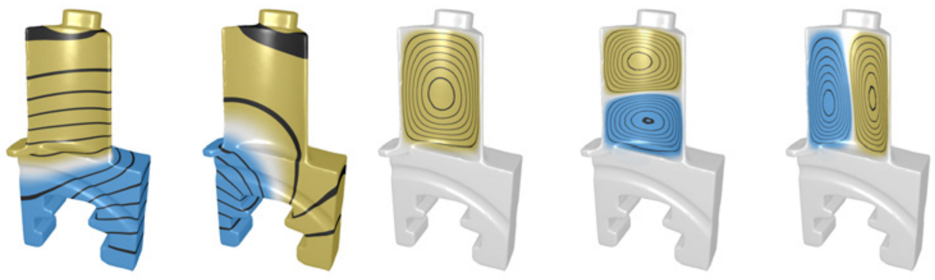

Figure 3.4: A qualitative comparison of modes of the modified Dirichlet energy and modes of the Laplacian is shown. The two left models illustrate two lower modes of the Laplacian. The three models on the right show lower modes of the modified Dirichlet energy, which respect the extrinsic features.

to oscillations of flat areas surrounded by sharp edges, Figure 3.3 right. For a third comparison, we choose a model without sharp edges: a CAD model, see Figure 3.4. As in the previous cases, we notice that the Laplacian eigenmodes oscillate equally over the whole surface, see Figure 3.4 left. In contrast, the eigenfunctions of E_D^N respect the extrinsic geometry features, e.g., they align to the smooth creases on the CAD model, see Figure 3.4 right. Furthermore, on the CAD model, eigenfunctions corresponding to the lower eigenvalues of the spectrum tend to concentrate on larger, less curved parts of the surface, e.g., the cylindrical part of the CAD model.

The eigenfunctions of discrete Laplace operator and E_D^N differ significantly, see Figure 3.2, 3.3 and 3.4. In our experiments, we observed that modes of E_D^N corresponding to lower eigenvalues hardly move in regions of high curvature.

A possible explanation for this behavior can be found by examining the energy changes caused by a variation $u \in S_h$ away from a minimum. With our assumptions on the entries of S and A, the energies $E_D^N(\cdot)$ and $E_D(\cdot)$ are minimal at $m = 0$. Then an energy change caused by a variation $u \in S_h$ at m is given by:

$$\begin{aligned} E_D(m+u) - E_D(m) &= \left\langle \nabla_m^2 E_D(u), u \right\rangle_M \\ &= u^T S u \\ E_D^N(m+u) - E_D^N(m) &= \left\langle \nabla_m^2 E_D^N(u), u \right\rangle_M \\ &= u^T S u + u^T C u. \end{aligned} \tag{3.14}$$

Hence the energy changes caused by a variation $u \in S_h$ differ by a term $u^T C u$. By our assumptions, $u^T C u$ is required to increase the energy for variations u that are nonzero within surface regions of high curvature. Because variations $\phi \in S_h$ in the direction of an eigenmode associated to small eigenvalues correspond to a direction with a small change in energy, we deduce that the modes of low eigenvalues of $\nabla_m^2 E_D^N$ tend to avoid the additional energy increase by $\phi^T C \phi$ occurring in curved regions.

3.4 Extended Dirichlet energy

Based on the modified Dirichlet energy from Section 3.3, we introduce a new family of extrinsic features sensitive energies. This energy class is parametrized by a scalar parameter $t \in \mathbb{R}$, which controls the feature sensitivity of the corresponding energy. We refer to this family as the extended Dirichlet energies defined by

Definition 7 (Extended Dirichlet Energy) *An extended Dirichlet energy is given by*

$$E_\Delta^X(f,t) = E_\Delta(f) + \frac{1}{2} t \int_\Sigma f^2(\kappa_1^2 + \kappa_2^2) dA,$$

the sum of a Dirichlet energy term and a scaled total curvature term.

Using E_Δ^N from Equation 3.10 we can represent E_Δ^X as a weighted sum of E_Δ and E_Δ^N. To isolate the total curvature term from E_Δ^N, we use the difference $(E_\Delta^N(u) - E_\Delta(u))$. Then E_Δ^X is given by

$$\begin{aligned} E_\Delta^X(f,t) &= E_\Delta(f) + t(E_\Delta^N(f) - E_\Delta(f)) \\ &= tE_\Delta^N(f) + (1-t)E_\Delta(f). \end{aligned} \tag{3.15}$$

Thus choosing a $t \neq 0$, the total curvature term influences the corresponding energy. That is, it becomes sensitive to extrinsic surface features. For $t = 0$ we get the Dirichlet energy from Equation 3.8, i.e., $E_\Delta^X(f,0) = E_\Delta^X(f)$. Setting $t = 1$, we get the modified Dirichlet energy from Equation 3.9, i.e., $E_\Delta^X(f,1) = E_\Delta^N(f)$. Choosing a larger t, the total curvature term, that is, the extrinsic feature sensitive part, starts to dominate the energy.

We define the discrete counterpart E_D^X to Definition 7 by

Definition 8 (Discrete Extended Dirichlet Energy) *A discrete extended Dirichlet energy is given by a weighted sum of the discrete Dirichlet energy and the modified Drichlet energy, that is,*

$$E_D^X(u,t) = tE_D^N(u) + (1-t)E_D(u). \tag{3.16}$$

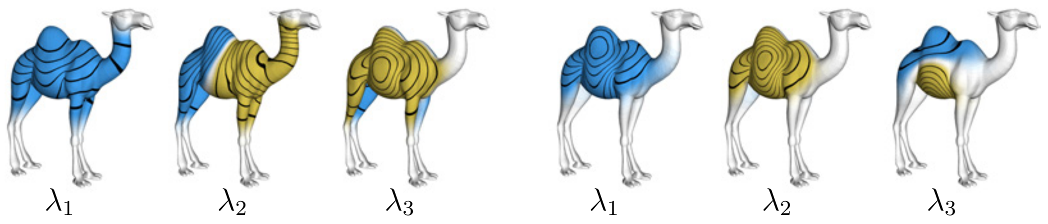

Figure 3.5: Lower modes derived from the Hessian of two discrete extended Dirichlet energy that have different t values. The first three eigenfunctions on the camel model are shown for $t = 0.05$ on the left left and $t = 0.5$ on the right.

By construction, the extended Dirichlet energy is a quadratic functional on S_h. We refer to $\nabla_t^2 E_D^X$ as the Hessian of a discrete extended Dirichlet energy for $t \in \mathbb{R}$.

The first three eigenfunctions of the spectra of $\nabla_{t_1}^2 E_D^X$ and $\nabla_{t_2}^2 E_D^X$ with $t_1 = 0.05$ and $t_2 = 0.5$ for the camel model are shown in Figure 3.5. Notice that the nonzero variations of the eigenfunctions for $\nabla_{t_2}^2 E_D^X$ are concentrated in the less curved parts compared to the variations described by eigenfunctions of $\nabla_{t_1}^2 E_D^X$.

3.5 Stability of discrete cmc-surfaces

In the continuous setting, properties of constant mean curvature surfaces or, in short, cmc-surfaces have been investigated intensively. We will denote by H and K the mean curvature and Gauss curvature, respectively. By $C_0^k(\Sigma) \subset C^k(\Sigma)$ we refer to the k-times differentiable compactly supported functions on Σ. The subset $C_{0,0}^k(\Sigma) \subset C_0^k(\Sigma)$ contains functions that satisfy $\int_\Sigma f dA = 0$. Following [Barbosa et al., 1988] and [Polthier and Rossmann, 2002] stable cmc-surfaces are then defined by

Definition 9 *A cmc-surface Σ is stable if $J(f) > 0$ for all $f \in C_{0,0}^1$, where the functional $J(\cdot)$ is given by $J(f) = \int_\Sigma \langle \nabla f, \nabla f \rangle - (4H^2 - 2K) f^2 dA$.*

As shown in [Barbosa et al., 1988] and [Polthier and Rossmann, 2002] the functional $J(f)$ corresponds to the second-order change in area when the set of allowed variations is restricted to volume preserving variations. To determine stability, two indices I and I_u are defined in [Polthier and Rossmann, 2002]. I is the geometric index and corresponds to the dimension of the subspace of allowed functions $f \in C_{0,0}^k(\Sigma)$ for which $J(f) < 0$. Hence for a stable cmc-surface, we have $I = 0$.

The unconstrained index I_u corresponds to the dimension of the space of unconstrained variations for which $J(f) < 0$ with $f \in C_0^1(\Sigma)$. Hence it can be used to estimate stability because $I_u \geq I$. We can rewrite the functional $J(f)$ used in Definition 9 by

$$\begin{aligned} J(f) &= E_\Delta(f) - \int_\Sigma (4H^2 - 2K) f^2 dA \\ &= E_\Delta(f) - \int_\Sigma (\kappa_1^2 + \kappa_2^2) f^2 dA. \end{aligned}$$

Hence the functional $J(f)$ can be represented as a certain energy from the class of extended Dirichlet energies given in Definition 7, that is, $J(f) = E_\Delta^X(f, -2)$. Denoting by $S_{h,0} \subset S_h$ the subset of functions that vanish on the boundary of $\mathcal{M}_h$ we define its discrete counterpart J_D by

$$\begin{aligned} J_D(u) &= E_D^X(u, -2) \qquad u \in S_{h,0} \\ &= \frac{1}{2} u^T B u, \end{aligned} \tag{3.17}$$

where the matrix B and its entries b_{ij} are given by

$$B = \frac{1}{2}(-2A + 3S) \qquad \text{and} \qquad b_{ij} = s_{ij}(3 - 2n_{ij}).$$

We refer to Λ_- as the set of all negative eigenvalues of $\nabla^2 J_D = M^{-1}B$. Denoting by $\mu(\lambda_i)$ the multiplicity of an eigenvalue λ_i, the dimension of the subspace of $h, 0$ for which $J(u) < 0$ is then given by

$$\bar{I}_u = \sum_{\lambda \in \Lambda_-} \mu(\lambda).$$

If $\mathcal{M}_h$ is a discrete cmc-surface we call $\bar{I}_u$ its energy based unconstrained index. Hence given a family of simplicial surfaces that point- and normal-converge to a smooth cmc surface we can estimate I_u by $\bar{I}_u$.

We determined the energy based unconstrained index $\bar{I}_u$ for three cmc-surfaces for the trinoid, catenoid, and Enneper surface. We approximated the smooth normal vector field by a normal vector field $N \in V_h$, where $N(p_i)$ is computed by averaging the incident triangle normals. The computed energy based unconstrained indices correspond to the unconstrained indices I_u given in [Polthier and Rossmann, 2002]. The eigenfunctions ϕ corresponding to $\lambda \in \Lambda_-$ are shown for the three surfaces in Figure 3.6 and Figure 3.7.

To determine the number of negative eigenvalues we computed the full spectrum and all eigenfunctions using the divide-and-conquer algorithm from LAPACK [Anderson et al., 1999].

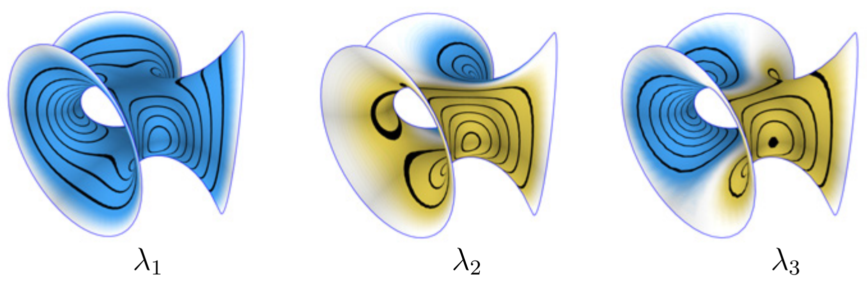

Figure 3.6: Illustration of the three-dimensional subspace for which $J(u) < 0$ for the trinoid, where $\lambda_1 < \lambda_2 \approx \lambda_3 < 0$.

