Discrete Differential Geometry: An Applied Introduction

SIGGRAPH ASIA 2008 COURSE NOTES

ORGANIZERS
Eitan Grinspun
Max Wardetzky

LECTURERS
Mathieu Desbrun
Peter Schröder
Max Wardetzky

Preface

The behavior of physical systems is typically described by a set of continuous equations using tools such as geometric mechanics and differential geometry to analyze and capture their properties. For purposes of computation one must derive discrete (in space and time) representations of the underlying equations. Researchers in a variety of areas have discovered that theories, which are discrete from the start, and have key geometric properties built into their discrete description can often more readily yield robust numerical simulations which are true to the underlying continuous systems: they exactly preserve invariants of the continuous systems in the discrete computational realm. Such theories make up the nascent field of *discrete differential geometry*.

This volume documents the full day course *Discrete Differential Geometry: An Applied Introduction* at SIGGRAPH Asia 2008 in Singapore on 12 December 2008. These notes supplement the lectures given by Mathieu Desbrun, Peter Schröder, and Max Wardetzky. These notes include contributions by Miklos Bergou, Mathieu Desbrun, Sharif Elcott, Akash Garg, Eitan Grinspun, David Harmon, Eva Kanso, Felix Kälberer, Saurabh Mathur, Ulrich Pinkall, Peter Schröder, Adrian Secord, Boris Springborn, Ari Stern, John M. Sullivan, Yiying Tong, Max Wardetzky, and Denis Zorin, and build on the ideas of many others.

Changes in the 3rd Edition

In this third offering of the course we have introduced and revised several chapters. A considerable effort was invested to ensure that chapters 1-3, 4-5, and 7-8 provide a structured, didactic introduction to discrete curvature, plate/shell simulation, and differential forms, respectively.

Our topic is advancing at a rapid pace, and these notes would not present a complete picture if we did not include some of the most recent advances. We have therefore decided to include four exact reprints of late-breaking relevant work. Rather than placing these as appendices, we have chosen to situate them by their natural relation to the progression of the notes.

A chapter-by-chapter synopsis

The course notes are organized similarly to the lectures. We introduce discrete differential geometry in the context of discrete curves and curvature (Chapter 1). The overarching themes introduced here, *convergence* and *structure preservation*, make repeated appearances throughout the entire volume. We ask the question of which quantities one should measure on a discrete object such as a triangle mesh, and how one should define such measurements (Chapter 2). This exploration yields a host of measurements such as length, area, mean curvature, etc., and these in turn form the basis for various applications described later on. We conclude the introduction with a summary of curvature measures for discrete surfaces (Chapter 3).

The discussion of immersed surfaces paves the way to the development of a physical model for thin, flexible surfaces (Chapter 4). The case of nearly inextensible surfaces is commonplace and allows for particularly fast cloth simulation and Willmore flow (Chapter 5). The mathematical modeling and simulation of elastic rods (flexible curves with twist) requires not only a discrete curvature but also discrete framed curves and parallel transport (Chapter 6).

At this point we shift down to explore the low-level approach of discrete exterior calculus: after an overview of the field (Chapter 7), we lay out the simple tools for implementing DEC (Chapter 8). With this in place, numerically robust and efficient simulations of the Navier-Stokes equations of fluids become possible (Chapter 9).

DEC can be used to formulate various differential operators, including the so-called cotangent Laplacian. While this discrete operator has long been used for parameterization based on harmonic functions, it reappears in an altogether different manner when one considers discrete conformal maps and the conformal equivalence of triangle meshes (Chapter 10).

The cotangent Laplacian preserves some (but not all) core structures of the smooth Laplace-Beltrami operator. Recent results suggest the impossibility of simultaneously preserving all the important smooth structures (Chapter 11).

Simulations of thin-shells, cloth, and fluids, and geometric modeling problems such as fairing and parameterization, require robust numerical time integration. We conclude with an overview of discrete geometric mechanics and variational time integrators (Chapter 12).

Eitan Grinspun, Mathieu Desbrun, Peter Schröder and Max Wardetzky
14 Sept 2008

Contents

Chapter 1: Introduction to Discrete Differential Geometry: The Geometry of Plane Curves

Eitan Grinspun
Columbia University

Adrian Secord
New York University

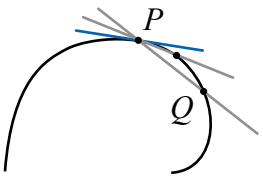
1 Introduction

The nascent field of discrete differential geometry deals with discrete geometric objects (such as polygons) which act as analogues to continuous geometric objects (such as curves). The discrete objects can be measured (length, area) and can interact with other discrete objects (collision/response). From a computational standpoint, the discrete objects are attractive, because they have been designed from the ground up with data-structures and algorithms in mind. From a mathematical standpoint, they present a great challenge: the discrete objects should have properties which are analogues of the properties of continuous objects. One important property of curves and surfaces is their curvature, which plays a significant role in many application areas (see, *e.g.*, Chapters 4 and 5). In the continuous domain there are remarkable theorems dealing with curvature; a key requirement for a discrete curve with discrete curvature is that it satisfies analogous theorems. In this chapter we examine the curvature of continuous and discrete curves on the plane.

The notes in this chapter draw from a lecture given by John Sullivan in May 2004 at Oberwolfach, and from the writings of David Hilbert in his book *Geometry and the Imagination*.

2 Geometry of the Plane Curve

Consider a plane curve, in particular a small piece of curve which does not cross itself (a *simple* curve). Choose two points, P and Q, on this curve and connect them with a straight line: a *secant*. Fixing P as the "hinge," rotate the secant about P so that Q slides along the curve toward Q. If the curve is sufficiently smooth ("*tangent-continuous* at P") then the secant approaches a definite line: the *tangent*. Of all the straight lines passing through P, the tangent is the best approximation to the curve. Consequently we define the *direction* of the curve at P to be the direction of the tangent, so that if two curves intersect at a point P their angle of intersection is given by the angle formed by their tangents at P. If both curves have identical tangents at P then we say "the curves are tangent at P." Returning to our single curve, the line perpendicular to the tangent and passing through P is called the *normal* to the curve at P. Together the tangent and normal form the axes of a local rectangular coordinate system. In addition, the tangent can be thought of as a local approximation to the curve at P.



A better approximation than the tangent is the *circle of curvature*: consider a circle through P and two neighboring points on the curve, and slide the neighboring points towards P. If the curve is sufficiently smooth ("*curvature-continuous* at P") then the circle thus approaches a definite position known as the circle of curvature or *osculating circle*; the center and radius of the osculating circle are the *center of curvature* and *radius of curvature* associated to point P on the curve. The inverse of the radius is κ, the *curvature* of the curve at P.

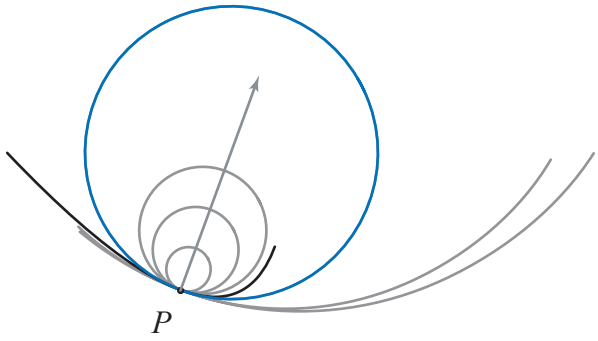


Figure 1: The family of tangent circles to the curve at point P. The circle of curvature is the only one crossing the curve at P.

If we also consider a sense of traversal along the curve segment (think of adding an arrowhead at one end of the segment) then we may measure the *signed* curvature, identical in magnitude to the curvature, but negative in sign whenever the curve is turning clockwise (think of riding a bicycle along the curve: when we turn to the right, it is because the center of curvature lies to the right, and the curvature is negative).

Another way to define the circle of curvature is by considering the infinite family of circles which are tangent to the curve at P (see Figure 1). Every point on the normal to the curve at P serves as the center for one circle in this family. In a small neighborhood around P the curve divides the plane into two sides. Every circle (but one!) in our family lies entirely in one side or the other. Only the circle of curvature however spans both sides, crossing the curve at P. It divides the family of tangent circles into two sets: those with radius smaller than the radius of curvature lying on one side, and those with greater radius lying on the other side. There may exist special points on the curve at which the circle of curvature does not locally cross the curve, and in general these are finite and isolated points where the curve has a (local) axis of symmetry (there are four such points on an ellipse). However on a circle, or a circular arc, the special points are infinitely many and not isolated.

That the circle of curvature crosses the curve may be reasoned by various arguments. As we traverse the curve past

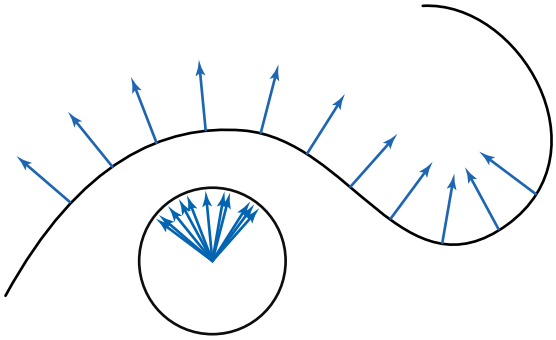


Figure 2: The Gauss map assigns to every point on the curve a corresponding point on the unit circle.

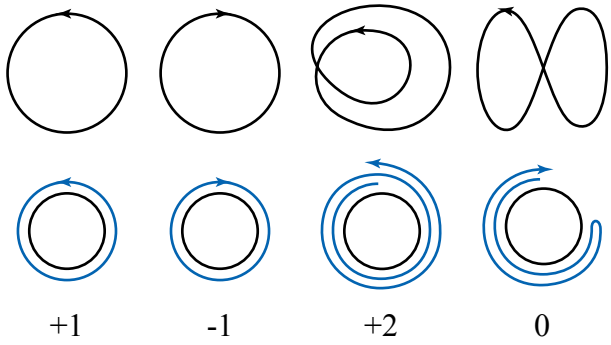


Figure 3: Turning numbers of various closed curves. *Top row:* Two simple curves with opposite sense of traversal, and two self-intersecting curves, one of which "undoes" the turn. *Bottom row:* Gaussian image of the curves, and the associated turning numbers.

point P, the curvature is typically either increasing or decreasing, so that in the local neighborhood of P, so that the osculating circle in comparison to the curve will have a higher curvature on one side and lower on the other. An alternative argument considers our three point construction. Trace along a circle passing through three consecutive points on the curve to observe that the circle must pass from side A to side B on the first point, B to A on the second, and A to B on the third. Similar reasoning of our two-point construction shows that in general the tangent does not cross the curve—the isolated exceptions are the *points of inflection*, where the radius of curvature is infinite and the circle of curvature is identical to the tangent.

Informally we say that P, the tangent at P, and the osculating circle at P have one, two, and three coincident points in common with the curve, respectively. Each construction in sequence considers an additional approaching point in the neighborhood of P and the so-called *order of approximation* (0, 1, and 2 respectively) is identical to the number of additional points.

In 1825 Karl F. Gauss introduced a new tool for thinking about the shape of curves and surfaces. Begin by fixing a sense of traversal for the curve, naturally inducing for every point on the curve a direction for the tangent. By convention, the normal points a quarter turn counterclockwise from tangent direction. Gauss's idea is to draw a unit circle on the plane of the curve, and for any point on the curve, to represent the normal by the radius of the circle parallel to the normal and having the same sense as the normal. To any point P on the curve, the *Gauss map* assigns a point Q on the unit circle, namely the point where the *radius* meets the circle (here, radius means the line segment from the center of the circle to a point on the circumference). Observe that the normal at P is parallel to the radius of the circle, and the tangent to the curve at P is parallel to the tangent to the circle at Q. That the tangent at P and Q are parallel is used to simplify important definitions in differential geometry (see, *e.g.*, the definition of the shape operator in the chapter on discrete shells). While the Gauss map assigns exactly one point on the unit circle to any point on the curve, there may be multiple points on the curve that map to the same point on the circle, *i.e.*the map is not one-to-one.

Consider the image of the curve under the Gauss map: the *Gaussian image* of a curve is the union of all points on the unit circle corresponding to all points on the given curve. For an open curve, the Gaussian image may be an arc or may be the unit circle. Consider a closed simple plane curve: the image is always the unit circle. If we allow the closed curve to intersect itself, we can count how many times the image completely "wraps around" the unit circle (and in which sense): this is the *turning number* or the *index of rotation*, denoted k. It is unity for a simple closed curve traversed counterclockwise. It is zero or ± 2 for curve that self-intersects once, depending on the sense of traversal and on whether or not the winding is "undone."

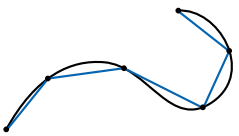
Turning Number Theorem. An old and well-known fact about curves is that the integral of signed curvature over a closed curve, Ω, is dependent *only* on the turning number:

$$\int_\Omega \kappa \, ds = 2\pi k \ .$$

No matter how much we wiggle and bend the curve, if we do not change its turning number we do not change its *total signed curvature*[1]. To change the total signed curvature of Ω we are forced to alter its turning number by adjusting the curve to introduce (or rearrange) self-intersecting loops. This theorem about the significance of the turning number is a piece of mathematical *structure*: together all the structure we discover embodies our understanding of differential geometry. Consequently, our computational algorithms will take advantage of this structure. In computing with discrete approximations of continuous geometry, we will strive to keep key pieces of structure intact.

3 Geometry of the Discrete Plane Curve

Given a curve, r, approximate it drawing an *inscribed polygon* p: a finite sequence of (point) *vertices*, $V_1, V_2, \ldots V_n$, ordered by a traversal of the curve, and line segments connecting successive vertices[2].



[1] Beware that in the context of space curves, the phrase "total curvature" is occasionally used to denote the Pythagorean sum of torsion and curvature—a pointwise quantity like curvature. In contrast, here we mean the integral of curvature over the curve.

[2] While we concern ourselves here only with plane curves, this treatment may be extended to curves in a higher-dimensional am-

The length of the inscribed polygon is given by

$$len(p) = \sum_{i=0}^{n} d(V_i, V_{i+1}) \ ,$$

where $d(\cdot,\cdot)$ measures the euclidean distance[3] between two points. We find the length of the continuous curve by taking the supremum over all possible inscriptions:

$$len(r) = \sup_{p \text{ inscribed in } r} len(p) \ .$$

Next, choose a sense of traversal along the curve, naturally inducing a sense for the inscribed polygon. The (discrete) *total signed curvature* of the inscribed polygon is given by

$$tsc(p) = \sum_{i=0}^{n} \alpha_i \ ,$$

where α_i is the signed *turning angle* at vertex V_i, measured in the sense that a clockwise turn has negative sign; if p is open then $\alpha_0 = \alpha_n = 0$. (N.B.: the turning *angle* is a local quantity at each vertex, whereas the turning *number* is a global quantity of a curve—these are two distinct concepts). Again, we may express the total signed curvature of the continuous curve by taking the supremum over all possible inscribed polygons:

$$tsc(r) = \sup_{p \text{ inscribed in } r} tsc(p) \ .$$

A definition based on suprema serves as an elegant foundation for defining the (integral quantities) length and total curvature of a smooth curve using only very simple polygonal geometry; however suprema are typically is not well suited for computation. For an equivalent, computationally meaningful definition, we construct an infinite sequence of inscribed polygons, p_1, p_2, p_3, ..., that approaches the position of r; analogous definitions of $len(r)$ and $tsc(r)$ are formulated as limits of measurements over elements of the sequence.

To clarify what we mean by "the inscribed polygon p approaches the position of r," define the *geometric mesh size* of p by the length of its longest line segment:

$$h(p) = \max_{0 \le i < n} d(V_i, V_{i+1}) \ .$$

Suppose that r is a smooth simple curve. By smooth we mean that every point on the curve has a unique well-defined tangent[4]. Then one can show that given a sequence p_1, p_2, p_3, ... such that $h(p_i)$ vanishes in the limit of the sequence, then $len(p_i)$ approaches $len(r)$. An analogous statement holds for total curvature, as summarized by the following statement:[5]

bient space, $M^m \subseteq \mathbb{R}^d$, by replacing line segments with shortest geodesics in this definition, and straight-line distance by length of geodesic in subsequent definitions.

[3]It measures distance using the metric of the ambient space, in our case $\mathbb{R}^2$.

[4]Observe that smoothness here is in a purely geometric sense—the notion of parametric smoothness in the context of parameterized curves is a different matter altogether.

[5]Note that there are sequences of pathological polygons whose mesh size vanishes yet the limit of the sequence does not approach the curve. For example, if the curve is a circle, consider a polygon whose vertices all cluster about a single point of the circle.

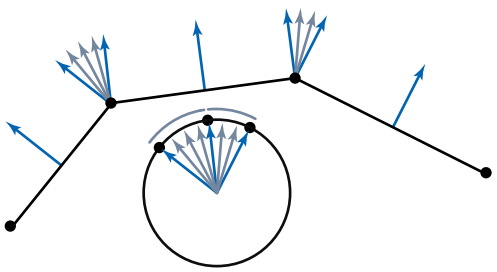


Figure 4: The discrete Gauss map assigns to every edge of the polygon a corresponding point on the unit circle, and to every vertex of the polygon a corresponding arc on the unit circle.

Convergence. A key recurring theme in discrete differential geometry is the *convergence* of a measurement taken over a sequence of discrete objects each better approximating a particular smooth object. In the case of a plane curve, a sequence of inscribed polygons, each closer in position to the curve, generates a sequence of measurements that approach that of the curve:

$$len(r) = \lim_{h(p_i) \to 0} len(p_i) \ ,$$

$$tsc(r) = \lim_{h(p_i) \to 0} tsc(p_i) \ .$$

Establishing convergence is a key step towards numerical computations which use discrete objects as approximations to continuous counterparts. Indeed, one might argue that the notion of continuous *counterpart* is only meaningful in the context of established convergence. Put simply, if we choose an inscribed polygon as our discrete analogue of a curve, then as the position of the approximating polygon approaches the curve, the measurements taken on the approximant should approach those of the underlying curve.

Next, consider the tangents, normals, and Gaussian image of a closed polygon p. Repeating the two-point limiting process we used to define the tangent for a point on the curve, we observe that every vertex of the polygon has two limiting tangents (thus two normals), depending on the direction from which the limit is taken (see Figure 4). Define the Gaussian image of p by assigning to every vertex V_i the arc on the unit circle whose endpoints are the two limiting normals and whose signed angle equals the signed turning angle α_i, *i.e.*, as if one "smoothly interpolated" the two normals in the Gaussian image. Every point on the polygon away from the vertices has a unique normal which corresponds in the Gaussian image to the meeting point of consecutive arcs. The sense of traversal along the polygon induces a natural sense of traversal along the arcs of the Gaussian image. With this construction in place, our definition of turning number for a smooth plane curve carries over naturally to the setting of closed polygons. Not that for for open polygons, the Gaussian image of vertices at the endpoints is a point on the unit circle (a degenerate arc).

As long as the length of the longest line segment shrinks, *i.e.* the polygon clusters more tightly around the point, then this sequence of polygons will satisfy our definition but will clearly not converge to a circle. One may introduce stronger requirements on the polygon sequence to exclude such pathological sequences.

Structure preservation. Does the Turning Number Theorem hold for discrete curves? Yes. Recall that the sum of exterior angles of a simple closed polygon is 2π. This observation may be generalized to show that $tsc(p) = 2\pi k$ where k is the turning number of the polygon. We stress a key point: the Turning Number Theorem is *not* a claim that the total signed curvature converges to a multiple of 2π in the limit of a finely refined inscribed polygon. The Turning Number Theorem is preserved *exactly* and it holds for *any* (arbitrarily coarse) closed polygon. Note, however, that the turning number of an inscribed polygon may not match that of the smooth curve, at least until sufficiently many vertices are added (in the right places) to capture the topology of the curve.

4 Parameterization of the Plane Curve

So far in our exploration of curves our arguments have never explicitly made reference to a system of coordinates. This was to stress the point that the geometry (or *shape*) of the curve can be described without reference to coordinates. Nevertheless, the idea of *parameterizing* a curve occurs throughout applied mathematics. Unfortunately, parameterization can sometimes obscure geometric insight. At the same time, it is an exceedingly useful computational tool, and as such we complete our exploration of curves with this topic.

In working with curves it is useful to be able to indicate particular points and their neighborhoods on the curve. To that end we *parameterize* a curve over a real interval mapping each parameter point, $t \in [0, a]$, to a point $R(t)$ on the plane:

$$R : [0, a] \to \mathbb{R}^2 \ .$$

Thus the endpoints of finite open curve are $R(0)$ and $R(a)$; for closed curves we require $R(0) = R(a)$.

The parameterization of a curve is not unique. Besides the geometric information encoded in the image of R, the parameterization also encodes a parameterization-dependent *velocity*. To visualize this, observe that moving the parameter at unit velocity slides a point $R(t)$ along the curve: the rate of change of $R(t)$, or velocity, is the vector $\vec{v}(t) = \frac{d}{dt}R(t)$. Indeed, given any strictly increasing function $t(s) : [0, b] \to [0, a]$ we *reparameterize* the curve as $R(t(s))$ so that moving along $s \in [0, b]$ generates the same points along the curve; the geometry remains the same, but by chain rule of the calculus the velocity is now $\vec{v}(s) = \frac{d}{ds}R(t(s)) = \frac{d}{dt}R(t(s))\frac{d}{ds}t(s)$: at every point the $R(t(s))$ reparameterization scales the velocity by $\frac{d}{ds}t(s)$.

Given a parameterized curve there is a unique reparameterization, $\hat{R}(s) = R(t(s))$, with the property that $\|\vec{v}(s)\| = 1,\ s \in [0, b]$. In *arc-length* parameterization of a curve, unit motion along the parameter s corresponds to unit motion along the length of the curve. Consequently, s is the length traveled along the curve walking from $\hat{R}(0)$ to $\hat{R}(s)$, therefore b is the length of the entire curve.

In the special setting of an arc-length parameterization the curvature at a point $R(s)$ is identical to the second derivative $\frac{d^2}{ds^2}R(s)$. It is a grave error to identify curvatures with second derivatives in general. The former is a geometric quantity only, and we defined it without reference to a parameterization; the latter encodes both geometry and velocity, and is parameterization-dependent. Here a spaceship analogy is helpful. If a spaceship travels at unit speed along a curved path, the curvature give the acceleration of the spaceship. Now if the spaceship travels at a nonuniform velocity along the path, then part of the acceleration is due to curvature, and part is due to speeding up and slowing down. A parameterization encodes velocity—this can be extremely useful for some applications.

Parameterization enables us to reformulate our statement of convergence. Given a sequence of parameter values, $0 = t_1 \leq t_2 \ldots \leq t_{n-1} \leq t_n = b$, for a "sufficiently well-behaved" parameterization of a "sufficiently well-behaved" curve[6], we may form an inscribed polygon taking $V_i = R(t_i)$. Then the *parametric mesh size* of the inscribed polygon is the greatest of all parameter intervals $[t_i, t_i + 1]$:

$$h_R(p) = \max_i (t_{i+1} - t_i) \ .$$

Unlike *geometric* mesh size, *parametric* mesh size is dependent on the chosen parameterization.

As before, consider a sequence of inscribed polygons, each sampling the curve at more parameter points, and in the limit sampling the curve at all parameter points: the associated sequences of discrete measurements approach their continuous analogs:

$$len(r) = \lim_{h_R(p_i)\to 0} len(p_i) \ ,$$

$$tsc(r) = \lim_{h_R(p_i)\to 0} tsc(p_i) \ .$$

5 Conclusion and Overview

So far we have looked at the geometry of a plane curve and demonstrated that it is possible to define its discrete analogue. The formulas for length and curvature of a discrete curve (a polygon) are immediately amenable to computation. *Convergence* guarantees that in the presence of abundant computational resources we may refine our discrete curve until the measurements we take match to arbitrary precision their counterparts on a smooth curve. We discussed an example of *structure preservation*, namely that the Turning Number Theorem holds exactly for discrete curves, even for coarse mesh sizes. If we wrote an algorithm whose correctness relied on the Turning Number Theorem, then the algorithm could be applied to our discrete curve.

The following chapters will extend our exploration of discrete analogues of the objects of differential geometry to the settings of surfaces and volumes and to application areas spanning physical simulation (thin shells and fluids) and geometric modeling (remeshing and parameterization). In each application area algorithms make use of mathematical structures that are carried over from the continuous to the discrete realm. We are *not* interested in preserving structure just for mathematical elegance—each application demonstrates that by carrying over the right structures from the continuous to the discrete setting, the resulting algorithms exhibit impressive computational and numerical performance.

[6] Indeed, the following theorems depend on the parameterization being *Lipschitz*, meaning that small changes in parameter value lead to small motions along the curve:

$$d(R(a), R(b)) \leq C|a - b| \ ,$$

for some constant C. The existence of a Lipschitz parameterization is equivalent to the curve being *rectifiable*, or having finite arclength. Further care must be taken in allowing non-continuous curves with finitely many isolated jump points.

Chapter 2: What Can We Measure?

Peter Schröder
Caltech

1 Introduction

When characterizing a shape or changes in shape we must first ask, what can we measure about a shape? For example, for a region in $\mathbb{R}^3$ we may ask for its volume or its surface area. If the object at hand undergoes deformation due to forces acting on it we may need to formulate the laws governing the change in shape in terms of measurable quantities and their change over time. Usually such measurable quantities for a shape are defined with the help of integral calculus and often require some amount of smoothness on the object to be well defined. In this chapter we will take a more abstract approach to the question of measurable quantities which will allow us to define notions such as mean curvature integrals and the curvature tensor for piecewise linear meshes without having to worry about the meaning of second derivatives in settings in which they do not exist. In fact in this chapter we will give an account of a classical result due to Hadwiger, which shows that for a convex, compact set in $\mathbb{R}^n$ there are only $n+1$ unique measurements if we require that the measurements be invariant under Euclidian motions (and satisfy certain "sanity" conditions). We will see how these measurements are constructed in a very straightforward and elementary manner and that they can be read off from a characteristic polynomial due to Steiner. This polynomial describes the volume of a family of shapes which arise when we "grow" a given shape. As a practical tool arising from these consideration we will see that there is a well defined notion of the curvature tensor for piecewise linear meshes and we will see very simple formulas for quantities needed in physical simulation with piecewise linear meshes. Much of the treatment here will initially be limited to convex bodies to keep things simple. This limitation that will be removed at the very end.

The treatment in this chapter draws heavily upon work by Gian-Carlo Rota and Daniel Klein, Hadwigers pioneering work, and some recent work by David Cohen-Steiner and colleagues.

2 Geometric Measures

To begin with let us define what we mean by a measure. A measure is a function μ defined on a family of subsets of some set S, and it takes on real values: $\mu : L \to \mathbb{R}$. Here L denotes this family of subsets and we require of L that it is closed under finite set union and intersection as well as that it contains the empty set, $\emptyset \in L$. The measure μ must satisfy two axioms: (1) $\mu(\emptyset) = 0$; and (2) $\mu(A \cup B) = \mu(A) + \mu(B) - \mu(A \cap B)$ whenever A and B are measureable. The first axiom is required to get anything that has a hope of being well defined. If $\mu(\emptyset)$ was not equal to zero the measure of some set $\mu(A) = \mu(A \cup \emptyset) = \mu(A) + \mu(\emptyset)$ could not be defined. The second axiom captures the idea that the measure of the union of two sets should be the sum of the measures minus the measure of their overlap. For example, consider the volume of the union of two sets which clearly has this property. It will also turn that the additivity property is the key to reducing measurements for complicated sets to measurements on simple sets. We will furthermore require that all measures we consider be invariant under Euclidian motions, *i.e.*, translations and rotations. This is so that our measurements do not depend on where we place the coordinate origin and how we orient the coordinate axes. A measure which depended on these wouldn't be very useful.

Let's see some examples. A well known example of such a measure is the volume of bodies in $\mathbb{R}^3$. Clearly the volume of the empty body is zero and the volume satisfies the additivity axiom. The volume also does not depend on where the coordinate origin is placed and how the coordinate frame is rotated. To uniquely tie down the volume there is one final ambiguity due to the units of measurement being used, which we must remove. To this end we enforce a normalization which states that the volume of the unit, coordinate axis aligned parallelipiped in $\mathbb{R}^n$ be one. With this we get

$$\mu_n^n(x_1, \ldots, x_n) = x_1 \cdot \ldots \cdot x_n$$

for x_1 to x_n the side lengths of a given axis aligned parallelipiped. The superscript n denotes this as a measure on $\mathbb{R}^n$, while the subscript denotes the type of measurement being taken. Clearly the definition of μ_n^n is translation invariant. It also does not depend on how we number our coordinate axes, *i.e.*, it is invariant under permutations of the coordinate axes. Finally if we rotate the global coordinate frame none of the side lengths of our parallelipiped change so neither does μ_n^n. Notice that we have only defined the meaning of μ_n^n for axis aligned parallelipipeds as well as finite unions and intersections of such parallelipipeds. The definition can be extended to more general bodies through a limiting process akin to how Riemann integration fills the domain with ever smaller boxes to approach the entire domain in the limit. There is nothing here that prevents us from performing the same limit process. In fact we will see later that once we add this final requirement, that the measure is continuous in the limit, the class of such measures is completely tied down. This is Hadwiger's famous theorem. But, more on that later.

Of course the next question is, are there other such invariant measures? Here is a proposal:

$$\mu_{n-1}^n(x_1, \ldots, x_n) = \\ x_1x_2 + x_1x_3 + \ldots + x_1x_n + x_2x_3 + \ldots + x_2x_n \ldots$$

For an axis aligned parallelipiped in $\mathbb{R}^3$ we'd get

$$\mu_2^3(x_1, x_2, x_3) = x_1x_2 + x_2x_3 + x_3x_1$$

which is just half the surface area of the paralellipiped with sides x_1, x_2, and x_3. Since we have the additivity property we can certainly extend this definition to more general bodies

through a limiting process and find that we get, up to normalization, the surface area.

Continuing in this fashion we are next led to consider

$$\mu_1^3(x_1, x_2, x_3) = x_1 + x_2 + x_3$$

(and similarly for μ_1^n). For a parallelipiped this function measures one quarter the sum of lengths of its bounding edges. Once again this new measure is clearly rigid motion invariant. What we need to check is whether it satisfies the additivity theorem. Indeed it does, which is easily checked for the union of two paralellipipeds. What is less clear is what this measure represents if we extend it to more general shapes where the notion of "sum of edge lengths" is not clear. The resulting continuous measure is sometimes referred to as the *mean width*.

From these simple examples we can see a pattern. For Euclidian n-space we can use the elementary symmetric polynomials in edge lengths to define n invariant measures

$$\mu_k^n(x_1, \ldots, x_n) = \sum_{1 \le i_1 < i_2 < \ldots < i_k \le n} x_{i_1} x_{i_2} \ldots x_{i_k}$$

for $k = 1, \ldots, n$ for parallelipipeds. To extend this definition to more general bodies we'll follow ideas from geometric probability. In particular we will extend these measures to the ring of compact convex bodies, *i.e.*, finite unions and intersections of compact convex sets in $\mathbb{R}^n$.

3 How Many Points, Lines, Planes,... Hit a Body?

Consider a compact convex set, a *convex body*, in $\mathbb{R}^n$ and surround it by a box. One way to measure its volume is to count the number of points which, when randomly thrown into the box, hit the body versus those that hit empty space inside the box. To generalize this idea we consider *affine subspaces* of dimension $k < n$ in $\mathbb{R}^n$. Recall that an affine subspace of dimension k is spanned by $k+1$ points $p_i \in \mathbb{R}^n$ (in general position), *i.e.*, the space consists of all points q which can be written as affine combinations $q = \sum_i \alpha_i p_i$, $\sum_i \alpha_i = 1$. Such an affine subspace is simply a linear subspace translated, *i.e.*, it does not necessarily go through the origin. For example, for $k = 1$, $n = 3$ we will consider all lines—a line being the set of points one can generate as affine combinations of two points on the line—in three space. Let $\lambda_1^3(R)$ be the measure of all lines going through a rectangle in $\mathbb{R}^3$. Then

$$\lambda_1^3(R) = c\mu_2^3(R),$$

i.e., the measure of all lines which meet the rectangle is proportional to the area of the rectangle. To see this, note that a given line (in general position) either meets the rectangle once or not at all. Conversely for a given point in the rectangle there is a whole set of lines—a sphere's worth—which "pierce" the rectangle in the given point. The measure of those lines is proportional to the area of the unit sphere. Since this is true for all points in the rectangle we see that the total measure of all such lines must be proportional to the area of the rectangle with a constant of proportionality depending on the measure of the sphere. For now such constants are irrelevant for our considerations so we will just set it to unity. Given a more complicated shape C in a plane nothing prevents from performaing a limiting process and we see that the measure of lines meeting C is

$$\lambda_1^3(C) = \mu_2^3(C),$$

i.e., it is proportional to the area of the region C. Given a union of rectangles $D = \cup_i R_i$, each living in a different plane, we get

$$\int X_D(\omega)\, d\lambda_1^3(\omega) = \sum_i \mu_2^3(R_i).$$

Here $X_D(\omega)$ counts the number of times a line ω meets the set D and the integration is performed over all lines. Going to the limit we find for some convex body E a measure proportional to its surface area

$$\int X_E(\omega)\, d\lambda_1^3(\omega) = \mu_2^3(E).$$

Using planes ($k = 2$) we can now generalize the mean width. For a straight line $c \in \mathbb{R}^3$ we find $\lambda_2^3(c) = \mu_1^3(c) = l(c)$, *i.e.*, the measure of all planes that meet the straight line is proportional—as before we set the constant of proportionality to unity—to the length of the line. The argument mimics what we said above: a plane either meets the line once or not at all. For a given point on the line there is once again a whole set of planes going through that point. Considering the normals to such planes we see that this set of planes is proportional in measure to the unit sphere without being more precise about the actual constant of proportionality. Once again this can be generalized with a limiting process giving us the measure of all planes hitting an arbitrary curve in space as proportional to its length

$$\int X_F(\omega)\, d\lambda_2^3(\omega) = \mu_1^3(F).$$

Here the integration is performed over all planes $\omega \in \mathbb{R}^3$, and X_F counts the number of times a given plane touches the curve F.

It is easy to see that this way of measuring recovers the perimeter of a parallelipiped as we had defined it before

$$\lambda_2^3(P) = \mu_1^3(P).$$

To see this consider the integration over all planes but taken in groups. With the parallelipiped having one corner at the origin—and being axis aligned—first consider all planes whose normal (n_x, n_y, n_z) has either all non-negative or non-positive entries (*i.e.*, the normal, or its negative, points into the first octant). Any such plane, if it meets the parallelipiped, meets it in a point along either x_1, x_2 or x_3 giving us the desired $\mu_1^3(P) = x_1 + x_2 + x_3$ as the measure of all such planes. The same argument holds for the remaining seven octants giving us the desired result up to a constant. We can now see that $\mu_1^3(E)$ for some convex body E can be written as

$$\int X_E(\omega)\, d\lambda_2^3(\omega) = \mu_1^3(E),$$

i.e., the measure of all planes which meet E. With this we have generalized the notion of perimeter to more general sets.

All this can be summarized as follows. Let μ be a measure which is Euclidean motion invariant. Then it can be written, up to normalization, as a linear combination of the measures $\mu_k^n(C)$ of all affine subspaces of dimension $n-k$ meeting $C \subset \mathbb{R}^n$ for $k = 1, \ldots, n$. These measures are called the *intrinsic volumes*.

Are these all such measures? It turns out there is one measure missing, which corresponds to the elementary symmetric function of order zero

$$\mu_0(x_1,\ldots,x_n) = \begin{cases} 1 & n > 0 \\ 0 & n = 0 \end{cases}$$

This very special measure is the Euler characteristic of a convex body. It takes the value 1 on all non-empty convex bodies. The main trick is to prove that μ_0 is indeed well defined. This can be done by induction. In dimension $n = 1$ we consider closed intervals $[a, b]$, $a < b$. Instead of working with the set directly we consider a functional on the characteristic function $f_{[a,b]}$ of the set which does the trick

$$\chi_1(f) = \int_{\mathbb{R}} f(\omega) - f(\omega+)\, d\omega.$$

Here $f(\omega+)$ denotes the right limiting value of f at ω: $\lim_{\epsilon\to 0} f(\omega+\epsilon)$, $\epsilon > 0$. For the set $[a, b]$, $f(\omega) - f(\omega+)$ is zero for all $\omega \in \mathbb{R}$ except b since $f(b) = 1$ and $f(b+) = 0$. For higher dimensions we proceed by induction. In $\mathbb{R}^n$ take a straight line L and consider the affine subspaces A_ω of dimension $n-1$ which are orthogonal to L and parameterized by ω along L. Letting f be the characteristic function of a convex body in $\mathbb{R}^n$ we get

$$\chi_n(f) = \int_{\mathbb{R}} \chi_{n-1}(f_\omega) - \chi_{n-1}(f_{\omega+})\, d\omega.$$

Here f_ω is the restriction of f to the affine space A_ω or alternatively the characteristic function of the intersection of A_ω and the convex body of interest. With this we define $\mu_0^n(G) = \chi_n(f)$ for any finite union of convex bodies G and f the characteristic function of the set $G \in \mathbb{R}^n$.