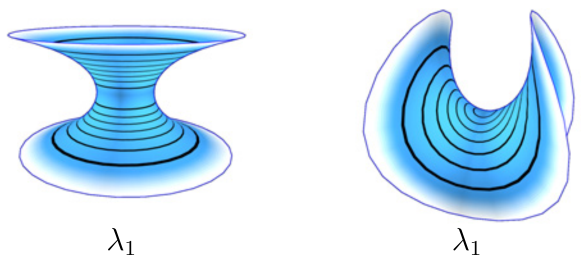

Figure 3.7: Illustration of the one-dimensional subspace for which $J(u) < 0$ for the catenoid and the enneper surface, that is, both $\lambda_1 < 0$.

3.6 Modal signatures

In this section we introduce a multi-scale surface signature. The feature signature is based on the eigenfunctions and eigenvalues of the modified discrete Dirichlet energy E_D^N. The construction of the signatures follows the construction of the heat kernel signature defined in [Sun et al., 2009].

Extending the heat diffusion process from Euclidean space to manifolds, the Laplace operator must be defined on these manifolds. The resulting operator is the Laplace–Beltrami operator. By describing the diffusion process by the Laplace–Beltrami operator, the geometry around a point determines its diffusion behavior. The corresponding signature captures the temperature change over time at any point in time. A signature exploits the fact that similar neighborhoods result in similar diffusion behavior.

The signature we consider is a multi-scale signature that takes a positive scale parameter t as its input. For every t, the signature is a function on the mesh, i.e., it associates a real value to every vertex of the mesh.

We define the feature signature for $p \in \mathcal{M}_h$ and scale $t \in \mathbb{R}^+$ by

$$S_t^F(p) = \sum_j e^{-\lambda_j t}\, \phi_j(p)^2, \tag{3.18}$$

where the λ_j are the eigenvalues and the $\phi_j \in S_h$ are the L^2-normalized eigenmodes of the Hessian of the modified discrete Dirichlet energy E_D^N.

The weights depend on the eigenvalues λ_i and on the scale parameter t. For increasing λ, the function $e^{-\lambda t}$ rapidly decreases. The modes with smaller eigenvalue receive higher weights than the modes with larger eigenvalues. Furthermore, for increasing t, all weights decrease and, more importantly, the weights of the eigenmodes with smaller eigenvalues increase relative to the weights of the modes with larger eigenvalues.

From the feature signature, we can construct the following multi-scale pseudo-metric on the mesh. Let $p, q \in \mathcal{M}_h$, then we define the corresponding pseudo-metric between p and q by

$$\delta_{[t_1,t_2]}(p,q) = \left(\int_{t_1}^{t_2} \left(\frac{S_t^F(p) - S_t^F(q)}{\sum_k e^{-\lambda_k t}} \right)^2 \mathrm{d}\log t \right)^{\frac{1}{2}}. \tag{3.19}$$

By construction, for any pair of scalar values $t_1 < t_2$, $\delta_{[t_1,t_2]}$ is positive semi-definite, symmetric and satisfies the triangle inequality. We call the pseudo-metric constructed from S_t^F the feature distance.

The idea behind the construction of the pseudo-metric is to use the integral $\int_{t_1}^{t_2} \left(S_t^F(p) - S_t^F(q)\right)^2 dt$. However, the actual definition additionally includes two heuristics. First, since the values $S_t^F(p)$ decreases for all p for increasing t, we normalize the value $\left(S_t^F(p) - S_t^F(q)\right)$ by dividing it by the discrete L^1-norm of S_t^F,

$$\left\| S_t^F \right\|_{L^1} = \sum_k e^{-\lambda_k t}.$$

Second, for a fixed vertex p, the signature $S_t^F(p)$ varies more for small values of t compared to large t.

To increase the discriminative power of the pseudo-metric, we associate a higher weight to the small t and a lower weight to the larger t. We achieved this by using a weighted integral with weight function $\mathrm{d}\log t = \frac{1}{t}\mathrm{d}t$. To discretize this weighted integral, we use a uniform decomposition of the logarithmically scaled interval $[t_1, t_2]$.

3.7 Applications

The feature signature and the feature distance can be used to identify features of the surface, like sharp bends or sharp corners. The signature could serve as an indicator function to surface segmentation algorithms. Figure 3.8 shows the feature signature on the rocker arm model for different scales. Vertices of the mesh that have a signature value close to zero are colored white in

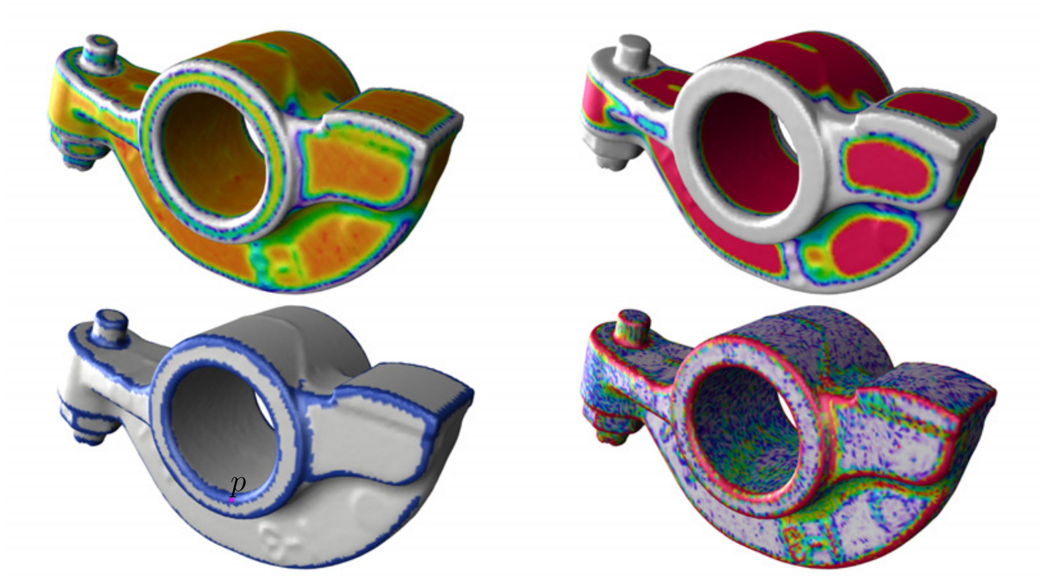

Figure 3.8: Results of the feature signature on the rocker arm model. The top row shows the feature signature for increasing scale values. The bottom row shows on the left the feature distance to the marked vertex $p \in \mathcal{M}_h$ binary colored by a threshold, and, on the right, the surface colored by curvature $(\sqrt{\kappa_1^2 + \kappa_2^2})$.

these images. The white areas seem to include the important features of the rocker arm model. The lower left image shows in blue all the vertices that are close (with respect to the feature distance) to a vertex on a sharp bend. For comparison, we also show a curvature plot $(\sqrt{\kappa_1^2 + \kappa_2^2})$ on the rocker arm.

Concerning the applicability as a feature indicator of the feature signature, a nice aspect of the feature signature compared to curvature is that the feature signature naturally comes with a scale parameter whereas for curvature a scale space needs to be constructed. Another interesting difference is that some areas of the rocker arm model have high curvature but do not indicate features. The curved area inside the hole has a much higher curvature than, for example, the flat parts on the sides of the model. Still, the feature distance associates similar function values to both of these parts.

Feature detection is an application that highlights differences between our feature signature and the heat kernel signature or, in short, HKS. This is due to the isometry invariance of the HKS, which implies that the HKS marks only features that remain features under arbitrary isometric deformations. For example, sharp bends (like the one in Figure 3.2) are not regarded as features by the HKS. Similar to the example shown in Figure 3.8, we select an initial point and mark all points that are close to the initial point with respect to a modal distance. For comparison, Figure 3.9 shows results obtained with our feature distance (colored in blue) and with the heat kernel distance (colored in green) on a collection of models. The first example, the blade model (top left) shows the features detected for three different initial points located on corners and sharp bends of the surface. The feature signature finds almost identical sets of features for all three initial points. The results the HKS produces vary strongly.

On the turbocharger model (top right), we select two different initial points: one located on a sharp bend and one point close to a bend. The images show that in the second case, the feature signature marks points that have similar distance to a feature as the initial point. Since the sharp bends on this surface are curved (in the tangential direction orthogonal to the bend), the HKS detects these features as well and produces comparable results to the feature signature. A second example where the feature signature and the HKS produce similar results is the bumpy plate (bottom right).

To determine the eigenfunctions ϕ_i, we need to solve a sparse generalized eigenproblem (see Equation 3.6). One way to solve such a generalized eigenvalue problem is to transform it to a standard eigenvalue problem. In our case, the mass matrix is a positive definite diagonal matrix. Therefore such a basis transformation requires only a rescaling of the Lagrange basis vectors. Details for this procedure can be found in [Vallet and Lévy, 2008].

To compute the feature signature and feature distance, only a fraction of the lower part of the spectrum is required because the weights $e^{-\lambda_j t}$ rapidly decrease with increasing eigenvalue. Typically, the first 300 eigenvalues and modes yield a faithful approximation of the signatures and distances. To efficiently compute a lower portion of the spectrum and its corresponding eigenvectors we employ the shift-and-invert Lanczos method which does not need the inverse matrix explicitly. Instead, only a matrix vector product has to be provided, which can be evaluated by solving a linear system of equations. We solve these systems using the sparse direct solver implemented in *MUMPS*, see [Amestoy et al., 2001]. Once the eigenvalues are computed, the evaluation of the signatures and distances is relatively fast. To discretize the integral in Equation 3.19, we use a numeric quadrature. We place the samples of the interval $[t_1, t_2]$ so that they are equidistant on the logarithmic scale, which yields equal weights for all points in the quadrature.