That this definition of μ_0^n amounts to the Euler characteristic is not immediately clear, but it is easy to show, if we convince ourselves that for any non empty convex body $C \in \mathbb{R}^n$

$$\mu_0^n(\text{Int}(C)) = (-1)^n.$$

For $n = 1$, *i.e.*, the case of open intervals on the real line, this statement is obviously correct. We can now apply the recursive definition to the characteristic function of the interior of C and get

$$\mu_0^n(\text{Int}(C)) = \int_\omega \chi_{n-1}(f_\omega) - \chi_{n-1}(f_{\omega+})\, d\omega.$$

By induction the right hand side is zero except for the first ω at which $A_\omega \cap C$ is non-empty. There $\chi_{n-1}(f_{\omega+}) = (-1)^{n-1}$, thus proving our assertion for all n.

The Euler-Poincaré formula for a polyhedron

$$|F| - |E| - |V| = 2(1-g)$$

which relates the number of faces, edges, and vertices to the genus now follows easily. Given a polyhedron simply write it as the non-overlapping union of the interiors of all its cells from dimension n down to dimension 0, where the interior of a vertex (0-cell) is the vertex itself. Then

$$\mu_0^n(P) = \sum_{c\in P} \mu_0^n(\text{Int}(c)) = c_0 - c_1 + c_2 - \ldots$$

where c_i equals the number of cells of dimension i. For the case of a polyhedron in $\mathbb{R}^3$ this is exactly the Euler-Poincaré formula.

4 The Intrinsic Volumes and Hadwiger's Theorem

The above machinery can now be used to define the intrinsic volumes as functions of the Euler characteristic alone for all finite unions of convex bodies G

$$\mu_k^n(G) = \int \mu_0^n(G \cap \omega)\, d\lambda_{n-k}^n(\omega).$$

Here $\mu_0^n(G \cap \omega)$ plays the role of $X_G(\omega)$ we used earlier to count the number of times ω hits G.

There is one final ingredient missing, continuity in the limit. Suppose C_n is a sequence of convex bodies which converges to C in the limit as $n \to \infty$. Hadwiger's theorem says that if a Euclidean motion invariant measure μ of convex bodies in $\mathbb{R}^n$ is continuous in the sense that

$$\lim_{C_n \to C} \mu(C_n) = \mu(C)$$

then μ must be a linear combination of the intrinsic volumes μ_k^n, $k = 0,\ldots,n$. In other words, the intrinsic volumes, under the additional assumption of continuity, are the only linearly independent, Euclidean motion invariant, additivie measures on finite unions and intersections of convex bodies in $\mathbb{R}^n$.

What does all of this have to do with the applications we have in mind? A consequence of Hadwiger's theorem assures us that if we want to take measurements of piecewise linear geometry (surface or volume meshes, for example) such measurements should be functions of the intrinsic volumes. This assumes of course that we are looking for additive measurements which are Euclidean motion invariant and continuous in the limit. For a triangle for example this would be area, edge length, and Euler characteristic. Similarly for a tetrahedron with its volume, surface area, mean width, and Euler characteristic. As the name suggests all of these measurements are intrinsic. For a 2-manifold mesh which is the boundary of a solid one of these measurements is an *extrinsic* quantity corresponding to the dihedral angle between triangles meeting at an edge (see below).

5 Steiner's Formula

We return now to questions of discrete differential geometry by showing that the intrinsic volumes are intricately linked to curvature integrals and represent their generalization to the non-smooth setting. This connection is established by Steiner's formula.

Consider a non-empty convex body $K \in \mathbb{R}^n$ together with its parallel bodies

$$K_\epsilon = \{x \in \mathbb{R}^n : d(x,K) \le \epsilon\}$$

where $d(x, K)$ denotes the Euclidean distance from x to the set K. In effect K_ϵ is the body K thickened by ϵ. Steiner's formula gives the volume of K_ϵ as a polynomial in ϵ

$$V(K_\epsilon) = \sum_{j=0}^{n} V(\mathbb{B}_{n-j}) V_j(K) \epsilon^{n-j}.$$

Here the $V_j(K)$ are the measures μ_k^n we have seen earlier. For this formula to be correct the $V_j(K)$ are normalized so that they compute the j-dimensional volume when restricted to a j-dimensional subspace of $\mathbb{R}^n$. $V(\mathbb{B}_{n-j}) = \pi^{n/2}/\Gamma(1+$

$1/2n$) denotes the $(n-j)$-volume of the $(n-j)$-unit ball. In particular we have $V(\mathbb{B}_0) = 1$, $V(\mathbb{B}_1) = 2$, $V(\mathbb{B}_2) = \pi$, and $V(\mathbb{B}_3) = 4\pi/3$.

In the case of a polyhedron we can verify Steiner's formula "by hand." Consider a tetrahedron in $T \in \mathbb{R}^3$ and the volume of its parallel bodies T_ϵ. For $\epsilon = 0$ we have the volume of T itself ($V_3(T)$). The first order term in ϵ, $2V_2(T)$, is controlled by area measures: above each triangle a displacement along the normal creates additional volume proportional to ϵ and the area of the triangle. The second order term in ϵ, $\pi V_1(T)$, corresponds to edge lengths. Above each edge the parallel bodies form a wedge with opening angle θ which is the exterior angle of the faces meeting at that edge and radius ϵ (this is the extrinsic measurement alluded to above). The volume of such as wedge is proportional to edge length, exterior angle, and ϵ^2. Finally the third order term in ϵ, $4\pi/3V_0(T)$, corresponds to the volume of the parallel bodies formed over vertices. Each vertex gives rise to additional volume spanned by the vertex and a spherical cap above it. The spherical cap corresponds to a spherical triangle formed by the three incident triangle normals. The volume of such a spherical wedge is proportional to its solid angle and ϵ^3.

If we have a convex body with a boundary which is C^2 we can give a different representation of Steiner's formula. Consider such a convex $M \in \mathbb{R}^n$ and define the offset function

$$g(p) = p + t\vec{n}(p)$$

for $0 \leq t \leq \epsilon$, $p \in \partial M$ and $\vec{n}(p)$ the outward normal to M in p. We can now directly compute the volume of M_ϵ as the sum of $V_n(M)$ and the volume between the surfaces ∂M and ∂M_ϵ. The latter can be written as an integral of the determinant of the Jacobian of g

$$\int_{\partial M} \left(\int_0^\epsilon \left| \frac{\partial g(p)}{\partial p} \right| dt \right) dp.$$

Since we have a choice of coordinate frame in which to do this integration we may assume wlog that we use principal curvature coordinates on ∂M, *i.e.*, a set of orthogonal directions in which the curvature tensor diagonalises. In that case

$$\begin{aligned} \left| \frac{\partial g(p)}{\partial p} \right| &= |\mathbb{I} + t\mathbb{K}(p)| \\ &= \prod_{i=1}^{n-1} (1 + \kappa_i(p)t) \\ &= \sum_{i=0}^{n-1} \mu_i^{n-1}(\kappa_1(p), \ldots, \kappa_{n-1}(p)) t^i. \end{aligned}$$

In other words, the determinant of the Jacobian is a polynomial in t whose coefficients are the elementary symmetric functions in the principal curvatures. With this subsitution we can trivially integrate over the variable t and get

$$V(M_\epsilon) = V_n(M) + \sum_{i=0}^{n-1} \frac{\epsilon^{i+1}}{i+1} \int_{\partial M} \mu_i^{n-1}(\kappa_1(p), \ldots, \kappa_{n-1}(p))\, dp.$$

Comparing the two versions of Steiner's formula we see that the intrinsic volumes generalize curvature integrals. For example, for $n = 3$ and some arbitrary convex body K we get

$$V(K_\epsilon) = 1V_3(K) + 2V_2(K)\epsilon + \pi V_1(K)\epsilon^2 + \frac{4\pi}{3} V_0(K)\epsilon^3$$

while for a convex body M with C^2 smooth boundary the formula reads as

$$\begin{aligned} V(M_\epsilon) = V_3(M) + & \underbrace{\left(\int_{\partial M} \underbrace{\mu_0^2(\kappa_1(p), \kappa_2(p))}_{=1}\, dp \right)}_{=A} \epsilon + \\ & \left(\int_{\partial M} \underbrace{\mu_1^2(\kappa_1(p), \kappa_2(p))}_{=2H}\, dp \right) \frac{\epsilon^2}{2} + \\ & \underbrace{\left(\int_{\partial M} \underbrace{\mu_2^2(\kappa_1(p), \kappa_2(p))}_{=K}\, dp \right)}_{=4\pi} \frac{\epsilon^2}{3}. \end{aligned}$$

6 What All This Machinery Tells Us

We began this section by considering the question of what additive, continuous, rigid motion invariant measurements there are for convex bodies in $\mathbb{R}^n$ and learned that the $n+1$ intrinsic volumes are the only ones and any such measure must be a linear combination of these. We have also seen that the intrinsic volumes in a natural way extend the idea of curvature integrals over the boundary of a smooth body to general convex bodies without regard to a differentiable structure. These considerations become one possible basis on which to claim that integrals of Gaussian curvature on a triangle mesh become sums over excess angle at vertices and that integrals of mean curvature can be identified with sums over edges of dihedral angle weighted by edge length. These quantities are always *integrals*. Consequently *they do not make sense as pointwise quantities*. In the case of smooth geometry we can define quantities such as mean and Gaussian curvature as pointwise quantities. On a simplicial mesh they are only defined as integral quantities.

All this machinery was developed for convex bodies. If a given mesh is not convex the additivity property allows us to compute the quantities no less by writing the mesh as a finite union and intersection of convex bodies and then tracking the corresponding sums and differences of measures. For example, $V(K_\epsilon)$ is well defined for an individual triangle K and we know how to identify the coefficients involving intrinsic volumes with the integrals of elementary polynomials in the principal curvatures. Glueing two triangles together we can perform a similar identification carefully teasing apart the intrinsic volumes of the union of the two triangles. In this way the convexity requirement is relaxed so long as the shape of interest can be decomposed into a finite union of convex bodies.

This machinery was used by Cohen-Steiner and Morvan to give formulas for integrals of a discrete curvature tensor. We give these here together with some fairly straightforward intuition regarding the underlying geometry.

Let P be a polyhedron with vertex set V and edge set E and B a ball in $\mathbb{R}^3$ then we can define integrated Gaussian and mean curvature measures as

$$\phi_P^G(B) = \sum_{v \in V \cap B} K_v \quad \text{and} \quad \phi_P^H(B) = \sum_{e \in E} l(e \cap B)\theta_e,$$

where $K_v = 2\pi - \sum_j \alpha_j$ is the excess angle sum at vertex v defined through all the incident triangle angles at v, while $l(.)$ denotes the length and θ_e is the signed dihedral angle at e made between the incident triangle normals. Its sign is positive for convex edges and negative for concave edges (note that this requires an orientation on the polyhedron). In essence this is simply a restatement of the Steiner polynomial coefficients restricted to the intersection of the ball B and the polyhedron P. To talk about the second fundamental form II_p at some point p in the surface, it is convenient to first extend it to all of $\mathbb{R}^3$. This is done by setting it to zero if one of its arguments is parallel to the normal p. With this one may define

$$\bar{II}_P(B) = \sum_{e \in E} l(e \cap B)\theta_e e_n \otimes e_n, \quad e_n = e/\|e\|.$$

The dyad $e_n \otimes e_n(u, v) = \langle u, e_n \rangle \langle v, e_n \rangle$ projects given vectors u and v along the normalized edge. What is the geometric interpretation of the summands? Consider a single edge and the associated dyad. The curvature along this edge is zero while it is θ orthogonal to the edge. A vector aligned with the edge is mapped to θ_e while one orthogonal to the edge is mapped to zero. These are the principal curvatures *except they are reversed*. Hence $\bar{II}_P(B)$ is an integral measure of the curvature tensor *with the principal curvature values exchanged*. For example we can assign each vertex a three by three tensor by summing the edge terms for each incident edge. As a tangent plane at the vertex, which we need to project the three by three tensor to the expected two by two tensor in the tangent plane, we may take a vector parallel to the area gradient at the vertex. Alternatively we could defined $\bar{II}_P(B)$ for balls containing a single triangle and its three edges each. In that case the natural choice for the tangent plane is the support plane of the triangle. In practice one often finds that noise in the mesh vertex positions makes these discrete computations noisy. It is then a simple matter of enlarging B to stabilize the computations.

Cohen-Steiner and Morvan show that this definition can be rigorously derived from considering the coefficients of the Steiner polynomial in particular in the presence of non-convexities (which requires some fancy footwork...). They also show that if the polyhedron is a sufficiently fine sample of a smooth surface the discrete curvature tensor integrals have linear precision with regards to continuous curvature tensor integrals. They also provide a formula for a discrete curvature tensor which does not have the principal curvatures swapped.

7 Further Reading

The material in this section only gives the rough outlines of what is a very fundamental theory in probability and geometric measure theory. In particular there are many other consequences which follow from relationships between intrinsic volumes which we have not touched upon. A rigorous derivation of the results of Hadwiger, but much shorter than the original can be found in [Klain 1995]. A complete and rigorous account of the derivation of intrinsic volumes from first principles in geometric probability can be found in the short book by Klain and Rota [Klain and Rota 1997], while the details of the discrete curvature tensor integrals can be found in [Cohen-Steiner and Morvan 2003]. Approximation results which discuss the accuracy of these measure vis-a-vis an underlying smooth surface are treated by Cohen-Steiner and Morvan in a series of tech reports available at http://www-sop.inria.fr/geometrica/publications/.

Acknowledgments This work was supported in part by NSF (DMS-0220905, DMS-0138458, ACI-0219979), DOE (W-7405-ENG-48/B341492), nVidia, the Center for Integrated Multiscale Modeling and Simulation, Alias, and Pixar.

References

COHEN-STEINER, D., AND MORVAN, J.-M. 2003. Restricted Delaunay Triangulations and Normal Cycle. In *Proceedings of the 19th Annual Symposium on Computational Geometry*, 312–321.

KLAIN, D. A., AND ROTA, G.-C. 1997. *Introduction to Geometric Probability*. Cambridge University Press.

KLAIN, D. A. 1995. A Short Proof of Hadwiger's Characterization Theorem. *Mathematika 42*, 84, 329–339.

Chapter 3: Curvature Measures for Discrete Surfaces

John M. Sullivan
sullivan@math.tu-berlin.de
Institut für Mathematik, TU Berlin MA 3–2
Str. des 17. Juni 136, 10623 Berlin

The curvatures of a smooth curve or surface are local measures of its shape. Here we consider analogous measures for discrete curves and surfaces, meaning polygonal curves and triangulated polyhedral surfaces. We find that the most useful analogs are those which preserve integral relations for curvature, like the Gauß–Bonnet theorem. For simplicity, we usually restrict our attention to curves and surfaces in euclidean space $\mathbb{R}^3$, although many of the results would easily generalize to other ambient manifolds of arbitrary dimension.

These notes are based on work by many people, but unfortunately do not include proper citations to the literature.

1 Smooth Curves and Surfaces

Before discussing discrete analogs, we briefly review the usual theory of curvatures for smooth curves and surfaces in space.

1.1 Smooth curves

The curvatures of a smooth curve γ are the local properties of its shape invariant under Euclidean motions. The only first-order information is the tangent line; since all lines in space are equivalent, there are no first-order invariants. Second-order information is given by the osculating circle, and the invariant is its curvature $\kappa = 1/r$.

For a plane curve given as a graph $y = f(x)$ let us contrast the notions of curvature and second derivative. At a point p on the curve, we can find either one by translating p to the origin, transforming so the curve is horizontal there, and then comparing to a standard set of reference curves. The difference is that for curvature, the transformation is a Euclidean rotation, while for second derivative, it is a shear $(x, y) \mapsto (x, y - ax)$. A parabola has constant second derivative f'' because it looks the same at any two points after a shear. A circle, on the other hand, has constant curvature because it looks the same at any two points after a rotation.

A plane curve is completely determined (up to rigid motion) by its (signed) curvature $\kappa(s)$ as a function of arclength s. For a space curve, however, we need to look at the third-order invariants, which are the torsion τ and the derivative κ' (which of course gives no new information). These are now a complete set of invariants: a space curve is determined by $\kappa(s)$ and $\tau(s)$. (Generally, for higher codimension, higher-order invariants are needed. For curves in $\mathbb{R}^n$, we need $n - 1$ curvatures, of order up to n, to characterize the shape.)

A smooth space curve γ is often described by its orthonormal *Frenet frame* (T, N, B). With respect to an arclength parameter s, the defining equations are $T := \gamma'$ and

$$\begin{pmatrix} T \\ N \\ B \end{pmatrix}' = \begin{pmatrix} 0 & \kappa & 0 \\ -\kappa & 0 & \tau \\ 0 & -\tau & 0 \end{pmatrix} \begin{pmatrix} T \\ N \\ B \end{pmatrix}.$$

For a curve γ lying on a surface M, it is often more useful to consider the *Darboux frame* (T, η, ν), adapted to this situation. This orthonormal frame includes the tangent vector T to γ and the normal vector ν to M. Its third element is thus $\eta := \nu \times T$, called the *cornormal*. The curvature vector of γ decomposes into parts tangent and normal to M as $T' = \kappa N = \kappa_g \eta + \kappa_n \nu$. Here in fact, κ_n measures the normal curvature of M in the direction T, and is independent of γ.

1.2 Smooth surfaces

Given a (two-dimensional, oriented) surface M (immersed) in $\mathbb{R}^3$, we understand its local shape by looking at the Gauß map $\nu : M \to \mathbb{S}^2$ given by the unit normal vector $\nu = \nu_p$ at each point $p \in M$. We can view its derivative at p as a linear endomorphism $-S_p : T_pM \to T_pM$, since T_pM and $T_{\nu_p}\mathbb{S}^2$ are naturally identified, being parallel planes in $\mathbb{R}^3$. The map S_p is called the shape operator (or Weingarten map).

The shape operator is the second-order invariant (or curvature) which completely determines the original surface M. However, it is usually more convenient to work with scalar quantities. The eigenvalues κ_1 and κ_2 of S_p are called principal curvatures, and (since they cannot be globally distinguished) it is their symmetric functions which have geometric meaning.

We define the *Gauß curvature* $K := \kappa_1\kappa_2$ as the determinant of S_p and the *mean curvature* $H := \kappa_1 + \kappa_2$ as its trace. Note that the sign of H depends on the choice of unit normal ν, and so often it is more natural to work with the *vector mean curvature* (or mean curvature vector) $\mathbf{H} := H\nu$. Note furthermore that some authors use the opposite sign on S_p and thus H, and many use $H = (\kappa_1 + \kappa_2)/2$, justifying the name *mean* curvature. Our conventions mean that the mean curvature vector for a convex surface points inwards (like the curvature vector for a circle). For a unit sphere oriented with inward normal, the Gauß map ν is the antipodal map, $S_p = I$, and $H = 2$.

The Gauß curvature is an intrinsic notion, depending only on the pullback metric on the surface M, and not on the immersion into space. That is, K is unchanged by bending the surface without stretching it. For instance, a developable surface like a cylinder or cone has $K = 0$ because it is obtained by bending a flat plane. One intrinsic definition of $K(p)$ is obtained by considering the circumferences C_ε of (intrinsic) ε-balls around p. We get

$$\frac{C_\varepsilon}{2\pi\varepsilon} = 1 - \frac{\varepsilon^2}{6}K + \mathcal{O}(\varepsilon^3).$$

Mean curvature is certainly not intrinsic, but it has a nice variational interpretation. Consider a variation vectorfield V on M, compactly supported away from any boundary. Then $H = -\delta \text{ Area} / \delta \text{ Vol}$ in the sense that

$$\delta_V \text{ Vol} = \int V \cdot \nu \, dA, \quad \delta_V \text{ Area} = -\int V \cdot H\nu \, dA.$$

With respect to the L^2 inner product $\langle U, V \rangle := \int U_p \cdot V_p \, dA$ on vectorfields, the vector mean curvature is the negative gradient of the area functional, often called the first variation of area: $\mathbf{H} = -\nabla \text{ Area}$. (Similarly, the negative gradient of length for a curve is κN.)

Just as κ is the geometric version of second derivative for curves, mean curvature is the geometric version of the Laplacian Δ. Indeed, if a surface M is written locally as the graph of a height function f over its tangent plane T_pM then $H(p) = \Delta f$. Alternatively, we can write $\mathbf{H} = \nabla_M \cdot \nu = \Delta_M \mathbf{x}$, where $\mathbf{x}$ is the position vector in $\mathbb{R}^3$ and Δ_M is Beltrami's surface Laplacian.

If we flow a curve or surface to reduce its length or area, by following these gradients κN and $H\nu$, the resulting parabolic heat flow is slightly nonlinear in a natural geometric way. This so-called *mean-curvature flow* has been extensively studied as a geometric smoothing flow.

1.3 Integral curvature relations for curves

The *total curvature* of a curve is $\int \kappa \, ds$. For closed curves, the total curvature is at least 2π (Fenchel) and for knotted space curves the total curvature is at least 4π (Fáry/Milnor). For plane curves, we can consider instead the signed curvature, and find that $\int \kappa \, ds$ is always an integral multiple of 2π.

Suppose we define (following Milnor) the total curvature of a polygonal curve simply to be the sum of the turning angles at the vertices. Then all the theorems for smooth curves mentioned in the previous paragraph remain true for polygonal curves. Our goal, when defining curvatures for polyhedral surfaces, will be to again ensure that integral relations for these curvatures remain exactly true.

1.4 Integral curvature relations for surfaces

For surfaces, the integral curvature relations we want to consider relate area integrals over a region $D \subset M$ to arclength integrals over the boundary $\gamma = \partial D$. The Gauß–Bonnet theorem says, when D is a disk,

$$2\pi - \iint_D K \, dA = \oint_\gamma \kappa_g \, ds = \oint_\gamma T' \cdot \eta \, ds = -\oint \eta' \cdot \, d\mathbf{x},$$

where $d\mathbf{x} = T \, ds$ is the vector line element along γ. This implies that the total Gauß curvature of D depends only on a collar neighborhood of γ: if we make any modification to D supported away from the boundary, the total curvature is unchanged (as long as D remains topologically a disk). We will extend the notion of Gauß curvature from smooth surfaces to more general surfaces (in particular polyhedral surfaces) by requiring this property to remain true.

The other relations are all proved by Stokes' Theorem, and thus only depend on γ being the boundary of D in a homological sense; for these D is not required to be a disk. First consider the vector area

$$\mathbf{A}_\gamma := \tfrac{1}{2} \oint_\gamma \mathbf{x} \times d\mathbf{x} = \iint_D \nu \, dA.$$

The right-hand side represents the total vector area of any surface spanning γ, and the relation shows this to depend only on γ (and this time not even on a collar neighborhood). The integrand on the left-hand side depends on a choice of origin for the coordinates, but because we integrate over a closed loop, the integral is independent of this choice. Both sides of this vector area formula can be interpreted directly for a polyhedral surface, and the equation remains true in that case.

A simple integral for curve γ from p to q says that

$$T(q) - T(p) = \int_p^q T'(s) \, ds = \int \kappa N \, ds.$$

This can be viewed as a balance between tension forces trying to shrink the curve, and sideways forces holding it in place. It is the relation used in proving that κ is the first variation of length.

The analog for a surface patch D is the mean curvature force balance equation

$$\oint_\gamma \eta \, ds = -\oint_\gamma \nu \times d\mathbf{x} = \iint_D H\nu \, dA = \iint_D \mathbf{H} \, dA.$$

Again this represents a balance between surface tension forces acting in the conormal direction along the boundary of D and what can be considered as pressure forces (espcially in the case of constant H) acting normally across D. We will use this equation to develop the analog of mean curvature for discrete surfaces.

Two other similar relations that we will not need later are the torque balance

$$\oint_\gamma \mathbf{x} \times \eta \, ds = \oint_\gamma \mathbf{x} \times (\nu \times d\mathbf{x}) = \iint_D H(\mathbf{x} \times \nu) \, dA$$

and the area relation

$$\oint_\gamma \mathbf{x} \cdot \eta \, ds = \oint_\gamma \mathbf{x} \cdot (\nu \times d\mathbf{x}) = \iint_D (\mathbf{H} \cdot \mathbf{x} - 2) \, dA.$$

2 Discrete Surfaces

For us, a discrete or polyhedral surface $M \subset \mathbb{R}^3$ will mean a triangulated surface with a PL map into space. In more detail, we start with an abstract combinatorial triangulation—a simplicial complex—representing a 2-manifold with boundary. We then pick positions $p \in \mathbb{R}^3$ for every vertex, which uniquely determine a linear map on each triangle; these fit together to form the PL map.

2.1 Gauß curvature

It is well known how the notion of Gauß curvature extends to such discrete surfaces M. Any two adjacent triangles (or, more generally, any simply connected region in M not including any vertices) can be flattened—developed isometrically into the plane. Thus the Gauß curvature is supported on the vertices $p \in M$. In fact, to keep the Gauß–Bonnet theorem true, we must take

$$\iint_D K \, dA := \sum_{p \in D} K_p; \qquad K_p := 2\pi - \sum_i \theta_i.$$

Here, the angles θ_i are the interior angles at p of the triangles meeting there, and K_p is often known as the angle defect

at p. If D is any neighborhood of p contained in $\mathrm{Star}(p)$, then $\oint_{\partial D} \eta\, ds = \sum \theta_i$; when the triangles are acute, this is most easily seen by letting ∂D be the path connecting their circumcenters and crossing each edge perpendicularly.

(Similar arguments lead to a notion of Gauß curvature—defined as a measure—for any rectifiable surface. For our polyhedral surface, this measure consists of point masses at vertices. Surfaces can also be built from intrinsically flat pieces joined along curved edges. Their Gauß curvature is spread out with a linear density along these edges. This technique is often used in designing clothes, where corners would be undesirable.)

Note that K_p is clearly an intrinsic notion, as it should be, depending only on the angles of each triangle and not on the precise embedding into $\mathbb{R}^3$. Sometimes it is useful to have a notion of combinatorial curvature, independent of all geometric information. Given just a combinatorial triangulation, we can pretend that each triangle is equilateral with angles $\theta = 60^\circ$, whether or not that geometry could be embedded in space. The resulting *combinatorial curvature* is $K_p = \frac{\pi}{3}(6 - \deg p)$. In this context, the global form $\sum K_p = 2\pi\chi(M)$ of Gauß–Bonnet amounts to nothing more than the definition of the Euler characteristic χ.

2.2 Vector area

The vector area formula

$$\mathbf{A}_\gamma := \tfrac{1}{2} \oint_\gamma \mathbf{x} \times d\mathbf{x} = \iint_D \nu\, dA$$

needs no special interpretation for discrete surfaces: both sides of the equation make sense directly, since the surface normal ν is well-defined almost everywhere. However, it is worth interpreting this formula for the case when D is the star of a vertex p. More generally, suppose γ is any closed curve (smooth or polygonal), and D is the cone from p to γ (the union of all line segments pq for $q \in \gamma$). Fixing γ and letting p vary, we find that the volume enclosed by this cone is a linear function of p, and $\mathbf{A}_p := \nabla_p \operatorname{Vol} D = \mathbf{A}/3 = \frac{1}{6} \oint_\gamma \mathbf{x} \times d\mathbf{x}$. We also note that any such cone D is intrinsically flat except at the cone point p, and that $2\pi - K_p$ is the cone angle at p.

2.3 Mean curvature

The mean curvature of a discrete surface M is supported along the edges. If e is an edge, and $e \subset D \subset \mathrm{Star}(e) = T_1 \cup T_2$, then

$$\mathbf{H}_e := \iint_D \mathbf{H}\, dA = \oint_{\partial D} \eta\, ds = e \times \nu_1 - e \times \nu_2 = J_1 e - J_2 e.$$

Here ν_i is the normal vector to the triangle T_i, and J_i is rotation by 90° in the plane of that triangle. Note that $|\mathbf{H}_e| = 2|e| \sin \frac{\theta_e}{2}$ where θ_e is the exterior dihedral angle along the edge, defined by $\cos\theta_e = \nu_1 \cdot \nu_2$.

No nonplanar discrete surface has $\mathbf{H}_e = 0$ along every edge. But this discrete mean curvature can cancel out around the vertices. We set

$$2\mathbf{H}_p := \sum_{e \ni p} \mathbf{H}_e = \iint_{\mathrm{Star}(p)} \mathbf{H}\, dA = \oint_{\mathrm{Link}(p)} \eta\, ds.$$

The area of the discrete surface is a function of the vertex positions; if we vary only one vertex p, we find that $\nabla_p \operatorname{Area}(M) = -\mathbf{H}_p$.

Suppose that vertices adjacent to p are $p_1, \ldots, p_n$. Then we have

$$\begin{aligned} 3\mathbf{A}_p = 3\nabla_p \operatorname{Vol} &= \iint_{\mathrm{Star}\, p} \nu\, dA \\ &= \tfrac{1}{2} \oint_{\mathrm{Link}\, p} \mathbf{x} \times d\mathbf{x} = \tfrac{1}{2} \sum_i p_i \times p_{i+1} \end{aligned}$$

and similarly

$$\begin{aligned} 2\mathbf{H}_p = \sum \mathbf{H}_{pp_i} = -2\nabla_p \operatorname{Area} &= \sum J_i(p_{i+1} - p_i) \\ &= \sum_i (\cot\alpha_i + \cot\beta_i)(p - p_i), \end{aligned}$$

where α_i and β_i are the angles opposite edge pp_i in the two incident triangles.

Note that if we change the combinatorics of a discrete surface M by introducing a new vertex p along an existing edge e, and subdividing the two incident triangles, then $\mathbf{H}_p$ in the new surface equals the original $\mathbf{H}_e$, independent of where along e we place p. This allows a variational interpretation of $\mathbf{H}_e$.

2.4 Minkowski mixed volumes

A somewhat different interpretation of mean curvature for convex polyhedra is suggested by Minkowski's theory of mixed volumes (which actually dates in this form well earlier). If X is a smooth convex body in $\mathbb{R}^3$ and $B_t(X)$ denotes its t-neighborhood, then

$$\operatorname{Vol}(B_t(X)) = \operatorname{Vol} X + t \operatorname{Area} X + \frac{t^2}{2} \int_X H\, dA + \frac{t^3}{3} \int_X K\, dA.$$

Here, the last integral is always 4π.

When X is instead a convex polyhedron, the only term that needs a new interpretation is $\int_X H\, dA$. The correct replacement for this term is then $\sum_e |e|\, \theta_e$. This suggests $H_e := |e|\, \theta_e$ as a notion of total mean curvature for the edge e.

We note the difference between this formula and our earlier $|\mathbf{H}_e| = 2|e| \sin\theta_e/2$. Either one can be derived by replacing the edge e with a sector of a cylinder of length $|e|$ and arbitrary (small) radius r. We find then

$$\iint \mathbf{H}\, dA = \mathbf{H}_e, \qquad \iint H\, dA = H_e.$$

The difference is explained by the fact that one formula integrates the scalar mean curvature while the other integrates the vector mean curvature.

2.5 CMC surfaces and Willmore surfaces

A smooth surface which minimizes area under a volume constraint has constant mean curvature; the constant H can be understood as the Lagrange multiplier for the constrained minimization problem. A discrete surface which minimizes area among surfaces of fixed combinatorial type and fixed volume will have constant discrete mean curvature H in the sense that at every vertex, $\mathbf{H}_p = H\mathbf{A}_p$, or equivalently $\nabla_p \operatorname{Area} = -H\nabla_p \operatorname{Vol}$. In general, of course, the vectors $\mathbf{H}_p$ and $\mathbf{A}_p$ are not even parallel: they give two competing notions of a normal vector at p.

Still,

$$h_p := \frac{|\nabla_p \operatorname{Area}|}{|\nabla_p \operatorname{Vol}|} = \frac{|\mathbf{H}_p|}{|\mathbf{A}_p|} = \frac{|\iint_{\operatorname{Star} p} \mathbf{H}\, dA|}{|\iint_{\operatorname{Star} p} \nu\, dA|}$$

gives a better notion of mean curvature near p than, say, the smaller quantity $3|\mathbf{H}_p|/\operatorname{Area}(\operatorname{Star}(p)) = |\iint \mathbf{H}\, dA|/\iint 1\, dA$.

For this reason, a good discretization of the Willmore elastic energy $\iint H^2\, dA$ is given by $\sum_p h_p^2 \frac{1}{3} \operatorname{Area}(\operatorname{Star}(p))$.

2.6 Relation to discrete harmonic maps

Discrete minimal surfaces minimize area, but also have other properties similar to those of smooth minimal surfaces. For instance, in a conformal parameterization, their coordinate functions are harmonic. We don't know when in general a discrete map should be considered conformal, but the identity map is certainly conformal. We have that M is discrete minimal if and only if $\mathrm{Id} : M \to \mathbb{R}^3$ is discrete harmonic. Here a PL map $f : M \to N$ is called discrete harmonic if it is a critical point for the Dirichlet energy $E(f) := \sum_T |\nabla f_T|^2 \operatorname{Area}_M(T)$. We find that $E(f) - \operatorname{Area} N$ is a measure of nonconformality. For the identity map, $E(\mathrm{Id}_M) = \operatorname{Area}(M)$ and $\nabla_p E(\mathrm{Id}_M) = \nabla_p \operatorname{Area}(M)$ confirming that M is minimal if and only if Id_M is harmonic.

Chapter 4:
A Discrete Model of Thin Shells

Eitan Grinspun
Columbia University

Figure 1: A measure of discrete strain is used to fracture a thin shell in this simulation of a shattering lightbulb.

Abstract

We describe a discrete model for the dynamics of thin flexible structures, such as hats, leaves, and aluminum cans, which are characterized by a curved undeformed configuration. Previously such *thin shell* models required complex continuum mechanics formulations and correspondingly complex algorithms. We show that a simple shell model can be derived geometrically for triangle meshes and implemented quickly by modifying a standard cloth simulator. Our technique convincingly simulates a variety of curved objects with materials ranging from paper to metal, as we demonstrate with several examples including a comparison of a real and simulated falling hat.

This chapter is based on the publication by Eitan Grinspun, Anil Hirani, Mathieu Desbrun, and Peter Schröder which appeared in the Proceedings of the Symposium for Computer Animation 2003*, and on subsequent collaborations with the group of Denis Zorin at New York University and Zoë Wood at Cal Poly San Luis Obispo.*

1 Introduction

Thin shells are thin flexible structures with a high ratio of width to thickness (> 100) [Ciarlet 2000]. While their well-known counterparts, thin *plates*, relax to a *flat* shape when unstressed, thin *shells* are characterized by a *curved undeformed configuration*. Cloth, recently studied in the computer animation literature, may be modeled as a thin plate, since garments are typically constructed from flat textiles. In stark contrast, thin-walled objects which are naturally curved (*e.g.*, leaves, fingernails), or put into that shape through plastic deformation (*e.g.*, hats, cans, carton boxes, pans, car bodies, detergent bottles) are thin shells and cannot be modeled using plate formulations.

Thin shells are remarkably difficult to simulate. Because of their degeneracy in one dimension, shells do not admit to straightforward tessellation and treatment as three-dimensional solids; indeed, the numerics of such approaches become catastrophically ill-conditioned, foiling numerical convergence and/or accuracy. Robust finite element methods for thin shell equations continue to be an active and challenging research area.

In this chapter we develop a simple model for thin shells with applications to computer animation. Our *discrete model* of shells captures the same characteristic behaviors as more complex models, with a surprisingly simple implementation. We demonstrate the realism of our approach through various examples including comparisons with real world footage (see Figure 4).

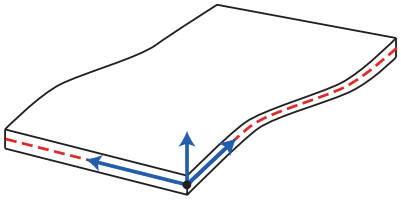


Figure 2: The local coordinate frame in a small neighborhood of a thin shell: two axes span the *middle surface*, and the normal *shell director* spans the thickness.