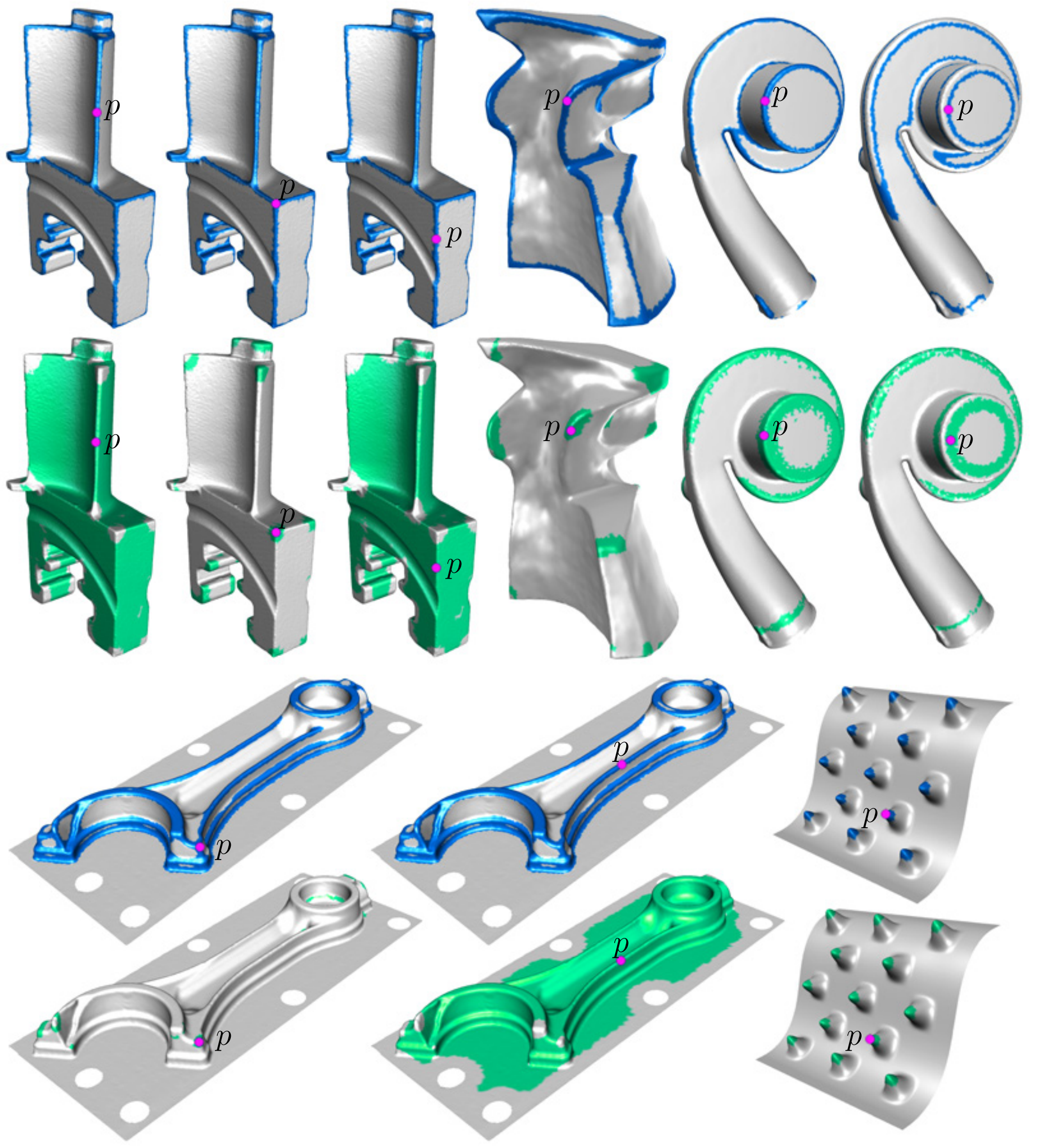

Figure 3.9: Comparison of our feature signature and the HKS on a collection of models are shown. For a selected point $p_i \in \mathcal{M}_h$ similar points are denoted by blue for our signature and by green for the HKS.

3.8 Summary

We presented an approach to derive operators from the Hessians of a certain type of a surface energy acting on surface functions. These operators are feature sensitive and have an easy to implement matrix representation.

We derived a modified Dirichlet energy that is quadratic and depends on the surface normals. In contrast to the spectrum and eigenfunctions of the Laplacian, its associated operator possesses a spectrum that is sensitive to extrinsic surface features, i.e., the squared total curvature. We demonstrate this sensitivity of the eigenfunctions on three examples.

Based on the definition of the modified energy we introduced a family of energies whose feature sensitivity can be controlled by a scalar. An energy of this class is given by a weighted sum of the Dirichlet energy and our modified Dirichlet energy. That means it has a simple matrix representation. By construction the corresponding family of operators, that is, the Hessians of these energies, possess feature sensitive spectra and eigenfunctions.

We showed that under certain assumptions for the normals of a surface mesh, a certain operator of this family can be used to approximate the unconstrained stability index of cmc-surfaces. We estimated the index for three minimal surfaces. A possible next step is to estimate the stability using the operator for non minimal cmc-surfaces, such as the wente torus.

Similar to the Heat Kernel Signature we constructed a point signature based on the spectra and eigenfunctions of the modified Dirichlet energy. We applied the signature on various models to identify similar points and compared it to point identification resulting from the HKS. Having the scale parameter to control an operator's feature sensitivity, the properties of the corresponding scaled signatures still needs to be explored.

Another application for the feature sensitive spectra and eigenfunctions is to use them in the generation of quadrangulations that align to salient surface features. Further possible research directions are the feature signature's potential to place point singularities for quadrangulations/parametrizations or to determine inter shape correspondence.

Chapter 4

Patch Layout from Feature Graph

Reverse engineering deals with the reconstruction of CAD surfaces typically from scanned 3D geometries. Current CAD system are based mainly on spline geometries. A scanned triangle mesh must be converted into a highly structured and segmented data structure that can then be used to determine an appropriate spline representation. Our algorithm helps to automate this reconstruction process. It consists of two general steps. In the first step, we generate an abstract description of the final patch layout. This description is encoded in a feature graph. This means there is a one-to-one relation between the feature graph elements and the regions of the patches of the final layout. Furthermore, the feature graph is a graph embedded on the surface and its smooth edges are oriented along geometric surface features. In the second step, a geometrically reasonable patch layout is generated from the feature graph. The resulting patches have a uniform curvature distribution and are encircled by smooth boundaries. For our decomposition approach, no primitive fitting, i.e., template matching, is required making it easy and fast to compute.

In Section 4.2, we explain our basic concepts and underlying notions of a feature graph and a patch layout. The procedure to generate a feature graph is contained in Section 4.3. The creation of the patch layout from a given feature graph is explained in Section 4.4. Finally, we present some results of our surface decomposition approach in Section 4.5.

4.1 Background

Our patch layout algorithm is related to many known techniques in surface segmentation and graph smoothing algorithms. A general overview of surface decomposition methods is given in [Shamir, 2006].

Surface segmentation has its roots in image processing, where surfaces are treated as height fields, i.e., there exists a canonical parametrization of the surface over a planar domain as used in [Sapidis and Besl, 1995]. The main part of [Shamir, 2006] contains a detailed overview of segmentation algorithms working on general triangulated surfaces showing their variety of applications and implementations. Segmentation algorithms can be used for various purposes ranging from remeshing, animation [Bergou et al., 2007], shape matching [Funkhouser et al., 2004], and mesh editing to geometry compression and other areas. We focus on the segmentation of CAD parts for reverse engineering.

Some related work focuses on surface segmentation by approximation with several kinds of predefined types of primitives. In [Cohen-Steiner et al., 2004], planes are fitted. [Wu and Kobbelt, 2005] uses a collection of CAD primitives, such as spheres or rolling ball blends. The use of parametrized shapes is suggested in [Joris S. M. Vergeest and Jelier, 2001].

Clustering vertices into groups belonging to a specific shape type using multiresolution is demonstrated in [Attene et al., 2006] and [Garland et al., 2001]

[Julius et al., 2005, Shatz et al., 2006] show a tiling of a given model into nearly-developable charts. This kind of chart tiling makes it possible to recreate the given surface as a paper craft model.

[Lévy et al., 2002] use a region growing algorithm for creating patches whose boundaries run along sharp features. In the first step, some surface features are detected; then a set of regions is constructed, which meet at these features.

There are many approaches to computing a feature layout using Morse theory. In [Dong et al., 2006], an eigenvector of the Laplacian is computed and used as a Morse function. The Morse complex which is then built from this function segments the surface into quads. In [Edelsbrunner et al., 2001, Cazals et al., 2003], the construction of a Morse-Smale complex is described. By prescribing an adequate Morse function that represents the important parts of the surface, one can control the alignment of the feature layout. In [Edelsbrunner, 2005], a curvature based Morse function is used to construct a Morse-Smale complex that aligns to surface features. This approach is applied to CAD models in [Várady et al., 2007].

[Huang et al., 2009] uses vibration modes of a surface to decompose it

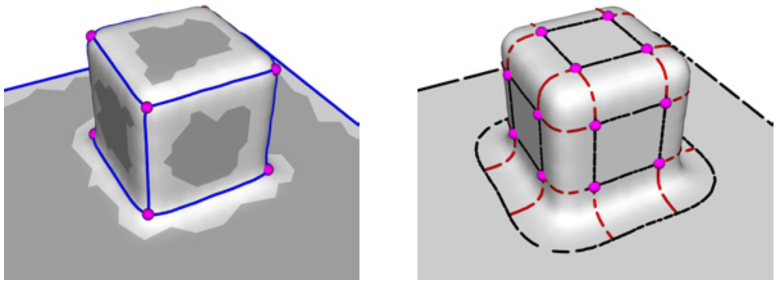

Figure 4.1: Feature graph on CAD part consisting of faces, feature edges and node points are shown on the left. The dark grey parts within each face indicate the plane-like or weakly curved parts and the light grey parts show where the surface starts to curve. Patch layout with face patches, fillets, and node patches, as well as offset and node curves and offset nodes are shown on the right.

into physically meaningful parts. They compute the modes of the surface from the Hessian of a simplification of the As-rigid-As-possible deformation energy, which was proposed by [Sorkine and Alexa, 2007].

The patch boundaries should be smooth curves. In [Lee and Lee, 2002] the use of snakes for the generation of smooth curves on triangulated surfaces is proposed. This approach requires the repeated projection of the actual curve onto a two-dimensional domain. The smoothness of the curve is controlled by an energy term. Recasting the problem of smooth curves on triangulated manifolds to a high dimensional optimization problem is described in [Hofer and Pottmann, 2004]. Furthermore, the alignment of curves along features can be driven by the use of the feature sensitive metric introduced in [Pottmann et al., 2004]. Thickening of smooth curves is also mentioned in [Várady et al., 2007] but without going into the actual details of the thickening procedure.

4.2 Setup

We consider simplicial surfaces $\mathcal{M}_h \subset \mathbb{R}^3$ (see Section 1.1) without self intersections. The two-dimensional cells, i.e., its triangles $\sigma_i^2 \in \mathcal{M}$, are denoted by T_i. Furthermore, to a set of triangles $A = \{T_{A_1}, \dots, T_{A_m}\}$ with $T_{A_i} \in \mathcal{M}$, we associate the subset $|A| \subset \mathcal{M}_h$ with $|A| = \bigcup_i |T_{A_i}|$.

The basic idea of our algorithm is to initially decompose the surface $\mathcal{M}_h$ into disjoint sets given by sets of triangles I_i with $\bigcup_i |I_i| \subseteq \mathcal{M}_h$. The sets I_i and $|I_i|$ are referred to as primitives. The final patch layout is generated based on this decomposition. The initial primitive guess is already a rough approximation of the final layout (Figure 4.1 left, Figure 4.2 left). They lack smooth boundaries and inherent connectivity information. To create the patch layout from the initial set of primitives I_i, we introduce the feature graph (Figure 4.1 left). The feature graph will provide the necessary connectivity information. Additionally the feature graph's significant points and curves will meet certain smoothness and alignment requirements to create the final patch layout (Figure 4.1 right). For a complete description of our layout generation method we define the feature graph and the patch layout.

A feature graph is a graph on the surface (Figure 4.1 left) that describes the underlying structure of a CAD surface. The feature graph is a net of smooth surface curves, which run along surface features. It consists of: faces, feature edges, and node points. A face is a surface region containing a primitive. These can be any kind of primitive, e.g., planes, cylinders, cones. We mainly focus on plane-like faces. Feature edges are smooth edges that separate two adjacent faces. Feature edges run along in cylindrical/conical regions. Node points are points on the surface where several feature edges meet, i.e., these points are also incident to more than one feature graph face. These are isolated points expected to be in spherical/hyperbolic regions.

A patch layout is an embedded graph on the surface that decomposes the geometry into various cells (Figure 4.1 right). Within each cell of the patch layout, heavily changing curvature is not allowed. We expect three possible cell types: almost planar or weakly curved parts called face patches, regions between two adjacent face patches called fillets, and regions that connect several fillets called node patches. Face patches, fillets, and node patches correspond to feature graph faces, edges, and node points, respectively. Usually, a fillet has a cylindrical or conical shape and node patches are spherical or hyperbolic.

We will encounter two types of boundaries between cells of a patch layout: offset curves and node curves. Offset curves encircle face patches. Each offset curve separates a face patch from an adjacent fillet. The feature edge that represents this fillet runs more or less parallel to the offset curve. Node curves separate fillets from node areas. In general, each node patch is bounded by a sequence of smooth node curves. Start and end points of these curves will be denoted as offset nodes.

4.3 Feature graph

The basis of a consistent layout is a feature graph representing the layout's final structure. So given a simplicial surface $\mathcal{M}_h$, we present a strategy to build all parts of a feature graph, such as faces, feature edges, and nodes.

Algorithm 3: Generate the feature graph

```
**Input**: surface $\mathcal{M}_h$
**Output**: feature graph
Compute principle curvatures (values and directions)
Detect initial regions $I_i$
Expand regions $I_i$ by a region growing process
Extract nodes and edge based face boundaries
Create smooth feature edges from edge based face boundaries
```

We focus on detecting plane-like regions. The basic idea is to detect these primitives $|I_i| \subset \mathcal{M}_h$ with $I_i \cap I_j = \emptyset$ and expand them to cover all of $\mathcal{M}_h$.

Having covered all of $\mathcal{M}_h$, we detect node points. The node points are connected by curves that separate adjacent expanded primitives. These curves are very jagged and run only along edges of the underlying mesh. We take these curves as the first approximation for the later feature edges. Thus, they need to be smoothed to meet our alignment and orientation requirements.

The result of this last step is a net of smooth curves on the surface – the feature graph. These curves encircle the feature graph faces containing the primitives $|I_i|$; in our case they describe the weakly curved or plane-like part of each face.

The steps to generate the feature graph are summarized in Algorithm 3. The details for the single steps are explained in detail in the following subsections.

4.3.1 Computing curvatures

The algorithm starts by computing the curvature information, i.e., principle curvature values and directions, at each vertex of the mesh. We use an approximation of the shape operator given in [Hildebrandt and Polthier, 2004]. It is a stable and reliable method and requires no fitting. Other methods, see [Cohen-Steiner and Morvan, 2003, Pottmann et al., 2007], can also be applied. Having curvature information at all vertices enables us to assign curvature information to each triangle $T_i \in \mathcal{M}$ by averaging the curvature information of all three incident vertices. Thus, for each triangle T_i, two averaged unit vectors pointing in principle curvature directions ($\pm X_{max,i}$, $\pm X_{min,i}$) are given together with their corresponding averaged curvature values ($\kappa_{max,i}$ and $\kappa_{min,i}$) with $|\kappa_{max,i}| \geq |\kappa_{min,i}|$.

4.3.2 Detecting primitives

Primitives I_i are taken as seeds for the generation of the feature graph faces (see Figure 4.1 left, Figure 4.2 left) by a region growing algorithm. We expect the primitives to be characterized by their curvature properties. Because we focus on the detection of almost planar primitives only one curvature threshold $\tau > 0$ is needed. Thus we define the following set of flat triangles $I := \{T_i \in \mathcal{M} \mid |\kappa_{max,i}| < \tau\}$. In general, I can be split into a set of simply connected disjoint components or primitives I_i, i.e., $I = \bigcup_i I_i$ with $I_i \cap I_j = \emptyset$. Apart from using τ to characterize initial regions, it can also be considered as a measure of acceptable noise within a primitive, that is a higher value of τ will ignore more noise.

4.3.3 Edge based feature graph

The result of the expansion of the primitives I_i is a rough approximation of the feature graph. It defines the combinatorial structure of the feature graph, that is, it determines the node points and provides connectivity information between those points. The detected node points are then fixed for the remainder of the patch layout generation. The primitives I_i are grown

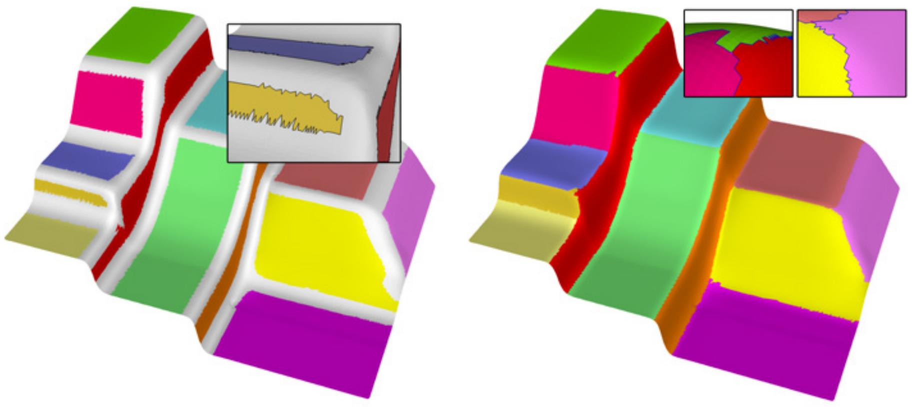

Figure 4.2: Illustration of the expansion step. The primitives $|I_i|$ are shown (left). The complete covering of $\mathcal{M}_h$ by expanded regions $|F_i|$ including parts of the unsmoothed feature graph are shown (right).

until the whole surface is covered. We denote F_i to be the triangle set that results from the expansion of the initial set I_i. That is after expansion we have $\bigcup_i |F_i| = \mathcal{M}_h$ with $F_i \cap F_j = \emptyset$. Various expansion strategies exist. Our experiments show that node points are best placed where the absolute

Gaussian curvature $|K|$ is high and that feature graph edges should be placed where $|\kappa_{max,i}|$ is high.

In order to ensure the correct placement of feature graph nodes and edges, we developed an expansion strategy based on curvature information. We use a region growing approach to realize the expansion of initial regions $|I_i|$ into the uncovered part of $\mathcal{M}_h$. The growing is controlled by a feature function, which assigns curvature related priority values to free triangles $T_i \notin I$.

We use $|\kappa_{max,i}|$ as the feature function for the expansion into cylindrical surface regions. However, when the surface becomes spherical or hyperbolic, the values of $\kappa_{min,i}$ and $\kappa_{max,i}$ become more and more similar. To ensure placement of node points where $|K|$ is high, we use $|K|$ as a feature function within spherical/hyperbolic regions. We use $|\kappa_{max,i}| - |\kappa_{min,i}|$ as an estimate for the stability of $|\kappa_{max,i}|$. We classify the free triangles $T_i \notin I$ into two

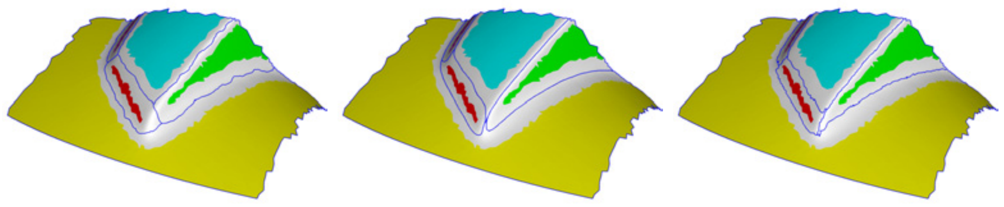
Figure 4.3: Results from expanding initial regions with varying growing strategies. From left to right: geodesic distance approximation by [Kimmel and Sethian, 1998], metric from third fundamental form by [Pottmann et al., 2004], our feature function.

groups: the unstable group N and stable group E, that is,

$$\begin{aligned} N &:= \{T_i \in \mathcal{M} \mid T_i \notin I, |\kappa_{max,i}| - |\kappa_{min,i}| < t\}, \quad t \in \mathbb{R} \\ E &:= \{T_i \in \mathcal{M} \mid T_i \notin (I \cup N)\}. \end{aligned}$$

Our region growing is driven by $|\kappa_{max,i}|$ for $T_i \in E$ and by $|K|$ for $T_i \in N$. A comparison of feature graph nodes placement by our region growing strategy to results using geodesic distance approximation from [Kimmel and Sethian, 1998] and a metric based on the third fundamental form from [Pottmann et al., 2004] are shown in Figure 4.3. Our method is similar to a watershed technique from image segmentation, see e.g., [Mangan and Whitaker, 1999], with a similar use of a priority queue. After the region growing finishes the node points of the feature graph are determined, i.e., points $p_i \in \mathcal{M}_h$ where more than two expanded primitives F_i meet, see Figure 4.2 right.

4.3.4 Smooth feature graph

The rough approximation to the final feature graph from the last step corresponds to the set of boundaries of the expanded primitives F_i, that is, a set of polygonal curves where each curve runs along edges of the underlying triangulation (Figure 4.2 right and Figure 4.4 left). Because a feature graph with smooth edges is necessary to generate a consistent patch layout, we need to smooth these curves (Figure 4.4 right). During the smoothing process, the node points are fixed. If the feature edges are smoothed using

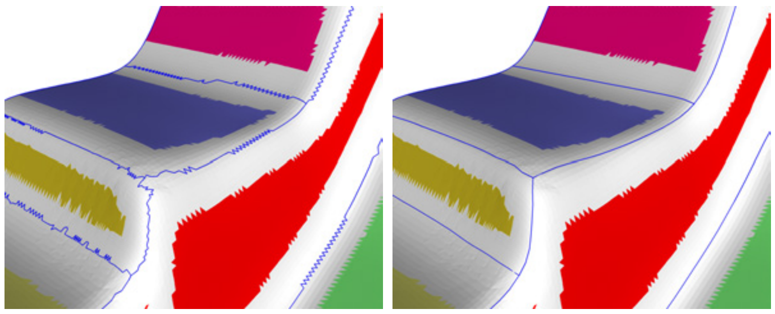
Figure 4.4: Smoothing with fixed end points. The jagged edge based region boundary resulting from expansion (left) and the corresponding smoothed region boundaries describing edges of the feature graph (right).

a standard method, e.g., Laplace smoothing, they cannot be guaranteed to stay in highly curved feature areas of the surface. Instead, we introduce an algorithm that alters a curve on a surface so that the curve is aligned to a given vector field. This approach can be applied to the field of minimal principle curvature directions $X_{min,i}$. In practice, this works well since in highly curved areas, the principal curvature directions are very stable and smooth. So we are looking for a smooth curve connecting the node points which is aligned to the $X_{min,i}$ field in its vicinity. In the smooth setting, an alignment energy for a smooth curve $\gamma : \mathbb{R} \to \Sigma$ can be defined with respect to a given smooth tangential vector field X on Σ by:

$$\begin{aligned} E(\gamma) &= \frac{1}{2} \int_{\gamma} \left(\cos\left(\dot{\gamma}, JX\right)\right)^2 \, ds \\ &= \frac{1}{2} \int_{\gamma} \left(\frac{\langle \dot{\gamma}, JX \rangle}{\|\dot{\gamma}\| \, \|X\|} \right)^2 \, ds, \end{aligned} \tag{4.1}$$

where J denotes the rotation by 90 degrees in the oriented tangent plane. The energy from Equation 4.1 measures how much a curve's tangents deviate from a vector field X. The energy vanishes if the curve tangents are parallel to the vector field X. In the discrete setting a feature curve γ is a polygonal curve, that is, $\gamma = \{q_1, \ldots, q_n\}$ with $q_i \in \mathcal{M}_h$. Points on the edges $\{q_i, q_{i+1}\}$ are not forced to stay on the surface. Then we define a discrete alignment energy by

$$E_D(\gamma) = \frac{1}{2} \sum_{i=1}^{n-1} \|e_i\| \left(\frac{\langle e_i, JX_i \rangle}{\|e\| \, \|X_i\|} \right)^2 , \tag{4.2}$$

where the edge e_i and the edge dependent vector X_i are given by

$$e_i = q_{i+1} - q_i, \qquad \text{and} \qquad X_i = (X(q_i) + X(q_{i+1}))/2.$$

The alignment energy in Equation 4.2 is non-linear in the vertex positions. To compute the descent direction at vertex $q_i \in \gamma$, we ignore the non-linear terms, that is, we approximate the first derivative by

$$\frac{\partial E_D(\gamma)}{\partial q_i} \approx \frac{\langle e_{i-1}, JX_{i-1} \rangle}{\|e_{i-1}\| \, \|X_{i-1}\|^2} JX_{i-1} - \frac{\langle e_i, JX_i \rangle}{\|e_i\| \, \|X_i\|^2} JX_i.$$

To find the minimum, we apply a simple gradient descent strategy in combination with equidistant resampling of the polygonal curve and projecting sample points onto $\mathcal{M}_h$ during the optimization. The points of the minimizing curve are connected by geodesics, that is, we get a polygonal curve lying in the surface mesh. A result of this optimization procedure is shown in Figure 4.4 right.