2 Kinematics

Since it is thin, the geometry of the shell is well described by its *middle-surface* (see Figure 2). At any point on the middle surface the local tangent plane and surface normal induce a coordinate frame in which to describe "motion along the surface" and "motion along thickness."

In the discrete setting, the topology of the middle surface is represented by the combinatorics of an oriented 2-manifold simplicial complex, $M = \{v, e, f\}$, where $v = \{v_1, v_2, \ldots\}$, $e = \{e_1, e_2, \ldots\}$, $f = \{f_1, f_2, \ldots\}$ are sets of vertices, edges and faces respectively. The geometry of the middle surface is given by the discrete *configuration map*, $C : v \mapsto \mathbb{R}^3$, which assigns to every vertex, v_i, a position, $C(v_i)$, in the ambient space. Together M and C correspond to the usual notion of a *triangle mesh* in $\mathbb{R}^3$; in our exposition we assume fixed combinatorics M, and discuss a temporally evolving configuration, C_t where the subscript refers to a specific instant in time.

Restricting our attention to elastic ("memory-less") materials, the physics can be understood in terms of the *undeformed* configuration (the stress-free shape) and the *deformed* configuration (the stressed shape at the current instant in time), C_0 and C_1, respectively. The elastic response of a material depends on the *change in*

Figure 3: Frames from the simulation of tumbling thin shell.

shape of the elastic body, and on the *constitutive laws* that describe the restoring forces associated to this change in shape. The former is a purely geometric quantity.

What is the change in shape between C_0 and C_1? Since rigid motions (translations and rotations) do not affect shape, the answer must be invariant under composition of C_0 (likewise C_1) with any rigid body transformation. A simple theorem is that any reasonable measure of change in shape, or *generalized strain*, may be written as a function of only the edge lengths and dihedral angles of C_0 and C_1. The proof lies in showing that the configuration can be completely recovered from the edge lengths and dihedral angles, up to an unknown (but here inconsequential) rigid body transformation. We will also expect our measure of strain to be zero when shape has not changed, and non-zero whenever shape has changed. In particular, strain should "see" *any local change in shape*.

Perhaps the simplest forms of generalized strain which satisfy these desiderata are two expressions that are evaluated at a specific edge e_i. Comparing C_0 to C_1, let $s^e(e_i)$ be the difference in length of edge e_i, and let $s^\theta(e_i)$ be the difference in dihedral angle at e_i.

While these are perhaps the simplest possible measures of generalized strain, other more complex formulas can offer attendant advantages. Recent research in discrete shell models has focused on functions evaluated over mesh faces which aggregate in one term the configuration of all the incident edge lengths and dihedral angles [Gingold et al. 2004]. Nevertheless, our goal here is to develop the simplest discrete model of thin shells capturing their qualitative elastic behavior.

3 Constitutive Model

Having defined the *geometry* of thin shells, we turn our attention to the governing *physical* equations. The *stored elastic energy* of a thin shell is at the heart of the equations which govern its response to elastic deformations. The stored energy, $W[C_0, C_1]$, should be a function of the local strain, integrated over the middle surface.

We choose the simplest expression for energy that is consistent with Hookean mechanics. In 1676 Robert Hooke stated

> The power [*sic.*] of any springy body is in the same proportion with the extension.

This statement was the birth of modern elasticity, which states that a first order approximation for the response of a material is a force proportional to strain, and consequently (by the definition of work as force over distance) that the first approximation of stored energy is quadratic in strain. We propose an energy with two kinds of terms, measuring stretching and bending modes respectively:

$$W_M[C_0, C_1] = \sum_{e_i \in M} k_i^e \cdot \left(s^e(e_i)\right)^2 + \sum_{e_k \in M} k_k^\theta \cdot \left(s^\theta(e_k)\right)^2 ,$$

This expression has several desirable properties. First, it is positive whenever the shapes of C_0 and C_1 differ, and zero otherwise. Second, evaluations over subsets of M satisfy the usual inclusion/exclusion principle: for $A, B \subset M$, $W_M = W_A + W_B - W_{A \cap B}$, which is consistent with continuum formulations in which energy is defined as an integral of energy density over the middle surface. Third, because strain is invariant under rigid body transformations of the undeformed and deformed configurations, Nöther's theorem guarantees that the resulting dynamics will conserve linear and angular momentum. Consider the following interpretations of the membrane and bending terms:

Membrane Elastic surfaces resist stretching (local change in length). While some materials such as rubber sheets may undergo significant deformations in the stretching or shearing (*membrane*) modes, we focus on inextensible shells which are characterized by nearly *isometric* deformations, *i.e.*, possibly significant deformations in bending but unnoticeable deformation in the membrane modes. Most membrane models for triangle meshes satisfy this small-membrane-strain assumption with choice of suitably large membrane stiffness coefficient, k_i^e.

Rewriting the membrane term in the following form permits an alternative interpretation:

$$W^e(e_i) = k^e \left(|e_i| - |\bar{e}_i|\right)^2 = k^e |\bar{e}_i|^2 \left(\frac{|e_i|}{|\bar{e}_i|} - 1\right)^2$$

where $|e_i|$ is the length of edge i, quantities with a bar (such as $\bar{e}_i$) refer to the undeformed configuration C_0 and remaining quantities are with respect to C_1; note that we have dropped the subscript on k_i^e indicating a uniform material stiffness over the domain of interest. This is a unitless strain measurement, squared, and then integrated over the area of the local neighborhood, and multiplied by the material-dependent parameters. Observe that under regular refinement of a triangle mesh, the local area indeed scales as $|\bar{e}_j|^2$, which has units of area. The units of the material parameters are energy per unit area, *i.e.*, surface energy density. In engineering models of shells, the material parameter is given as a volume energy density, and the energy is integrated over shell thickness yielding a surface energy density. Efficient implementations of this formula precompute the quantities $k^e|\bar{e}_i|^2$, which depend only on the undeformed configuration.

Bending Consider the proposed discrete bending energy in relation to its continuous analogues. Models in mechanics are typically based on tensors, and in particular shell models use the difference of the second fundamental forms [Gray 1998] in the deformed and undeformed configurations (pulling back the deformed tensor onto the undeformed configuration). These treatments derive tensorial expressions over smooth manifolds, and as a final step discretize to carry out the numerics.

The shape operator [Gray 1998] is the derivative of the Gauss map[1]: geometrically, it measures the local curvature at a point on a smooth surface. Our bending energy is an extrinsic measure of the difference between the shape operator evaluated on the deformed and undeformed surfaces. We express this difference as the *squared difference of mean curvature*:

$$\left[\mathrm{Tr}(\varphi^* \mathrm{S}) - \mathrm{Tr}(\bar{\mathrm{S}})\right]^2 = 4(H \circ \varphi - \bar{H})^2 , \qquad (1)$$

where $\bar{\mathrm{S}}$ and S are the shape operators evaluated over the undeformed and deformed configurations respectively; likewise $\bar{H}$ and H are the mean curvatures; $\varphi^*\mathrm{S}$ is the pull-back of S onto $\bar{\Omega}$, and we use $\mathrm{Tr}(\varphi^* S) = \varphi^* \mathrm{Tr}(S) = \mathrm{Tr}(S) \circ \varphi = H \circ \varphi$ for a diffeomorphism φ. This measure is *extrinsic*: it sees only changes in the *embedding* of the surface in $\mathbb{R}^3$. Integrating (1) over the reference domain we find the continuous flexural energy $\int_{\bar{\Omega}} 4(H \circ \varphi - \bar{H})^2 d\bar{A}$. Next, we discretize this integral over the *piecewise linear mesh* that represents the shell.

We derive the discrete, integral *mean-curvature squared* operator as follows. We first partition the undeformed surface

[1] This is the map from the surface to the unit sphere, mapping each surface point to its unit surface normal.

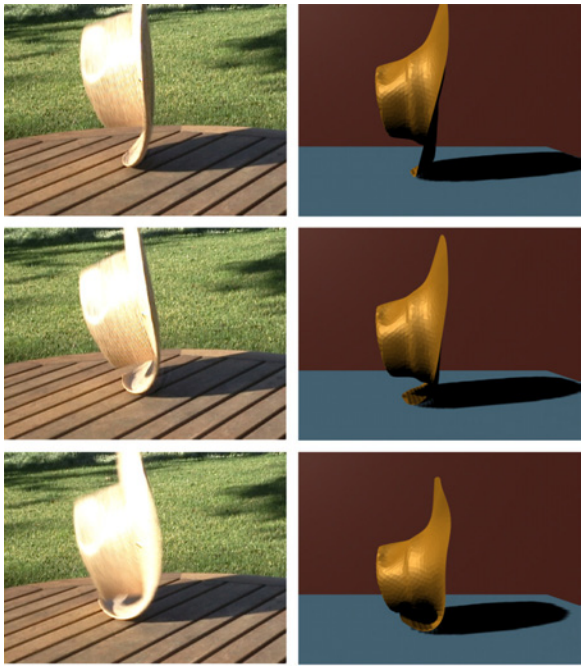
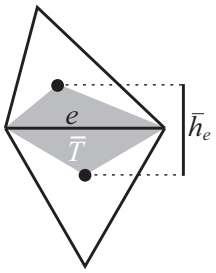


Figure 4: Real footage vs. Simulation: left, a real hat is dropped on a table; right, our shell simulation captures the bending of the brim. Notice that volumetric-elasticity, plate, or cloth simulations could not capture this behavior, while earlier work on shell simulation required significant implementation and expertise.

into a disjoint union of diamond-shaped tiles, $\bar{T}$, associated to each mesh edge, e, as indicated on the side figure. Following [Meyer et al. 2003], one can use the barycenter of each triangle to define these regions—or alternatively, the circumcenters. Over such a diamond, the mean curvature integral is $\int_{\bar{T}} \bar{H} d\bar{A} = \bar{\theta}|\bar{e}|$ (for a proof see [Cohen-Steiner and Morvan 2003]). A similar argument leads to: $\int_{\bar{T}} (H \circ \varphi - \bar{H}) d\bar{A} = (\theta - \bar{\theta})|\bar{e}|$. Using the notion of area-averaged value from [Meyer et al. 2003], we deduce that $(H \circ \varphi - \bar{H})|_{\bar{T}} = (\theta - \bar{\theta})/\bar{h}_e$, where $\bar{h}_e$ is the span of the undeformed tile, which is one sixth of the sum of the heights of the two triangles sharing $\bar{e}$. For a sufficiently fine, non-degenerate tessellation approximating a smooth surface, the average over a tile (converging pointwise to its continuous counterpart) *squared* is equal to the squared average, leading to: $\int_{\bar{T}} (H \circ \varphi - \bar{H})^2 d\bar{A} = (\theta - \bar{\theta})^2 |\bar{e}|/\bar{h}_e$.

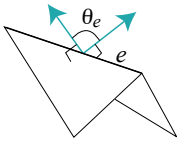


We might instead consider a formula of the form $(\theta - \bar{\theta})^2 |\bar{e}|$. Here the energy functional becomes dependent only on its piecewise planar geometry *not* on the underlying triangulation. An attractive claim, this is appealing in that a material's physical energy should depend on its shape, *not* on the discretization of the shape. Unfortunately, there is no discretization of (1) that simultaneously is (a) dependent only on the geometry *not* its triangulation, and (b) converges to its continuous equivalent under refinement. Indeed, the area integral of (1) is in general unbounded for a piecewise planar geometry! A discrete energy satisfying both (a) and (b) may exist for smoother surfaces, but our focus is piecewise planar (triangle mesh) geometry.

Following the argument found in [Meyer et al. 2003], there may be numerical advantages in using circumcenters instead of barycenters for the definition of the diamond tiles (except in triangles with obtuse angles). This affects the definition of $\bar{h}_e$ and of the lumped mass. Since we only need to compute these values for the undeformed shape, the implementation and performance *only of initialization code* would be affected. Bobenko notes that when circumcenters are used, this formulation of discrete shells coincides (for flat undeformed configurations) with the derivation of the discrete Willmore energy based on circle packing [Bobenko 2004],.

As we have just seen, we can express our *discrete flexural energy* as a summation over mesh edges,

$$W^\theta(e_k) = K^\theta \left(\theta_k - \bar{\theta}_k\right)^2 \frac{|\bar{e}_k|}{\bar{h}_k}\,, \tag{2}$$



where the term for edge e_k is where θ_k and $\bar{\theta}_k$ are corresponding complements of the dihedral angle of edge e_k measured in the deformed and undeformed configuration respectively, K^θ is the material bending stiffness, and $\bar{h}_k$ is a third of the average of the heights of the two triangles incident to the edge e_k (see the appendix for another possible definition of $\bar{h}_k$). Note that the unit of K^θ is energy (not surface energy density). This formulation is consistent with the physical scaling laws of thin shells: if the (deformed and undeformed) geometry of a thin shell is uniformly scaled by λ along each axis, then surface area scales as λ^2 as does the total membrane energy, *however* the total bending (curvature squared) energy is *invariant* under uniform scaling.

Following the reasoning for (1), we could have formed a second energy term taking the determinant instead of the trace of S. This would lead to a difference of Gaussian curvatures, but this is always zero under isometric deformations (pure bending). This is not surprising, as Gaussian curvature is an *intrinsic* quantity, *i.e.*, it is independent of the embedding of the two-dimensional surface into its ambient three-dimensional space. In contrast, flexural energy measures *extrinsic* deformations.

4 Dynamics

The treatment of the temporal evolution of a thin shell is beyond the scope of this chapter; we briefly summarize the basic components required to simulate the motion of thin shells.

Our dynamic system is governed by the ordinary differential equation of motion $\ddot{\mathbf{x}} = -\mathbf{M}^{-1} \nabla W(\mathbf{x})$ where $\mathbf{x}$ is the vector of unknown DOFs (*i.e.*, the vertices of the deformed geometry) and $\mathbf{M}$ is the mass matrix. We use the conventional simplifying hypothesis that the mass distribution is lumped at vertices: the matrix M is then diagonal, and the mass assigned to a vertex is a third of the total area of the incident triangles, scaled by the area mass density.

Newmark Time Stepping We adopt the Newmark scheme [Newmark 1959] for ODE integration,

$$\begin{aligned} \mathbf{x}_{i+1} &= \mathbf{x}_i + \Delta t_i \dot{\mathbf{x}}_i + \Delta t_i^2 \left((1/2 - \beta)\ddot{\mathbf{x}}_i + \beta \ddot{\mathbf{x}}_{i+1}\right), \\ \dot{\mathbf{x}}_{i+1} &= \dot{\mathbf{x}}_i + \Delta t_i \left((1 - \gamma)\ddot{\mathbf{x}}_i + \gamma \ddot{\mathbf{x}}_{i+1}\right), \end{aligned}$$

where Δt_i is the duration of the i^{th} timestep, $\dot{\mathbf{x}}_i$ and $\ddot{\mathbf{x}}_i$ are configuration velocity and acceleration at the beginning of the i^{th} timestep, respectively, and β and γ are adjustable parameters linked to the accuracy and stability of the time scheme. Newmark is either an explicit ($\beta = 0$) or implicit ($\beta > 0$) integrator: we used $\beta = 1/4$ for final production, and $\beta = 0$ to aid in debugging. Newmark gives control over numerical damping via its second parameter γ. We obtained the best results by minimizing numerical damping

($\gamma = 1/2$); this matches Baraff and Witkin's observation that numerical damping causes undesirable effects to rigid body motions. See also [West et al. 2000] for a discussion of the numerical advantages of the Newmark scheme.

Dissipation Shells dissipate energy via flexural oscillations. To that end we complete our model with an *optional* damping force proportional to $(\dot{\theta} - \dot{\bar{\theta}})\nabla\theta$ where $\dot{\bar{\theta}} = 0$ for elastic deformations but is in general non-zero for plastoelastic deformations. This is consistent with standard derivations of Rayleigh-type damping forces using the strain rate tensor [Baraff and Witkin 1998].

Discussion This discrete flexural energy (2) generalizes established formulations for (*flat*) plates both continuous and discrete: (a) Ge and coworkers presented a geometric argument that the stored energy of a continuous inextensible plate has the form $\int_{\bar{\Omega}} c_H H^2 + c_K K dA$ for material-specific coefficients c_H and c_K [Ge et al. 1996]; (b) Haumann used a discrete hinge energy [Haumann 1987], similarly Baraff and Witkin used a discrete constraint-based energy [Baraff and Witkin 1998], of the form $W_B(\mathbf{x}) = \sum_{\bar{e}} \theta_{\bar{e}}^2$. Our approach generalizes both (a) and (b), and produces convincing simulations beyond the regime of thin plate and cloth models (see Section 5).

The proposed discrete model has three salient features: (a) the energy is invariant under rigid body transformation of both the undeformed and the deformed shape: our system conserves linear and angular momenta; (b) the piecewise nature of our geometry description is fully captured by the purely intrinsic membrane terms, and the purely extrinsic bending term; most importantly, (c) it is *simple* to implement.

5 Results

We exercised our implementation on various problems, including fixed beams, falling hats, and pinned paper. Computation time, on a 2GHz Pentium 4 CPU, ranged from 0.25s–3.0s per frame; timings are based on a research implementation that relies on automatic differentiation techniques.

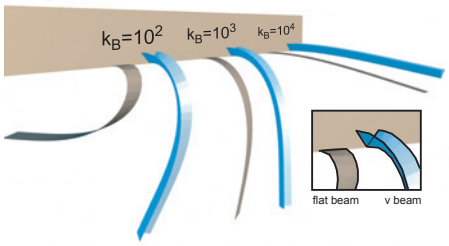


Figure 5: Three pairs of flat and v-beams with increasing flexural stiffness K^θ (left to right) of 100, 1000, and 10000; the non-flat cross section of the v-beam contributes to structural rigidity.

Beams We pinned to a wall one end of a v-beam, and released it under gravity. Figure 5 demonstrates the effect of varying flexural stiffness on oscillation amplitude and frequency. The flexural energy coefficient has a high dynamic range; extreme values (from pure-membrane to near-rigid) remain numerically and physically well-behaved. Observe that increasing flexural stiffness augments structural rigidity. Compare the behavior of beams: the non-flat cross section of the v-beam contributes to structural rigidity. This difference is most pronounced in the operating regime of low flexural stiffness (but high membrane stiffness). Here the material does not inherently resist bending, but a V-shaped cross-section effectively converts a bending deformation into a stretching deformation.

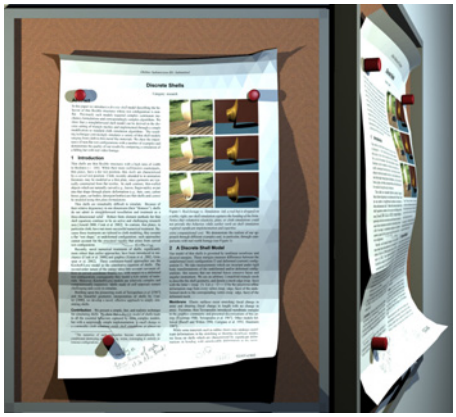


Figure 6: Modeling a curled, creased, and pinned sheet of paper: by altering dihedral angles of the reference configuration, we effect plastic deformation. While the rendering is texture-mapped we kept flat-shaded triangles to show the underlying mesh structure.

One can mimic this experiment by holding a simple paper strip by its end; repeat after folding a v-shaped cross-section.

Elastic hats We dropped both real and virtual hats and compared (see Figure 4): the deformation is qualitatively the same, during impact, compression, and rebound. Adjusting the damping parameter, we capture or damp away the brim's vibrations. Adjusting the flexural stiffness, we can make a hat made of hard rubber or textile of a nearly-rigid hat and a floppy hat).

Plastoelasticity As discussed in the early work of Kergosien and coworkers, a compelling simulation of paper would require a mechanical shell model [Kergosien et al. 1994]. Using our simple shell model, we can easily simulate a sheet of paper that is rolled, then creased, then pinned (see Figure 6). Here the physics require *plastic* as well as elastic deformations. We begin with a flat surface, and gradually increase the undeformed angles, $\bar{\theta}_e$. Notice: modifying the *undeformed* configuration effects a *plastic* deformation. The kinematics of changing $\bar{\theta}_e$ span only physically-realizable bending, *i.e.*, *inextensible* plastic deformations. In contrast, directly modifying $\bar{\mathbf{x}}$ could introduce plastic deformations with unwanted membrane modes. We introduced elastic effects by applying three pin constraints to the *deformed* configuration. Observe the half-crease on the left side. The energy of the undeformed state is no longer zero! The (plastically-deformed) left and (untouched) right halves have incompatible undeformed shapes, consequently the undeformed configuration is *not* stress-free.

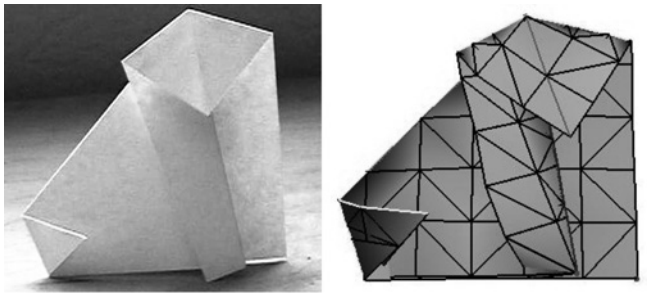


Figure 7: Virtual origami: user-guided simulated folding of a paper sheet produces a classical origami dog.

Recent extensions More recently, we demonstrated that simple, discrete models of thin shells can also produce striking examples of shattering glass (see Figure 1) [Gingold et al. 2004], and paper origami (see Figure 7) [Burgoon et al. 2006].

Implementation An attractive practical aspect of the proposed model is that it may be easily incorporated into working code of a standard cloth or thin-plate simulator such as those commonly used by the computer graphics community [Baraff and Witkin 1998]. One must replace the bending energy with (2). From an implementation point of view, this involves minimal work. For example, consider that [Baraff and Witkin 1998] already required all the computations relating to θ_e. These and other implementation details were outlined in [Grinspun et al. 2003].

6 Further Reading

A comprehensive survey of this expansive body of literature is far beyond the scope of this chapter; as a starting point see [Arnold 2000; Cirak et al. 2002] and references therein. Here we highlight only a few results from the graphics and engineering literature.

Recently, novel numerical treatments of shells, significantly more robust than earlier approaches, have been introduced in mechanics by Cirak *et al.* [Cirak et al. 2000] and in graphics by Green *et al.*and Grinspun *et al.* [Green et al. 2002; Grinspun et al. 2002] among others. These continuum-based approaches use the Kirchoff-Love constitutive equations, whose energy captures curvature effects in curved coordinate frames; consequently they model a rich variety of materials. In contrast, thin *plate* equations assume a flat undeformed configuration. Thin plate models are commonly used for cloth and garment simulations and have seen successful numerical treatment in the computer graphics literature (see [House and Breen 2000] and references therein). Thin plates have also been useful for variational geometric modeling [Celniker and Gossard 1991; Greiner 1994; Welch and Witkin 1992] and intuitive direct manipulation of surfaces [Qin and Terzopoulos 1996; Terzopoulos and Qin 1994]. In graphics, researchers have used two kinds of approaches to modeling plates: finite-elements and mass-spring networks. In the latter resistance to bending is effected by springs connected to opposite corners of adjacent mesh faces. Unfortunately, this simple approach does not carry over to curved undeformed configurations: the diagonal springs are insensitive to the sign of the dihedral angles between faces.

In this chapter we have developed a very simple discrete model of thin shells. One price that must be paid for this simplicity is that, while we have taken care to ensure the correct scaling factors for each energy term, for an arbitrary triangle mesh we cannot guarantee the convergence of this model to its continuum equivalent. In [Grinspun et al.] we present experimental results comparing the convergence of the discrete shell and other discrete curvature operators.

7 Acknowledgments

The work described here is the fruit of a collaboration with Mathieu Desbrun, Anil Hirani, and Peter Schröder [Grinspun et al. 2003]. Recent work on modeling thin shells is part of an active collaboration with Miklos Bergou, Rob Burgoon, Akash Garg, David Harmon, Adrian Secord, Yotam Gingold, Jason Reisman, Max Wardetzky, Zoë Wood, and Denis Zorin. The author is indebted to Alexander Bobenko, Jerry Marsden, and Anastasios Vayonakis for insightful discussions. Pierre Alliez, Ilja Friedel, Harper Langston, Anthony Santoro and Steven Schkolne were pivotal in the production of the images shown here. The author is grateful for the generous support provided by the National Science Foundation (MSPA-MCS 0528402), Elsevier, and nVidia.

References

ARNOLD, D., 2000. Questions on shell theory. Workshop on Elastic Shells: Modeling, Analysis, and Computation. Mathematical Sciences Research Institute, Bekeley.

BARAFF, D., AND WITKIN, A. 1998. Large steps in cloth simulation. In *Proceedings of SIGGRAPH*, 43–54.

BOBENKO, A. I. 2004. A Conformal Energy for Simplicial Surfaces. Published online at http://arxiv.org/abs/math.DG/0406128, August.

BURGOON, R., GRINSPUN, E., AND WOOD, Z. 2006. Discrete shells origami. In *Proceedings of CATA*.

CELNIKER, G., AND GOSSARD, D. 1991. Deformable Curve and Surface Finite Elements for Free-Form Shape Design. *Computer Graphics (Proceedings of SIGGRAPH 91) 25*, 4, 257–266.

CIARLET, P. 2000. *Mathematical Elasticity. Vol. III*, vol. 29 of *Studies in Mathematics and its Applications*. Amsterdam. Theory of shells.

CIRAK, F., ORTIZ, M., AND SCHRÖDER, P. 2000. Subdivision surfaces: A new paradigm for thin-shell finite-element analysis. *Internat. J. Numer. Methods Engrg. 47*, 12, 2039–2072.

CIRAK, F., SCOTT, M., ANTONSSON, E., ORTIZ, M., AND SCHRÖDER, P. 2002. Integrated modeling, finite-element analysis, and engineering design for thin-shell structures using subdivision. *CAD 34*, 2, 137–148.

COHEN-STEINER, D., AND MORVAN, J.-M. 2003. Restricted delaunay triangulations and normal cycle. In *Proc. 19th Annu. ACM Sympos. Comput. Geom.*, 237–246.

GE, Z., KRUSE, H. P., AND MARSDEN, J. E. 1996. The limits of hamiltonian structures in three-dimensional elasticity, shells, and rods. *Journal of Nonlinear Science 6*, 19–57.

GINGOLD, Y., SECORD, A., HAN, J. Y., GRINSPUN, E., AND ZORIN, D. 2004. Poster: A discrete model for inelastic deformation of thin shells. In *ACM/Eurographics Symposium on Computer Animation '04*.

GRAY, A. 1998. *Modern Differential Geometry of Curves and Surfaces*. Second edition. CRC Press.

GREEN, S., TURKIYYAH, G., AND STORTI, D. 2002. Subdivision-based multilevel methods for large scale engineering simulation of thin shells. In *Proceedings of ACM Solid Modeling*, 265–272.

GREINER, G. 1994. Variational design and fairing of spline surfaces. *Computer Graphics Forum 13*, 3, 143–154.

GRINSPUN, E., GINGOLD, Y., REISMAN, J., AND ZORIN, D. Computing discrete shape operators on general meshes. submitted for publication.

GRINSPUN, E., KRYSL, P., AND SCHRÖDER, P. 2002. CHARMS: a simple framework for adaptive simulation. *ACM Transactions on Graphics 21*, 3 (July), 281–290.

GRINSPUN, E., HIRANI, A., DESBRUN, M., AND SCHRÖDER, P. 2003. Discrete shells. In *ACM SIGGRAPH Symposium on Computer Animation*. to appear.

HAUMANN, R. 1987. Modeling the physical behavior of flexible objects. In *Topics in Physically-based Modeling,* Eds. Barr, Barrel, Haumann, Kass, Platt, Terzopoulos, and Witkin*, SIGGRAPH Course Notes*.

HOUSE, D. H., AND BREEN, D. E., Eds. 2000. *Coth Modeling and Animation*. A.K. Peters.

KERGOSIEN, Y. L., GOTODA, H., AND KUNII, T. L. 1994. Bending and creasing virtual paper. *IEEE Computer Graphics and Applications*, 40–48.

MEYER, M., DESBRUN, M., SCHRÖDER, P., AND BARR, A. H. 2003. Discrete differential-geometry operators for triangulated 2-manifolds. In *Visualization and Mathematics III*, H.-C. Hege and K. Polthier, Eds. Springer-Verlag, Heidelberg, 35–57.

NEWMARK, N. M. 1959. A method of computation for structural dynamics. *ASCE J. of the Engineering Mechanics Division 85*, EM 3, 67–94.

QIN, H., AND TERZOPOULOS, D. 1996. D-NURBS: A physics-based framework for geometric design. *IEEE Transactions on Visualization and Computer Graphics 2*, 1, 85–96.

TERZOPOULOS, D., AND QIN, H. 1994. Dynamic NURBS with Geometric Constraints for Interactive Sculpting. *ACM Transactions on Graphics 13*, 2, 103–136.

WELCH, W., AND WITKIN, A. 1992. Variational Surface Modeling. *Computer Graphics (Proceedings of SIGGRAPH 92) 26*, 2, 157–166.

WEST, M., KANE, C., MARSDEN, J. E., AND ORTIZ, M. 2000. Variational integrators, the newmark scheme, and dissipative systems. In *International Conference on Differential Equations 1999*, World Scientific, Berlin, 1009 – 1011.

Chapter 5:
Simple and Efficient Implementation of Discrete Plates and Shells

Max Wardetzky, University of Göttingen
Miklós Bergou, Columbia University
Akash Garg, Columbia University
David Harmon, Columbia University
Denis Zorin, New York University
Eitan Grinspun, Columbia University

Figure 1: Snapshots from our simulation of a billowing flag. Despite its economy of cost, the proposed bending model achieves qualitatively the same dynamics as popular nonlinear models.

Abstract

Efficient computation of curvature-based energies is important for practical implementations of geometric modeling and physical simulation applications. Building on a simple geometric observation, we propose a hinge-based bending model that is simple to implement, efficient to compute, and offers a great number of effective material parameters. Our formulation builds on two mathematical observations: (a) the bending energy of a thin flexible plate (resp. shell) can be expressed as a *quadratic (resp. cubic)* polynomial of surface positions provided that the surface does not stretch; (b) a general class of anisotropic materials—those that are *orthotropic*—is captured by appropriate choice of a single stiffness per hinge. We provide two approaches for deriving our isometric bending model (IBM): a purely geometric view and a derivation based on finite elements. By offering a highly efficient treatment of force Jacobians, our model impacts the speed of a general range of surface animation applications, from isotropic cloth and thin plates, over orthotropic fracturing of thin shells, to Willmore-type surface fairing.

The present notes are condensed from previous articles by the authors: [Bergou et al. 2006], [Garg et al. 2007], and [Wardetzky et al. 2007]—augmented by a finite element treatment that was not present in these earlier works.

1 Introduction

Many animation applications require simple and efficient simulation of a general class of elastic surfaces. This class includes objects that are (a) flat (plates) or curved (shells) in their undeformed state, (b) flexible or nearly rigid and (c) isotropic or anisotropic in their response to bending. In mesh-based simulation, *hinge-based* methods are preferred for their simplicity and economy of computation. Considering every two triangles meeting at an edge to be a bending hinge, such methods require that some function of the dihedral angle is subject to a restoring force. Empirically, hinge-based models are known to work well for isotropic materials, and for geometric models of anisotropy based on Euler's curvature formula [Baraff and Witkin 1998; Volino and Magnenat-Thalmann 2006]. For a survey of bending models used in animation, see [Thomaszewski and Wacker 2006].

Nearly Inextensible Surfaces In this chapter we explore the interrelation between isometry, differential operators, and curvature energy. Specifically, we present what seems to be the simplest and most efficient hinge-based bending model to encompass the spectrums (a), (b), and (c), at the cost of restricting our attention to nearly *inextensible* surfaces, *i.e.*, surfaces that prefer to bend much more than to stretch. The importance of isometry for simplification of energy was previously acknowledged in the context of surface fairing and modeling, *e.g.*, by Desbrun *et al.* [1999] and later Schneider *et al.* [2001]. We focus primarily on the simulation of inextensible plates and shells where physics dictates quasi-isometry: membrane stiffness typically is greater than bending stiffness by four or more orders of magnitude (cf. Koiter's model [Ciarlet 2000]) and the advantages of isometry can be exploited in full.

First, we introduce a hinge-based bending model that preserves a key property of the smooth setting: the bending energy of a thin plate (resp. shell) is *quadratic (resp. cubic)* under isometric deformations. To understand the consequences of this statement, consider that bending energy is in general a highly nonlinear functional of the surface position; therefore, the implementation of implicit solvers for thin shells involves relatively complex derivation and costly computation of the nonlinear bending force Jacobian. However, under the assumption of quasi-isometry, we show that implementation can be reduced to a one-time precomputation of an approximate Jacobian matrix, and implicit time stepping routines can use an inexact Newton method for significant performance gains. Shells simulated with this method conserve both linear and angular momenta.

Second, we treat anisotropy, where mechanical response depends on the direction of applied strain. We present a bending model for *orthotropic* materials, exposing an important physical parameter—the *shear modulus*—which affects the *drapability* of a fabric [Sidabraite and Masteikaite 2002]. Perhaps due to their small stencil, hinge-based models in computer animation have eluded the incorporation of orthotropic response until recently.

Overview By first considering *isotropic* bending, we expose the quadratic and cubic nature of bending energies of smooth plates and shells, respectively (§2.1). We then take this analysis to the discrete settings (§2.2), and establish the *quadratic and cubic hinge* as our discrete building block. We offer two parallel views: a purely geometric one (§3.1) and a derivation based on finite elements (§3.2). Next, we introduce *orthotropy* (§4) and show that it is captured by a scalar hinge stiffness (§4.1). Finally, we describe the efficient implementation of our model (§5), and discuss a battery of experiments demonstrating its efficiency and efficacy (§6–§8).

2 Isotropic Bending

2.1 The Isometric Bending Model (IBM)

Consider a smooth surface deformed away from its natural (*undeformed*) shape. To this deformation we associate the *isotropic* bending energy

$$E_{\mathrm{b}}(\mathbf{x}) = \frac{1}{2}\int_S (H-\bar{H})^2 \mathrm{d}A\,. \tag{1}$$

This is the integral, over the undeformed surface, of the squared change in mean curvature. Here $\mathbf{x}$ denotes the position of the deformed surface, *i.e.*, $\mathbf{x}(p)$ is a 3-vector for each point $p \in S$, and H is the mean curvature function of the deformed surface. A bar (*e.g.*, $\bar{H}$) denotes the corresponding quantity evaluated on the undeformed surface. The special case of flat undeformed shapes, or *plates*, corresponds to $\bar{H} = 0$.

Although not immediately apparent from (1), $E_{\mathrm{b}}(\mathbf{x})$ is actually a *cubic polynomial* in $\mathbf{x}$ under *isometric deformations*, *i.e.*, if the surface is allowed to bend but not to stretch. For the special case of $\bar{H} = 0$, the energy is a *quadratic polynomial*. To see this, rewrite (1) as

$$E_{\mathrm{b}}(\mathbf{x}) = \frac{1}{2}\int_S \left(\langle \mathbf{H},\mathbf{H}\rangle - 2\langle \mathbf{H}, \bar{H}\hat{\mathbf{n}}\rangle + \bar{H}^2\right)\mathrm{d}A\,. \tag{2}$$

Here $\langle\cdot,\cdot\rangle$ denotes the standard inner product in 3-space, and $\mathbf{H} = H\hat{\mathbf{n}}$ stands for the mean curvature *normal*, a vector parallel to the surface's unit normal, $\hat{\mathbf{n}}$. The mean curvature vector can be expressed as

$$\mathbf{H} = \Delta\mathbf{x}\,, \tag{3}$$

the surface's intrinsic Laplacian applied to the (3-component) position of the surface. For isometric deformations of the surface, two facts follow: (F1) Since Δ is intrinsic, and therefore *invariant* under isometric surface deformations, $\mathbf{H}$ is *linear in surface position*, thus $\langle \mathbf{H},\mathbf{H}\rangle$ is quadratic in $\mathbf{x}$, as observed in [Bergou et al. 2006]. (F2) The surface normal is quadratic[1] in $\mathbf{x}$; therefore, $\langle \mathbf{H}, \bar{H}\hat{\mathbf{n}}\rangle$ is cubic. It follows that $E_{\mathrm{b}}(\mathbf{x})$ is a cubic polynomial in $\mathbf{x}$, because $\langle \mathbf{H}, \bar{H}\hat{\mathbf{n}}\rangle$ is cubic, $\langle \mathbf{H},\mathbf{H}\rangle$ is quadratic, and $\bar{H}^2$ does not depend on $\mathbf{x}$. While (F1) applies to both thin plates and shells, (F2) is unique to shells ($\bar{H} \neq 0$).