In regions near the two ends of the edges, that is, in regions of hyperbolic/spherical type, the principle curvature directions become unstable. We perform a matching operation introduced in [Kälberer et al., 2007] to decide how to extent the stable $X_{min,i}$ directions from the fillet regions into these unstable regions.

4.4 Patch layout

Given a feature graph, we are now able to create the final patch layout, i.e., the structure which decomposes the surface into its functional parts such as faces, fillets, and node areas. Our patch layout generation process can be visualized as thickening the feature graph edges back into its faces then cutting off the areas around its nodes. The proposed thickening procedures ensures the alignment of face boundaries to nearby feature lines. Furthermore, we connect feature oriented boundaries in the vicinity of node areas at offset nodes. After having computed a consistent loop of smooth offset curves around each face, we cut out the node areas by node curves. An illustration of the whole process is given in Figure 4.5.

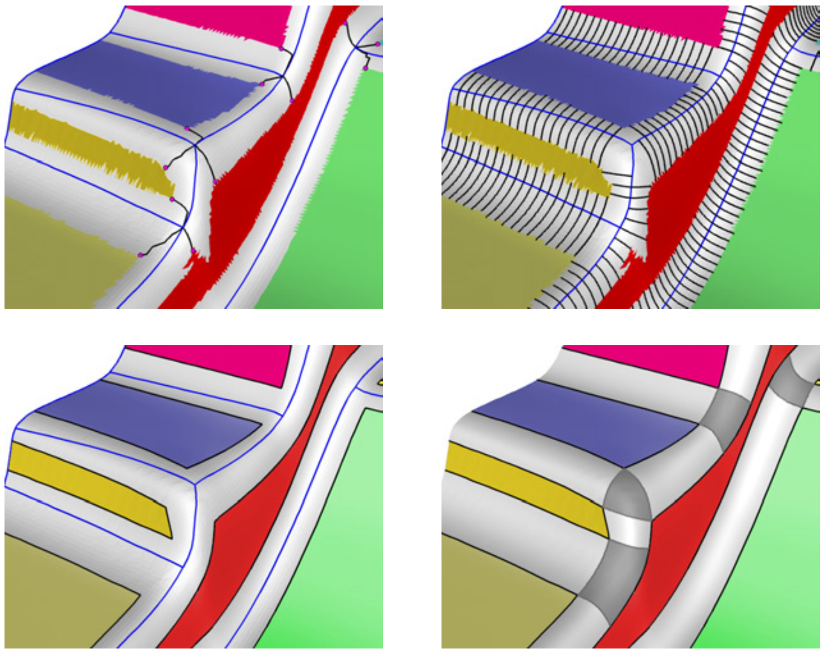

Figure 4.5: Generation of offset curves. Compute offset nodes as nearest points to feature nodes in each adjacent initial patch I_i (top left). Regard distance to initial patch I_i as a one dimensional graph over the feature line (top right). Smooth and aligned offset lines (bottom left). Final offset layout after smoothing the distance function (bottom right).

4.4.1 Offset nodes

Offset nodes are points $p_i \in \mathcal{M}_h$ where offset curves and node curves meet. For each node point, we determine an offset node for each incident face F_j, see Figure 4.5 top left. Our choice for an offset node within a certain face $|F_j|$ is the point that is contained in the corresponding primitive part $|I_j|$ and is closest to the node point of the feature graph. We use Dijkstra distances to determine these points. We refer to an offset node within a face by n_{ij}, where the indices i and j denote the feature graph node and the primitive part, respectively (Figure 4.6 left). In practice, offset nodes of two adjacent feature nodes may fall onto the same geometric position on the surface, i.e., the closest points of different node points coincide. This occurs especially for nearby feature nodes that are connected by a very short feature edge. In this case, the corresponding feature edge will not be offset so that the final offset layout will not be spoiled by these nearby feature nodes, see Figure 4.9 bottom right.

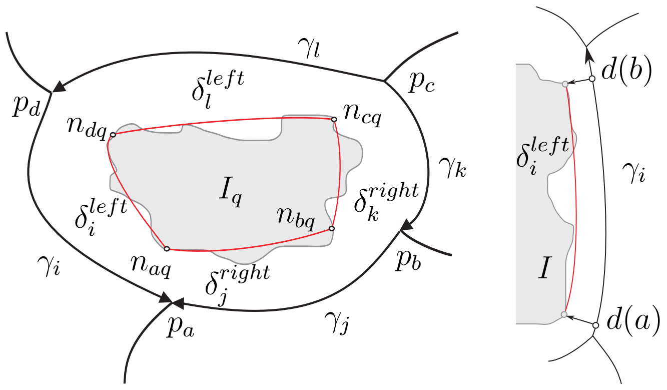

Figure 4.6: Initial patch I_q of one face with adjacent feature curves γ_i and corresponding offset curves δ_i^{dir} and offset nodes n_{ij} on the left. Offsetting a feature curve γ_i, definition of the distance function $d_i^{dir}(t)$ on the right.

4.4.2 Offset curves

Each feature edge γ_i is offset into its two adjacent faces resulting in two offset curves δ_i^{left} and δ_i^{right}, see Figure 4.6 left. The upper index refers to the offset direction as seen from γ_i, whereas the lower index indicates the feature curve to which this offset curve belongs. Each of these curves is created from a scalar function denoted by $d_i^{left}(t)$ and $d_i^{right}(t)$ defined along γ_i. Here the indices of $d_i^j(t)$ are defined in the same manner as for the offset curves. In the remaining sections, we skip the indices, i.e., $d_i^j(t)$ becomes $d(t)$, to shorten the notation.

The distance between a feature edge and the primitive part of the corresponding adjacent face is measured by $d(t)$. This is in general a nonsmooth function, so we apply a convolution to get rid of spikes within the set of distance values. The resulting distance values then encode points on the final offset curve.

The parameterized feature edge $\gamma(t)$ is represented by a set of points uniformly distributed along the feature edge. From each of these points, a geodesic ray is shot perpendicularly to the curve until the corresponding primitive part is hit. To extend a ray geodesically, see [Polthier and Schmies, 1998]. The length of the ray then gives the distance $d(t)$ (Figure 4.5 top right). If the ray does not intersect the edge based polygon connecting the offset nodes, defining the start and end of the offset curve to be generated, the value of $d(t)$ is set to be undefined. The result is a set of distance values $\{d(a), \ldots, d(b)\}$. We smooth the set of distance values $d(t)$ by convolution with a hat function with large support, e.g., half of the length of the feature curve. By construction, the function values $d(a)$ and $d(b)$ are distance samples close to the offset nodes (Figure 4.6 right). By keeping these values fixed during the smoothing process, the resulting smoothed distance values will still resemble the distance near the offset nodes. Thus, we have to convolute $d(t)$ with a hat function and fix the function values at the endpoints. Therefore we extend $d(t)$ to a larger domain in $\mathbb{R}$ by mirroring at the endpoints.

$$\begin{aligned} d(a-x) &:= 2d(a) - d(a+x), \\ d(b+x) &:= 2d(b) - d(b-x), \quad x \in [0, b-a] \end{aligned}$$

Convolution of this function with a hat function will (by symmetry) not change the function values at a and b. These convoluted distances define a sequence of points along the feature curve. The polygonal offset curve lying on the surface is created from this sequence (Figure 4.5, bottom left) and is connected to the corresponding node points.

4.4.3 Node curves

There is one node area for each node point of the feature graph. In most cases, a node area is encircled by a sequence of node lines which start and end in offset nodes. As illustrated in Figure 4.7, let p_l be a feature graph node and γ_i a feature line emanating from p_l. In general, there are two offset curves δ_i^{left} and δ_i^{right}, that arise from offsetting γ_i into the two adjacent faces. So a node curve needs to be created between the two corresponding offset nodes n_{la}, n_{lb} to separate the node area from the fillet area. This is done by

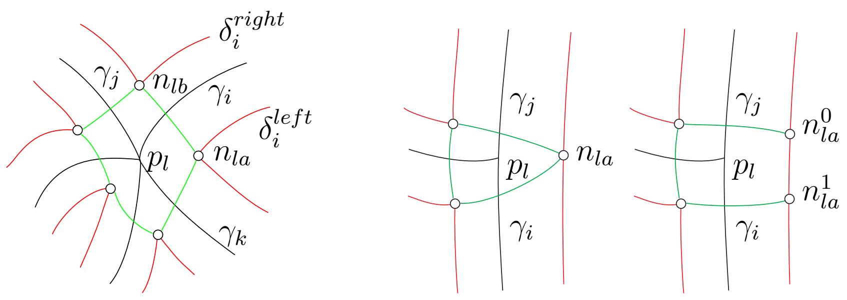

Figure 4.7: A general node area type where the node lines (green) connect the endpoints of offset curves on the left. Two feature edges meet at node point with angle close to 180, then corresponding offset node gets split into two new ones on the right.

constructing a plane out of the two points n_{la}, n_{lb} and their normals. The plane is defined to contain the vector connecting n_{la} and n_{lb} and the average

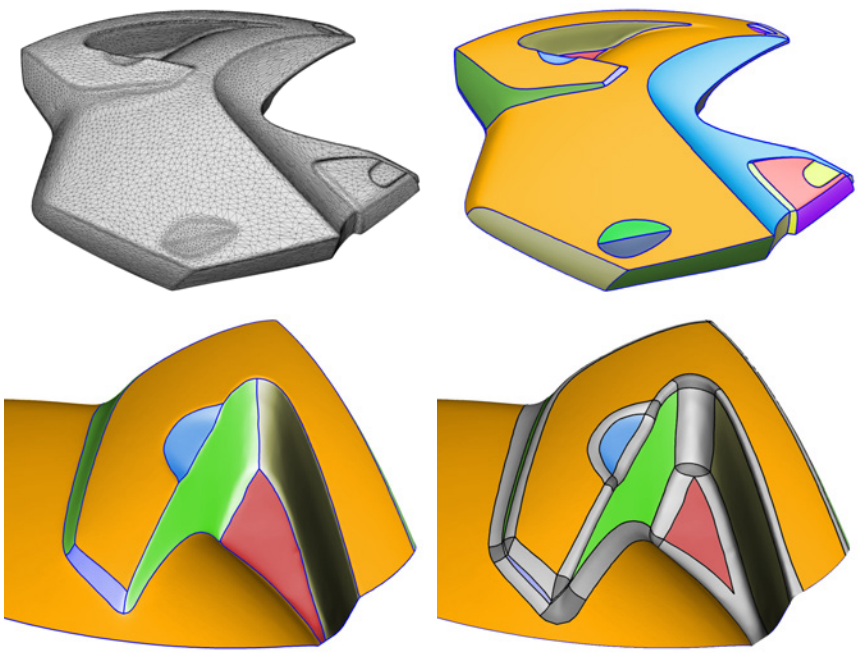

Figure 4.8: Patch layout computed on a motorcycle part. The given triangulated model, top left. The generated feature graph with our method, top right and bottom left, and the final patch layout, bottom right.

of the two normals. The intersection curve of this plane and the mesh will then be our actual node curve. If two feature curves γ_i and γ_j meet at a node point with an angle close to 180 degrees (Figure 4.7 right), the corresponding offset node n_{la} is split into two new ones. The two new nodes n_{la}^0 and n_{la}^1 are found using the distance map. We look for the first point within a valid range of $d(t)$. The actual node curve is constructed as in the usual case. In the actual implementation, we used a threshold of 120 degrees.