2.2 The Discrete IBM

From a differential geometry point of view, the above observations offer no surprises. However, they have a number of very practical consequences when carried over to the discrete case. In this setting one begins with a discrete notion of mean curvature or a discrete

[1] To see that $\hat{\mathbf{n}}$ is quadratic, write $\hat{\mathbf{n}} = \frac{\partial\mathbf{x}}{\partial u} \times \frac{\partial\mathbf{x}}{\partial v}$, and observe that orthonormality of tangent vectors, $\frac{\partial\mathbf{x}}{\partial u}$ and $\frac{\partial\mathbf{x}}{\partial v}$, is preserved under isometric deformations.

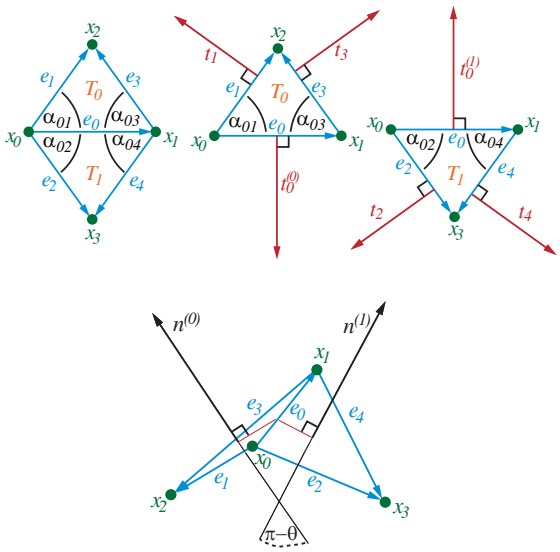


Figure 2: *Top-left:* Hinge stencil for an interior edge. *Top-middle and -right:* Perpendicular vectors used in the computation. *Bottom:* Hinge in 3-space with corresponding labels.

Laplacian if one seeks a discrete version of (1), (2), and (3). While DDG operators are extensively studied [Pinkall and Polthier 1993; Mercat 2001; Meyer et al. 2003; Bobenko 2005; Cohen-Steiner and Morvan 2006], our aim here is to seek operators that satisfy a discrete IBM: a bending energy quadratic in positions for plates and cubic for shells under the class of discrete isometric deformations. Therefore we must begin with a reasonable definition of isometry, which in the case of triangulated surfaces we take to be that (a) bending occurs only along edges, and (b) edges may not stretch. In reality, the second clause is relaxed, *i.e.*, we consider discrete quasi-isometry.

Hinge-based Bending Energy Consider a triangle mesh whose shape before deformation is given by the vector of vertex positions $\bar{\mathbf{x}}$. When the mesh is deformed to position $\mathbf{x}$, the usual hinge-based bending energy [Baraff and Witkin 1998; Bridson et al. 2003; Grinspun et al. 2003; Bergou et al. 2006; Thomaszewski and Wacker 2006] is given by[2]

$$E_{\mathrm{b}}(\mathbf{x}) = \frac{1}{2}\sum_i \frac{3|\bar{\mathbf{e}}_i|^2}{\bar{A}_i}\left(2\sin\frac{\theta_i-\bar{\theta}_i}{2}\right)^2\,. \tag{4}$$

Here the sum is taken over all interior edges, $\bar{A}_i$ denotes the combined area of the two triangles incident to edge $\bar{\mathbf{e}}_i$, and θ_i denotes the dihedral angle at edge i.

Our goal is to establish an important, and previously overlooked, property of (4): under isometric deformations, it is a cubic polynomial in $\mathbf{x}$, and in fact quadratic for the case of discrete plates ($\bar{\theta} = 0$). This observation will expose a much more efficient implementation than is directly evident from (4); for details on this implementation, refer directly to §5.

3 The Quadratic & The Cubic Hinge

The discrete energy (4) can be written as a sum over contributions from every hinge (indexed by i). Using the identity $2\sin^2 u =$

[2] For planar rest state ($\bar{\theta} = 0$) we have $\lim_{\theta\to 0} 2\sin\theta/2 = \lim_{\theta\to 0} 2\tan\theta/2 = \theta$, so that models that use $2\sin\theta/2$, $2\tan\theta/2$, or θ coincide in the limit of an appropriate refinement sequence.

$1-\cos 2u$, with $u = (\theta - \bar{\theta})/2$, we arrive at an expression of energy associated to the i^{th} hinge:

$$E_{\text{b}}(\mathbf{x})_i = \frac{3|\bar{\mathbf{e}}_i|^2}{\bar{A}_i}(1-\cos(\theta_i - \bar{\theta}_i)) \,. \tag{5}$$

In this section, we prove that for isometric deformations this hinge energy is cubic in $\mathbf{x}$. We offer two parallel proofs: a geometric one and a finite element derivation.

3.1 Geometric View

Our derivation uses the geometric construction in Figure 2, which defines *local indices* of the triangles, $\{T_0, T_1\}$, edge vectors, $\{\mathbf{e}_0, \ldots, \mathbf{e}_4\}$, and vertex positions, $\{\mathbf{x}_0, \ldots, \mathbf{x}_3\}$, associated to a hinge. Note the construction of *perpendiculars* $\mathbf{t}_i$: each edge vector $\mathbf{e}_i$ is rotated 90 degrees, on the plane of the triangle, to point outwards (thus $|\mathbf{t}_i| = |\mathbf{e}_i|$ and $\langle \mathbf{t}_i, \mathbf{e}_i \rangle = 0$). Since $\mathbf{e}_0$ is associated to two triangles, we construct two perpendiculars, $\mathbf{t}_0^{(0)}$ and $\mathbf{t}_0^{(1)}$, on the planes of triangles T_0 and T_1 respectively.

Expressing the perpendiculars $\mathbf{t}_0^{(0)}$ (resp. $\mathbf{t}_0^{(1)}$) in the edge basis $\{\mathbf{e}_1, \mathbf{e}_3\}$ (resp. $\{\mathbf{e}_2, \mathbf{e}_4\}$) we obtain

$$\mathbf{t}_0^{(0)} = -\cot\alpha_{03}\mathbf{e}_1 - \cot\alpha_{01}\mathbf{e}_3 \,, \tag{6}$$

$$\mathbf{t}_0^{(1)} = -\cot\alpha_{04}\mathbf{e}_2 - \cot\alpha_{02}\mathbf{e}_4 \,. \tag{7}$$

Under isometric deformation, interior angles remain constant, therefore $\mathbf{t}_0^{(0)}$ and $\mathbf{t}_0^{(1)}$ are *linear expressions* in the position of the mesh. Furthermore, the geometry of Fig. 2-bottom reveals that[3]

$$\cos\theta = \frac{-\langle \mathbf{t}_0^{(0)}, \mathbf{t}_0^{(1)} \rangle}{|\mathbf{e}_0|^2}, \quad \text{and} \quad \sin\theta = -\frac{\beta[\mathbf{e}_0, \mathbf{e}_1, \mathbf{e}_2]}{|\mathbf{e}_0|^2} \,,$$

where $[\mathbf{e}_i, \mathbf{e}_j, \mathbf{e}_k]$ is the scalar triple product $(\mathbf{e}_i \times \mathbf{e}_j) \cdot \mathbf{e}_k$, and $\beta = \frac{1}{|\mathbf{e}_0|}(\cot\alpha_{01} + \cot\alpha_{03})(\cot\alpha_{02} + \cot\alpha_{04})$.

Expanding (5) by the identity $\cos(\theta - \bar{\theta}) = \cos\theta\cos\bar{\theta} + \sin\theta\sin\bar{\theta}$, substituting the above expressions for $\cos\theta$ and $\sin\theta$, and simplifying the resulting expression by using the isometry assumption ($|\bar{\mathbf{e}}_i| = |\mathbf{e}_i|$) we find that (5) equals

$$\underbrace{\frac{3}{\bar{A}_0}\left(|\bar{\mathbf{e}}_0|^2 + \langle \mathbf{t}_0^{(0)}, \mathbf{t}_0^{(1)} \rangle \cos\bar{\theta}\right)}_{\text{thin plate component}} + \underbrace{\frac{3\bar{\beta}}{\bar{A}_0}[\mathbf{e}_0, \mathbf{e}_1, \mathbf{e}_2]\sin\bar{\theta}}_{\text{cubic term}} \,. \tag{8}$$

Under isometry, the energy associated to an individual hinge is a cubic polynomial in $\mathbf{x}$. We have thus established that—as in the smooth picture—the discrete bending energy (4) is cubic. Note that for the special case $\bar{\theta} = 0$, we recover the quadratic bending thin *plate* energy of [Bergou et al. 2006] simply by substituting (6) and (7) into (8).

3.2 Finite Element Derivation

Our discrete IBM may be explained in terms of so-called nonconforming *Crouzeix-Raviart* (CR) finite elements. This analysis is primarily a theoretical contribution that achieves two goals: (a) to retroactively understand the bending models of [Baraff and Witkin 1998; Bridson et al. 2003; Grinspun et al. 2003] in terms of finite elements, and (b) to provide the theoretical foundation for the anisotropic model of §4.

CR elements are non-conforming elements which are linear on triangles and have their degrees of freedom (DOFs) associated with edges. A CR basis, $\{\phi_i\}$, is given by those functions ϕ_i for which $\phi_i(j) = \delta_{ij}$, where δ_{ij} is the Kronecker delta. The corresponding stiffness and mass matrices take the respective forms

$$\mathrm{K}_{ij} = \int_S \langle \nabla\phi_i, \nabla\phi_j \rangle \mathrm{d}A \quad \text{and} \quad \mathrm{M}_{ij} = \int_S \phi_i\phi_j \mathrm{d}A \,, \tag{9}$$

where ∇ denote the gradient operator.

Similarly to (3), the *pointwise* mean curvature vector is a vector-valued function over the surface whose coefficient vector in the above CR basis is given by

$$\mathbf{H} = \mathrm{M}^{-1}\,\mathrm{K}\,\mathrm{T}\,\mathbf{x} \,, \tag{10}$$

where the linear transformation operator T takes the mesh embedding, $\mathbf{x}$, having vertex-based DOFs, to the space of CR functions having edge-based DOFs (see below). To better understand (10), consider the *weak-form formulation* of (3), *i.e.*,

$$\int_S \phi\mathbf{H}\,\mathrm{d}A = \int_S \phi\Delta\mathbf{x}\,\mathrm{d}A = \int_S \langle \nabla\phi, \nabla\mathbf{x} \rangle\,\mathrm{d}A \,, \tag{11}$$

where ϕ is a *scalar* (test) function on S and $\langle \nabla\phi, \nabla\mathbf{x} \rangle$ denotes the vector-valued function evaluated separately for each of the three components of the embedding function $\mathbf{x}$. The apparently missing minus sign in the last equality of (11) is due to our sign convention for the Laplace operator, which we require to be positive, and not negative, semi-definite. Moving to the discrete case and using the CR basis, we can write

$$\mathbf{x} = \sum_i \phi_i (\mathrm{T}\,\mathbf{x})_i \quad \text{and} \quad \mathbf{H} = \sum_i \phi_i \mathbf{H}_i \,, \tag{12}$$

where $(\mathrm{T}\,\mathbf{x})_i$ and $\mathbf{H}_i$ are edge-based coefficient vectors. Notice that our discrete mesh is given by vertex-based DOFs; therefore we need the linear transformation T to map those DOFs to edge-based ones. Consequently, if $\mathbf{x}_p$ and $\mathbf{x}_q$ denote two mesh vertices that are connected by edge $\mathbf{e}_i$, then $(\mathrm{T}\,\mathbf{x})_i = (\mathbf{x}_p + \mathbf{x}_q)/2$. Finally, by plugging (12) into (11) and using (9), we immediately arrive at (10).

Each coefficient vector in (12) corresponds to the edge-based mean curvature vector

$$\mathbf{H}_i = \frac{3|\mathbf{e}_i|}{A_i}\left(2\sin\frac{\theta_i}{2}\right)\hat{\mathbf{n}}_i \,, \tag{13}$$

where A_i is the area of $\mathbf{e}_i$'s hinge stencil as before, and $\hat{\mathbf{n}}_i$ is the angle-bisecting normal of the dihedral angle at edge $\mathbf{e}_i$. This formula is obtained from an explicit calculation of the entries of the mass and stiffness matrix in (9). In particular, the CR mass matrix is *diagonal* and satisfies $\mathrm{M}_{ii} = A_i/3$; therefore, as observed in [Wardetzky et al. 2007], the *thin plate energy* can be written as

$$E_{\text{b}}(\mathbf{x}) = \frac{1}{2}\sum_i \mathrm{M}_{ii}\,\mathbf{H}_i^2 = \sum_i \frac{A_i}{6}\mathbf{H}_i^2 \,.$$

However, without the intuition gained from the geometric picture in §3.1, it is not trivial to extend the CR viewpoint to the thin shell setting. From the smooth energy, (2), and the corresponding discrete energy, (8), it follows that one needs to treat three terms in the FE discretization. The CR formulation of the first and third terms

[3] The expression for $\sin\theta$ was derived from

$$\sin\theta = \frac{\langle \mathbf{t}_0^{(0)}, \mathbf{n}^{(1)} \rangle}{|\mathbf{e}_0|^2} = -\frac{[\mathbf{t}_0^{(0)}, \mathbf{t}_0^{(1)}, \mathbf{e}_0]}{|\mathbf{e}_0|^3} = -\frac{\beta[\mathbf{e}_0, \mathbf{e}_1, \mathbf{e}_2]}{|\mathbf{e}_0|^2} \,,$$

where $\mathbf{n}^{(1)} = (|\mathbf{e}_0|/|\mathbf{e}_4 \times \mathbf{e}_2|)\mathbf{e}_4 \times \mathbf{e}_2$ is a scaled triangle normal. The last step is obtained by using (6) and (7) to replace $\mathbf{t}_0^{(0)}$ and $\mathbf{t}_0^{(1)}$, followed by observing that $\mathbf{e}_4 = \mathbf{e}_2 - \mathbf{e}_0$ and $\mathbf{e}_3 = \mathbf{e}_1 - \mathbf{e}_0$.

follows directly from the thin plate setting:

$$\int_S H^2 \mathrm{d}A \sim \sum_i \frac{A_i}{3} \mathbf{H}_i^2 \quad \left(= \sum_i \frac{3|\mathbf{e}_i|^2}{A_i} \left(2 \sin \frac{\theta_i}{2} \right)^2 \right) , \tag{14}$$

$$\int_S \bar{H}^2 \mathrm{d}A \sim \sum_i \frac{\bar{A}_i}{3} \bar{\mathbf{H}}_i^2 \quad \left(= \sum_i \frac{3|\bar{\mathbf{e}}_i|^2}{\bar{A}_i} \left(2 \sin \frac{\bar{\theta}_i}{2} \right)^2 \right) . \tag{15}$$

The cubic term, $H\bar{H} = \langle \mathbf{H}, \bar{H}\hat{\mathbf{n}} \rangle$, requires special treatment in the discrete case: $\mathbf{H}_i$ and $\bar{\mathbf{H}}_i$ are in general not parallel vectors, thus $H\bar{H} \neq \langle \mathbf{H}_i, \bar{\mathbf{H}}_i \rangle$. We *approximate* collinearity by conceptually applying a rigid transformation to the isometrically deformed mesh such that the positions of the triangles $\bar{T}_0$ and T_0 coincide[4]. Such a rigid transformation does not alter elastic energy, but it in the limit of mesh refinement it attains $\mathbf{H}_i \| \bar{\mathbf{H}}_i$.

With this (conceptual) transformation in effect, we may evaluate $H\bar{H} \approx \langle \mathbf{H}_i, \bar{\mathbf{H}}_i \rangle = |\mathbf{H}_i||\bar{\mathbf{H}}_i| \cos \angle(\mathbf{H}_i, \bar{\mathbf{H}}_i)$. The lengths $|\mathbf{H}_i|$ and $|\bar{\mathbf{H}}_i|$ are obtained via (13). To obtain $\angle(\mathbf{H}_i, \bar{\mathbf{H}}_i)$, observe that edge-based mean curvature vectors bisect the dihedral angle of their incident flap triangles; therefore, $\angle(\mathbf{H}_i, \bar{\mathbf{H}}_i) = (\theta_i - \bar{\theta}_i)/2$. Assembling these observations, and using that under isometry we have $|\mathbf{e}_i| = |\bar{\mathbf{e}}_i|$ and $A_i = \bar{A}_i$, the cubic term is given by $H\bar{H} \approx$

$$\langle \mathbf{H}_i, \bar{\mathbf{H}}_i \rangle = \frac{9|\bar{\mathbf{e}}_i|^2}{\bar{A}_i^2} \left(2 \sin \frac{\theta_i}{2} \right) \left(2 \sin \frac{\bar{\theta}_i}{2} \right) \cos \frac{\theta_i - \bar{\theta}_i}{2} . \tag{16}$$

Together, (14), (15), and (16) simplify to $E_\mathrm{b}(\mathbf{x})_i =$

$$\frac{\bar{A}_i}{6} \left(\mathbf{H}_i^2 + \bar{\mathbf{H}}_i^2 - 2\langle \mathbf{H}_i, \bar{\mathbf{H}}_i \rangle \right) = \frac{3|\bar{\mathbf{e}}_i|^2}{2\bar{A}_i} \left(2 \sin \frac{\theta_i - \bar{\theta}_i}{2} \right)^2 ,$$

which exactly corresponds to the geometric formulation (4).

4 Orthotropic Bending

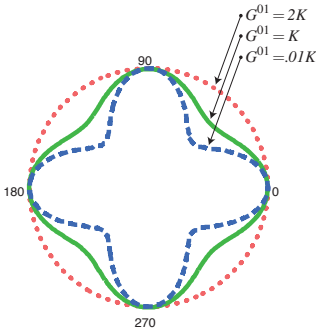
We now generalize our bending model to *orthotropic* materials—an important class of anisotropic materials whose elastic properties depend on the direction along which they are measured [Ventsel and Krauthammer 2001]. A fully general linear elasticity model for deformable surfaces has six parameters (not counting the choice of anisotropy axes) some of which are hard to interpret intuitively. We focus on a more restricted *orthotropic* elasticity model whose four parameters have more intuitive meaning and appear to be useful for material behavior control in animation. Most common man-made materials are orthotropic, for example, cloth, plastic reinforced by fibers, sheet metal, and paper. Non-orthotropic thin materials are less common (*e.g.*, thin sheets obtained by cutting a 3D orthotropic material at an angle, or composite materials). For cloth, orthotropic approximation naturally matches most of the parameters of the Kawabata cloth evaluation system [Kawabata 1980], a commonly used system for characterizing cloth properties, as explained below.

We are primarily interested in how these material parameters affect *bending*, rather than in-plane deformations. From the four parameters of orthotropic materials, one parameter can be eliminated if we assume that bending the surface along material directions does not have any effect on bending on the other direction (this corresponds to having zero Poisson ratio in the isotropic case). For simplicity we will assume that this is the case, and briefly discuss the role of the Poisson ratio at the end of this section.

[4] Existence of this transformation is ensured by the isometry assumption.

The most obvious of the remaining three parameters are two Young's moduli, Y^0 and Y^1, which determine bending stiffness along two material directions. The third parameter is the *shear modulus*, G^{01}, which, for in-plane deformations, determines the resistance to shear. In the case of bending, the shear modulus allows for additional directional variability of bending stiffness. Specifically, bending stiffness at an angle α with respect to the material axis 0, is given by (up to a scale factor):

$$Y^0 \cos^4 \alpha + Y^1 \sin^4 \alpha + 2G^{01} \sin^2 \alpha \cos^2 \alpha . \tag{17}$$



Consider the adjacent plot of *directional stiffness* as a function of α, for three materials all sharing $Y^0 = Y^1$, but with shear moduli $0.1K$, K, and $2K$. Note that for low shear, typical for cloth, the maximal to minimal bending stiffness ratio is 2. As shown in Figure 9, this has a significant effect on drapability, and *cannot* be achieved by a simple model using just two parameters. Similarly, the shear modulus affects resistance of cloth to *twisting*, even if $Y^0 = Y^1$, as twisting may lead to diagonal bending.

Directional stiffness is particularly important in the case of *quasi-inextensible* flat sheets. Such sheets have small Gaussian curvature (the product of the two principal curvatures), which implies that strong bending can be present only in a single direction.

In contrast to directional stiffness, forces arising from the interaction of *multiple* bending directions (*e.g.*, due to the Poisson ratio in the isotropic case) appear to be qualitatively less important. In fact, one can show that for directional stiffness this interaction results in effective increase in the shear modulus. One can prove that in the simple case of bending of a plate with fixed boundaries the bending forces (unlike stretching) are independent of the Poisson ratio.

The Kawabata system characterizes cloth by its resistance to bending along warp and along weft directions, and by tensile, shearing and compressive stiffnesses measured as functions of deformation. In the orthotropic elasticity model, stiffnesses are assumed to be independent of the deformation, which appears to be a good approximation for qualitative modeling. Furthermore, plasticity effects are ignored, and the compressive and tensile stiffnesses are assumed to be equal. As pointed out in [Breen and Donald 1994], linear elasticity provides a good approximation for Kawabata measurements for small deformations, although it ignores some of the subtler initial-resistance effects.

Given Kawabata measurements of a fabric, Y^1 and Y^0 can be inferred from directional bending stiffnesses and tensile measurements, and G^{01} from shear measurements. Alternatively, Y^1, Y^0, G^{01}, and the material axes may be directly controlled by an artist, adjusting parameters based on the notion that Y^1 and Y^0 determine resistance to bending along the two orthogonal material directions and G^{01} determines the resistance of the material to draping over an object.

Given the material parameters, and applying a standard formulation of Hooke's law for orthotropic materials, we can express the bending energy density of a deformed surface as a function of bending strain ϵ, using one additional material parameter $Y^{01} = \nu_{01} Y^1 = \nu_{10} Y^0$, where ν_{10} and ν_{01} are Poisson ratios (which always satisfy $\nu^{01}/Y^0 = \nu^{10}/Y^1$). The energy density is

$$c \left(Y^0 \epsilon_{00}^2 + Y^1 \epsilon_{11}^2 + 2Y^{01} \epsilon_{00} \epsilon_{11} \right) + 2hG^{01} \epsilon_{01}^2 , \tag{18}$$

where $c = hY^0Y^1/(Y^0Y^1 - Y^{01}Y^{01})$, $h = \tau^3/12$, and τ is the sheet thickness. The bending strain is the shape operator [do Carmo 1992] for surface, which for a quadratic bending energy reduces to the matrix of second partial derivatives. For the special case of isotropic materials, $Y^0 = Y^1 = 2(1+\nu)G^{01} = Y^{01}/\nu$.

In the following section, we show that orthotropic effects can be captured by weighting each hinge stiffness by a scalar factor, $\bar{\lambda}_i$. We again offer two parallel views, a geometric and a finite element one. For convenience of implementation, the results of this derivation are summarized in §5.

4.1 Hinge-based Orthotropy

In §2.2 we discussed a simple model of discrete bending energy for isotropic shells. We based this model on summing the contributions of squared mean curvatures over interior edges. While discrete mean curvatures suffice to cover the isotropic case, we must use a discretization of the full shape operator in the anisotropic case. Indeed, smooth anisotropic energy density (18) requires expressing the discrete shape operator as a symmetric 2×2 matrix in the material frame—as the entries of this matrix measure bending and shearing stiffness in principal material directions. A hinge-based discrete shape operator $\mathbf{S}$ has been successfully applied in geometric modeling [Hildebrandt and Polthier 2004]. Here we augment the geometric modeling view by three aspects: (a) we express $\mathbf{S}$ in a material frame as opposed to its original expression in a principal curvature frame, (b) we show how $\mathbf{S}$ leads to a scalar stiffness-correcting factor per edge covering the discrete orthotropic case, and (c) we offer a derivation of the discrete shape operator based on non-conforming CR elements.

Geometric View We briefly recall the geometric view of discrete shape operator based on discrete principal curvatures taken in [Hildebrandt and Polthier 2004]. It is well-known from the smooth case [do Carmo 1992] that S corresponds to a quadratic form whose (orthogonal) eigenvectors correspond to the two principal directions, and its eigenvalues correspond to the principal curvatures, κ_1 and κ_2. In the discrete *edge-based* view, [Hildebrandt and Polthier 2004] define principal directions as the two directions along and perpendicular to a given edge, respectively. In this view, there is no curvature along the edge ($\kappa_1 = 0$). Consequently, in the edge-based principal curvature frame, the discrete shape operator takes the form

$$\mathrm{S} = \begin{pmatrix} 0 & 0 \\ 0 & H \end{pmatrix} ,$$

where $H = \kappa_1 + \kappa_2$ denotes mean curvature. We note that this hinge-based shape operator is mesh dependent (in particular, principal directions are tied to edge directions). However, if the edge directions are distributed evenly, the true shape operator is well approximated on average.

In order to express S in the material frame, we treat the mesh area associated with each hinge stencil as a *homogeneous* piece of material, *i.e.*, we assume that the material parameters cY^0, cY^1, cY^{01}, hG^{01}, and the material axes are *constant* over hinge stencils. Denoting by γ_0 the angle between edge $\mathbf{e}_i$ and the first material axis, $\hat{\mathbf{y}}_a$, and using the fact that the shape operator transforms like a quadratic form, we obtain that S takes the form

$$\mathrm{S}_i = H_i \begin{pmatrix} \sin^2\gamma_0 & -\sin\gamma_0\cos\gamma_0 \\ -\sin\gamma_0\cos\gamma_0 & \cos^2\gamma_0 \end{pmatrix} , \qquad (19)$$

when expressed in the material frame of the edge i. Finally, using the smooth energy density (18) and setting $\epsilon = \mathbf{S}$, it follows from basic trigonometric identities that the discrete orthotropic density can be written as

$$\underbrace{\left((cY^0)\sin^4\gamma_0 + (cY^1)\cos^4\gamma_0 + \frac{1}{2}\left(cY^{01} + hG^{01}\right)\sin^2(2\gamma_0)\right)}_{\bar{\lambda}_i} \cdot H_i^2 ,$$

providing a way to adjust hinge weights to effect orthotropic response.

Finite Element View In §3.2 we discussed the derivation of the edge-based mean curvature vector in terms of the CR finite elements. We now offer a derivation of the discrete shape operator given by (19) in terms of the non-conforming elements.

In material frame coordinates, the representation of the shape operator is given by the second directional derivatives,

$$\frac{\partial^2}{\partial y_a \partial y_b} = (\hat{\mathbf{y}}_a \cdot \nabla)(\hat{\mathbf{y}}_b \cdot \nabla) ,$$

with $a, b \in \{0, 1\}$, applied to the immersion of the surface [do Carmo 1992]. We require the CR discretization of these operators. As for the case of mean curvature, the corresponding weak form formulation is obtained by integration by parts. Using that the material axes are assumed to be constant on hinges, we obtain

$$-\int_S \phi_i(\hat{\mathbf{y}}_a \cdot \nabla)(\hat{\mathbf{y}}_b \cdot \nabla)\phi_j \,\mathrm{d}A = \int_S (\hat{\mathbf{y}}_a \cdot \nabla\phi_i)(\hat{\mathbf{y}}_b \cdot \nabla\phi_j)\,\mathrm{d}A .$$

Notice that this expression is in general not symmetric in a and b. In order to match the smooth case, where partial derivatives with constant coefficients commute, we symmetrize it, obtaining the stiffness matrices, K^{ab}, whose entries are given by

$$\mathrm{K}_{ij}^{ab} = \frac{1}{2}\int_S (\hat{\mathbf{y}}_a \cdot \nabla\phi_i)(\hat{\mathbf{y}}_b \cdot \nabla\phi_j) + (\hat{\mathbf{y}}_b \cdot \nabla\phi_i)(\hat{\mathbf{y}}_a \cdot \nabla\phi_j)\mathrm{d}A .$$

In perfect analogy to the mean curvature definition (10), we obtain the discrete shape operator by

$$\mathbf{S}^{ab} = \mathrm{M}^{-1}\,\mathrm{K}^{ab}\,\mathrm{T}\,\mathbf{x} .$$

Notice that this expression is vector-valued, as was (10). Indeed, $\mathbf{S}$ is precisely the vector-valued version of (19), *i.e.*, $\mathbf{S}_i = \mathrm{S}_i\hat{\mathbf{n}}_i$, where $\hat{\mathbf{n}}_i$ is the angle-bisecting normal of the dihedral angle at the edge.

This completes the requisite derivations for orthotropic cubic shells. In the following, we discuss the implementation of our model (§5) and demonstrate the generality of the model with various simulation examples (§6).

5 Implementing the Discrete IBM

In this section we describe the computation of forces and force Jacobians for the discrete bending energy (8). Consider a mesh with n vertices and coordinate vectors $\mathbf{x}^x, \mathbf{x}^y, \mathbf{x}^z \in \mathbb{R}^n$. *Hinge-centric* computations are expressed in terms of the hinge's *local indices* (Fig. 2) for triangles, $\{T_0, T_1\}$, edge vectors, $\{\mathbf{e}_0, \ldots, \mathbf{e}_4\}$, and vertex positions, $\{\mathbf{x}_0, \ldots, \mathbf{x}_3\}$.

Precomputation Recall that quantities that depend only on the undeformed positions are decorated with a bar, *e.g.*, $\bar{\mathbf{x}}$. Compute barred quantities once, before the simulation.

One-time Matrix Assembly Assemble the global $n \times n$ matrix, $\bar{\mathbf{Q}}$, by iterating over hinge stencils. In the usual style of *stiffness matrix assembly* [Zienkiewicz and Taylor 1989], $\bar{\mathbf{Q}}$ accumulates contributions from each *local* 4×4 matrix, $\bar{\mathbf{Q}}_i$:

$$\bar{\mathbf{Q}}_i = \left(\frac{3\bar{\lambda}_i}{\bar{A}_i}\right) \bar{K}_i^T\,\bar{K}_i ,$$

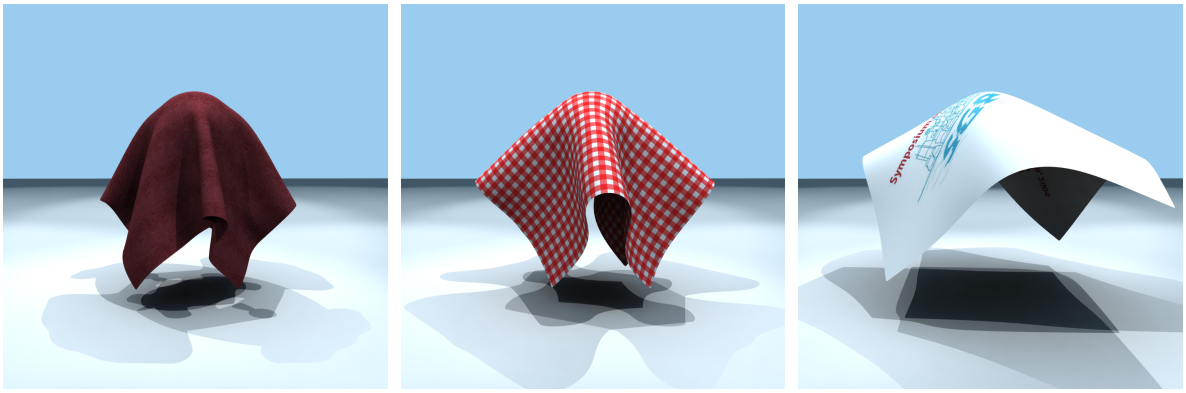


Figure 3: Deformations are primarily characterized by the ratio of bending to membrane stiffness. Despite having forces linear in positions, the quadratic bending model is valid over a broad range of stiffness values. Shown here *(left to right)*: $10^{-5} : 1$, $10^{-3} : 1$, and $10^{-2} : 1$.

where $\bar{\lambda}_i$ is the stiffness of the hinge and $\bar{A}_i$ is the combined area of the two hinge triangles, In local indices, we have $\bar{K} = (\bar{c}_{03} + \bar{c}_{04}, \bar{c}_{01} + \bar{c}_{02}, -\bar{c}_{01} - \bar{c}_{03}, -\bar{c}_{02} - \bar{c}_{04}) \in \mathbb{R}^4$, where $\bar{c}_{jk} = \cot \angle(\bar{\mathbf{e}}_j, \bar{\mathbf{e}}_k)$.

Force Computation Compute forces by adding thin plate and (if dealing with shells) nonflat contributions. Compute thin plate contributions globally as the matrix-vector products $\mathbf{f}^x = \bar{\mathbf{Q}}\mathbf{x}^x$, $\mathbf{f}^y = \bar{\mathbf{Q}}\mathbf{x}^y$, $\mathbf{f}^z = \bar{\mathbf{Q}}\mathbf{x}^z$. Compute nonflat contributions by accumulating local hinge contributions

$$\begin{aligned} \mathbf{f}_0 &= -\mathbf{f}_1 - \mathbf{f}_2 - \mathbf{f}_3 , & \mathbf{f}_1 &= \bar{k}(\mathbf{e}_1 \times \mathbf{e}_2) , \\ \mathbf{f}_2 &= \bar{k}(\mathbf{e}_2 \times \mathbf{e}_0) , & \mathbf{f}_3 &= \bar{k}(\mathbf{e}_0 \times \mathbf{e}_1) , \end{aligned}$$

where $\bar{k} = 3\bar{\lambda}_i(\bar{c}_{01} - \bar{c}_{03})(\bar{c}_{04} - \bar{c}_{02})[\bar{\mathbf{e}}_0, \bar{\mathbf{e}}_1, \bar{\mathbf{e}}_2]/(\bar{A}_0|\bar{\mathbf{e}}_0|^3)$.

Force Jacobian Computation The exact force Jacobian ($\mathbb{R}^{3n} \times \mathbb{R}^{3n}$) is obtained by adding the thin plate and (in the case of shells) the nonflat contributions. The thin plate contributions are given by $\partial \mathbf{f}^x/\partial \mathbf{x}^x = \partial \mathbf{f}^y/\partial \mathbf{x}^y = \partial \mathbf{f}^z/\partial \mathbf{x}^z = \bar{\mathbf{Q}}$. For best performance (without sacrifice of accuracy), the nonflat contribution is omitted (as explained in §6); for completeness of exposition the nonflat contribution is detailed in the Appendix A.

Orthotropic Hinge The simple expressions above are all that is required to implement a cubic hinge-based bending force, given a stiffness value $\bar{\lambda}_i$ per hinge. For homogeneous isotropic materials, $\bar{\lambda}_i = \bar{\lambda}$ does not vary over the mesh, and the above derivation suffices; if a general class of orthotropic materials is desired, then $\bar{\lambda}_i$ should be computed by the formula derived in (§4.1); recall $\bar{\lambda}_i =$

$$(cY^0)\sin^4\gamma_0 + (cY^1)\cos^4\gamma_0 + \frac{1}{2}\left(cY^{01} + hG^{01}\right)\sin^2(2\gamma_0) ,$$

where γ_0 is the angle between the hinge edge and the first material axis, $\hat{\mathbf{y}}_0$, and Y^0, Y^1, Y^{01}, and G^{01} are the material parameters described above. The above expression is all that is needed to implement orthotropic bending for cubic shells, and indeed for any existing hinge-based model.

In our implementation, the (spatially-varying) direction of $\hat{\mathbf{y}}_0$ is encoded by a coordinate function over a given parameterization of the surface. In particular, we use the color values of a texture map to encode the direction of the material axis.

6 Efficient Plate & Shell Dynamics

Whether in the high-fidelity or interactive setting, rapid simulation requires an efficient numerical integrator for second order initial value problems (IVPs) of the form

$$M\ddot{\mathbf{x}}(t) = \mathbf{f}(t, \mathbf{x}(t), \dot{\mathbf{x}}(t)) , \tag{20}$$

where M is the (physical) mass matrix; $\mathbf{f}(t, \mathbf{x}(t), \dot{\mathbf{x}}(t))$ are forces depending on time, position, and velocity; and the initial position and velocity of the cloth are prescribed [Hauth 2004]. For analysis one often introduces velocity as a separate variable, rewriting the above as a coupled first-order system,

$$\begin{pmatrix} Id & 0 \\ 0 & M \end{pmatrix} \begin{pmatrix} \dot{\mathbf{x}}(t) \\ \dot{\mathbf{v}}(t) \end{pmatrix} = \begin{pmatrix} \mathbf{v}(t) \\ \mathbf{f}(t, \mathbf{x}(t), \mathbf{v}(t)) \end{pmatrix} ,$$

with appropriate initial conditions. The temporal discretization of this system is a well-studied and active area of applied mathematics and computational mechanics, with a host of attendant methods. For treatments tailored to cloth simulation we refer the reader to [Baraff and Witkin 1998; Etzmuss et al. 2000; Hauth and Etzmuss 2001; Choi and Ko 2002; Ascher and Boxerman 2003; Bridson et al. 2003; Hauth et al. 2003; Boxerman and Ascher 2004; Hauth 2004]).

As a general observation, each force may be treated *explicitly* or *implicitly*. An explicit treatment requires the (possibly repeated) evaluation of the force per discrete time step; an implicit method requires additionally the (possibly repeated) evaluation of the force Jacobian per discrete time step. In the case of a conservative force, such as elastic bending, the force Jacobian is minus the energy Hessian. In the case of a dissipative Rayleigh force, such as damping of the bending modes, the Jacobian is expressed as a linear combination of the physical mass and energy Hessian. Etzmuss, Hauth *et al.* [2000; 2001; 2003; 2004] as well as Ascher and Boxerman [2003; 2004] analyzed the behavior of time integration schemes and presented arguments for the adoption of implicit-explicit (IMEX) integrators; Bridson *et al.* [2003] presented a semi-implicit method that treats damping implicitly and elastic forces explicitly. Our results indicate that IBM accelerates both explicit or implicit treatments of bending.

Among common discrete time-stepping schemes [Baraff and Witkin 1998; Hauth 2004; Hairer et al. 2006], implicit schemes are popular in animation due to their stability. Implicit schemes advance time by solving a (typically nonlinear) system of equations. The system is usually solved by repeated Newton iterations (although *semi-implicit* methods complete a single Newton iteration) [Press et al. 1992]. Each Newton iteration requires an evaluation of the force, $-\nabla E$, as well as the force Jacobian.

Inexact Newton's Method The thin plate case (§6.1) is distinguished by the property that the force Jacobian is *constant*, and

Draping problem		regular mesh (resolution, in no. vertices)				irregular mesh (resolution, in no. vertices)			
		400	1600	6400	25600	450	2100	6500	22500
Gradient	nonlinear hinge	0.937	3.45	16.4	66.6	1.10	5.43	17.6	67.8
cost (ms)	quadratic IBM	0.081	0.338	2.19	9.15	0.098	0.494	2.32	9.68
Hessian	nonlinear hinge	12.8	54.2	218	890.	15.2	77.2	246	888
cost (ms)	quadratic IBM	0.237	0.963	3.87	15.7	0.266	1.28	3.99	13.6
Explicit step	nonlinear hinge	3.81	6.64	27.5	112.	2.16	9.53	31.4	140.
cost (ms)	quadratic IBM	2.63	2.90	11.9	48.8	0.964	4.35	15.2	76.5
Implicit step	nonlinear hinge	28.6	138	470.	1730	33.9	219	557	1880
cost (ms)	quadratic IBM	11.0	62.7	168	505	13.6	103	219	612

Flag problem		regular mesh (resolution, in no. vertices)				irregular mesh (resolution, in no. vertices)			
		400	1600	6400	25600	450	2100	6500	22500
Gradient	nonlinear hinge	0.975	3.99	16.0	64.0	1.10	5.43	17.8	68.7
cost (ms)	quadratic IBM	0.085	0.341	2.14	8.75	0.099	0.490	2.31	9.28
Hessian	nonlinear hinge	13.4	54.8	212	849	15.2	77.4	247	887
cost (ms)	quadratic IBM	0.251	0.974	3.79	14.99	0.267	1.30	3.96	13.7
Explicit step	nonlinear hinge	1.73	7.05	27.7	112.	1.97	9.80	32.7	134
cost (ms)	quadratic IBM	0.780	3.26	13.3	53.4	0.900	4.54	16.1	70.0
Implicit step	nonlinear hinge	27.6	106	420.	1680	33.5	155	513	1880
cost (ms)	quadratic IBM	9.53	32.9	127	490	12.5	50.4	166	608

Table 1: Computational cost per time step for a variety of regular- and irregular-mesh resolutions, comparing our quadratic energy to the popular nonlinear hinge energy. Costs reported include the time required for collision detection and response. These tests were conducted on a single process, Pentium D 3.4GHz, 2GB RAM.

can be pre-computed *once* (see §5). In the cubic shells case (§6.2), however, the force Jacobian is no longer constant. Here the following observation comes to the rescue. Consider Newton's method, applied to an implicit time-stepping scheme,

$$\mathbf{G}(\mathbf{x}_{k+1}) = \mathbf{x}_{k+1} - \mathbf{x}_k - h\mathbf{f}(\mathbf{x}_{k+1}) = 0\,.$$

The fixed points of this method remain unaltered when the Jacobian, $\mathbf{J} = \nabla\mathbf{G}$, is replaced by any invertible matrix, $\tilde{\mathbf{J}} \approx \mathbf{J}$. This observation justifies the *inexact* Newton's method: Instead of a costly $\mathbf{J}$, use any good but efficient approximation $\tilde{\mathbf{J}}$ [Morini 1999]. Approximating the Jacobian *may* affect (the rate or radius of) convergence, but it will not affect the limit value of converging iterations. In contrast, approximating $\mathbf{G}$ *will* affect the value of the fixed points; therefore, inexact Newton approximates only the Jacobian $\nabla\mathbf{G}$, not the function $\mathbf{G}$. The isometric deformation assumption gives the constant approximant, $\tilde{\mathbf{J}}$, to the flow Jacobian, $\mathbf{J}$, where

$$\mathbf{J} = \mathrm{Id} - h\,\mathrm{H}ess(E_b)(\mathbf{x}) \approx \mathrm{Id} - h\,\bar{\mathbf{Q}} = \tilde{\mathbf{J}}\,,$$

where $\bar{\mathbf{Q}}$ is the constant matrix computed in §5.

Hauth [Hauth 2004] used an approximation of the in-plane stretching force Jacobian, and an inexact Newton framework, to accelerate cloth simulations; our work is similar, but we approximate the bending Jacobian. We will revisit the inexact Newton's method in §7, where it enables an acceleration of Willmore-type fairing applications.

6.1 Thin Plates

We compare the computational cost and visual quality of the isometric bending model to the widely-used bending hinge-based energy described in [Baraff and Witkin 1998; Bridson et al. 2003; Grinspun et al. 2003], for regular- and irregular-meshes ranging from 400 to 25600 vertices, on a draping problem as well as a dynamic billowing flag. Figures 1-4 provide a qualitative point of comparison between IBM and the nonlinear hinge model.

Implementing our discrete IBM for thin plates is straightforward, as explained in §5. In contrast, an efficient implementation of a nonlinear model such as a hinge spring requires both significant (human and computer) gradient calculations, or leads to adoption of costly automatic-differentiation techniques [Grinspun et al. 2003]. For comparison of computational efficiency, our implementation of the hinge forces and Jacobians was hand tuned and not based on automated methods.

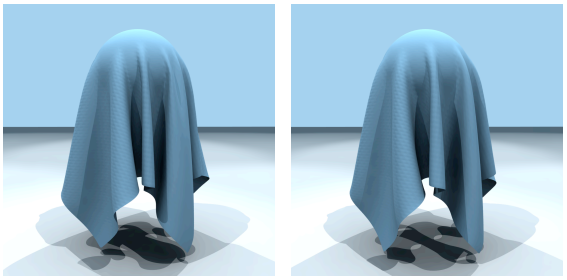


Figure 4: Final rest state of a cloth draped over a sphere, for *(left)* the proposed isometric bending model and *(right)* the widely-adopted nonlinear hinge model.

Replacing the nonlinear hinge model with the quadratic IBM reduces bending force computation cost by seven- to eleven-fold. Where a force Jacobian is required, IBM's constant Hessian is precomputed once and stored in a sparse matrix datastructure, eliminating the computational cost of bending Jacobian assembly. Of course, the actual net performance improvement in any given application depends on the fraction of total computation associated to bending.

Test Setup To estimate this, we implemented both the implicit solver framework of [Baraff and Witkin 1998] as well as a simple explicit Euler solver. Our choice of solvers is motivated by a desire to estimate profitability of incorporating IBM whether or not a framework requires bending force Jacobians. The test framework incorporates: (a) the usual constant strain linear finite element for membrane response [Zienkiewicz and Taylor 1989; Hughes 1987], which computationally is at least as costly as the membrane model proposed in [Baraff and Witkin 1998]; (b) the robust collision detection and response methods of [Bridson et al. 2002], augmented with k-DOP trees [Klosowski et al. 1998]; (c) the PETSc solver library [Balay et al. 2001]. Further optimization of our collision detection using [Govindaraju et al. 2005] is likely to reduce the overhead of collision computations for large meshes. For a more

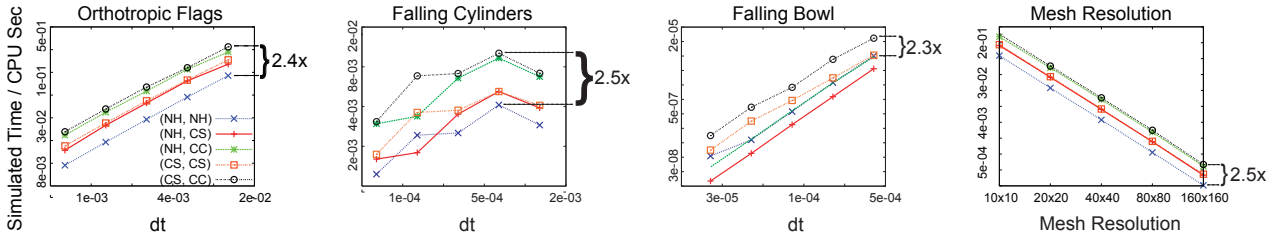


Figure 5: (*Left:*) Performance measured in completed simulation time per CPU second (y-axes) as a function of discrete time step (x-axes). Higher values mean better performance. In an inexact Newton framework, Cubic Shells dominate performance for all tested scenarios, including (left-to-right) Orthotropic Flags (Fig. 10), Falling Cylinders (Fig. 8), and Falling Bowl (Fig. 12). *Right:* Performance as a function of mesh resolution (from 10×10 to 160×160 regular grids), demonstrating that the benefits of cubic shells are not resolution-dependent. Timing numbers include timestepping and collision detection/response.

detailed comparison, see Table 1. We observe a consistent reduction in total computational cost compared to the nonlinear hinge model, across the two problem setups, two mesh types, and resolutions ranging from 400 to 25600 vertices.

Draping Cloth We simulated the draping of a square sheet over a sphere. As shown in Figure 4, the draped cloths are qualitatively similar in their configuration and distribution of wrinkles and folds. Since only the final draped state was important, we introduced a dissipative Rayleigh force allowing for larger time steps [Hughes 1987]. For this example we might also consider a quasistatic method, as recently considered in [Teran et al. 2005]. Since quasistatics requires repeated evaluation of forces and their Jacobian, the proposed bending model will also be profitable in the quasitatic setting. Together Figures 4-3 capture the behaviour of IBM under a range of bending stiffnesses.

Billowing Flag We simulated the dynamics of a flag under wind. As shown in the accompanying movie (see also Fig. 1) the motion of the flag is qualitatively unaffected when we substitute the more economical quadratic bending energy. Furthermore, we found that there is no need to readjust material parameters when switching from the nonlinear hinge to the IBM model; this is not unexpected in light of the link between the two energies, as discussed in Footnote 2. We modeled wind by a constant homogeneous velocity field, with force proportional to the projection of the wind velocity onto the area-weighted surface-normal. For more sophisticated effects of wind fields on cloth see [Keckeisen et al. 2004]. Purely for contrast with the draping example, we did not introduce any specific damping forces, although the implicit Euler method is known to exhibit numerical damping.

IMEX Methods We stress that the proposed model is not specialized to our test framework. Consider for example the integration of the IBM into the framework described by Bridson *et al.* [2003], which treats elastic (position-dependent, velocity-independent) forces explicitly, and treats damping forces implicitly. With IBM both force and Jacobian computation is greatly reduced; therefore explicit, implicit, and semi-implicit treatments are all prime candidates. Bridson *et al.* describe a very efficient matrix-free solver strategy, involving as few as one to two conjugate gradient iterations: this heightens the relevance of fast (elastic and damping) bending force computations, since with fewer conjugate gradient iterations, the balance of computation shifts further to the computation of the cached nonlinear position-dependent terms as well as the algebraic system's right hand side. In incorporating IBM in this context, the requisite matrix-vector multiplication may be distributed, $(\mathbf{K}_m + \mathbf{K}_b)\vec{\mathbf{x}} = \mathbf{K}_m\vec{\mathbf{x}} + \mathbf{K}_b\vec{\mathbf{x}}$, where the precomputed K_b is a multiple of the constant bending Jacobian, and K_m is the membrane Jacobian.

6.2 Thin Shells

The separability of the force Jacobian (see §5) of our cubic bending model into a constant and linear term, combined with an inexact Newton's method, opens several interesting possibilities for efficient time integration of thin shell dynamics.

Experiment We consider the application of the inexact Newton's method with approximations to the bending force Jacobian. To provide a standard and easily-reproducible point of reference, we consider the Euler method, fully-implicit on stretching and bending elastic forces, without any additional damping forces.

In our experiments we pair two possible bending forces with three possible force Jacobians. For the two forces we consider the *nonlinear hinge* (NH)—the force resulting from treating bending energy fully nonlinearly (*i.e.*, by dropping the isometry assumption)—and the *cubic shells force* (CS) arising from our cubic energy as discussed in the previous section. For the three force Jacobians we consider the *nonlinear hinge Jacobian* (NH), the *entire* cubic shells Jacobian (CS), and the *constant part* (CC) of the cubic shells Jacobian. A particular choice of (inexact) Newton is denoted by a 2-tuple, *e.g.*, (CS,CC) denotes the use of the cubic shell bending forces paired with an approximate constant Jacobian; (NH,NH) denotes the usual fully-implicit implementation of the nonlinear hinge.

As a baseline, we run (NH,NH) at the maximum stable step size, and four smaller step sizes; these step sizes are reused in runs of (NH, CS), (NH, CC), (CS, CS), (CS, CC). We measure performance of these runs for various problem scenarios (see §6.3).

We record the performance of each method, measured as the ratio *completed simulation time per CPU second*, *i.e.*, higher values mean better performance (see Figure 5–left). The cubic shells with an approximate *constant* Jacobian, (CS,CC), dominates all methods for all problem scenarios. Note that the performance gains observed for (CS, CC) are independent of mesh resolution (see Figure 5–right).

6.3 Visual Impact of Orthotropic Effects

Orthotropic parameters provide additional artistic control over the dynamics of wrinkles and folds. Consider for example a flag tailored by cutting a rectangular pattern from an orthotropic textile. The orientation of warp and weft relative to the flag's pattern, as well as the values of Young's and shear moduli, have considerable visual impact on the resulting dynamics (see Fig. 10).

The orthotropic shear modulus plays an important role in the *drapability* of a fabric [Sidabraite and Masteikaite 2002]; the shear modulus is *not* captured by geometric models that employ Euler's formula, such as [Baraff and Witkin 1998; Volino and Magnenat-Thalmann 2006], or elliptic interpolation, such as [Choi and Ko

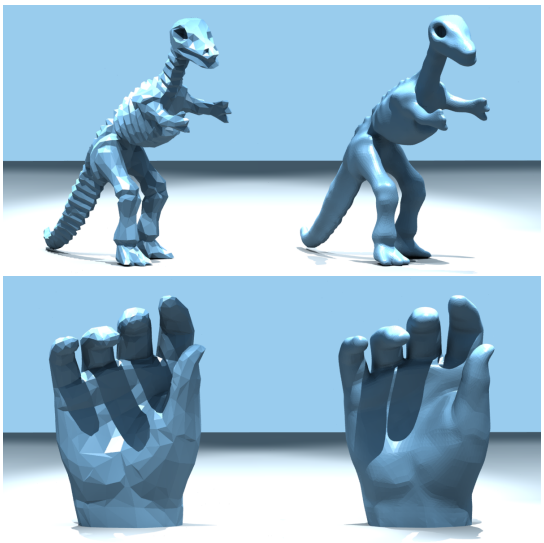


Figure 6: Initial and final frames of Willmore flow applied to smooth *(top)* a 44928 triangle dinosaur and *(bottom)* a 24192 triangle hand at interactive rates. 16 smoothing steps require a total of 7.47s and 4.42s, with one-time factorization costing 8.77s and 5.31s, for the dinosaur and the hand, respectively. Images rendered with flat shading.

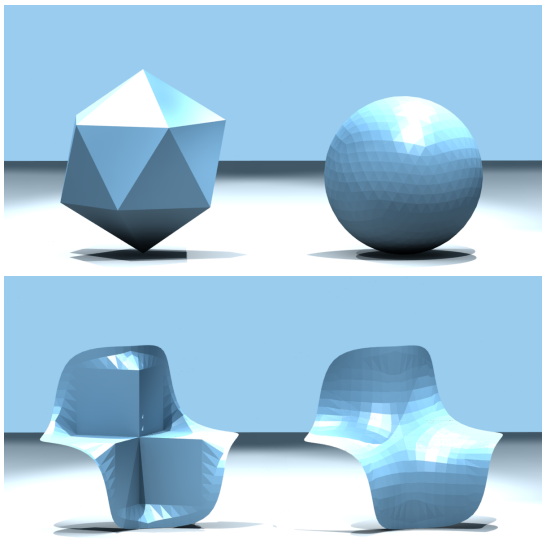


Figure 7: Initial and final frames of Willmore flow applied to solve two tasks posed by Bobenko and Schröder. *(top)* Smoothing a 4-times subdivided icosahedron into a sphere and *(bottom)* a hole filling problem. The sphere converged in 120ms (12ms $\times$ 10 smoothing steps), with 200ms for Hessian prefactorization. The hole filling problem required 640ms after 120ms for prefactorization. Rendered with flat shading.

2003]. A high shear modulus tends to align folds and wrinkles to the material axes; a lower shear modulus allows the fabric to fold along other directions, and therefore to obtain a closer fit to the body. The first two frames of Fig. 9 compare a high and low shear modulus, respectively, illustrating the intuitive notion of drapability. When a high shear modulus is desired (leftmost frame), the Young's modulus is a poor substitute—it is unable to capture stiff extrusion of the poncho around the shoulders without introducing extraneous stiffness elsewhere.

Furthermore, orthotropy has a strong effect on the interaction between a material and its environment. We simulate the fall and bounce of four elastic cylinders, each with a different orientation for its principal material axis. The resulting animation (Fig. 8) reveals the extreme variations in deformation and overall trajectory that arise purely from changing the orthotropic parameters.

Orthotropy indirectly enriches any other technologies implemented in the simulator, such as viscoelasticity or fracture [Terzopoulos and Fleischer 1988]. For example, the anisotropic bending resulting from collisions produces distinctive fracture patterns both for plates (Fig. 11) and shells (Fig. 12). For a summary of the fracture algorithm refer to Appendix B.

6.4 Sensitivity to the Inextensibility Assumption

Implicit to the IBM is the assumption that the surface deforms isometrically, *i.e.*, without stretching. A natural question, then, is how much stretching is permissible in practice, and what are the failure modes of the model under excessive stretching? To explore these questions, we re-simulate the falling cylinder and falling bowl using progressively lower stretching resistance. Both examples remain well-behaved so long as the stretching stiffness is two orders of magnitude greater than the bending stiffness; during the simulation, the meshes stretch by as much as 15%.

If we reduce stretching stiffness even further, then for the thin *shell* examples we observe a slow, noticeable stretching of the surface, in particular, in an expanding (not oscillatory) mode. This is perhaps not surprising, since a cubic energy is not bounded from below; under normal (quasi-isometric) circumstances, the stretching resistance prevents this infinite well from being exploited.

We repeat the above experiment with the billowing flag. Since the flag has a flat undeformed configuration, the cubic energy term vanishes, leaving a quadratic energy bounded from below. We observe that the flag simulation is well-behaved even when strong wind induces stretching of 300%.

7 Geometric Modeling

The *Willmore energy* of a surface is given as

$$E_W(\mathbf{x}) = \int_S (H^2 - K)\,\mathrm{d}A \;=\; \frac{1}{4}\int_S (\kappa_1 - \kappa_2)^2\,\mathrm{d}A\,.$$

As noted in [Bobenko et al. 2005], immersions that minimize Willmore energy are of interest in a range of areas, including the study of conformal geometry [Blascke 1929; Willmore 2000], physical modeling of fluid membranes [Canham 1970; Helfrich 1973], and our focus in this example, geometric modeling. For surfaces without boundary and surfaces whose boundary is fixed up to first order, the Willmore functional is variationally equivalent to the thin plate ($\bar{H} = 0$) version of the functional (1). The corresponding geometric flow

$$\dot{\mathbf{x}} = -\nabla E_{\mathrm{b}}(\mathbf{x}),$$

has spurred many applications for surface fairing and surface restoration [Bobenko et al. 2005; Clarenz et al. 2004; Schneider and Kobbelt 2001; Yoshizawa and Belyaev 2002]: Hole-filling applications (see Figure 7-bottom) integrate the flow to its stationary limit (when such a limit exists). Smoothing applications (see Figures 6 & 7-top) integrate the flow over a prescribed duration, with longer integration times smoothing progressively coarser spatial frequencies.

Implementations of discrete Willmore flow were reported by several authors. Ken Brakke's *Surface Evolver* [Brakke 1992; Hsu et al. 1992] used a discretized version of mean curvature as a building block for Willmore energy. Yoshizawa *et al.* [Yoshizawa and Belyaev 2002] discretized directly the energy gradient, using the cotangent formula. The latter introduced an additional tangential force to improve the quality of the evolving mesh. Clarenz *et al.* [Clarenz et al. 2004] discretized the variation of the Willmore energy in terms of linear Lagrange elements and treated the corresponding L^2-flow by a coupled system of second order equations. Finally, Bobenko *et al.* [Bobenko et al. 2005] presented a discrete version of the fact that the integrand of the smooth Willmore energy is conformally invariant. However, existing approaches did not focus on the economy that arises from assuming (or rather pretending) that deformations are isometric.

To compute a geometric flow, one must integrate the flow trajectory over time. This may be achieved via explicit or implicit methods, as described in §6. Following the direction laid out by Desbrun and coworkers [Desbrun et al. 1999], who note that the stability of implicit integration methods improves the performance of geometric flows, we implemented an implicit method, in particular using the *inexact* Newton's method.

Discrete IBM in Willmore Solver In contrast to our treatment of cloth and inextensible thin plates, in applications of Willmore energy the presence of an accompanying isometry-enforcing term is notably absent. Indeed, the deformations governed by Willmore flow are generally *not* isometric so that the energy gradient will not be linear, as it was for cloth. It may therefore seem surprising that our proposed Willmore flow application, while incorporating an isometry assumption, gains speed without paying in visual quality. The *inexact Newton Method* serves as the numerical framework in which this phenomenon can explained.

Using the PETSc [Balay et al. 1996] and PARDISO [Schenk and Gärtner 2004] solver libraries, we implemented the backward Euler method with an inexact Newton solver. We found that a *semi-implicit* treatment, which stops Newton's method after one iteration, exhibits the best trade-off between stable time step size and cost per time step, for a prescribed level of accuracy.

We briefly discuss the details that led to efficient Newton iterations. To make clear the structure of the linear solve, we write the Newton iteration as

$$(\nabla \mathbf{G})(\mathbf{x}_{k+1}^{(i+1)} - \mathbf{x}_{k+1}^{(i)}) = -\mathbf{G}(\mathbf{x}_{k+1}^{(i)}) \, .$$

For the right hand side, we compute the full nonlinear expression $\nabla E_b(\mathbf{x})$; consult Appendix C for a derivation. For the left hand side, we *replace* $\nabla \mathbf{G}$ by the constant precomputed matrix $\bar{\mathbf{Q}}$ (see §5). We used PARDISO's LL^T solver, which factors $\bar{\mathbf{Q}}$ symbolically and numerically. Since $\bar{\mathbf{Q}}$ is known at program start, the symbolic factorization step—the bulk of the linear solver's computation—can be executed just once at startup. The pre-factorization of $\bar{\mathbf{Q}}$, and the elimination of repeated matrix assembly, accounts for the speedup we observed.

Results To evaluate the performance of our method, we duplicated several problem scenarios presented by Bobenko *et al.* [Bobenko et al. 2005] (see Figures 6 & 7). Whereas Bobenko's work preserves the Möbius symmetries of the underlying continuous system, our focus is on rapid computation at nearly interactive rates, maintaining good surface quality, while retaining invariance under a subset of all Möbius transformations: rigid transformations and uniform scaling.

Figure 7-bottom shows the result of a six-sided hole filling problem (compare with [Bobenko et al. 2005]). In this problem setup, the boundary conditions are taken from a smooth Loop subdivision surface, and the interior triangles are initialized with a trivial non-smooth solution. We fix two rings of vertices to enforce the prescribed boundary conditions up to first order, and integrate the geometric flow until it reaches a stationary point. The solution required 760ms, which includes 120ms for Jacobian pre-factorization.

Due to its constant, pre-factored Jacobian, our method scales well to larger meshes. We applied Willmore flow to smooth several large meshes, including the dinosaur (45k triangles) and hand (24k triangles), shown in Figure 6. Unlike Laplacian smoothing, for which fast implicit methods have been demonstrated [Desbrun et al. 1999], Willmore flow is derived from a scale-invariant energy, hence it is not biased toward shrinking the surface. The inexact Newton's method enabled us to accelerate computation by several orders of magnitude. The near-interactive times are reported using a notebook computer, suggesting that fully-interactive Willmore flow for large meshes is well within the reach of the discrete IBM.

8 Discussion

Limitations The IBM does not overcome those limitations that are shared by all hinge-based approaches. In particular, the simplicity of hinge-based models comes at the cost of limited convergence behavior and meshing-dependence [Grinspun et al. 2006]. Beyond those limitations inherited from the general class of hinge-based methods, the approach here is built on the assumption of inextensibility. We performed a cursory evaluation of the sensitivity of the method to violations of this assumption (recall §6.4); a more rigorous evaluation is required to complete the picture.

Conserved Momenta In all circumstances, and in particular even when parasitic stretching is observed due to a broken isometry assumption, the simulated model conserves linear and angular momenta. To see this, note that the quadratic forces arising from the cubic energy are *not* approximated (only their Jacobian is); since the cubic energy is invariant under rigid body transformations of the deformed (and also the undeformed) configuration, forces induced by this energy do not apply a global acceleration or torque.

Who Should Use This Method? In any practical application, the final decision on incorporating a proposed technique hinges on the implementation cost and performance benefit of the method. We have presented a model for bending that involves simply computing the entries of a sparse matrix. This computation is straightforward, as shown in §6. Finally, incorporation of the proposed model is a minimally-invasive task. Given these considerations, we suggest that it will be straightforward to evaluate the efficacy of the proposed model within the reader's preferred cloth simulation framework.

Shortly after the work described in this chapter was completed, Goldenthal *et al.* [2007] proposed a fast projection method for enforcing inextensibility of a quadrilateral surface mesh. Building on this, English and Bridson [2008] observed that fast projection could be applied to enforce inextensibility of triangle meshes, by constraining the edge lengths of the dual-mesh; one interpretation is that the discontinuous surface given by the Crouzeix-Raviart elements deforms isometrically. This view illuminates the natural connection between the projection used to enforce inextensibility, and the IBM forces described in this chapter. English and Bridson observe this connection when they select the IBM as the simple, natural choice of bending model for the simulation of developable triangle meshes.

Acknowledgments We thank David Eberle for his feedback. This work was supported in part by the DFG Research Center MATHEON in Berlin, the NSF (MSPA-IIS-05-28402, CSR-CNS-06-14770, CCR-0093390, CAREER-CCF-06-43268), Autodesk, Elsevier, mental images, NVIDIA, and an IBM Faculty Parternship award.

Appendix A

Neglected Term in Force Jacobian As discussed in §6, we advocate the use of an inexact Newton's method, using only the constant portion of the force Jacobian given in §5. Indeed, for optimal performance (and no loss of accuracy) one should omit calculation of the linear component arising from the nonflat undeformed configuration. However, for completeness of presentation, we include the expression for this linear component below.

With respect to the local position vector $(\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3) \in \mathbb{R}^{12}$, and the corresponding local force vector $\mathbf{f} = (\mathbf{f}_0, \mathbf{f}_1, \mathbf{f}_2, \mathbf{f}_3) \in \mathbb{R}^{12}$, the local force Jacobian ($\mathbb{R}^{12} \times \mathbb{R}^{12}$) is given by

$$\bar{k} \begin{pmatrix} 0 & (\mathbf{e}_1 - \mathbf{e}_2)^* & (\mathbf{e}_2 - \mathbf{e}_0)^* & (\mathbf{e}_0 - \mathbf{e}_1)^* \\ -(\mathbf{e}_1 - \mathbf{e}_2)^* & 0 & -\mathbf{e}_2^* & \mathbf{e}_1^* \\ -(\mathbf{e}_2 - \mathbf{e}_0)^* & \mathbf{e}_2^* & 0 & -\mathbf{e}_0^* \\ -(\mathbf{e}_0 - \mathbf{e}_1)^* & -\mathbf{e}_1^* & \mathbf{e}_0^* & 0 \end{pmatrix},$$

where each entry represents a 3×3 skew-symmetric subblock; the asterisk applied to a vector, $\mathbf{v}^*$, produces the matrix

$$\mathbf{v}^* = \begin{pmatrix} 0 & -\mathbf{v}_z & \mathbf{v}_y \\ \mathbf{v}_z & 0 & -\mathbf{v}_x \\ -\mathbf{v}_y & \mathbf{v}_x & 0 \end{pmatrix},$$

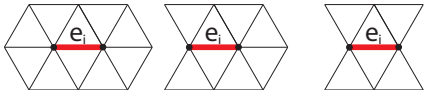
corresponding to the cross product operation $\mathbf{v} \times (\cdot)$. The local force Jacobian is assembled into the global $\mathbb{R}^{3n} \times \mathbb{R}^{3n}$ Jacobian in the usual manner. While it is most easily expressed by a skew-symmetric matrix of skew-symmetric subblocks, note that $\operatorname{Hess} G(\mathbf{e}_0)$ is indeed symmetric.

Appendix B

Implementing Fracture The material fractures when internal strain exceeds a threshold. For simplicity we only consider fracture along existing mesh edges; however, this limitation can be easily removed, see *e.g.*, [O'Brien and Hodgins 1999; Molino et al. 2004; Gingold et al. 2004]. A hinge edge, $\mathbf{e}_i$, fractures if the bending strain, $\frac{|\mathbf{e}_i|}{A_i}(\theta_i - \bar{\theta}_i)$, exceeds a material-dependent threshold.

Fracture may be viewed as the transition when an interior edge becomes a boundary edge. A mesh edge has one of three states: INTERIOR→FRACTURED-INTERIOR→BOUNDARY, where the arrows indicate allowable state transitions. An INTERIOR edge becomes FRACTURED-INTERIOR if the strain threshold is exceeded; a FRACTURED-INTERIOR edge becomes BOUNDARY only as a consequence of explicit changes to mesh connectivity, as explained below.

A BOUNDARY vertex is a vertex incident on a FRACTURED-INTERIOR or BOUNDARY edge. A FRACTURED-INTERIOR edge, e_i, may be in one of three configurations, distinguished by the number of incident BOUNDARY vertices:



In each case, any BOUNDARY vertices incident on e_i are split into two. Specifically, in case: (**B0**), no action is taken; (**B1**) and (**B2**), one and two vertices are split, respectively, and the resulting change to mesh connectivity causes at least one edge (e_i) but possibly other incident edges to transition FRACTURED-INTERIOR→BOUNDARY.

Appendix C

Fully Nonlinear Thin Plate Forces In the thin plate case, we derive fully non-linear forces, as they are required in our Willmore flow framework. To compute these forces, we drop the assumption of isometric deformations and allow for arbitrary variations of vertices. According to Equation (5), we can write the thin plate bending energy as a sum over contributions from individual edges, where

$$E_{\mathrm{b}}(\mathbf{x})_i = \frac{3|\mathbf{e}_i|^2}{A_i}(1 - \cos(\theta_i)) .$$

Here A_i denotes the combined area of the two triangles meeting at edge $\mathbf{e}_i$, and θ_i is $\mathbf{e}_i$'s dihedral angle. With regards to Figure 2, we focus on the edge $\mathbf{e}_0$ and its hinge stencil, consisting of the two triangles meeting at $\mathbf{e}_0$. We shall drop the subscript 0 wherever this causes no confusion. The non-linear forces arising from edge $\mathbf{e}_0$ with respect to variations associated with vertex $\mathbf{x}_i$ are

$$\mathbf{f}_i^{nl} = \underbrace{3(1 - \cos\theta)\nabla_{\mathbf{x}_i}\left(\frac{|\mathbf{e}_0|^2}{A_0}\right)}_{\mathbf{f}_i^P} + \underbrace{\frac{3|\mathbf{e}_0|^2}{A_0}\sin\theta\,\nabla_{\mathbf{x}_i}\theta}_{\mathbf{f}_i^B} .$$

Notice that we decompose the force $\mathbf{f}_i^{nl}$ into a sum of two parts—a component $\mathbf{f}_i^P$, corresponding to in-plane deformations, and another component, $\mathbf{f}_i^B$, corresponding to pure bending modes. We now provide closed expression for these components.

For the nonlinear bending forces, it has been shown [Wardetzky et al. 2007] that

$$\begin{aligned}
\nabla_{\mathbf{x}_0}\theta &= \frac{-1}{|\mathbf{e}_0|}\left(\cot\alpha_{03}\,\mathbf{n}^{(0)} + \cot\alpha_{04}\,\mathbf{n}^{(1)}\right) , \\
\nabla_{\mathbf{x}_1}\theta &= \frac{-1}{|\mathbf{e}_0|}\left(\cot\alpha_{01}\,\mathbf{n}^{(0)} + \cot\alpha_{02}\,\mathbf{n}^{(1)}\right) , \\
\nabla_{\mathbf{x}_2}\theta &= \frac{1}{|\mathbf{e}_0|}\left(\cot\alpha_{01} + \cot\alpha_{03}\right)\mathbf{n}^{(0)} = \frac{|\mathbf{e}_0|}{2A(T_0)}\mathbf{n}^{(0)} , \\
\nabla_{\mathbf{x}_3}\theta &= \frac{1}{|\mathbf{e}_0|}\left(\cot\alpha_{02} + \cot\alpha_{04}\right)\mathbf{n}^{(1)} = \frac{|\mathbf{e}_0|}{2A(T_1)}\mathbf{n}^{(1)} ,
\end{aligned}$$

where $A(T_i)$ denotes the area of triangle T_i. We note that the above formulas for $\nabla\theta$ are well-known in the literature, see *e.g.*, the work of Bridson *et al.* [2003]; indeed, they can be derived from the fact that $\nabla\theta$ causes no in-plane stretching and is orthogonal to all rigid body modes. This completes our derivation of $\mathbf{f}^B$. The final step of the derivation [Wardetzky et al. 2007] yields the nonlinear in-plane forces

$$\begin{aligned}
\nabla_{\mathbf{x}_0}\left(\frac{|\mathbf{e}_0|^2}{A_0}\right) &= \frac{-2}{A_0}\mathbf{e}_0 + \frac{|\mathbf{e}_0|^2}{2A_0^2}(\mathbf{t}_3 + \mathbf{t}_4) , \\
\nabla_{\mathbf{x}_1}\left(\frac{|\mathbf{e}_0|^2}{A_0}\right) &= \frac{2}{A_0}\mathbf{e}_0 + \frac{|\mathbf{e}_0|^2}{2A_0^2}(\mathbf{t}_1 + \mathbf{t}_2) , \\
\nabla_{\mathbf{x}_2}\left(\frac{|\mathbf{e}_0|^2}{A_0}\right) &= \frac{|\mathbf{e}_0|^2}{2A_0^2}\mathbf{t}_0^{(0)} , \\
\nabla_{\mathbf{x}_3}\left(\frac{|\mathbf{e}_0|^2}{A_0}\right) &= \frac{|\mathbf{e}_0|^2}{2A_0^2}\mathbf{t}_0^{(1)} .
\end{aligned}$$

References

ASCHER, U. M., AND BOXERMAN, E. 2003. On the modified conjugate gradient method in cloth simulation. *Visual Computer 19*, 7-8, 526–531.

BALAY, S., GROPP, W. D., MCINNES, L. C., AND SMITH, B. F. 1996. PETSc 2.0 users manual. Tech. rep., Argonne National Laboratory.