4.5 Results

We tested the algorithm on several CAD parts provided by Tebis AG. Here we discuss two parts in detail. The first part belongs to a scan of a motorcycle (Figure 4.8). We can see that the algorithm finds a suitable decomposition of the complex surface. The lower right picture shows the patch layout on the part. The surface contains approximately 100k triangles and the whole patch layout generation process took about 1 minute. The main part was the curve smoothing, which took about 40 seconds. The second part is a scan of a deformed metal plate, see Figure 4.9. Our decomposition algorithm also

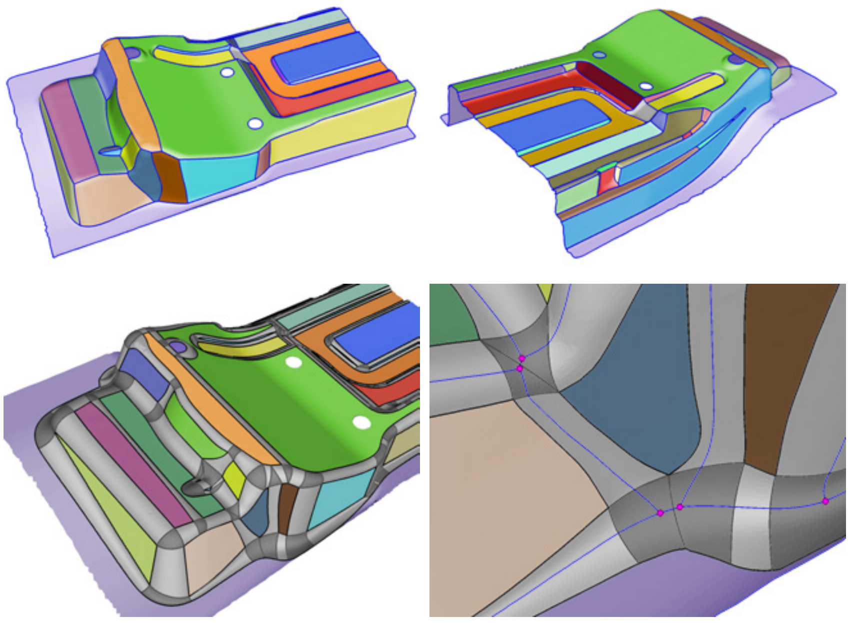

Figure 4.9: Feature graph on a complex CAD model, top left and top right. The final patch layout, bottom left. Offset curves and feature graph in regions with nearby node points, bottom right.

took less than 2 minutes foe this example. We can see that this geometry contains fillets with varying thickness, see Figure 4.9 bottom left, which are well detected by our method. The final patch layout also contains nearby node points showing the ability of our method to work in cases of degenerated offset curves, see Figure 4.9 bottom right.

4.6 Summary

We presented a surface decomposition method consisting of two steps. The first starts with the detection of planar like regions that are expanded by a curvature dependent region growing process. This results in a net of jagged surface curves. We then proposed a method to smooth these curves by a simple gradient descent procedure. In the second step, the actual decomposition is generated. We proposed an offsetting procedure that is applied to smooth curves found in the first step. Finally, we presented results for CAD geometries that consists of more or less smoothly connected planar surface parts.

As a possible extension of the method, we can use more complex types of initial primitives such as cones or cylinders. The method is designed to work with geometries having round features, thus we like to also allow sharp edges, i.e., edges where the corresponding fillet part is not well defined. Furthermore we could consider the inclusion of a feature sensitive point signature, as presented in Chapter 3 to drive the growing process and to control the placement of the node points of the feature graph.

Chapter 5

Outlook

We presented a technique based on the spacetime constraints paradigm that generalizes traditional spline interpolation. Our method can be used to generate and/or enhance an animation at interactive response times even for objects with a large number of degrees of freedom, e.g., deformable objects. For a detailed summary, see Section 2.9. Some related open problems are:

- to include partial keyframes
- the integration of contact and self collision handling
- to extend the approach to other physical systems, like fluids or smoke, and for rigged characters
- to include a varying rest state
- to determine a rest state for a given set of poses, similar to mean shape computation.

We introduced a new family of feature sensitive operators and introduced two possible applications: the stability index estimation for cmc-surfaces and a feature sensitive point signature. For a detailed summary, see Section 3.8. Some related open problems are:

- quadrangulations that align with salient surface features based on feature sensitive eigenfunctions
- surface decomposition based on feature sensitive eigenfunctions
- the estimation of the unconstrained index for nonzero cmc-surfaces
- to further explore the family multi-scale point signatures and distances.

We presented a method that helps to automate the decomposition of a given surface into patches appropriate for low order spline fitting. Just a few parameters, e.g., the curvature threshold, are necessary to drive the generation process, i.e., to control the look of the final layout. For a detailed summary, see Section 4.6. Some related open problems are:

- to extend the set of initial primitives, e.g., cylinders, spheres, cones
- to consider feature signature for feature graph generation, e.g., placement of node points
- to include sharp edges.

Appendix

We describe the mass matrix used in the n-chain examples from Section 2.3.1. In particular we use the notation given in Figure 2.1 for the planar rod and the Luxo model.

5.1 Planar rod

The generalized coordinates and velocities for the points of the planar rod consisting of n points are given by $x = (x_1, y_1, \theta_1, \ldots, \theta_n)$ and $\dot{x} = (\dot{x}_1, \dot{y}_1, \dot{\theta}_1, \ldots, \dot{\theta}_n)$, respectively. We refer to the point masses as m_i with $i \in \{1, \ldots, n\}$. Then the mass matrix used to set up the generalized eigenvalue problem is given by

$$M(x) = D^T(x) N D(x)$$

with

$$D = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & & 0 \\ 0 & 1 & 0 & 0 & 0 & & 0 \\ 1 & 0 & l_1 \cos\theta_1 & l_2 \cos\theta_2 & 0 & & 0 \\ 0 & 1 & -l_1 \sin\theta_1 & -l_2 \sin\theta_2 & 0 & & 0 \\ 1 & 0 & l_1 \cos\theta_1 & l_2 \cos\theta_2 & l_3 \cos\theta_3 & & 0 \\ 0 & 1 & -l_1 \sin\theta_1 & -l_2 \sin\theta_2 & -l_3 \sin\theta_3 & & 0 \\ & & & & & \ddots & \\ 1 & 0 & l_1 \cos\theta_1 & l_2 \cos\theta_2 & l_3 \cos\theta_3 & & l_n \cos\theta_n \\ 0 & 1 & -l_1 \sin\theta_1 & -l_2 \sin\theta_2 & -l_3 \sin\theta_3 & & -l_n \sin\theta_n \end{bmatrix}$$

and

$$N = \begin{bmatrix} m_1 & 0 & 0 & 0 & 0 \\ 0 & m_1 & 0 & 0 & 0 \\ 0 & 0 & \ddots & 0 & 0 \\ 0 & 0 & 0 & m_n & 0 \\ 0 & 0 & 0 & 0 & m_n \end{bmatrix}.$$

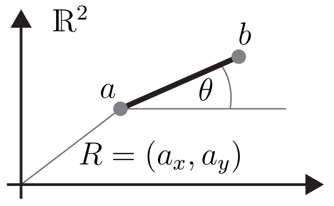

Figure 5.1: Illustration of a two-dimensional rod given by two points $a, b \in \mathbb{R}^2$.

5.2 Luxo

A single planar rigid body is illustrated in Figure 5.1. With the length $l = \|b - a\|$ fixed, we choose as generalized coordinates $x = (R, \theta)$ with $R = (a_x, a_y)$. We assume for the planar body, a linear varying density. If ρ_a denotes the density at a and ρ_b at b, we have

$$\rho(r) = \rho_a + \frac{r}{l}(\rho_a - \rho_b) \quad 0 \le r \le l.$$

The density dependent scalar quantites needed to determine the kinetic energy are the total mass

$$m = \int_0^l \rho(r)\,dr = \frac{l}{2}(\rho_a + \rho_b)$$

and the two density distribution dependent quantities

$$I^1 = \int_0^l r\rho(r)\,dr = \frac{l^2}{3}\left(\frac{\rho_a}{2} + \rho_b\right)$$
$$I^2 = \int_0^l r^2\rho(r)\,dr = \frac{l^3}{4}\left(\frac{1}{3}\rho_a + \rho_b\right).$$

We also need the vector quantity

$$e = I^1 \frac{v}{\|v\|}, \quad \text{with } v = b - a.$$

According to [Josef Honerkamp, 1993], the kinetic energy of a two-dimensional rigid body with respect to the spatial velocity $\dot{R}$ and the angular velocity $\dot{\theta}$

is then given by

$$\begin{aligned}
E_{kin}(\dot{R}, \dot{\theta}) &= E_1(\dot{R}, \dot{R}) + E_2(\dot{R}, \dot{\theta}) + E_3(\dot{\theta}, \dot{\theta}) \\
E_1(\dot{R}, \dot{R}) &= \dot{R}^T \begin{bmatrix} m & 0 \\ 0 & m \end{bmatrix} \dot{R} \\
E_2(\dot{R}, \dot{\theta}) &= \dot{R}^T \begin{bmatrix} -e_y \\ e_x \end{bmatrix} \dot{\theta} \\
E_3(\dot{\theta}, \dot{\theta}) &= \dot{\theta} I^2 \dot{\theta}.
\end{aligned}$$

The Luxo model consits of 6 rigid bodies as shown in Figure 2.1. Hence its kinetic energy corresponds to

$$E_{kin}^{Luxo}(\dot{R}_1, \dot{\theta}_1, \dots, \dot{R}_6, \dot{\theta}_6) = \sum_i^6 E_{kin}(\dot{R}_i, \dot{\theta}_i).$$

We describe the kinetic energy with respect to the generalized coordinates $x = (x_1, y_1, \theta_1, \theta_2, \theta_3, \theta_4)$ introduced in Section 2.3.1. Hence the generalized velocities are given by $\dot{\mathbf{x}} = (\dot{x}_1, \dot{y}_1, \dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3, \dot{\theta}_4)$. We denote by P the projection onto the θ-components, that is,

$$P = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

The mapping from generalized velocities to spatial velocities is given by

$$D(x) = \begin{bmatrix}
1 & 0 & -l_1 \sin\theta_1 & 0 & 0 & 0 \\
0 & 1 & l_1 \cos\theta_1 & 0 & 0 & 0 \\
1 & 0 & -l_1 \sin\theta_1 & -l_2 \sin\theta_2 & 0 & 0 \\
0 & 1 & l_1 \cos\theta_1 & l_2 \cos\theta_2 & 0 & 0 \\
1 & 0 & -l_1 \sin\theta_1 & -l_2 \sin\theta_2 & -l_3 \sin\theta_3 & 0 \\
0 & 1 & l_1 \cos\theta_1 & l_2 \cos\theta_2 & l_3 \cos\theta_3 & 0 \\
1 & 0 & -l_1 \sin\theta_1 & -l_2 \sin\theta_2 & -l_3 \sin\theta_3 & -l_4 \sin\theta_4 \\
0 & 1 & l_1 \cos\theta_1 & l_2 \cos\theta_2 & l_3 \cos\theta_3 & l_4 \cos\theta_4
\end{bmatrix}.$$

Then we can define for the i-th planar rigid body, the scalars m_i, I_i^1, and I_i^2, and the vector e_i with $i \in \{1, 2, 3, 4, 5, 6\}$. Then we can set

$$\begin{aligned}
E_{kin}^{Luxo}(x) &= E_{kin}^{Luxo}(\dot{R}_1, \dot{\theta}_1, \dots, \dot{R}_6, \dot{\theta}_6) \\
&= E_1^{Luxo}(\dot{x}, \dot{x}) + E_2^{Luxo}(\dot{x}, \dot{x}) + E_3^{Luxo}(\dot{x}, \dot{x}).
\end{aligned}$$