BALAY, S., BUSCHELMAN, K., GROPP, W. D., KAUSHIK, D., KNEPLEY, M., MCINNES, L. C., SMITH, B. F., AND ZHANG, H. 2001. PETSc homepage. http://www.mcs.anl.gov/petsc.

BARAFF, D., AND WITKIN, A. 1998. Large steps in cloth simulation. In *SIGGRAPH*, ACM Press, 43–54.

BERGOU, M., WARDETZKY, M., HARMON, D., ZORIN, D., AND GRINSPUN, E. 2006. A Quadratic Bending Model for Inextensible Surfaces. In *Siggraph/Eurographics Sympos. Geom. Processing*, 227–230.

BLASCKE, W. 1929. *Vorlesungen über Differentialgeometrie*. Springer.

BOBENKO, A. I., , AND SCHRÖDER, P. 2005. Discrete Willmore Flow. *Europgaphics Symposium on Geometry Processing*, 101–110.

BOBENKO, A. I. 2005. A conformal energy for simplicial surfaces. *Combinatorial and Computational Geometry*, 133–143.

BOXERMAN, E., AND ASCHER, U. 2004. Decomposing cloth. In *SCA '04: Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, ACM Press, New York, NY, USA, 153–161.

BRAKKE, K. 1992. The surface evolver. *Exper. Math. 1*, 2, 141–165.

BREEN, D., AND DONALD, H. 1994. House, and Michael J. Wozny. Predicting the drape of woven cloth using interacting particles. *Proceedings of SIGGRAPH '94*, 365–372.

BRIDSON, R., FEDKIW, R., AND ANDERSON, J. 2002. Robust treatment of collisions, contact and friction for cloth animation. *ACM TOG 21*, 3, 594–603.

BRIDSON, R., MARINO, S., AND FEDKIW, R. 2003. Simulation of clothing with folds and wrinkles. *SCA*, 28–36.

CANHAM, P. 1970. The Minimum Energy of Bending as a Possible Explanation of the Biconcave Shape of the Human Red Blood Cell. *Journal of Theoretical Biology 26*, 61–81.

CHOI, K.-J., AND KO, H.-S. 2002. Stable but responsive cloth. *ACM Transactions on Graphics 21*, 3 (July), 604–611.

CHOI, K.-J., AND KO, H.-S. 2003. Extending the immediate buckling model to triangular meshes for simulating complex clothes. In *Eurographics 2003 Short Presentations*, 187–191.

CIARLET, P. 2000. *Mathematical Elasticity, Vol III: Theory of Shells*. North-Holland.

CLARENZ, U., DIEWALD, U., DZIUK, G., RUMPF, M., AND RUSU, R. 2004. A finite element method for surface restoration with smooth boundary conditions. *CAGD*, 427–445.

COHEN-STEINER, D., AND MORVAN, J.-M. 2006. Second fundamental measure of geometric sets and local approximation of curvatures. *J. Differential Geom. 73*, 3, 363–394.

DESBRUN, M., MEYER, M., SCHRÖDER, P., AND BARR, A. H. 1999. Implicit fairing of irregular meshes using diffusion and curvature flow. *SIGGRAPH'99 Conference Proceedings*, 317–324.

DO CARMO, M. 1992. *Riemannian Geometry*. Mathematics: Theory and Applications. Birkhäuser.

ENGLISH, E., AND BRIDSON, R. 2008. Animating developable surfaces using nonconforming elements. In *SIGGRAPH '08: ACM SIGGRAPH 2008 papers*, ACM, New York, NY, USA, 1–5.

ETZMUSS, O., EBERHARDT, B., AND HAUTH, M. 2000. Implicit-explicit schemes for fast animation with particle systems. In *Computer Animation and Simulation 2000*, 138–151.

GARG, A., GRINSPUN, E., WARDETZKY, M., AND ZORIN, D. 2007. Cubic Shells. In *Siggraph/Eurographics Sympos. Comput. Anim.*, 91–98.

GINGOLD, Y., SECORD, A., HAN, J. Y., GRINSPUN, E., AND ZORIN, D. 2004. A Discrete Model for Inelastic Deformation of Thin Shells. Tech. rep., Aug.

GOLDENTHAL, R., HARMON, D., FATTAL, R., BERCOVIER, M., AND GRINSPUN, E. 2007. Efficient Simulation of Inextensible Cloth. *SIGGRAPH (ACM Transactions on Graphics) 26*, 3.

GOVINDARAJU, N. K., KNOTT, D., JAIN, N., KABUL, I., TAMSTORF, R., GAYLE, R., LIN, M. C., AND MANOCHA, D. 2005. Interactive collision detection between deformable models using chromatic decomposition. *ACM Transactions on Graphics 24*, 3 (Aug.), 991–999.

GRINSPUN, E., HIRANI, A. N., DESBRUN, M., AND SCHRÖDER, P. 2003. Discrete shells. *SCA*, 62–67.

GRINSPUN, E., GINGOLD, Y., REISMAN, J., AND ZORIN, D. 2006. Computing discrete shape operators on general meshes. *Comput. Graph. Forum 25*, 3.

HAIRER, E., LUBICH, C., AND WANNER, G. 2006. *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations*. Springer.

HAUTH, M., AND ETZMUSS, O. 2001. A high performance solver for the animation of deformable objects using advanced numerical methods. *Computer Graphics Forum 20*, 3, 319–328.

HAUTH, M., ETZMUSS, O., AND STRASSER, W. 2003. Analysis of numerical methods for the simulation of deformable models. *Visual Computer 19*, 7-8, 581–600.

HAUTH, M. 2004. *Visual Simulation of Deformable Models*. PhD thesis, University of Tübingen.

HELFRICH, W. 1973. Elastic Properties of Lipid Bilayers: Theory and Possible Experiments. *Zeitschrift fr Naturforschung Teil C 28*, 693–703.

HILDEBRANDT, K., AND POLTHIER, K. 2004. Anisotropic filtering of non-linear surface features. *CGF 23*, 3, 391–400.

HSU, L., KUSNER, R., AND SULLIVAN, J. 1992. Minimizing the squared mean curvature integral for surfaces in space forms. *Experiment. Math. 1*, 3, 191–207.

HUGHES, T. J. R. 1987. *Finite Element Method - Linear Static and Dynamic Finite Element Analysis*. Prentice-Hall, Englewood Cliffs.

KAWABATA, S. 1980. *The Standardization and Analysis of Hand Evaluation*. The Hand Evaluation and Standardization Committee, The Textile Machinery Society of Japan.

KECKEISEN, M., KIMMERLE, S., THOMASZEWSKI, B., AND WACKER, M. 2004. Modelling Effects of Wind Fields in Cloth Animations. In *Journal of WSCG*, vol. Vol. 12, 205–212.

KLOSOWSKI, J. T., HELD, M., MITCHELL, J. S. B., SOWIZRAL, H., AND ZIKAN, K. 1998. Efficient collision detection using bounding volume hierarchies of k-dops. *IEEE TVCG 4*, 1 (January-March), 21–36.

MERCAT, C. 2001. Discrete Riemann surfaces and the Ising model. *Communications in Mathematical Physics 218*, 1, 177–216.

MEYER, M., DESBRUN, M., SCHRÖDER, P., AND BARR, A. H. 2003. Discrete differential-geometry operators for triangulated 2-manifolds. In *Visualization and Mathematics III*, H.-C. Hege and K. Polthier, Eds. Springer-Verlag, Heidelberg, 113–134.

MOLINO, N., BAO, Z., AND FEDKIW, R. 2004. A virtual node algorithm for changing mesh topology during simulation. In *SIGGRAPH*, 385–392.

MORINI, B. 1999. Convergence behaviour of inexact newton methods. *Math. of Comp. 68*, 228, 1605–1613.

O'BRIEN, J. F., AND HODGINS, J. K. 1999. Graphical modeling and animation of brittle fracture. In *SIGGRAPH*, 137–146.

PINKALL, U., AND POLTHIER, K. 1993. Computing discrete minimal surfaces and their conjugates. *Experim. Math. 2*, 15–36.

PRESS, W. H., TEUKOLSKY, S. A., VETTERLING, W. T., AND FLANNERY, B. P. 1992. *Numerical Recipes in C: The Art of Scientific Computing*. Cam. Univ. Press.

SCHENK, O., AND GÄRTNER, K. 2004. Solving unsymmetric sparse systems of linear equations with PARDISO. *Journal of Future Generation Computer Systems 20*, 3, 475–487.

SCHNEIDER, R., AND KOBBELT, L. 2001. Geometric Fairing of Irregular Meshes for Free-From Surface Design. *Computer Aided Geometric Design 18*, 359–379.

SIDABRAITE, V., AND MASTEIKAITE, V. 2002. A preliminary study for evaluation of skirt asymmetric drape. *International Journal of Clothing Science and Technology 14*, 5, 286–298.

TERAN, J., SIFAKIS, E., IRVING, G., AND FEDKIW, R. 2005. Robust quasistatic finite elements and flesh simulation. In *2005 ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, 181–190.

TERZOPOULOS, D., AND FLEISCHER, K. 1988. Modeling inelastic deformation: Viscoelasticity, plasticity, fracture. In *SIGGRAPH*, 269–278.

THOMASZEWSKI, B., AND WACKER, M. 2006. Bending Models for Thin Flexible Objects. In *WSCG Short Comm.*

VENTSEL, E., AND KRAUTHAMMER, T. 2001. *Thin Plates and Shells*. CRC Press.

VOLINO, P., AND MAGNENAT-THALMANN, N. 2006. Simple linear bending stiffness in particle systems. In *SCA*, 101–106.

WARDETZKY, M., BERGOU, M., HARMON, D., ZORIN, D., AND GRINSPUN, E. 2007. Discrete Quadratic Curvature Energies. *Computer Aided Geometric Design 24*, 499–518.

WILLMORE, T. J. 2000. Surfaces in conformal geometry. *Annals of Global Analysis and Geometry 18*, 255–264.

YOSHIZAWA, S., AND BELYAEV, A. G. 2002. Fair Triangle Mesh Generation with Discrete Elastica. *Proceedings of the 2nd Biennial International Conference on Geometric Modeling and Processing*, 119–123.

ZIENKIEWICZ, O. C., AND TAYLOR, R. L. 1989. *The finite element method*. McGraw Hill.

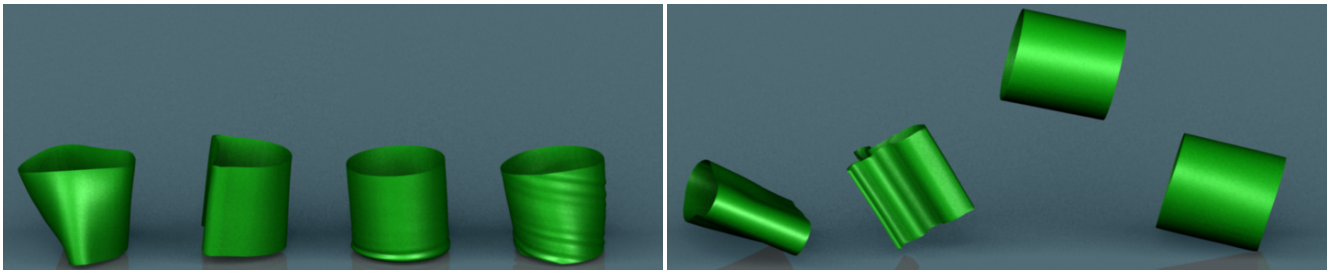


Figure 8: Thin-shell simulation of falling orthotropic cylinders. Material axes vary from left-to-right: isotropic, vertical, horizontal, diagonal. Left Image: stress formed on the cylinders upon impact. Right Image: the effects of anisotropy after impact. Note the extreme flatenning of the cylinder with axis aligned vertically and the high bounce of the cylinder with axis aligned horizontally.

Figure 9: Shear vs. no bending shear: a high bending shear modulus tends to align folds and wrinkles to the material axes (leftmost); a lower shear modulus allows the fabric to fold along other directions, and therefore to obtain a closer fit to the body (second frame). Shearing effects cannot be reproduced by simply tuning Young moduli (3 unsuccessful attempts shown on the right).

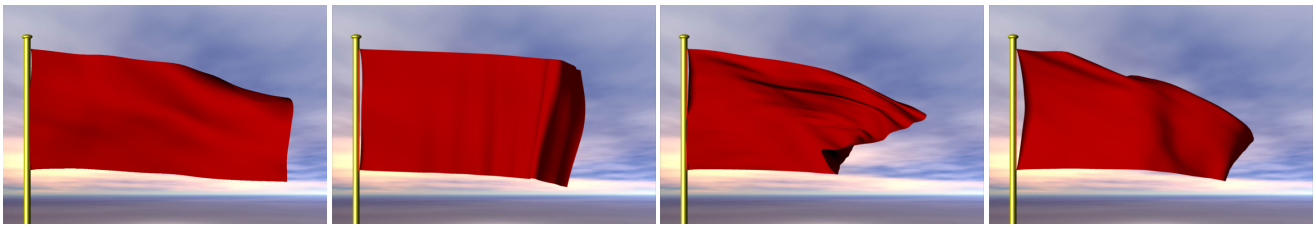


Figure 10: Varying material axes at no additional cost. From left-to-right: isotropic, vertical, horizontal and diagonal.

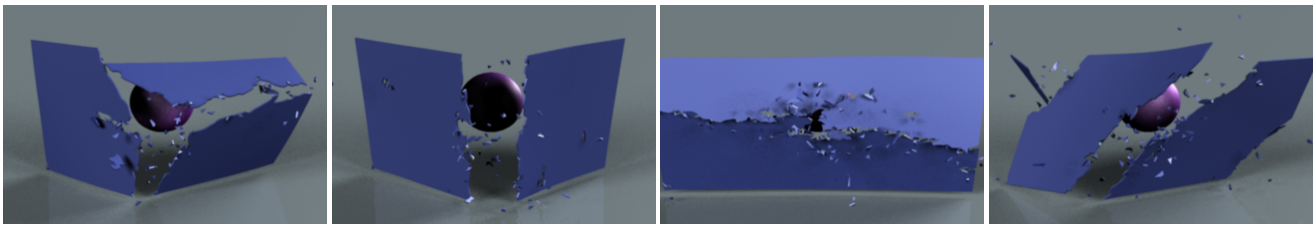


Figure 11: Thin plates: adjustable directions of fracture patterns. Left-to-right: isotropic, vertical, horizontal and diagonal.

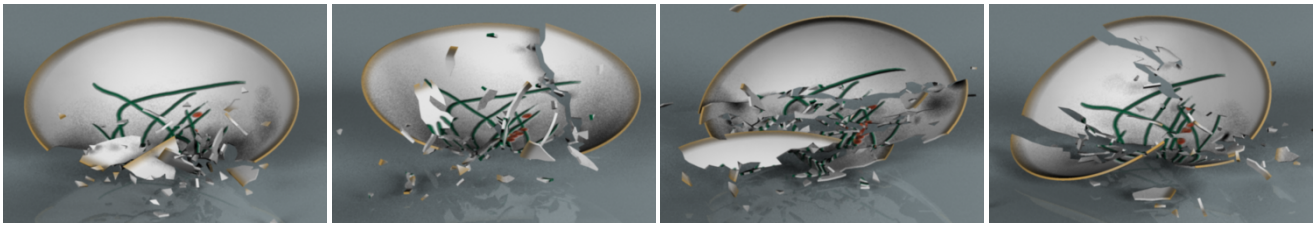


Figure 12: Thin shells: adjustable directions of fracture patterns. Left-to-right: isotropic, vertical, horizontal and diagonal.

Chapter 6: Discrete Elastic Rods

Miklós Bergou, Columbia University
Max Wardetzky, University of Göttingen
Stephen Robinson, Columbia University
Basile Audoly, CNRS / UPMC Univ Paris 06
Eitan Grinspun, Columbia University

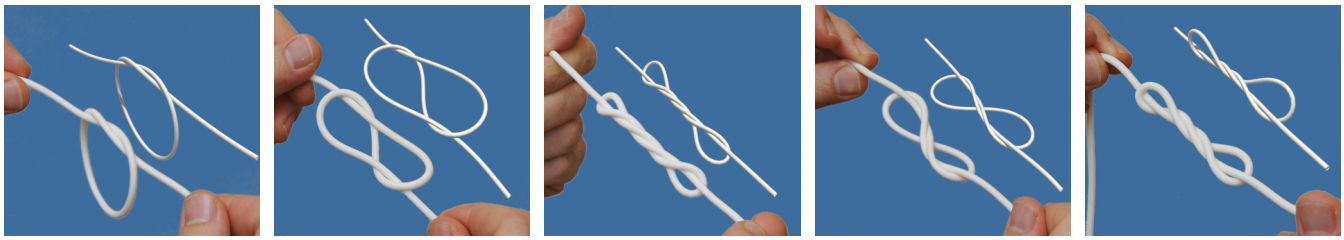


Figure 1: **Experiment and simulation:** A simple (trefoil) knot tied on an elastic rope can be turned into a number of fascinating shapes when twisted. Starting with a twist-free knot (*left*), we observe both continuous and discontinuous changes in the shape, for both directions of twist. Using our model of Discrete Elastic Rods, we are able to reproduce experiments with high accuracy.

Abstract

We present a discrete treatment of adapted framed curves, parallel transport, and holonomy, thus establishing the language for a discrete geometric model of thin flexible rods with arbitrary cross section and undeformed configuration. Our approach differs from existing simulation techniques in the graphics and mechanics literature both in the kinematic description—we represent the material frame by its angular deviation from the natural Bishop frame—as well as in the dynamical treatment—we treat the centerline as dynamic and the material frame as quasistatic. Additionally, we describe a manifold projection method for coupling rods to rigid-bodies and simultaneously enforcing rod inextensibility. The use of quasistatics and constraints provides an efficient treatment for stiff twisting and stretching modes; at the same time, we retain the dynamic bending of the centerline and accurately reproduce the coupling between bending and twisting modes. We validate the discrete rod model via quantitative buckling, stability, and coupled-mode experiments, and via qualitative knot-tying comparisons.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation

Keywords: rods, strands, discrete holonomy, discrete differential geometry

1 Introduction

Recent activity in the field of discrete differential geometry (DDG) has fueled the development of simple, robust, and efficient tools for geometry processing and physical simulation. The DDG approach to simulation begins with the laying out of a physical model that is discrete from the ground up; the primary directive in designing this model is a focus on the preservation of key geometric structures that characterize the actual (smooth) physical system [Grinspun 2006].

Notably lacking is the application of DDG to physical modeling of *elastic rods*—curve-like elastic bodies that have one dimension ("length") much larger than the others ("cross-section"). Rods have many interesting potential applications in animating knots, sutures, plants, and even kinematic skeletons. They are ideal for modeling deformations characterized by stretching, bending, and twisting. Stretching and bending are captured by the deformation of a curve called the *centerline*, while twisting is captured by the rotation of a *material frame* associated to each point on the centerline.

1.1 Goals and contributions

Our goal is to develop a principled model that is (a) simple to implement and efficient to execute and (b) easy to validate and test for convergence, in the sense that solutions to static problems and trajectories of dynamic problems in the discrete setup approach the solutions of the corresponding smooth problem. In pursuing this goal, this paper advances our understanding of discrete differential geometry, physical modeling, and physical simulation.

Elegant model of elastic rods We build on a representation of elastic rods introduced for purposes of analysis by Langer and Singer [1996], arriving at a *reduced coordinate* formulation with a minimal number of degrees of freedom for extensible rods that represents the centerline of the rod explicitly and represents the material frame using only a scalar variable (§4.2). Like other reduced coordinate models, this avoids the need for stiff constraints that couple the material frame to the centerline, yet unlike other (*e.g.*, curvature-based) reduced coordinate models, the explicit centerline representation facilitates collision handling and rendering.

Efficient quasistatic treatment of material frame We additionally emphasize that the speed of sound in elastic rods is much faster for twisting waves than for bending waves. While this has long been established, to the best of our knowledge it has not been used to simulate general elastic rods. Since in most applications the slower waves are of interest, we treat the material frame *quasistatically* (§5). When we combine this assumption with our reduced coordinate representation, the resulting equations of motion (§7) become very straightforward to implement and efficient to execute.

Geometry of discrete framed curves and their connections Because our derivation is based on the concepts of DDG, our discrete model retains very distinctly the geometric structure of the smooth setting—in particular, that of parallel transport and the forces induced by the variation of holonomy (§6). We introduce

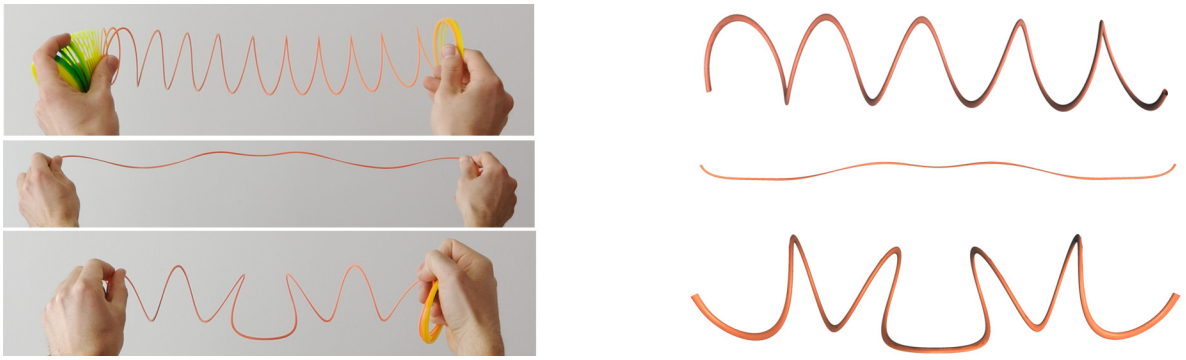


Figure 2: **Helical perversion in experiment and simulation:** starting from the natural shape of a Slinky® (top) we first remove writhe by pinching the flat cross-section between two fingers and traveling from one end to the other, arriving at a fully stretched, almost flat ribbon (middle). By bringing the ends together, a persistent perversion forms (bottom), whose shape is surprisingly different from the natural shape.

simple algebraic tools and mnemonic diagrams that make it possible to carry out in a methodical manner derivations involving the discrete connection induced by parallel transport. These tools are the central building blocks for deriving forces associated to displacements of the rod's centerline and twist of the material frame.

Inextensibility and encapsulated coupling with rigid-bodies When simulating inextensible rods, it becomes profitable to enforce the rod's inextensibility via constraints rather than stiff penalty forces. Also, in general simulation applications, and in computer animation and gaming, it is often useful to enforce boundary conditions by *coupling* the rod to another physical system—most naturally, to a rigid-body, which carries exactly the degrees of freedom as any point on an elastic rod (*i.e.*, position and orientation). Our approach in §8 handles both types of constraints, emphasizing physical correctness as well as an important software reuse principle: the method of coupling allows our rod simulation to interact with any existing rigid-body simulation without internal modification of either the rod or rigid-body codes.

Validation We show how to model and simulate inextensible elastic rods with arbitrary curved undeformed centerline and anisotropic bending response (§7), and we show the simplifications that occur for naturally straight rods with isotropic bending response. We validate our model against known analytic solutions and present empirical evidence supporting the good convergence behavior of our discrete model to its smooth counterpart (§9).

2 Related work

Mathematical analysis, modeling, and simulation of elastic rods is an active field in mechanics [van der Heijden et al. 2003; Goyal et al. 2007], numerical analysis [Falk and Xu 1995], and geometry [Bobenko and Suris 1999; Lin and Schwetlick 2004] with applications spanning medicine [Lenoir et al. 2006], biology [Yang et al. 1993; Klapper 1996], the study of knots [Phillips et al. 2002; Brown et al. 2004], and computer graphics [Chang et al. 2007]. It is impossible to survey the many works in this area in a brief space, so we discuss only the most closely-related works. For a broader starting point, refer to the expositions of Rubin [2000] and Maddocks [1984; 1994]. For the state of the art in strand and hair simulation, refer to the survey by Ward *et al.* [2007] and the course notes of Hadap *et al.* [2007].

In mechanical engineering, elastic rods are typically treated with a finite difference [Klapper 1996] or finite element [Yang et al. 1993; Goyal et al. 2007] discretization of the smooth equations. Goldstein and Langer [1995] observed that using the Bishop frame simplifies both the analytical formulation and the numerical implementation of the dynamics of symmetric rods.

In graphics, Terzopoulos *et al.* [1987] introduced tensorial treatments of elastica, and Pai [2002] applied a discretization of the Cosserat rod model to simulate a strand. Bertails *et al.* [2006] used a piecewise helical discretization to produce compelling animations of curly hair using few elements per strand. Hadap [2006] considered a serial multi-body chain and used differential algebraic equations to treat the attendant numerical stiffness. These recent works used an *implicit* centerline representation based on reduced (curvature) coordinates.

By contrast, Choe *et al.* [2005] represented the centerline *explicitly* by a sequence of edges connected by linear and torsional springs. Grégoire and Schömer [2007] proposed an explicit centerline discretization coupled to a quaternionic material frame representation. Spillmann and Teschner [2007; 2008] built on this idea in their development of algorithms for dynamic contact. Theetten *et al.* [2006] presented a geometric spline-based approach that can model large-displacement plastic deformations. These authors argue that an explicit (displacement) representation of the centerline facilitates the simulation of complex contact scenarios and looping phenomena. We share this view.

3 Overview

In many applications, the rod tends to bend or twist rather than to stretch; therefore, the case of *inextensible* rods is prevalent and im-

Algorithm 1 Discrete elastic rod simulation

Require: $\mathbf{u}^0$ *// Bishop frame vector in frame* $\{\mathbf{t}^0, \mathbf{u}^0, \mathbf{v}^0\}$ *at edge 0*
Require: $\bar{\mathbf{x}}_0 \dots \bar{\mathbf{x}}_{n+1}$ *// position of centerline in rest state*
Require: $(\mathbf{x}_0, \dot{\mathbf{x}}_0) \dots (\mathbf{x}_{n+1}, \dot{\mathbf{x}}_{n+1})$ *// initial position/velocity of centerline*
Require: boundary conditions *// free, clamped or body-coupled ends*

precompute $\bar{\boldsymbol{\omega}}_i^j$ using (2)
set quasistatic material frame (§5.1)
while simulating **do**
  apply torque to rigid-body (§8.2)
  integrate rigid-body (external library) *// [Smith 2008]*
  compute forces on centerline (§7.1)
  integrate centerline (§7.2) *// [Hairer et al. 2006]*
  enforce inextensibility and rigid-body coupling (§8)
  collision detection and response *// [Spillmann and Teschner 2008]*
  update Bishop frame (§4.2.2)
  update quasistatic material frame (§5.1)
end while

portant. We visualize an inextensible rod as an infinitesimally thin, fixed centerline that can bend but not stretch and that is surrounded by a finite, but thin, elastic material. We consider the general case of naturally *curved* rods with *anisotropic* bending response and note the simplifications that occur in the special case of naturally *straight* rods with *isotropic* bending response. We also present a simple and efficient method for attaching a rod to a rigid-body (§8). The different components of our method are summarized in Algorithm 1. Barred quantities are precomputed and subsequently held fixed.

4 Kirchhoff rods

4.1 Smooth setting

Framed-curve representation We describe the configuration of a rod by an *adapted framed curve* $\Gamma = \{\boldsymbol{\gamma}; \mathbf{t}, \mathbf{m_1}, \mathbf{m_2}\}$ (see Fig. 3). Here $\boldsymbol{\gamma}(s)$ is an arc length parameterized curve in $\mathbb{R}^3$ describing the rod's *centerline*; the assignment of an orthonormal material frame $\{\mathbf{t}(s), \mathbf{m_1}(s), \mathbf{m_2}(s)\}$ to each point on the centerline contains the requisite information for measuring twist. The material frame satisfies $\mathbf{t}(s) = \boldsymbol{\gamma}'(s)$, *i.e.*, it is *adapted* to the centerline such that the first material axis is tangent to the curve. We will refer to $\boldsymbol{\kappa} = \mathbf{t}'$ as the centerline's *curvature (normal) vector*.

Elastic energy The Kirchhoff theory of elastic rods assigns an elastic energy, $E(\Gamma)$, to any adapted framed curve Γ. This energy is assembled from three scalar functions that measure *strain*—given by the change of the orthonormal frame $\{\mathbf{t}(s), \mathbf{m_1}(s), \mathbf{m_2}(s)\}$ expressed in its own coordinates:

$$\omega_1 = \mathbf{t}' \cdot \mathbf{m_1}, \qquad \omega_2 = \mathbf{t}' \cdot \mathbf{m_2}, \quad \text{and} \quad m = \mathbf{m_1}' \cdot \mathbf{m_2} .$$

Notice that since $\mathbf{t}' = \boldsymbol{\kappa}$, the first two of the above terms, ω_1 and ω_2, represent the rod's curvature vector expressed in material coordinates and measure the *bending* of material frame. The last term, m, refers to the *twist* of the material frame around the tangent. Accordingly, total elastic energy contains bending and twisting contributions:

$$E(\Gamma) = E_{\text{bend}}(\Gamma) + E_{\text{twist}}(\Gamma) .$$

The classical Kirchhoff equations for rods are obtained from this type of energy using Lagrangian mechanics (see, *e.g.*, [Audoly and Pomeau 2008]). The goal of the present paper is to derive a discrete form of these equations.

Our assumption that s is an arc length parameterization implies that the rod is *inextensible*; therefore, we do not include a stretching energy. Instead, we enforce inextensibility via an auxiliary constraint (§8). It is straightforward to drop this assumption by also including a stretching term.

4.1.1 Bending energy

When the rod's undeformed configuration is *straight* (as opposed to curved) and the bending response is *isotropic* (as opposed to giving preference to some bending directions over others), the *bending energy* takes the simple form

$$E_{\text{bend}}(\Gamma) = \frac{1}{2}\int \alpha \boldsymbol{\omega}^2 \mathrm{d}s = \frac{1}{2}\int \alpha \boldsymbol{\kappa}^2 \mathrm{d}s ,$$

where the 2-vector $\boldsymbol{\omega} = (\omega_1, \omega_2)^T$ represents the centerline curvature vector expressed in the material frame coordinates and α is the rod's bending modulus.

We generalize this to *anisotropic* bending response by replacing the (isotropic) dot product with a general quadratic form $\overline{B}$ (a symmetric positive definite 2×2 matrix), so that $\boldsymbol{\omega}^T \overline{B} \boldsymbol{\omega}$ is the bending energy density. We do not assume that $\overline{B}$ is diagonal—in fact, we determine $\overline{B}$ by requiring that the undeformed material frame is a Bishop frame (see below). We also generalize to *naturally curved rods* by subtracting away the undeformed centerline curvature $\overline{\boldsymbol{\omega}}$, *i.e.*, using $(\boldsymbol{\omega} - \overline{\boldsymbol{\omega}})$ in place of $\boldsymbol{\omega}$ above.

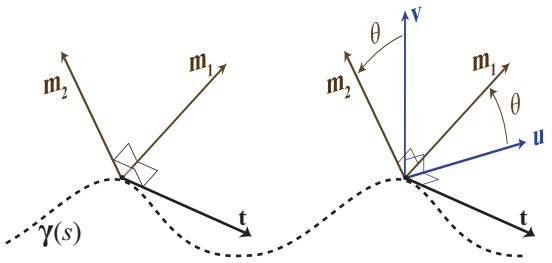


Figure 3: **Adapted framed curve** *(Left)* The configuration of an elastic rod is represented by a curve $\boldsymbol{\gamma}(s)$ and a material frame $\{\mathbf{t}(s), \mathbf{m_1}(s), \mathbf{m_2}(s)\}$. *(Right)* The material frame is encoded by an angle of rotation θ relative to the natural Bishop frame $\{\mathbf{t}(s), \mathbf{u}(s), \mathbf{v}(s)\}$.

Putting all this together, we have

$$E_{\text{bend}}(\Gamma) = \frac{1}{2}\int (\boldsymbol{\omega} - \overline{\boldsymbol{\omega}})^T \overline{B} (\boldsymbol{\omega} - \overline{\boldsymbol{\omega}}) \mathrm{d}s ,$$

where barred quantities refer to the undeformed configuration. The particular case of an isotropic, naturally straight rod is recovered by taking $\overline{B} = \alpha\, Id_{2\times 2}$ and $\overline{\boldsymbol{\omega}} = \mathbf{0}$.

4.1.2 Twisting energy

Letting $m = \mathbf{m_1}' \cdot \mathbf{m_2}$ denote the *twist* of the material frame about the centerline, the *twisting energy* is given by

$$E_{\text{twist}}(\Gamma) = \frac{1}{2}\int \beta m^2 \mathrm{d}s .$$

The formula $m = \mathbf{m_1}' \cdot \mathbf{m_2}$ gives an expression for the twist in terms of material vectors immersed in ambient space. We now seek an equivalent expression that exposes a reduced set of coordinates.

Parallel transport and the Bishop (natural) frame Given a fixed centerline, consider the task of assigning to it the geometrically most *natural* (and physically the most *relaxed*) frame. The *Bishop frame* $\{\mathbf{t}(s), \mathbf{u}(s), \mathbf{v}(s)\}$ for a given centerline is an adapted frame with zero twist uniformly, *i.e.*, $\mathbf{u}' \cdot \mathbf{v} = -\mathbf{v}' \cdot \mathbf{u} = 0$. The assignment of an adapted frame to one point on the curve uniquely fixes the Bishop frame throughout the curve. Our convention is to assign the Bishop frame at the first endpoint ($s = 0$) of the curve.

Consider traversing the centerline from one end to the other at unit speed. The evolution of the Bishop frame (and any other orthonormal frame) can be described in terms of its *Darboux vector*, $\Omega(s)$:

$$\mathbf{t}' = \Omega \times \mathbf{t} , \quad \mathbf{u}' = \Omega \times \mathbf{u} , \quad \text{and} \quad \mathbf{v}' = \Omega \times \mathbf{v} .$$

Since by the definition of the Bishop frame, $\mathbf{u}' \cdot \mathbf{v} = 0$, it follows from the second equation that Ω has no tangential component (along $\mathbf{t}$). This, together with the first equation implies that $\Omega = \kappa \mathbf{b}$, where $\kappa \mathbf{b} = \mathbf{t} \times \boldsymbol{\kappa}$ is the *curvature binormal* along the centerline.

The Darboux vector of the Bishop frame serves to define *parallel transport*, a concept that plays a central role in the remainder of this paper. We parallel transport a vector $\mathbf{x}$ from one point on the centerline to another by integrating the equation $\mathbf{x}' = \kappa \mathbf{b} \times \mathbf{x}$.

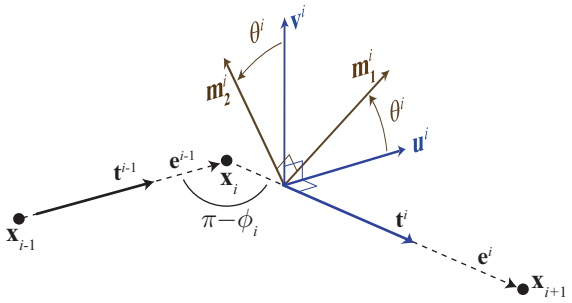


Figure 4: **Notation** Angles and vectors used in our discrete model.

Thus, infinitesimally, parallel transport corresponds to a *rotation about the binormal*—a concept that we will use again in our discrete model. Parallel transport keeps the tangential component of $\mathbf{x}$ tangential, and evolves the cross-sectional component of $\mathbf{x}$ via a tangential velocity, in particular without rotating the cross-section about the centerline. Observe that the three Bishop axes evolve under parallel transport.

Curve-angle representation The (twist-free) Bishop frame allows for a simple parameterization of the material frame [Langer and Singer 1996]. Let $\theta(s)$ be the scalar function that measures the rotation about the tangent of the material frame relative to the Bishop frame (see Fig. 3):

$$\mathbf{m_1} = \cos\theta\cdot\mathbf{u} + \sin\theta\cdot\mathbf{v}\,, \qquad \mathbf{m_2} = -\sin\theta\cdot\mathbf{u} + \cos\theta\cdot\mathbf{v}\,.$$

The key observation is that twist, m, can be expressed in terms of θ by $m(s) = \theta'(s)$. This relation is easily derived using the facts that $m = (\mathbf{m_1})'\cdot\mathbf{m_2}$ and $\mathbf{u}'\cdot\mathbf{v} = 0$. Hence, we write the twist energy as

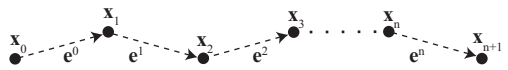
$$E_{\text{twist}}(\Gamma) = \frac{1}{2}\int \beta(\theta')^2 \mathrm{d}s\,.$$

Observe that we have expressed the elastic energy of Kirchhoff rods by two dominant players: the position of the centerline, $\boldsymbol{\gamma}$, and the angle of rotation, θ, between the Bishop and the material frame. This reduced coordinate representation is the cornerstone of our discrete theory.