The three terms in the kinetic energy are

$$
\begin{aligned}
E_1^{Luxo}(\dot{\mathbf{x}}) &= \frac{1}{2}\dot{\mathbf{x}}^T D^T(x) N D(x)\dot{\mathbf{x}} \\
E_2^{Luxo}(\dot{\mathbf{x}}) &= \dot{\mathbf{x}}^T D(x)\, C(x)\, P\dot{\mathbf{x}} \\
E_3^{Luxo}(\dot{\mathbf{x}}) &= \frac{1}{2}\dot{\mathbf{x}} P^T I P\dot{\mathbf{x}}
\end{aligned}
$$

with

$$
N = \begin{bmatrix}
m_1+m_5+m_6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & m_1+m_5+m_6 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & m_2 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & m_2 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & m_3 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & m_3 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & m_4 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & m_4
\end{bmatrix},
$$

$$
C(x) = \begin{bmatrix}
-e_{1,y}-e_{5,y}-e_{6,y} & 0 & 0 & 0 \\
e_{1,x}+e_{5,x}+e_{6,x} & 0 & 0 & 0 \\
0 & -e_{2,y} & 0 & 0 \\
0 & e_{2,x} & 0 & 0 \\
0 & 0 & -e_{3,y} & 0 \\
0 & 0 & e_{3,x} & 0 \\
0 & 0 & 0 & -e_{4,y} \\
0 & 0 & 0 & e_{4,x}
\end{bmatrix},
$$

and

$$
I = \begin{bmatrix}
I_1^2+I_5^2+I_6^2 & 0 & 0 & 0 \\
0 & I_2^2 & 0 & 0 \\
0 & 0 & I_3^2 & 0 \\
0 & 0 & 0 & I_4^2
\end{bmatrix}.
$$

Then the final mass matrix $M(x)$ used to set up the generalized eigenvalue problem is given by

$$
\begin{aligned}
A(x) &= P^T I P + D(x)C(x)P + D^T(x)ND(x) \\
M(x) &= A^T(x) + A(x).
\end{aligned}
$$

Zusammenfassung

In der mathematischen Geometrieverarbeitung nutzt eine Vielzahl moderner und etablierter Methoden die Eigenschaften von Spektren und Eigenvektoren. So wurden unter anderem Verfahren zur Segmentierung, Parametrisierung und Deformation von Oberflächen entwickelt, die das Lösen generalisierter Eigenwertprobleme erfordern.

In dieser Arbeit werden zwei Anwendungen vorgestellt, die von energiebasierten Spektren und Eigenfunktionen abhängen.

Im Bereich der physikalisch basierten Computeranimation präsentieren wir eine Verallgemeinerung der üblichen Spline Interpolationsmethode von Keyframes. Unsere Verfahren ermöglicht die interaktive Kontrolle über eine Reihe von animationsrelavanten Parametern selbst für komplexe Formen, d.h. unter anderem für diskrete Geometrien die deformierbare Objekte, wie z.B. dünne Schalen oder elastische Körper, beschreiben. Die Interaktivität wird durch eine geeignete Problemreduktion und einer expliziten Darstellung der Wiggly Splines erreicht. Die Reduktion und die Darstellung der Wiggly Splines erfordert das Lösen generalisierter Eigenwertprobleme. Wir demonstrieren die Vielseitigkeit unseres Verfahrens an einer Reihe von Animationen für ein-, zwei- und dreidimensionalen Formen.

Im Bereich der Oberflächenanalyse stellen wir eine neue Familie von Operatoren für Funktionen auf Oberflächen vor. Im Gegensatz zum Laplace-Operator besitzen diese Operatoren merkmals-sensitive Spektren und Eigenfunktionen. Wir konstruieren eine Punkt-Signatur, die auf den merkmalssensitiven Spektren und Eigenfunktionen basiert. Wir vergleichen die gefundenen ähnlichen Punkte bzgl. dieser Signatur mit Ergebnissen der Diffusionssignatur, d.h. einer Signatur die aus dem Laplacespektrum und Eigenfunktionen konstruiert wird. Weiterhin zeigen wir, dass das Spektrum eines bestimmten Operators dieser Familie zur Abschätzung des Stabilitätsindexes einer diskreten Fläche konstanter mittlerer Krümmung verwendet werden kann.

Abschliessend stellen wir noch eine Methode vor, die eine merkmalssensitive Oberflächenstrukturierung ermöglicht.

Verfassererklärung

Gemäß § 7 (4) der Promotionsordnung versichere ich hiermit, diese Arbeit selbständig verfasst zu haben. Ich habe alle bei der Erstellung dieser Arbeit benutzten Hilfsmittel und Hilfen angegeben.

Bibliography

[Amestoy et al., 2001] Amestoy, P. R., Duff, I. S., L'Excellent, J.-Y. and Koster, J. (2001). A Fully Asynchronous Multifrontal Solver Using Distributed Dynamic Scheduling. SIAM Journal on Matrix Analysis and Applications *23*, 15–41.

[Anderson et al., 1999] Anderson, E., Bai, Z., Bischof, C., Blackford, S., Demmel, J., Dongarra, J., Croz, J. D., Greenbaum, A., Hammarling, S., McKenney, A. and Sorensen, D. (1999). LAPACK Users' Guide. SIAM.

[Attene et al., 2006] Attene, M., Falcidieno, B. and Spagnuolo, M. (2006). Hierarchical mesh segmentation based on fitting primitives. The Visual Computer *22*, 181–293.

[Baraff and Witkin, 1998] Baraff, D. and Witkin, A. (1998). Large steps in cloth simulation. In Proceedings of SIGGRAPH.

[Barbič et al., 2009] Barbič, J., da Silva, M. and Popović, J. (2009). Deformable object animation using reduced optimal control. ACM Transactions on Graphics *28*, 53:1–53:9.

[Barbič and James, 2005] Barbič, J. and James, D. L. (2005). Real-Time Subspace Integration for St. Venant-Kirchhoff Deformable Models. ACM Transactions on Graphics *24*, 982–990.

[Barbosa et al., 1988] Barbosa, J. L., do Carmo, M. and Eschenburg, J. (1988). Stability of Hypersurfaces of Constant Mean Curvature in Riemannian Manifolds. Mathematische Zeitschrift *197*, 127 – 138.

[Barzel, 1997] Barzel, R. (1997). Faking Dynamics of Ropes and Springs. IEEE Computer Graphics and Applications *17*, 31–39.

[Bergou et al., 2007] Bergou, M., Mathur, S., Wardetzky, M. and Grinspun, E. (2007). TRACKS: Toward Directable Thin Shells. In Proceedings of SIGGRAPH.

[Capell et al., 2002a] Capell, S., Green, S., Curless, B., Duchamp, T. and Popović, Z. (2002a). A multiresolution framework for dynamic deformations. In Proceedings of SIGGRAPH/Eurographics Symposium on Computer Animation.

[Capell et al., 2002b] Capell, S., Green, S., Curless, B., Duchamp, T. and Popović, Z. (2002b). Interactive skeleton-driven dynamic deformations. ACM Transactions on Graphics *21*, 586–593.

[Cazals et al., 2003] Cazals, F., Chazal, F. and Lewiner, T. (2003). Molecular shape analysis based upon the morse-smale complex and the connolly function. In Proceedings of Symposium on Computational Geometry.

[Chai and Hodgins, 2007] Chai, J. and Hodgins, J. K. (2007). Constraint-based motion optimization using a statistical dynamic model. ACM Transactions on Graphics *26*, 00–00.

[Chao et al., 2010] Chao, I., Pinkall, U., Sanan, P. and Schröder, P. (2010). A simple geometric model for elastic deformations. ACM Transactions on Graphics *29*, 38:1–38:6.

[Ciarlet, 1997] Ciarlet, P. G. (1997). Mathematical Elasticity Volume II: Theory of Plates. North Holland.

[Ciarlet, 2004] Ciarlet, P. G. (2004). Mathematical Elasticity Volume I: Three Dimensional Elasticity. North Holland.

[Ciarlet, 2005] Ciarlet, P. G. (2005). An Introduction to Differential Geometry with Applications to Elasticity. Springer.

[Clements and Zhang, 2006] Clements, A. and Zhang, H. (2006). Robust 3D Shape Correspondence in the Spectral Domain. In Proceedings of Shape Modeling International.

[Cohen, 1992] Cohen, M. F. (1992). Interactive spacetime control for animation. SIGGRAPH Comput. Graph. *26*, 293–302.

[Cohen-Steiner et al., 2004] Cohen-Steiner, D., Alliez, P. and Desbrun, M. (2004). Variational shape approximation. In Proceedings of SIGGRAPH.

[Cohen-Steiner and Morvan, 2003] Cohen-Steiner, D. and Morvan, J.-M. (2003). Restricted delaunay triangulations and normal cycle. In Proceedings of Symposium on Computational Geometry.

[Dey et al., 2010] Dey, T. K., Li, K., Luo, C., Ranjan, P., Safa, I. and Wang, Y. (2010). Persistent heat signature for pose-oblivious matching of incomplete models. Computer Graphics Forum *29*, 1545–1554.

[Dong et al., 2006] Dong, S., Bremer, P.-T., Garland, M., Pascucci, V. and Hart, J. C. (2006). Spectral surface quadrangulation. ACM Transactions on Graphics *25*, 1057–1066.

[Dziuk, 1988] Dziuk, G. (1988). Finite elements for the Beltrami operator on arbitrary surfaces. In Partial Differential Equations and Calculus of Variations pp. 142–155. Springer.

[Edelsbrunner, 2005] Edelsbrunner, H. (2005). Surface tiling with differential topology. In Proceedings of Symposium on Geometry Processing.

[Edelsbrunner et al., 2001] Edelsbrunner, H., Harer, J. and Zomorodian, A. (2001). Hierarchical morse complexes for piecewise linear 2-manifolds. In Proceedings of Symposium on Computational Geometry.

[Fang and Pollard, 2003] Fang, A. C. and Pollard, N. S. (2003). Efficient synthesis of physically valid human motion. ACM Transactions on Graphics *22*, 417–426.

[Funkhouser et al., 2004] Funkhouser, T., Kazhdan, M., Shilane, P., Min, P., Kiefer, W., Tal, A., Rusinkiewicz, S. and Dobkin, D. (2004). Modeling by example. ACM Transactions on Graphics *23*, 652–663.

[Garland et al., 2001] Garland, M., Willmott, A. and Heckbert, P. S. (2001). Hierarchical face clustering on polygonal surfaces. In Proceedings of Symposium on Interactive 3D Graphics.

[Gebal et al., 2009] Gebal, K., Bærentzen, J. A., Aanæs, H. and Larsen, R. (2009). Shape Analysis Using the Auto Diffusion Function. Computer Graphics Forum *28*, 1405–1413.

[Gleicher, 1997] Gleicher, M. (1997). Motion editing with spacetime constraints. In Proceedings of Symposium on Interactive 3D Graphics.

[Griewank et al., 1996] Griewank, A., Juedes, D. and Utke, J. (1996). Algorithm 755: ADOL-C: a package for the automatic differentiation of algorithms written in C/C++. ACM Trans. Math. Softw. *22*, 131–167.

[Grinspun et al., 2003] Grinspun, E., Hirani, A. N., Desbrun, M. and Schröder, P. (2003). Discrete shells. In Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation.

[Guenter, 2007] Guenter, B. (2007). Efficient symbolic differentiation for graphics applications. ACM Transactions on Graphics *26*, 00–00.

[Hildebrandt, 2013] Hildebrandt, K. (2013). Discretization and approximation of the shape operator, the Laplace-Beltrami operator, and the Willmore energy of surfaces. PhD thesis, Freie Universität Berlin.

[Hildebrandt and Polthier, 2004] Hildebrandt, K. and Polthier, K. (2004). Anisotropic Filtering of Non-Linear Surface Features. Computer Graphics Forum *23*, 391–400.

[Hildebrandt et al., 2006] Hildebrandt, K., Polthier, K. and Wardetzky, M. (2006). On the convergence of metric and geometric properties of polyhedral surfaces. Geometricae Dedicata *123*, 89–112.