4.2 Discrete Kirchhoff rods

When painting the discrete picture, our guiding principle is to seek a viewpoint that builds on the *same geometric principles* as the corresponding smooth theory.

Primal vs. dual We distinguish between *primal quantities*, associated with vertices and decorated with *lower indices*, and *dual quantities*, associated with edges and decorated with *upper indices*. This diagram summarizes our indexing and notation conventions:



Discrete framed curves A discrete framed curve, Γ, consists of a centerline comprised of $(n+2)$ vertices $\mathbf{x}_0,\ldots,\mathbf{x}_{n+1}$ and $(n+1)$ straight edges $\mathbf{e}^0,\ldots,\mathbf{e}^n$ such that $\mathbf{e}^i = \mathbf{x}_{i+1} - \mathbf{x}_i$, together with an assignment of *material frames* $\mathbf{M}^i = \{\mathbf{t}^i, \mathbf{m}_\mathbf{1}^i, \mathbf{m}_\mathbf{2}^i\}$ per edge (see Fig. 4). We consider *adapted* frames, where $\mathbf{t}^i = \mathbf{e}^i/|\mathbf{e}^i|$ is the *unit tangent* vector per edge. Notice that tangent vectors of polygonal curves are uniquely defined at edges, whereas their definition at vertices would be ambiguous. Since tangent vectors belong to the triad that makes up frames on curves, we naturally assign *frames* to *edges*, rather than to vertices.

Pointwise vs. integrated quantities In the smooth setting, we consider certain quantities (*i.e.*, curvature and twist) at each point on the rod, and we define the energy by integrating over the length of the rod. By contrast, in the discrete setting, certain quantities often emerge naturally in an *integrated* rather than a pointwise way. For example, relating discrete curvature to the turning angle between incident edges corresponds to an integration over the curve's Gauss image [Grinspun 2006].

We refer to an *integrated* quantity as a measure that associates a number to a domain $\mathscr{D}$ of the curve, corresponding to an integral of a function over that domain. To convert an integrated quantity back to a pointwise one, we simply divide by the length, $|\mathscr{D}|$, of the domain of integration. In the discrete case, we define $\mathscr{D}_i$ as the Voronoi region associated to each vertex, having length $|\mathscr{D}_i| = l_i/2$, where $l_i = |\mathbf{e}^{i-1}| + |\mathbf{e}^i|$.

4.2.1 Discrete bending energy

Recall that the key player for formulating elastic bending energy in the smooth case was the curvature of the centerline.

Discrete curvature Since each edge is straight, it follows that discrete curvature is naturally associated with vertices. Letting ϕ_i denote the turning angle between two consecutive edges (see Fig. 4) we define (integrated) curvature by

$$\kappa_i = 2\tan\frac{\phi_i}{2}\,.$$

We will see in §6 that this definition of discrete curvature emerges naturally from an analysis of the geometry of discrete rods, and to be consistent throughout, we will use this definition in the bending energy. Note that $\kappa_i \to \infty$ as incident edges bend toward each other, so this measure of curvature will penalize sharp kinks in the rod.

Discrete curvature binormal We define the curvature binormal at a vertex (an *integrated quantity*) by

$$(\kappa\mathbf{b})_i = \frac{2\mathbf{e}^{i-1}\times\mathbf{e}^i}{|\bar{\mathbf{e}}^{i-1}||\bar{\mathbf{e}}^i| + \mathbf{e}^{i-1}\cdot\mathbf{e}^i}\,. \tag{1}$$

Observe that the vector $(\kappa\mathbf{b})_i$ is orthogonal to the osculating plane passing through two consecutive edges and has magnitude $2\tan(\phi_i/2)$. In particular, this quantity is well-defined in the case of collinear edges, even though the binormal itself is not.

Discrete bending energy We now have all the pieces to assemble the bending energy of a discrete naturally straight, isotropic rod:

$$E_{\text{bend}}(\Gamma) = \frac{1}{2}\sum_{i=1}^{n}\alpha\left(\frac{\kappa\mathbf{b}_i}{\bar{l}_i/2}\right)^2\frac{\bar{l}_i}{2} = \sum_{i=1}^{n}\frac{\alpha(\kappa\mathbf{b}_i)^2}{\bar{l}_i}\,.$$

The division by $\bar{l}_i/2$ is due to converting the integrated quantity $\kappa\mathbf{b}_i$ into a pointwise one (see above), whereas the multiplication by $\bar{l}_i/2$ takes care of the measure of the domain of integration, $\mathscr{D}_i$.

Recall that for an anisotropic bending response, we require the *material curvatures* given by inner products between curvature vectors and material frame vectors. Using the curvature *binormal* $(\kappa\mathbf{b})_i$ as defined by (1), we express the material curvatures as

$$\boldsymbol{\omega}_i^j = \left((\kappa\mathbf{b})_i\cdot\mathbf{m}_\mathbf{2}^j, -(\kappa\mathbf{b})_i\cdot\mathbf{m}_\mathbf{1}^j\right)^T \quad \text{for} \quad j\in\{i-1,i\}\,. \tag{2}$$

Here, following our primal-dual notation, the double indices—upper and lower—correspond to an edge-vertex pair. Barred quantities we denote evaluation on the undeformed configuration.

With these definitions, the bending energy of a discrete rod is

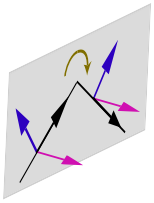
$$E_{\text{bend}}(\Gamma) = \sum_{i=1}^{n} \frac{1}{2\bar{l}_i} \sum_{j=i-1}^{i} \left(\boldsymbol{\omega}_i^j - \overline{\boldsymbol{\omega}}_i^j\right)^T \overline{B}^j \left(\boldsymbol{\omega}_i^j - \overline{\boldsymbol{\omega}}_i^j\right) . \quad (3)$$

This formulation allows for a general (anisotropic) bending response and a general (curved) undeformed configuration. As in the smooth case, the simple expression of the special case is recovered by taking $\overline{B}^j = \alpha\, Id_{2\times 2}$ and $\overline{\boldsymbol{\omega}}_i^j = \mathbf{0}$.

4.2.2 Discrete twisting energy

To formulate the discrete twisting energy, we follow the same progression as in the smooth setting, first laying out the notions of parallel transport and the Bishop frame, and then introducing the angle of rotation to the material frame.

Discrete parallel transport and Bishop frame We define discrete parallel transport in analogy to the smooth case, *i.e.*, as a rotation P_i about the curvature binormal,



$$P_i\left(\mathbf{t}^{i-1}\right) = \mathbf{t}^i, \quad P_i\left(\mathbf{t}^{i-1}\times\mathbf{t}^i\right) = \mathbf{t}^{i-1}\times\mathbf{t}^i .$$

By convention, P_i is the identity if $\mathbf{t}^{i-1} = \mathbf{t}^i$, whereas P_i is not defined if $\mathbf{t}^{i-1} = -\mathbf{t}^i$. Parallel transport is a key notion that allows us to generalize the notion of twist from the smooth to the discrete setting.

In order to define *Bishop frames*, we draw upon the smooth case by transporting a unit vector $\mathbf{u}^0$, which is orthogonal to $\mathbf{t}^0$, along the rod using our rotation operators P_i, *i.e.*, we iteratively define

$$\mathbf{u}^i = P_i\left(\mathbf{u}^{i-1}\right) .$$

The third axis of each Bishop frame is then $\mathbf{v}^i = \mathbf{t}^i \times \mathbf{u}^i$.

Bishop frame update The Bishop frame is by definition *adapted* to the centerline. This requires that $\mathbf{u}^0 \perp \mathbf{t}^0$, a condition that must be maintained during simulation. Each simulation step moves the centerline to a new position, so that in general after Algorithm step 9 the requirement $\mathbf{u}^0 \perp \mathbf{t}^0$ may be violated (the exception is when $\mathbf{t}^0$ is clamped). We re-adapt the frame by parallel transporting the Bishop vector (in time)—in analogy to the parallel transport along the centerline—by computing a rotation operation.

Material frame representation Since both the material frame and Bishop frame are defined at edges, let θ^i denote the *angle* needed to rotate the Bishop frame into the material frame at edge $\mathbf{e}^i$ (see Fig. 4). Then the material frame *vectors* at edge i are

$$\mathbf{m}_\mathbf{1}^i = \cos\theta^i \cdot \mathbf{u}^i + \sin\theta^i \cdot \mathbf{v}^i , \text{ and } \mathbf{m}_\mathbf{2}^i = -\sin\theta^i \cdot \mathbf{u}^i + \cos\theta^i \cdot \mathbf{v}^i .$$

Discrete twisting energy In perfect analogy to the *curve-angle representation* in the smooth case, we adopt the twisting energy

$$E_{\text{twist}}(\Gamma) = \sum_{i=1}^{n} \beta \frac{(\theta^i - \theta^{i-1})^2}{\bar{l}_i} = \sum_{i=1}^{n} \frac{\beta m_i^2}{\bar{l}_i} ,$$

where the scalar $m_i = \theta^i - \theta^{i-1}$ is the (*integrated*) *discrete twist*. The variable m_i measures the angle between two adapted frames: the result of parallel transporting the material frame from edge $\mathbf{e}^{i-1}$ to $\mathbf{e}^i$ and the material frame at edge $\mathbf{e}^i$ itself.

5 Quasistatic material frame postulation

Our goal is to arrive at equations describing the time-evolution of the centerline of the rod. We first pave the way with an important simplifying assumption.

The speed of a twist wave increases as the rotational inertia of the cross section vanishes. By contrast, bending waves are dispersive [Sommerfeld 1964]—their speed depends on their wavelength, λ—with velocity vh/λ, where h is the rod radius, and v is solid material's speed of sound. Consequently, twist waves travel faster than bending waves, for bending waves with large wavelengths $\lambda \gg h$, *i.e.*, much larger than the rod radius. For the problems we consider, the relevant *dynamics* are in (large-wavelength) bending modes, and it is wasteful of computation to resolve the temporal evolution of the twist waves.

Consider the limit of a vanishing cross-sectional rotational inertia: twist waves propagate instantly. At any instant in time, the material frames are the minimizer of elastic energy, subject to any given boundary conditions on the material frame:

$$\frac{\partial E(\Gamma)}{\partial \theta^j} = 0 , \quad (4)$$

Note that the quasistatic treatment of the material frame eliminates the θ^j variables as degrees of freedom from the system: given the rod's centerline and the boundary conditions on the material frame, (4) is used to determine what the material frame must be.

Boundary conditions The value of θ^j for an edge may be *prescribed* by a given boundary condition on the material frame. In practice, we consider *stressfree* ends (*i.e.*, no boundary conditions) or *clamped* ends (*i.e.*, assigning the material frame for $j = 0$ and $j = n$). We use (4) for all θ^j variables whose value is not prescribed by a boundary condition, ensuring that the number of equations matches the number of unknowns for the angles θ^j.

5.1 Quasistatic update

During simulation, we enforce the quasistatic nature of the material frame by ensuring that (4) is satisfied before forces are computed.

Special case of naturally straight, isotropic rods For an isotropic rod with a naturally straight undeformed configuration, (4) implies that a clamped rod has uniform twist,

$$\frac{m_i}{\bar{l}_i} = \frac{\theta^n - \theta^0}{2\bar{L}} = constant , \quad (5)$$

where $2\bar{L} = \sum_{i=1}^{n} \bar{l}_i$, and the pointwise twist $m_i/\bar{l}_i$ (recall §4.2.2) depends only on the *boundary conditions*: the difference of angles of the end edges, $\theta^n - \theta^0$. Substituting the above into the formula for $E(\Gamma)$ gives the simplified expression

$$E(\Gamma) = \sum_{i=1}^{n} \frac{\alpha(\kappa\mathbf{b})_i^2}{\bar{l}_i} + \frac{\beta\left(\theta^n - \theta^0\right)^2}{2\bar{L}} . \quad (6)$$

Observe that the twist energy depends only on the angle between material and Bishop frame at the *boundary edges* of the rod. For the naturally straight, isotropic case, the above implies that a rod with stressfree ends has no twist.

General case For the general case, we use Newton's method to minimize the elastic energy $E(\Gamma)$ with respect to the material

Figure 5: **Asymmetry of twist:** anisotropy of the cross-sections and natural curvature of the centerline "conspire" to produce non-uniform twist distribution. *From top to bottom*: reference configuration, moderate twist, large twist.

frames, which requires both the gradient and Hessian of the energy with respect to the θ^j variables. The gradient is given by

$$\frac{\partial E(\Gamma)}{\partial \theta^j} = \frac{\partial}{\partial \theta^j}\left(W_j + W_{j+1}\right) + 2\beta\left(\frac{m_j}{\bar{l}_j} - \frac{m_{j+1}}{\bar{l}_{j+1}}\right), \qquad (7)$$

$$\text{where} \quad \frac{\partial}{\partial \theta^j} W_i = \frac{1}{\bar{l}_i}(\boldsymbol{\omega}_i^j)^T \mathrm{J}\bar{B}^j(\boldsymbol{\omega}_i^j - \bar{\boldsymbol{\omega}}_i^j).$$

To derive the latter identity, note that $\partial \mathbf{m}_1^j/\partial\theta^j = \mathbf{m}_2^j$ and $\partial \mathbf{m}_2^j/\partial\theta^j = -\mathbf{m}_1^j$ to obtain $\partial \boldsymbol{\omega}_i^j/\partial\theta^j = -\mathrm{J}\boldsymbol{\omega}_i^j$, where J acts on two dimensional vectors by counter-clockwise $\pi/2$ rotation. As expected, (5) is a special case of (7).

The Hessian, H, is obtained by differentiating (7) with respect to θ^{j-1}, θ^j, and θ^{j+1}. The resulting components of the Hessian are:

$$H_{j,j-1} = -\frac{2\beta}{\bar{l}_j}, \qquad H_{j,j+1} = -\frac{2\beta}{\bar{l}_{j+1}},$$

$$H_{j,j} = \frac{2\beta}{\bar{l}_j} + \frac{2\beta}{\bar{l}_{j+1}} + \frac{\partial^2}{(\partial\theta^j)^2}\left(W_j + W_{j+1}\right),$$

$$\text{where} \quad \frac{\partial^2}{(\partial\theta^j)^2} W_i = \frac{1}{\bar{l}_i}(\boldsymbol{\omega}_i^j)^T \mathrm{J}^T \bar{B}^j \mathrm{J}(\boldsymbol{\omega}_i^j) - \frac{1}{\bar{l}_i}(\boldsymbol{\omega}_i^j)^T \bar{B}^j(\boldsymbol{\omega}_i^j - \bar{\boldsymbol{\omega}}_i^j).$$

Note that the Hessian is tridiagonal and so it can be factored in $O(n)$ time. Additionally, in order to accelerate the convergence of Newton's method, we combine it with a line search along the Newton direction [Press et al. 2002]. Finally, we note that Newton's method converges quickly: the θ^j variables do not change significantly between time steps, so that previous values are a good initial guess.

Forward We presented the bending and twisting energy of a smooth Kirchhoff rod and constructed by direct analogy the corresponding energies for a discrete Kirchhoff rod. The challenge ahead is to derive the *elastic forces* induced by the gradients of these energies. These forces will include terms that arise from the dependence of the vectors $\mathbf{u}^j$ and $\mathbf{v}^j$ on centerline positions, $\mathbf{x}_i$. Since these Bishop vectors are defined via parallel transport, we must consider the variation of parallel transport with respect to moving the centerline. For this, we turn to the concept of holonomy.

6 Discrete holonomy

Holonomy is a classical concept from differential geometry: it measures the deficit of geometric data (frames) to close up when parallel transported around a closed loop. In our case (of discrete rods), holonomy depends—just like parallel transport itself—only on the centerline; it measures the (scalar) angle when parallel transporting an adapted frame along a closed loop of discrete edges. Indeed, the fact that holonomy can be expressed as a *scalar* is the reason why it is such a useful concept for computing forces.

Consider the variation of two consecutive edges $\mathbf{e}^{i-1}(\varepsilon)$ and $\mathbf{e}^i(\varepsilon)$ with tangents $\mathbf{t}^{i-1}(\varepsilon)$ and $\mathbf{t}^i(\varepsilon)$, and parallel transport $P_i(\varepsilon)$, where ε denotes the variation parameter. We denote by $\tilde{P}^i(\varepsilon)$ the parallel transport from $\mathbf{t}^i(0)$ to the deformed configuration, $\mathbf{t}^i(\varepsilon)$, *i.e.*, the rotation that satisfies $\tilde{P}^i(0) = Id$ and

$$\tilde{P}^i(\varepsilon)\left(\mathbf{t}^i(0)\right) = \mathbf{t}^i(\varepsilon) \quad \text{and} \quad \tilde{P}^i(\varepsilon)\left(\mathbf{t}^i(0) \times \mathbf{t}^i(\varepsilon)\right) = \mathbf{t}^i(0) \times \mathbf{t}^i(\varepsilon).$$

Now consider the concatenation of parallel transports given by

$$R^{i-1}(\varepsilon) = \left(\tilde{P}^{i-1}(\varepsilon)\right)^T \circ \left(P_i(\varepsilon)\right)^T \circ \tilde{P}^i(\varepsilon) \circ P_i(0), \qquad (8)$$

which we depict by the following *mnemonic diagram*, where arrows represent parallel transport:

$$\begin{array}{ccc} \mathbf{t}^{i-1}(\varepsilon) & \longrightarrow & \mathbf{t}^i(\varepsilon) \\ \uparrow & \psi_i(\varepsilon) & \uparrow \\ \mathbf{t}^{i-1}(0) & \longrightarrow & \mathbf{t}^i(0) \end{array}$$

Observe that $R^{i-1}(\varepsilon)$ corresponds to traversing this diagram in counter-clockwise order, starting at $\mathbf{t}^{i-1}(0)$. It follows from the definitions that $R^{i-1}(\varepsilon)$ maps the tangent $\mathbf{t}^{i-1}(0)$ to itself and is therefore a rotation by angle $\psi_i(\varepsilon)$ about axis $\mathbf{t}^{i-1}(0)$. In the language of differential geometry, $\psi_i(\varepsilon)$ is the *holonomy* of the *connection* induced by parallel transport around the depicted closed loop.

The ingredient that we require is the *gradient* of ψ_i for $1 \le i \le n$. Building on the literature of a related quantity known as the *writhe* of polygonal curves [de Vries 2005], we obtain the variation of ψ_i with respect to a centerline displacement $\delta\mathbf{x}$:

$$\delta\psi_i = \frac{-2\mathbf{t}_{i-1} \times \mathbf{t}_i}{1 + \mathbf{t}_{i-1} \cdot \mathbf{t}_i} \cdot \left(\frac{1}{2}\frac{\delta\mathbf{x}_i - \delta\mathbf{x}_{i-1}}{|\mathbf{e}^{i-1}|} + \frac{1}{2}\frac{\delta\mathbf{x}_{i+1} - \delta\mathbf{x}_i}{|\mathbf{e}^i|}\right).$$

It can be shown that the corresponding expression in the smooth setting is $\delta(\int \psi \mathrm{d}s) = -\int \kappa\mathbf{b} \cdot (\delta\mathbf{x})_s \mathrm{d}s$ where the subscript s denotes differentiation with respect to s. Compare the discrete expression to the integrand of the smooth case; the second factor of the discrete expression is a finite-difference approximation of $(\delta\mathbf{x})_s$. In analogy to the smooth form, we take the first factor to be the integrated curvature binormal. Note that this definition coincides with (1), and allows us to rewrite the discrete result as

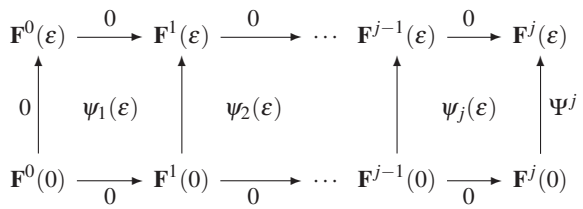
$$\nabla_{i-1}\psi_i = \frac{(\kappa\mathbf{b})_i}{2|\bar{\mathbf{e}}^{i-1}|}, \qquad \nabla_{i+1}\psi_i = -\frac{(\kappa\mathbf{b})_i}{2|\bar{\mathbf{e}}^i|}, \qquad (9)$$

and $\nabla_i\psi_i = -(\nabla_{i-1} + \nabla_{i+1})\psi_i$.

6.1 Variation of parallel transport

With our new tool in hand, we can derive the change in the Bishop frame when varying the rod's centerline. Since the Bishop frame is

by requirement adapted to the curve, we are interested in the *angle* that the frame is rotated by about the tangent. This situation is again depicted in the following diagram:



Here, the Bishop frames $\mathbf{F}^i = \{\mathbf{t}^i, \mathbf{u}^i, \mathbf{v}^i\}$ are displayed in place of the tangents to show that we are parallel-transporting these frames from one edge to the next. Additionally, each labeled arrow indicates the angle needed to align the *result* of parallel transporting the frame at the tail of the arrow with the frame at the head. Thus, the horizontal arrows are labeled with 0 to indicate that *no* twist is required to align $P_i(\mathbf{F}^{i-1})$ to $\mathbf{F}^i$; the first vertical arrow is also labeled with 0 because we ensure that $\mathbf{F}^0$ is always updated via parallel transport (see §4.2.2). We are interested in the angle of rotation, Ψ^j, required to align $\tilde{P}^j(\varepsilon)\big(\mathbf{F}^j(0)\big)$ to $\mathbf{F}^j(\varepsilon)$.

Since parallel transport commutes with twist and holonomy is additive under concatenation of loops, it follows from traversing this diagram in counter-clockwise order that the resulting angle is $\Psi^j = \sum_{i=1}^{j} \psi_i$. For computing forces, we require the gradient of this angle with respect to vertex positions, which is given by

$$\nabla_i \Psi^j = \sum_{k=1}^{j} \nabla_i \psi_k \, . \tag{10}$$

By (9), this sum will have at most three non-zero terms.

Discussion In hindsight, we view the relation of holonomy to the profile of the centerline as an extension of the celebrated Călugăreanu-White-Fuller theorem [1971] for discrete curves. Fuller [1978] noted that this theorem is applicable to the equilibria of *closed elastic rods with isotropic cross-sections*. Our development extends to the general case of open boundaries and even to anisotropic cross-sections.

7 Equations of motion

We are now ready to write the equations governing the time-evolution of the rod's centerline. Since the material frames depend on the rod's centerline and are *not* independent degrees of freedom (see §5), we must consider this dependence as well when computing the centerline forces.

7.1 Forces on centerline

In deriving the forces on the rod's centerline, we must consider how the energy depends on the $\mathbf{x}_i$ variables—both directly and indirectly by considering the effect that moving a vertex has on the rod's material frame. The force acting on vertex $\mathbf{x}_i$ is therefore given by

$$-\frac{\mathrm{d}E(\Gamma)}{\mathrm{d}\mathbf{x}_i} = -\frac{\partial E(\Gamma)}{\partial \mathbf{x}_i} - \sum_{j=0}^{n} \frac{\partial E(\Gamma)}{\partial \theta^j} \frac{\partial \theta^j}{\partial \mathbf{x}_i} \, .$$

Here, the *total derivative* $\mathrm{d}E/\mathrm{d}\mathbf{x}_i$ takes into account the implicit dependence of potential energy on centerline positions, whereas the *partial derivative* only takes into account explicit dependence.

Recall from (4) that $\partial E/\partial \theta^j$ vanishes for all edges for which θ^j is *not* prescribed by a boundary condition. Therefore, for stressfree boundary conditions, the sum is zero. For clamped boundaries, only one component of this sum could be non-zero, allowing us to write the force as

$$-\frac{\partial E}{\partial \mathbf{x}_i} - \frac{\partial E}{\partial \theta^n} \frac{\partial \theta^n}{\partial \mathbf{x}_i} \, .$$

To evaluate $\partial \theta^n / \partial \mathbf{x}_i$, recall from §6.1 that varying the centerline rotates the Bishop frame at edge $\mathbf{e}^n$ by angle Ψ^n about the tangent $\mathbf{t}^n$. Therefore, to keep the material frame aligned to the boundary condition, we must subtract this angle from θ^n to obtain

$$-\frac{\partial E}{\partial \mathbf{x}_i} + \frac{\partial E}{\partial \theta^n} \sum_{j=1}^{n} \frac{\partial \psi_j}{\partial \mathbf{x}_i} \, .$$

For clamped boundaries, we give the components required to evaluate this force for both the special and general case.

Special case For naturally straight, isotropic rods, the forces on vertex $\mathbf{x}_i$ are given by up to three contributions:

$$-\frac{2\alpha}{\bar{l}_j} \left(\nabla_i (\kappa \mathbf{b})_j\right)^T (\kappa \mathbf{b})_j + \frac{\beta(\theta^n - \theta^0)}{\bar{L}} \nabla_i \psi_j \quad i-1 \le j \le i+1 \, ,$$

with the gradient of holonomy given by (9) and the gradient of the curvature binormal given by

$$\begin{aligned}
\nabla_{i-1}(\kappa \mathbf{b})_i &= \frac{2[\mathbf{e}^i] + (\kappa \mathbf{b}_i)(\mathbf{e}^i)^T}{|\bar{\mathbf{e}}^{i-1}||\bar{\mathbf{e}}^i| + \mathbf{e}^{i-1} \cdot \mathbf{e}^i} \, , \\
\nabla_{i+1}(\kappa \mathbf{b})_i &= \frac{2[\mathbf{e}^{i-1}] - (\kappa \mathbf{b}_i)(\mathbf{e}^{i-1})^T}{|\bar{\mathbf{e}}^{i-1}||\bar{\mathbf{e}}^i| + \mathbf{e}^{i-1} \cdot \mathbf{e}^i} \, , \\
\nabla_i (\kappa \mathbf{b})_i &= -(\nabla_{i-1} + \nabla_{i+1})(\kappa \mathbf{b})_i \, .
\end{aligned}$$

Here $[\mathbf{e}]$ is a skew-symmetric 3×3 matrix acting on 3-vectors $\mathbf{x}$ by $[\mathbf{e}] \cdot \mathbf{x} = \mathbf{e} \times \mathbf{x}$. In deriving these gradients we have used the assumption of *inextensibility* of the rod's edges.

General case For anisotropic rods with a curved rest shape, the required expressions for the forces are given by:

$$\begin{aligned}
\frac{\partial E}{\partial \theta^n} &= \frac{1}{\bar{l}_n} (\boldsymbol{\omega}_n^n)^T J \overline{B}^n (\boldsymbol{\omega}_n^n - \overline{\boldsymbol{\omega}}_n^n) + \frac{2\beta m_n}{\bar{l}_n} \quad \text{and} \\
\frac{\partial E}{\partial \mathbf{x}_i} &= \sum_{k=1}^{n} \frac{1}{\bar{l}_k} \sum_{j=k-1}^{k} \left(\nabla_i \boldsymbol{\omega}_k^j\right)^T \overline{B}^j \left(\boldsymbol{\omega}_k^j - \overline{\boldsymbol{\omega}}_k^j\right) \, ,
\end{aligned}$$

where the gradient of the material-frame curvature $\boldsymbol{\omega}_k^j$ is given by

$$\nabla_i \boldsymbol{\omega}_k^j = \begin{pmatrix} (\mathbf{m}_2^j)^T \\ -(\mathbf{m}_1^j)^T \end{pmatrix} \nabla_i (\kappa \mathbf{b})_k - \mathrm{J} \boldsymbol{\omega}_k^j (\nabla_i \Psi^j)^T \, . \tag{11}$$

This last result is readily apparent from the definition of material-frame curvature in (2) and the variation of the Bishop frame in §6.1.

7.2 Integrating the centerline

Since the material frame is always updated to be in quasistatic equilibrium (see §5.1), we only need to update the centerline based on the forces derived above. The equations of motion are

$$M \ddot{\mathbf{x}} = -\frac{\mathrm{d}E(\Gamma)}{\mathrm{d}\mathbf{x}} \, ,$$

where M is a $3(n+2) \times 3(n+2)$ (diagonal) mass matrix associated to centerline positions. We discretize this equation using the symplectic Euler method [Hairer et al. 2006]. We use a manifold-projection method to enforce the inextensibility constraint.

8 Constraints

For simulating extensible rods, it would suffice to include a stretching component to the energy; however, simulating inextensible rods using stretch forces would lead to unnecessarily stiff equations. In addition, many interesting physical systems can be modeled by the coupling of elastic rods to rigid-bodies. Canonical examples include the torsional and Wilberforce pendulums, comprised of a rigid-body suspended under gravity by a thin elastic rod. To avoid numerical stiffness associated with maintaining inextensibility and rigid-body coupling, we add auxiliary constraints to our system.

Inextensibility constraints For each edge of the rod, we use the quadratic constraint equations $\mathbf{e}^i \cdot \mathbf{e}^i - \bar{\mathbf{e}}^i \cdot \bar{\mathbf{e}}^i = 0$, where (the precomputed) $\bar{\mathbf{e}}^i$ refers to the undeformed configuration.

Rigid-body coupling constraints The welding of the body to the rod requires the body's position and orientation, and the rod edge's position and material frame, to be in one-to-one correspondence. Attaching the rigid-body to the first edge of the rod gives three constraints:

$$\mathbf{q}\cdot\mathbf{q} - 1 = 0\,, \qquad \mathbf{q}\,\bar{\mathbf{x}}_0\,\mathbf{q}^* + \mathbf{r} - \mathbf{x}_0 = 0\,, \qquad \mathbf{q}\,\bar{\mathbf{x}}_1\,\mathbf{q}^* + \mathbf{r} - \mathbf{x}_1 = 0\,,$$

where $\mathbf{r} \in \mathbb{R}^3$ is the translation vector and $\mathbf{q} \in \mathbb{H}$ is the unit quaternion [Hanson 2006] mapping the body's center of mass and orientation, respectively, from the reference to the current configuration, and $\mathbf{q}^*$ is the conjugate of $\mathbf{q}$. The first equation ensures that $\mathbf{q}$ is unit length, so $\mathbf{q}\bar{\mathbf{x}}_0\mathbf{q}^*$ is a rotation of $\bar{\mathbf{x}}_0$ that is summed with $\mathbf{r}$ using vector addition. The second and third equations ensure that the rod's first edge and the rigid-body transform identically.

8.1 Enforcing constraints

Numerous approaches are available in the literature for maintaining constraints acting on a mechanical system [Shabana 2001]. Perhaps the most simple and straightforward of these is to use the penalty method; alas, as mentioned above, the penalty forces required to closely enforce the constraints are unacceptably stiff and require small time steps to ensure stability [Goldenthal et al. 2007].

An alternative is to employ an augmented Lagrangian formulation. Such formulations in general introduce additional variables (*i.e.*, Lagrange multipliers) to enforce the constraints during the time integration step of the algorithm. The exception are *manifold projection* methods [Hairer et al. 2006], which perform constraint enforcement as a post-integration step. We choose to maintain the constraints using this approach.

A manifold projection method integrates a mechanical system by alternating between an unconstrained time integration step and a constraint enforcement step. For the unconstrained step, we use an explicit symplectic Euler integrator (Algorithm step 7), and we call the ODE [Smith 2008] library to time step the rigid-body (Algorithm step 5).

Manifold projection allows us to reuse an existing rigid-body code without modifying its time integration, collision response, or other internal structures. However, any other method for enforcing these constraints could be used, and we include a discussion of our approach for completeness.

Fast manifold projection For the enforcement step, we adopt and extend the *fast projection* method of Goldenthal *et al.* [2007]. This method takes an unconstrained configuration and finds a nearby constrained configuration. The notion of "nearby" is made precise by the natural metric on the configuration manifold. We extend the application of fast projection to coupled systems involving both positional as well as rotational degrees of freedom by considering the metric induced by the kinetic energy $\frac{1}{2}\mathbf{y}\tilde{M}\mathbf{y}^T$, where the $(3n+12)\times(3n+12)$ *generalized mass matrix* and the *generalized velocity* $\mathbf{y} \in \mathbb{R}^{3n+12}$ are defined respectively by

$$\tilde{M} = \begin{pmatrix} 4\cdot I & & \\ & \mathscr{M}\cdot Id_{3\times 3} & \\ & & M \end{pmatrix} \qquad \text{and} \qquad \mathbf{y} = (\mathbf{q}^{-1}\dot{\mathbf{q}}, \dot{\mathbf{r}}, \dot{\mathbf{x}})\,.$$

Here M is the rod's (diagonal) $3(n+2)\times 3(n+2)$ mass matrix, $\mathscr{M}$ is the body's total (scalar) mass, and I is the body's (symmetric) moment of inertia tensor expressed as a 3×3 matrix in the reference coordinates. Since $\mathbf{q}$ is a *unit* quaternion, $\mathbf{q}^{-1}\dot{\mathbf{q}}$ corresponds to a vector in $\mathbb{R}^3$ [Hanson 2006]. The kinetic energy has contributions from the body rotation, body translation, and centerline translation.

We initialize the first iteration of fast projection with the results of an unconstrained time step. Each step of fast projection improves on the guess by computing the first Newton iteration for the minimization of the functional $\mathbf{y}\tilde{M}\mathbf{y}^T - \mathbf{C}^T\boldsymbol{\lambda}$, where (using Goldenthal's notation) $\mathbf{C}$ is the vector of constraint equations, and $\boldsymbol{\lambda}$ is the vector of corresponding Lagrange multipliers. Thus, by formulating the kinetic energy using a generalized mass matrix in a high-dimensional configuration space, we are able to directly apply fast projection iterations to rigid-body coupling. In all of our examples, we found that after 3–5 steps the method converged to within a tolerance of 10^{-8} in satisfying the constraint $\mathbf{C} = 0$.

After the iterations converge, fast projection requires a velocity update [Goldenthal et al. 2007]. Using our generalized coordinates, the appropriate update is

$$(\dot{\mathbf{q}}, \dot{\mathbf{r}}, \dot{\mathbf{x}}) \leftarrow (\dot{\mathbf{q}}, \dot{\mathbf{r}}, \dot{\mathbf{x}}) - \frac{1}{h}(2\mathbf{q}_0\mathbf{q}^{-1}, \mathbf{r}_0 - \mathbf{r}, \mathbf{x}_0 - \mathbf{x})\,.$$

Our discussion above immediately generalizes to the case of multiple rigid-bodies attached to multiple rods. As a corollary, we can couple a rigid-body to *any* edge on the rod simply by splitting the rod at that edge. However, care must be taken to understand the *induced boundary conditions*: each rigid-body's position and orientation serves to clamp the position and orientation of the centerline and material frame for each coupled rod end-edge. After fast projection, the material frame *vector* induced by the rigid body orientation is $\mathbf{m}_\mathbf{I}^0 = \mathbf{q}\bar{\mathbf{m}}_\mathbf{I}^0\mathbf{q}^*$.

8.2 Torque transfer

We consider a methodical categorization of those interface forces that are automatically transferred between the rod and rigid-body via projection and those that remain to be transferred explicitly. Observe that forces that correspond to gradients of the independent variables, $\mathbf{q}, \mathbf{r}, \mathbf{x}$, are automatically transferred between the rod and the body by the projection step. The material frame is *not* an independent variable—it is a function of the centerline position and of the boundary conditions (see §5). Therefore, in Algorithm step 4, we explicitly transfer the torque acting on the material frame $\mathbf{m}_\mathbf{I}^0$, to the coupled rigid-body. Note that the rigid-body's relatively large moment of inertia ensures that the explicit coupling is non-stiff.