[Hildebrandt et al., 2011] Hildebrandt, K., Schulz, C., von Tycowicz, C. and Polthier, K. (2011). Interactive surface modeling using modal analysis. ACM Transactions on Graphics *30*, 119:1–119:11.

[Hildebrandt et al., 2012] Hildebrandt, K., Schulz, C., von Tycowicz, C. and Polthier, K. (2012). Modal shape analysis beyond Laplacian. Computer Aided Geometric Design *29*, 204–218.

[Hofer and Pottmann, 2004] Hofer, M. and Pottmann, H. (2004). Energy-minimizing splines in manifolds. In Proceedings of ACM SIGGRAPH.

[Huang et al., 2006] Huang, J., Shi, X., Liu, X., Zhou, K., Wei, L.-Y., Teng, S.-H., Bao, H., Guo, B. and Shum, H.-Y. (2006). Subspace gradient domain mesh deformation. ACM Transactions on Graphics *25*, 00–00.

[Huang et al., 2008] Huang, J., Zhang, M., Ma, J., Liu, X., Kobbelt, L. and Bao, H. (2008). Spectral quadrangulation with orientation and alignment control. ACM Transactions on Graphics *27*, 1–9.

[Huang et al., 2009] Huang, Q., Wicke, M., Adams, B. and Guibas, L. (2009). Shape Decomposition Using Modal Analysis. Computer Graphics Forum *28*, 407–416.

[Idelsohn and Cardona, 1985] Idelsohn, S. R. and Cardona, A. (1985). A reduction method for nonlinear structural dynamic analysis. Computer Methods in Applied Mechanics and Engineering *49*, 253 – 279.

[Joris S. M. Vergeest and Jelier, 2001] Joris S. M. Vergeest, Sander Spanjaard, I. H. and Jelier, J. J. O. (2001). Fitting Freeform Shape Patterns to Scanned 3D Objects. Journal of Computing and Information Science in Engineering *1*, 218–224.

[Josef Honerkamp, 1993] Josef Honerkamp, H. R. (1993). Klassische Mechanik. Springer.

[Jost, 2008] Jost, J. (2008). Riemannian geometry and geometric analysis. Springer Verlag.

[Julius et al., 2005] Julius, D., Kraevoy, V. and Sheffer, A. (2005). D-Charts: Quasi-Developable Mesh Segmentation. In Proceedings of Eurographics.

[Kälberer et al., 2007] Kälberer, F., Nieser, M. and Polthier, K. (2007). QuadCover - Surface Parameterization using Branched Coverings. Comput. Graph. Forum *26*, 375–384.

[Kass and Anderson, 2008] Kass, M. and Anderson, J. (2008). Animating oscillatory motion with overlap: wiggly splines. ACM Transactions on Graphics *27*, 28:1–28:8.

[Kilian et al., 2007] Kilian, M., Mitra, N. J. and Pottmann, H. (2007). Geometric modeling in shape space. ACM Transactions on Graphics *26*, 00–00.

[Kim and James, 2009] Kim, T. and James, D. L. (2009). Skipping steps in deformable simulation with online model reduction. ACM Transactions on Graphics *28*, 123:1–123:9.

[Kimmel and Sethian, 1998] Kimmel, R. and Sethian, J. A. (1998). Computing Geodesic Paths on Manifolds. In National Academy of Sciences USA pp. 8431–8435,.

[Krysl et al., 2001] Krysl, P., Lall, S. and Marsden, J. E. (2001). Dimensional Model Reduction in Non-linear Finite Element Dynamics of Solids and Structures. Int. J. Numer. Meth. Eng. *51*, 479–504.

[Lee and Lee, 2002] Lee, Y. and Lee, S. (2002). Geometric Snakes for Triangular Meshes. Computer Graphics Forum *21*, 229–238.

[Lehoucq et al., 1998] Lehoucq, R. B., Sorensen, D. C. and Yang, C. (1998). ARPACK users' guide - solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods. SIAM.

[Lévy et al., 2002] Lévy, B., Petitjean, S., Ray, N. and Maillot, J. (2002). Least squares conformal maps for automatic texture atlas generation. In Proceedings of SIGGRAPH Computer graphics and interactive techniques.

[Lévy and Zhang, 2009] Lévy, B. and Zhang, H. (2009). Spectral mesh processing. In Proceedings of ACM SIGGRAPH ASIA.

[Mangan and Whitaker, 1999] Mangan, A. P. and Whitaker, R. T. (1999). Partitioning 3D Surface Meshes Using Watershed Segmentation. IEEE Transactions on Visualization and Computer Graphics *5*, 308–321.

[McNamara et al., 2004] McNamara, A., Treuille, A., Popović, Z. and Stam, J. (2004). Fluid control using the adjoint method. ACM Transactions on Graphics *23*, 449–456.

[Munkres, 1984] Munkres, J. R. (1984). Elements of Algebraic Topology. Perseus Publishing Cambridge, Massachusetts.

[Nickell, 1976] Nickell, R. (1976). Nonlinear dynamics by mode superposition. Computer Methods in Applied Mechanics and Engineering *7*, 107–129.

[Ovsjanikov et al., 2010] Ovsjanikov, M., Mérigot, Q., Mémoli, F. and Guibas, L. (2010). One point isometric matching with the heat kernel. Computer Graphics Forum *29*, 1555–1564.

[Ovsjanikov et al., 2008] Ovsjanikov, M., Sun, J. and Guibas, L. (2008). Global Intrinsic Symmetries of Shapes. Computer Graphics Forum *27*, 1341–1348.

[Pentland and Williams, 1989] Pentland, A. and Williams, J. (1989). Good vibrations: modal dynamics for graphics and animation. SIGGRAPH Comput. Graph. *23*, 207–214.

[Pinkall and Polthier, 1993] Pinkall, U. and Polthier, K. (1993). Computing Discrete Minimal Surfaces and Their Conjugates. Experimental Mathematics *2*, 15–36.

[Polthier, 2002] Polthier, K. (2002). Polyhedral Surfaces of Constant Mean Curvature. Habilitation, Technische Universität Berlin.

[Polthier and Rossmann, 2002] Polthier, K. and Rossmann, W. (2002). Discrete constant mean curvature surfaces and their index. Journal für reine und angewandte Mathematik *549*, 47–77.

[Polthier and Schmies, 1998] Polthier, K. and Schmies, M. (1998). Straightest Geodesics on Polyhedral Surfaces. In Mathematical Visualization pp. 135–150. Springer Verlag.

[Popović et al., 2003] Popović, J., Seitz, S. M. and Erdmann, M. (2003). Motion sketching for control of rigid-body simulations. ACM Transactions on Graphics *22*, 1034–1054.

[Pottmann et al., 2007] Pottmann, H., andYong Liang Yang, J. W., Lai, Y.-K. and Hu, S.-M. (2007). Principal curvatures from the integral invariant viewpoint. Comput. Aided Geom. Design *24*, 428–442.

[Pottmann et al., 2004] Pottmann, H., Steiner, T., Hofer, M., Haider, C. and Hanbury, A. (2004). The isophotic metric and its application to feature sensitive morphology on surfaces. In Computer Vision ECCV 2004 pp. 560–572,.

[Reuter et al., 2005] Reuter, M., Wolter, F.-E. and Peinecke, N. (2005). Laplace-Spectra as Fingerprints for Shape Matching. In Proceedings of the ACM Symposium on Solid and Physical Modeling.

[Reuter et al., 2006] Reuter, M., Wolter, F.-E. and Peinecke, N. (2006). Laplace-Beltrami spectra as "Shape-DNA" of surfaces and solids. Computer-Aided Design *38*, 342–366.

[Ritchie et al., 2005] Ritchie, K., Callery, J. and Biri, K. (2005). The Art of Rigging. CG Toolkit.

[Rustamov, 2007] Rustamov, R. M. (2007). Laplace-Beltrami eigenfunctions for deformation invariant shape representation. In Proceedings of Eurographics/ACM SIGGRAPH Symposium on Geometry Processing.

[Saad, 1992] Saad, Y. (1992). Numerical Methods for Large Eigenvalue Problems. Manchester University Press.

[Safonova et al., 2004] Safonova, A., Hodgins, J. K. and Pollard, N. S. (2004). Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces. ACM Transactions on Graphics *23*, 514–521.

[Sapidis and Besl, 1995] Sapidis, N. S. and Besl, P. J. (1995). Direct construction of polynomial surfaces from dense range images through region growing. ACM Transactions on Graphics *14*, 171–200.

[Scheck, 2007] Scheck, F. (2007). Theoretische Physik 1. Springer.

[Shabana, 1997] Shabana, A. A. (1997). Theory of Vibration II: Vibration of Discrete and Continuous Systems. Springer Verlag.

[Shamir, 2006] Shamir, A. (2006). Segmentation and Shape Extraction of 3D Boundary Meshes. In Proceedings of EUROGRAPHICS.

[Shatz et al., 2006] Shatz, I., Tal, A. and Leifman, G. (2006). Paper craft models from meshes. Vis. Comput. *22*, 825–834.

[Sorkine and Alexa, 2007] Sorkine, O. and Alexa, M. (2007). As-Rigid-As-Possible Surface Modeling. In Proceedings of Eurographics/ACM SIGGRAPH Symposium on Geometry Processing.

[Sulejmanpašić and Popović, 2005] Sulejmanpašić, A. and Popović, J. (2005). Adaptation of performed ballistic motion. ACM Transactions on Graphics *24*, 165–179.

[Sun et al., 2009] Sun, J., Ovsjanikov, M. and Guibas, L. J. (2009). A Concise and Provably Informative Multi-Scale Signature Based on Heat Diffusion. Computer Graphics Forum *28*, 1383–1392.

[Terzopoulos et al., 1987] Terzopoulos, D., Platt, J., Barr, A. and Fleischer, K. (1987). Elastically deformable models. In Proceedings of SIGGRAPH.

[Treuille et al., 2006] Treuille, A., Lewis, A. and Popović, Z. (2006). Model reduction for real-time fluids. ACM Transactions on Graphics *25*, 826–834.

[Treuille et al., 2003] Treuille, A., McNamara, A., Popović, Z. and Stam, J. (2003). Keyframe control of smoke simulations. ACM Transactions on Graphics *22*, 716–723.

[Vallet and Lévy, 2008] Vallet, B. and Lévy, B. (2008). Spectral Geometry Processing with Manifold Harmonics. In Proceedings of Eurographics.

[Várady et al., 2007] Várady, T., Facello, M. A. and Terék, Z. (2007). Automatic extraction of surface structures in digital shape reconstruction. Comput. Aided Des. *39*, 379–388.

[Wardetzky et al., 2007] Wardetzky, M., Bergou, M., Harmon, D., Zorin, D. and Grinspun, E. (2007). Discrete quadratic curvature energies. Computer Aided Geometric Design *24*, 499–518.

[Wicke et al., 2009] Wicke, M., Stanton, M. and Treuille, A. (2009). Modular bases for fluid dynamics. ACM Transactions on Graphics *28*, 39:1–39:8.

[Wirth et al., 2009] Wirth, B., Bar, L., Rumpf, M. and Sapiro, G. (2009). Geodesics in Shape Space via Variational Time Discretization. In Proceedings of International Conference on Energy Minimization Methods in Computer Vision and Pattern Recognition.

[Witkin and Kass, 1988] Witkin, A. and Kass, M. (1988). Spacetime constraints. SIGGRAPH Comput. Graph. *22*, 159–168.

[Wojtan et al., 2006] Wojtan, C., Mucha, P. J. and Turk, G. (2006). Keyframe control of complex particle systems using the adjoint method. In Proceedings of SIGGRAPH/Eurographics Symposium on Computer Animation.

[Wu and Kobbelt, 2005] Wu, J. and Kobbelt, L. (2005). Structure Recovery via Hybrid Variational Surface Approximation. Computer Graphics Forum *24*, 277–284.