The force acting on the material frame corresponds to the gradient of energy with respect to the *dependent* angles, θ^i. By the quasistatic material frame assumption, the torque, $\tau = |\tau|\mathbf{t}^0$, exerted by the rod on the body is equal and opposite to the torque exerted by the body on the rod. To derive the magnitude of the torque we turn to the principle of virtual work [Lanczos 1986]. Holding the centerline fixed, consider a *virtual displacement*, $\delta\theta^0$, which varies the material frame about $\mathbf{e}^0$. Note that $\delta\theta^0$ also affects the body's orientation, due to the rod-body constraint. The

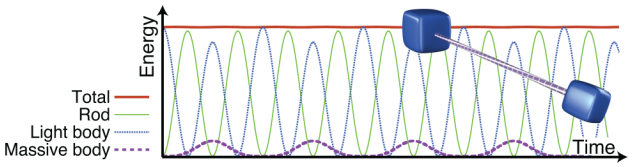


Figure 6: **Torque transducer:** two rigid-bodies coupled by a rod. The total energy, broken up into contributions from the bodies and the rod, is plotted.

work performed by the body on the rod equates to the change in the rod's stored energy: $|\tau|\delta\theta^0 = (\mathrm{d}E/\mathrm{d}\theta^0)\delta\theta^0$. Since this holds for *any* $\delta\theta^0$, it follows that $|\tau| = \mathrm{d}E/\mathrm{d}\theta^0$. In the isotropic case, this yields $|\tau| = \beta(\theta^n - \theta^0)/L$. In the anisotropic case, we expand the full derivative as $\mathrm{d}E/\mathrm{d}\theta^0 = \sum_i (\partial E/\partial\theta^i)(\partial\theta^i/\partial\theta^0)$. By the quasistatic assumption, most terms in this summation are zero, leading to $|\tau| = \partial E/\partial\theta^0$, which can be evaluated using (7).

9 Validation

The interaction between twisting and bending modes produces surprising instability phenomena, such as spontaneous buckling of rods. These instabilities are important for a variety of applications, such as for modeling DNA. When applied to some of these phenomena, our model not only reproduces the correct behavior qualitatively, but in fact shows good convergence to analytical solutions (for those models where an analytical treatment is available in the literature). In the following we describe several example problems; refer to Table 1 for corresponding performance measurements.

9.1 Convergence

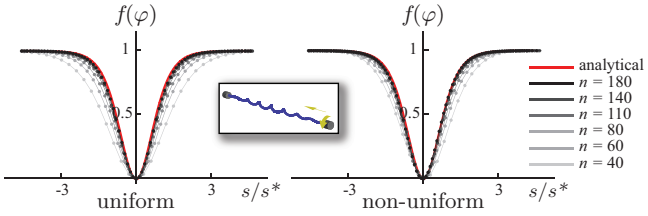


Figure 7: **Localized buckling** of a naturally straight, isotropic rod with an imposed angle of twist. The rod buckles into a helix with modulated amplitude (*inset*). Convergence of the envelope of this helix toward the analytical solution in the smooth case is observed for both a uniformly sampled rod (*left*) and a rod where one half is twice as refined as the other (*right*). The parameters for the simulation are: rod length $L = 9.29$, bending modulus $\alpha = 1.345$, twist modulus $\beta = .789$, clamp rotated by 27 turns, imposed axial shortening of 0.3 units, corresponding to a theoretical maximal deviation $\varphi_0 = 0.919$.

Localized helical buckling When a naturally straight, isotropic rod is clamped at its two ends with one end rotated by a given angle, the rod buckles as the two ends are quasi-statically translated towards one another. In the smooth case, the buckling of a long rod under twist is described by an exact solution of the equations of equilibrium. This analytical solution describes nonlinear (finite amplitude) buckling away from the straight configuration and mixes bending and twisting effects. We studied convergence of the equilibria of our discrete model towards this solution in the limit of refinement. The geometry is shown in the inset of Figure 7. We denote by s the arc length, $\mathbf{e}_x$ the unit vector parallel to the axis passing through the clamps, $\varphi(s) = \cos^{-1}(\mathbf{t}\cdot\mathbf{e}_x)$ the angular deviation of the tangent away from this axis, and $\varphi_0 = \max_s \varphi(s)$ the maximal deviation at the center of the pattern. The envelope of the helix is given by $f(\varphi(s)) = \tanh^2\left(\frac{s}{s^*}\right)$, where s/s^* is the dimensionless arc length given by $s/s^* = \left(\frac{\beta m}{2\alpha}\sqrt{\frac{1-\cos\varphi_0}{1+\cos\varphi_0}}\right) s$, which simplifies to $f(\varphi) = \frac{\cos\varphi - \cos\varphi_0}{1-\cos\varphi_0}$ (see *e.g.* [van der Heijden and Thompson 2000]). With a fixed loading geometry, we obtain convergence towards this analytical solution under refinement (see Fig. 7).

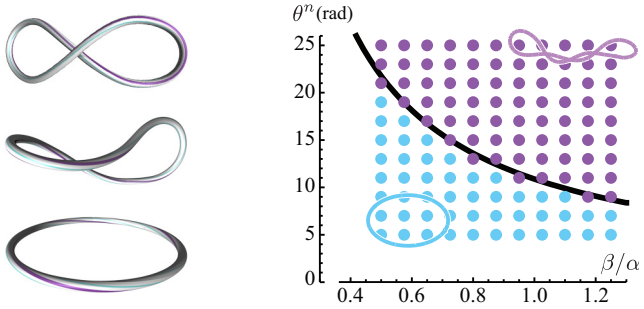


Figure 8: **Michell's buckling instability** of an elastic ring with imposed internal twist θ^n. *Left:* above a critical value of θ^n, the planar, circular shape loses stability and buckles to a non-planar shape. *Right:* domain of stability of the circular shape with radius $R = 1$: simulations (dots) compared to theoretical threshold (black curve). Each dot corresponds to a simulation run with particular values of β and θ^n ($\alpha = 1$ and $n = 50$ are fixed), initialized with a slightly perturbed circular shape; dots are colored in light blue when the amplitude of the perturbation decreases in time (stable) and in purple when it increases (unstable).

Michell's instability Another famous example of a rod becoming unstable when subjected to twist is Michell's instability, also known as Zajac's instability (see Fig. 8). When the ends of a naturally straight, isotropic rod are closed into a ring in a twist-free manner, the resulting shape is circular. When these ends are twisted by an angle θ^n before they are closed up, and when this angle is progressively increased starting from zero, the ring suddenly starts to writhe (buckle) out of plane, at a well-defined *critical* value of twist. This effect is a perfect illustration of the coupling of the twist with the shape of the centerline. This critical twist can be computed analytically [Goriely 2006] as $\theta^n_c = 2\pi\sqrt{3}/(\beta/\alpha)$. In Figure 8–*right*, we study the stability of the circular shape numerically for different values of the rod parameters and compare to the analytical prediction. Our model displays excellent agreement with the predictions of the smooth theory.

9.2 Special case

Instability of knots Although knots have been extensively studied as mathematical objects, the understanding of their mechanical behavior is far less advanced. Very recently, an analytical solution has been obtained describing the equilibrium shape of a loose trefoil knot tied on a naturally straight, isotropic rod [Audoly et al. 2007] in the case where *no twist* is applied. We ran a simulation of a knotted rod with both ends clamped and were able to reproduce the typical equilibrium shape of the knot, which is made of a large, almost circular loop connected to two flat tails by a braid region (see Fig. 1–*left*). We next twisted the ends of the rod and found that the shape of the knot first changes smoothly and then jumps at a critical value of the twist. This jump happens both for positive and negative applied twist, although the final shapes of the knot are qualitatively

different (see Fig. 1). This fascinating behavior will be studied in detail in a future paper [Audoly et al. 2008]—it can be easily reproduced using a small tube of an elastic material (a silicone wire in our experiments), and we encourage the reader to try it!

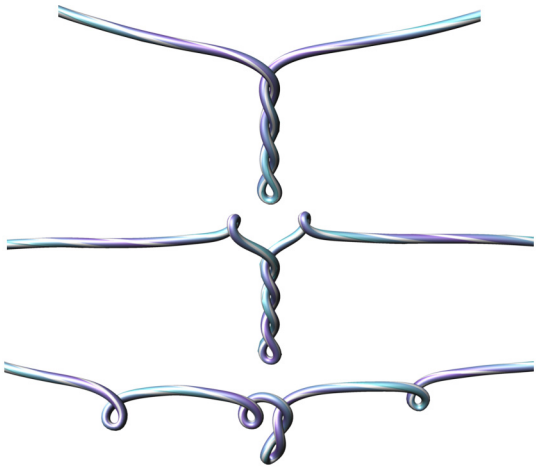


Figure 9: **Plectoneme formation:** When the ends of a hanging elastic rod are twisted, it takes on structures known as *plectonemes*. The formation of plectonemes is governed by physical parameters, such as the twist rate, viscosity of the ambient fluid, and gravity.

Plectonemes A rod generally forms entangled structures called plectonemes when subjected to large twist. Typically, the pattern formed by a twist-driven instability such as Michell's instability grows from a weakly helical shape at short times, to fully developed plectonemes at long time. Such structures have been well-studied, for instance in the context of DNA supercoiling [Yang et al. 1993]. In this experiment, we started with a naturally straight rod and deformed it into a parabola in a twist-free manner. Fixing the positions of the endpoints of the rod and progressively increasing twist, we find that the rod starts to writhe out of the plane. Letting the instability develop fully, we observe plectonemes (see Fig. 9). Depending on the viscous drag and gravity, we found that one or many plectonemes can be obtained. Single plectonemes have been described both analytically [van der Heijden et al. 2003] and numerically [Goyal et al. 2007] in several papers; we observed an interesting phenomenon of plectonemes merging at long times, which has seemingly not yet been studied.

The simulation of plectonemes requires the treatment of rod self-contact, which is outside the scope of this paper; for the state of the art, refer to the treatment by Spillmann and Teschner [2008].

9.3 General case

The coupling of twisting and bending modes of naturally curved or anisotropic rods is fairly more complex than in the isotropic case, as demonstrated by in following experiments.

Asymmetry of twist The combination of anisotropy and non-zero rest curvature can result in a non-uniform distribution of twist along the rod. In Figure 5, we show this effect for a rod with anisotropic cross section, whose centerline is either V-shaped or semi-circular in the reference configuration. We clamp one end of each of these rods and twist the other. In the V-shaped case, we observe that twist is first confined on one half of the rod—it takes a significant amount of twist before twist manages to "jump over" the kink in the middle of the rod. A similar phenomenon occurs with the semi-circular shape, but it is less marked. In either case, the twist concentrates near the end that is rotated, although the rod properties are uniform along its length; this symmetry-breaking points to the fact that that the equations governing the quasi-static twist are nonlinear.

Helical perversion: Perversion is a classical pattern that can be observed in naturally curved rods. It consists of a junction between two helices with opposite chiralities. It has been described by Darwin in the tendrils of climbing plants and can be often observed in tangled phone cords [Goriely and Tabor 1998]. Perversions can be produced by first flattening a helical spring such as a Slinky® into a flat, straight ribbon by pulling on both of its ends (see Fig. 2–*middle*); next, by progressively releasing its ends from this straight configuration, one obtains a shape made of two mirror-symmetric helices (see Fig. 2–*bottom*). The final shape is surprisingly different from the natural one as the chirality has been reversed in a half of the spring, as revealed by comparison with Figure 2–*top*. The existence of perversions is due to the nonlinear behavior of the rod—perversions belong to a general class of solutions that can be derived in nonlinear analysis by connecting two competing equilibrium configurations, which are here the right-handed and left-handed helices in the presence of natural curvature.

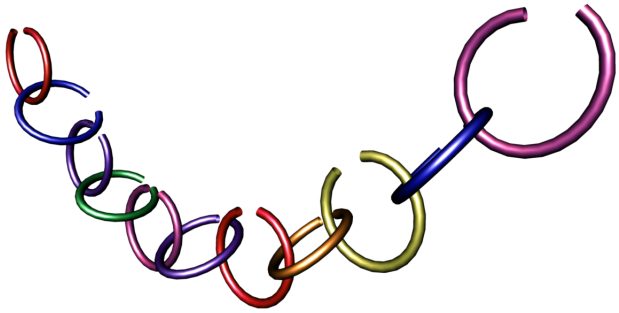


Figure 10: **Hanging chain:** A chain consisting of curved elastic rods as links hangs under the influence of gravity. The material parameters for each rod are $\overline{B} = 2Id_{2\times 2}$ and $\beta = 2$, with each link in the chain having a radius of approximately 1 unit.

Hanging chain: free boundaries In Figure 10, we show a chain hanging from its two ends under the influence of gravity. Each link is an elastic rod with curved undeformed configuration with stressfree boundary conditions on the material frames. As shown in the accompanying animation, the two ends are pulled apart until the chain breaks and the links fly apart. Even though we are not applying a twist to any of the links in the chain, we still need the material frame to represent the undeformed curvature of the rod—recall that $\boldsymbol{\omega}$ is defined as a 2-vector in material-frame coordinates.

9.4 Rigid-body coupling

When coupling rods with rigid-bodies, the transfer of energy between these systems results in complex motions—in particular, through the interplay between bending and (non-uniform) twisting of the rod and the translational and rotational moment of the attached mass. In order to validate our model, we have in particular tested its ability to faithfully reproduce real-world experiments.

Wilberforce pendulum This experiment reveals fascinating aspects of how bending and twisting interact and thereby affect the motion of an attached rigid-body. Consider a helicoidal spring (an isotropic rod whose reference state is a helix) whose upper end is

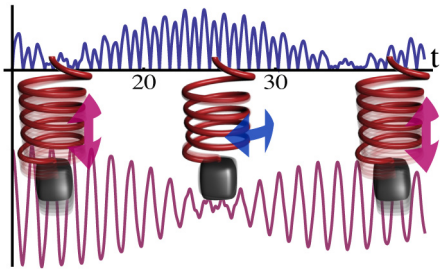


Figure 11: **Wilberforce pendulum:** Due to a weak coupling between the bending and twisting modes of a stretched spring, the energy of the system is transferred back and forth between the translational (red) and angular (blue) modes of oscillations.

fig. no.	verts.	time-step	forces	contact	quas. update	fast proj.	total
1	60	4.0	0.026	0.05	0.021	0.21	0.31
5	49	1.0	0.025	0.018	0.021	0.12	0.18
6	200	2.0	0.072	0.09	0.064	0.81	1.0
9	275	1.0	0.13	0.19	0.17	0.42	0.92
7	75	2.0	0.043		0.1	0.2	0.34
8	67	50.0	0.039		0.096	0.28	0.42
10	31	1.0	0.016		0.13	0.13	0.2
11	20	1.0	0.0036		0.0043	0.019	0.027
12	2158	0.1	0.87		0.64	21.0	22.0

Table 1: **Performance evaluation:** This table summarizes timing information (in milliseconds per simulation step) for examples depicted in the figures, as measured on a single-threaded application running on a 2.66GHz Core 2 Duo. For examples run without collision detection, collision timings are omitted.

fixed, and attach a rigid-body to its lower end. The weight of the body stretches the spring when the whole system is at rest. Moving the mass slightly upwards from this equilibrium and releasing it, first leads to vertical oscillations. Progressively, the system switches to twist oscillations and the vertical ones are extinguished; later, the twist oscillations start to decrease and the vertical ones reappear, and so on (see Fig. 11). The nonlinear behavior of the spring, which is captured accurately by our model, is responsible for this energy transfer between the two modes [Sommerfeld 1964]: stretching a spring affects its eigenmodes. Note the presence of stationary waves in the spring, which are also clearly visible in our simulation movie.

Torque transducer Rods can be used to couple rigid-bodies, with the rod effectively acting as a "torque transducer" between them. In Figure 6, we plot the kinetic and potential energy of the two rigid-bodies and the rod (one of which has much higher mass than the other), observing the energy transfer among the stored potential energy of the rod and the rotational energy of the two rigid bodies.

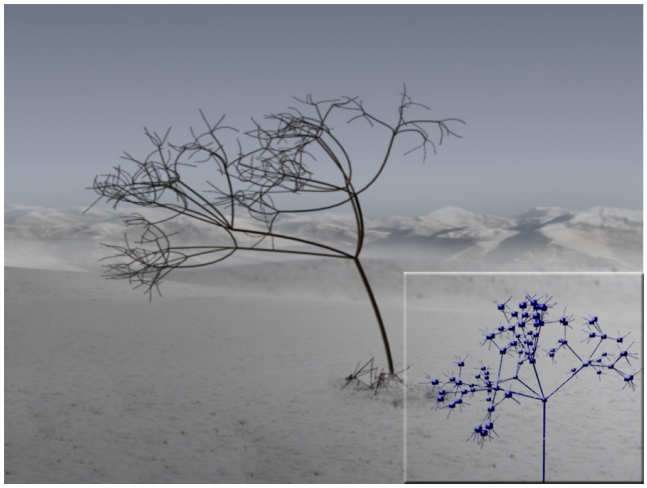


Figure 12: **Tree in wind:** rigid-bodies connecting multiple rods together at a single point (reference configuration shown in inlay). Using this method, we can simulate a tree bending and twisting in response to a strong arctic wind. Note that without resistance to twist, the tree would start spinning due to vortices in the wind.

Rigid-bodies for rigid couples Coupling rods with rigid-bodies opens the possibility for complex simulations—in particular, for simulating tree-like one-dimensional configurations with several T-junctions. In Figure 12, we show an example of such coupling, where we additionally used external forces (wind and gravity) to increase the dramatic effect. Here the mass of the rigid-bodies (but not necessarily their spatial extension) can be tuned to achieve a variety of dramatic effects. This example shows the power of coupling rods with rigid-bodies, indicating the attractiveness of this approach to be used in a wide range of graphics applications, such as for simulating plant motion, or the skeletal dynamics of rigged characters.

10 Conclusion

Limitations and future work Our use of the Fast Projection algorithm leads to energy being dissipated during the constraint-enforcement step. In many applications, the energy behavior of the system is of interest, so we would like to explore alternate methods for enforcing constraints that are both efficient and have good energy behavior.

The formulation of discrete rods allows us to impose boundary conditions on the material frame at any edge along the rod. We have presented boundary conditions that arise due to explicitly clamping certain edges or coupling them to rigid-bodies. We are currently considering the effect of frictional contact on the material frame and would like to have an implementation that also allows contact constraints to dictate boundary conditions for the material frame.

We are also interested in employing adaptivity in our current model and believe that the ideas of Spillmann and Teschner [2008] pave the way for this. In addition, an interesting related topic would be in providing higher-order methods for simulating elastic rods. The main issues involved would be in defining convergent discrete operators, such as parallel transport and curvature, on higher-order elements, and in ensuring the resulting order of accuracy.

Acknowledgments We would like to thank Michael Herrmann for his indispensable insights on unifying the isotropic and anisotropic picture, Sébastien Neukirch for his help in reviewing the literature on elastic rods, Jonas Spillmann and Matthias Teschner for sharing their code for CoRdE with us, David Harmon for his help with collision detection and response, Kori Valz for her artistic contributions, and Aner Ben-Artzi, Mathieu Desbrun, and Jerrold E. Marsden for early discussions regarding this project. This work was supported in part by the NSF (MSPA Award No. IIS-05-28402, CSR Award No. CNS-06-14770, CAREER Award No. CCF-06-43268) and the DFG Research Center Matheon in Berlin, as well as by Autodesk, Elsevier, mental images, NVIDIA, and Walt Disney Animation Studios.

References

AUDOLY, B., AND POMEAU, Y. 2008. *Elasticity and geometry: from hair curls to the nonlinear response of shells*. Oxford University Press. To appear.

AUDOLY, B., CLAUVELIN, N., AND NEUKIRCH, S. 2007. Elastic knots. *Physical Review Letters 99*, 16, 164301.

AUDOLY, B., CLAUVELIN, N., AND NEUKIRCH, S. 2008. Instabilities of elastic knots under twist. In preparation.

BERTAILS, F., AUDOLY, B., CANI, M.-P., QUERLEUX, B., LEROY, F., AND LÉVÊQUE, J.-L. 2006. Super-helices for predicting the dynamics of natural hair. In *ACM TOG*, 1180–1187.

BOBENKO, A. I., AND SURIS, Y. B. 1999. Discrete time lagrangian mechanics on lie groups, with an application to the lagrange top. *Comm. in Mathematical Physics 204*, 1, 147–188.

BROWN, J., LATOMBE, J.-C., AND MONTGOMERY, K. 2004. Real-time knot-tying simulation. *Vis. Comp. V20*, 2, 165–179.

CHANG, J., SHEPHERD, D. X., AND ZHANG, J. J. 2007. Cosserat-beam-based dynamic response modelling. *Computer Animation and Virtual Worlds 18*, 4–5, 429–436.

CHOE, B., CHOI, M. G., AND KO, H.-S. 2005. Simulating complex hair with robust collision handling. In *SCA '05*, 153–160.

DE VRIES, R. 2005. Evaluating changes of writhe in computer simulations of supercoiled DNA. *J. Chem. Phys. 122*, 064905.

FALK, R. S., AND XU, J.-M. 1995. Convergence of a second-order scheme for the nonlinear dynamical equations of elastic rods. *SJNA 32*, 4, 1185–1209.

FULLER, F. B. 1971. The writhing number of a space curve. *PNAS 68*, 4, 815–819.

FULLER, F. B. 1978. Decomposition of the linking number of a closed ribbon: a problem from molecular biology. *PNAS 75*, 8, 3557–3561.

GOLDENTHAL, R., HARMON, D., FATTAL, R., BERCOVIER, M., AND GRINSPUN, E. 2007. Efficient simulation of inextensible cloth. In *ACM TOG*, 49.

GOLDSTEIN, R. E., AND LANGER, S. A. 1995. Nonlinear dynamics of stiff polymers. *Phys. Rev. Lett. 75*, 6 (Aug), 1094–1097.

GORIELY, A., AND TABOR, M. 1998. Spontaneous helix hand reversal and tendril perversion in climbing plants. *Physical Review Letters 80*, 7 (Feb), 1564–1567.

GORIELY, A. 2006. Twisted elastic rings and the rediscoveries of Michell's instability. *Journal of Elasticity 84*, 281–299.

GOYAL, S., PERKINS, N. C., AND LEE, C. L. 2007. Non-linear dynamic intertwining of rods with self-contact. *International Journal of Non-Linear Mechanics In Press, Corrected Proof*.

GRÉGOIRE, M., AND SCHÖMER, E. 2007. Interactive simulation of one-dimensional flexible parts. *CAD 39*, 8, 694–707.

GRINSPUN, E., Ed. 2006. *Discrete differential geometry: an applied introduction*. ACM SIGGRAPH 2006 Courses.

HADAP, S., CANI, M.-P., LIN, M., KIM, T.-Y., BERTAILS, F., MARSCHNER, S., WARD, K., AND KAČIĆ-ALESIĆ, Z. 2007. *Strands and hair: modeling, animation, and rendering*. ACM SIGGRAPH 2007 Courses, 1–150.

HADAP, S. 2006. Oriented strands: dynamics of stiff multi-body system. In *SCA '06*, 91–100.

HAIRER, E., LUBICH, C., AND WANNER, G. 2006. *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations*. Series in Comp. Math. Springer.

HANSON, A. J. 2006. *Visualizing Quaternions*. MK/Elsevier.

KLAPPER, I. 1996. Biological applications of the dynamics of twisted elastic rods. *J. Comp. Phys. 125*, 2, 325–337.

LANCZOS, C. 1986. *The Variational Principles of Mechanics*, 4th ed. Dover.

LANGER, J., AND SINGER, D. A. 1996. Lagrangian aspects of the Kirchhoff elastic rod. *SIAM Review 38*, 4, 605–618.

LENOIR, J., COTIN, S., DURIEZ, C., AND NEUMANN, P. 2006. Interactive physically-based simulation of catheter and guidewire. *Computer & Graphics 30*, 3, 417–423.

LIN, C.-C., AND SCHWETLICK, H. R. 2004. On the geometric flow of Kirchhoff elastic rods. *SJAM 65*, 2, 720–736.

MADDOCKS, J. H., AND DICHMANN, D. J. 1994. Conservation laws in the dynamics of rods. *Journal of elasticity 34*, 83–96.

MADDOCKS, J. H. 1984. Stability of nonlinearly elastic rods. *Archive for Rational Mechanics and Analysis 85*, 4, 311–354.

PAI, D. K. 2002. STRANDS: interactive simulation of thin solids using Cosserat models. *CGF 21*, 3.

PHILLIPS, J., LADD, A., AND KAVRAKI, L. 2002. Simulated knot tying. In *Proceedings of ICRA 2002*, vol. 1, 841–846.

PRESS, W. H., VETTERLING, W. T., TEUKOLSKY, S. A., AND FLANNERY, B. P. 2002. *Numerical Recipes in C++: the art of scientific computing*. Cambridge University Press.

RUBIN, M. B. 2000. *Cosserat Theories: Shells, Rods and Points*. Kluwer Academic Publishers.

SHABANA, A. 2001. *Computational Dynamics*, 2nd ed. John Wiley & Sons, New York.

SMITH, R., 2008. Open dynamics engine. http://www.ode.org/.

SOMMERFELD, A. 1964. *Mechanics of deformable bodies*, vol. 2 of *Lectures on theoretical physics*. Academic Press.

SPILLMANN, J., AND TESCHNER, M. 2007. Corde: Cosserat rod elements for the dynamic simulation of one-dimensional elastic objects. In *SCA '07*, 63–72.

SPILLMANN, J., AND TESCHNER, M. 2008. An adaptative contact model for the robust simulation of knots. *CGF 27*.

TERZOPOULOS, D., PLATT, J., BARR, A., AND FLEISCHER, K. 1987. Elastically deformable models. In *Proceedings of SIGGRAPH '87*, 205–214.

THEETTEN, A., GRISONI, L., ANDRIOT, C., AND BARSKY, B. 2006. Geometrically exact dynamic splines. Tech. rep., INRIA.

VAN DER HEIJDEN, G. H. M., AND THOMPSON, J. M. T. 2000. Helical and localised buckling in twisted rods: A unified analysis of the symmetric case. *Nonlinear Dynamics 21*, 1, 71–99.

VAN DER HEIJDEN, G. H. M., NEUKIRCH, S., GOSS, V. G. A., AND THOMPSON, J. M. T. 2003. Instability and self-contact phenomena in the writhing of clamped rods. *Int. J. Mech. Sci. 45*, 161–196.

WARD, K., BERTAILS, F., KIM, T.-Y., MARSCHNER, S. R., CANI, M.-P., AND LIN, M. C. 2007. A survey on hair modeling: Styling, simulation, and rendering. *IEEE TVCG 13*, 2 (Mar./Apr.), 213–234.

YANG, Y., TOBIAS, I., AND OLSON, W. K. 1993. Finite element analysis of DNA supercoiling. *J. Chem. Phys. 98*, 2, 1673–1686.

Chapter 7: Discrete Differential Forms for Computational Modeling

Mathieu Desbrun Eva Kanso* Yiying Tong†
Applied Geometry Lab
Caltech‡

1 Motivation

The emergence of computers as an essential tool in scientific research has shaken the very foundations of differential modeling. Indeed, the deeply-rooted abstraction of smoothness, or *differentiability*, seems to inherently clash with a computer's ability of storing only finite sets of numbers. While there has been a series of computational techniques that proposed discretizations of differential equations, the geometric structures they are simulating are often lost in the process.

1.1 The Role of Geometry in Science

Geometry is the study of space and of the properties of shapes in space. Dating back to Euclid, models of our surroundings have been formulated using simple, geometric descriptions, formalizing apparent *symmetries* and experimental *invariants*. Consequently, geometry is at the foundation of many current physical theories: general relativity, electromagnetism (E&M), gauge theory as well as solid and fluid mechanics all have strong underlying geometrical structures. Einstein's theory for instance states that gravitational field strength is directly proportional to the *curvature of space-time*. In other words, the physics of relativity is *directly modelled* by the shape of our 4-dimensional world, just as the behavior of soap bubbles is modeled by their shapes. Differential geometry is thus, de facto, the mother tongue of numerous physical and mathematical theories.

Unfortunately, the inherent geometric nature of such theories is often obstructed by their formulation in vectorial or tensorial notations: the traditional use of a coordinate system, in which the defining equations are expressed, often obscures the underlying structures by an overwhelming usage of indices. Moreover, such complex expressions entangle the topological and geometrical content of the model.

1.2 Geometry-based Exterior Calculus

The geometric nature of these models is best expressed and elucidated through the use of the *Exterior Calculus of Differential Forms*, first introduced by Cartan [Cartan 1945]. This geometry-based calculus was further developed and refined over the twentieth century to become the foundation of modern differential geometry. The calculus of exterior forms allows one to express differential and integral equations on smooth and curved spaces in a consistent manner, while revealing the geometrical invariants at play. For example, the classical operations of gradient, divergence, and curl as well as the theorems of Green, Gauss and Stokes can all be expressed concisely in terms of differential forms and an operator on these forms called the exterior derivative—hinting at the generality of this approach.

Compared to classical tensorial calculus, this exterior calculus has several advantages. First, it is often difficult to recognize the coordinate-independent nature of quantities written in tensorial notation: local and global invariants are hard to notice by just staring at the indices. On the other hand, invariants are easily discovered when expressed as differential forms by invoking either Stokes' theorem, the Poincaré lemma, or by applying exterior differentiation. Note also that the exterior derivative of differential forms—the antisymmetric part of derivatives—is one of the most important parts of differentiation, since it is invariant under coordinate system change. In fact, Sharpe states in [Sharpe 1997] that every differential equation may be expressed in term of the exterior derivative of differential forms. As a consequence, several recent initiatives have been aimed at formulating the physical laws in terms of differential forms. For recent work along these lines, the reader is invited to refer to [Burke 1985; Abraham et al. 1988; Lovelock and Rund 1993; Flanders 1990; Morita 2001; Carroll 2003; Frankel 2004] for books offering a theoretical treatment of various physical theories using differential forms.

1.3 Differential vs. Discrete Modeling

We have seen that a large amount of our scientific knowledge relies on a deeply-rooted differential (*i.e.*, smooth) comprehension of the world. This abstraction of differentiability allows researchers to model complex physical systems via concise equations. With the sudden advent of the digital age, it was therefore only natural to resort to computations based on such differential equations.

However, since digital computers can only manipulate finite sets of numbers, their capabilities seem to clash with the basic foundations of differential modeling. In order to overcome this hurdle, a first set of computational techniques (*e.g.*, finite difference or particle methods) focused on satisfying the continuous equations at a discrete set of spatial and temporal samples. Unfortunately, focusing on accurately discretizing the local laws often fails to respect important global structures and invariants. Later methods such as Finite Elements (FEM), drawing from developments in the calculus of variations, remedied this inadequacy to some extent by satisfying local conservation laws on average and preserving some important invariants. Coupled with a finer ability to deal with arbitrary boundaries, FEM became the de facto computational tool for engineers. Even with significant advances in error control, convergence, and stability of these finite approximations, the underlying structures of the simulated continuous systems are often destroyed: a moving rigid body may gain or loose momentum; or a cavity may exhibit fictitious eigenmodes in an electromagnetism (E&M) simulation. Such examples illustrate some of the loss of fidelity that can follow from a standard discretization process, failing to preserve some fundamental geometric and topological structures of the underlying continuous models.

The cultural gap between theoretical and applied science communities may be partially responsible for the current lack of proper discrete, computational modeling that could mirror and leverage

*Now at the University of Southern California.
†Now at Michigan State University.
‡E-mail: {mathieu|eva|yiying}@caltech.edu

the rich developments of its differential counterpart. In particular, it is striking that the calculus of differential forms has not yet had an impact on the mainstream computational fields, despite excellent initial results in E&M [Bossavit 1998] or Lagrangian mechanics [Marsden and West 2001]. It should also be noticed that some basic tools necessary for the definition of a discrete calculus already exist, probably initiated by Poincaré when he defined his cell decomposition of smooth manifolds. The study of the structure of ordered sets or simplices now belongs to the well-studied branch of mathematics known as *Combinatorial Differential Topology and Geometry*, which is still an active area of research (see, e.g., [Forman 2003] and [Björner and Welker 1995] and references therein).

1.4 Calculus ex Geometrica

Given the overwhelming geometric nature of the most fundamental and successful calculus of these last few centuries, it seems relevant to *approach computations from a geometric standpoint*.

One of the key insights that percolated down from the theory of differential forms is rather simple and intuitive: one needs to recognize that different physical quantities have different properties, and must be treated accordingly. Fluid mechanics or electromagnetism, for instance, make heavy use of line integrals, as well as surface and volume integrals; even physical measurements are performed as specific local integrations or averages (think flux for magnetic field, or current for electricity, or pressure for atoms' collisions). Pointwise evaluations or approximations for such quantities are not the appropriate discrete analogs, since the defining geometric properties of their physical meaning cannot be enforced naturally. Instead, *one should store and manipulate those quantities at their geometrically-meaningful location*: in other words, we should consider values on vertices, edges, faces, and tetrahedra as proper discrete versions of respectively pointwise functions, line integrals, surface integrals, and volume integrals: only then will we be able to manipulate those values without violating the symmetries that the differential modeling tried to exploit for predictive purposes.

1.5 Similar Endeavors

The need for improved numerics have recently sprung a (still limited) number of interesting related developments in various fields. Although we will not try to be exhaustive, we wish to point the reader to a few of the most successful investigations with the same "flavor" as our discrete geometry-based calculus, albeit their approaches are rarely similar to ours. First, the field of *Mimetic Discretizations of Continuum Mechanics*, led by Shashkov, Steinberg, and Hyman [Hyman and Shashkov 1997], started on the premise that spurious solutions obtained from finite element or finite difference methods often originate from inconsistent discretizations of the operators div, curl, and grad, and that addressing this inconsistency pays off numerically. Similarly, *Computational Electromagnetism* has also identified the issue of field discretization as the main reason for spurious modes in numerical results. An excellent treatment of the discretization of the Maxwell's equations resulted [Bossavit 1998], with a clear relationship to the differential case. Finally, recent developments in *Discrete Lagrangian Mechanics* have demonstrated the efficacy of a proper discretization of the Lagrangian of a dynamical system, rather than the discretization of its derived Euler-Lagrange equations: with a discrete Lagrangian, one can ensure that the integration scheme satisfies an exact discrete least-action principle, preserving all the momenta directly for arbitrary orders of accuracy [Marsden and West 2001]. Respecting the defining geometric properties of both the fields and the governing equations is a common link between all these recent approaches.

1.6 Advantages of Discrete Differential Modeling

The reader will have most probably understood our bias by now: we believe that the systematic construction, inspired by Exterior Calculus, of **differential, yet readily discretizable computational foundations** is a crucial ingredient for numerical fidelity. Because many of the standard tools used in differential geometry have discrete combinatorial analogs, the *discrete versions of forms or manifolds* will be formally identical to (and should partake of the same properties as) the continuum models. Additionally, such an approach should clearly maintain the separation of the topological (*metric-independent*) and geometrical (*metric-dependent*) components of the quantities involved, keeping the geometric picture (*i.e.*, intrinsic structure) intact.

A *discrete differential modeling approach to computations* will also be often much simpler to define and develop than its continuous counterpart. For example, the discrete notion of a differential form will be implemented simply as values on mesh elements. Likewise, the discrete notion of orientation will be more straightforward than its continuous counterpart: while the differential definition of orientation uses the notion of equivalence class of atlases determined by the sign of the Jacobian, the orientation of a mesh edge will be one of two directions; a triangle will be oriented clockwise or counterclockwise; a volume will have a direction as a right-handed helix or a left-handed one; no notion of atlas (a collection of consistent coordinate charts on a manifold) will be required.

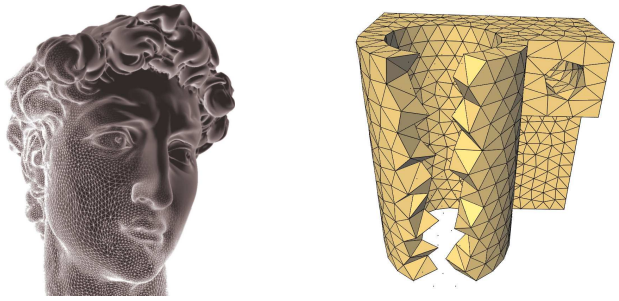


Figure 1: *Typical 2D and 3D meshes: although the David head appears smooth, its surface is made of a triangle mesh; tetrahedral meshes (such as this mechanical part, with a cutaway view) are some typical examples of irregular meshes on which computations are performed. David's head mesh is courtesy of Marc Levoy, Stanford.*

1.7 Goal of This Chapter

Given these premises, this chapter was written with several purposes in mind. First, we wish to demonstrate that the foundations on which powerful methods of computations can be built are quite approachable—and are not as abstract as the reader may fear: the ideas involved are very intuitive as a side effect of the simplicity of the underlying geometric principles.

Second, we wish to help bridge the gap between applied fields and theoretical fields: we have tried to render the theoretical bases of our exposition accessible to computer scientists, and the concrete implementation insights understandable by non-specialists. For this very reason, the reader should not consider this introductory exposition as a definite source of knowledge: it should instead be considered as a portal to better, more focused work on related subjects. We only hope that we will ease our readers into foundational concepts that can be undoubtedly and fruitfully applied to all sorts of computations—be it for graphics or simulation.

With these goals in mind, we will describe the background needed to develop a principled, geometry-based approach to computational modeling that gets around the apparent mismatch between differential and discrete modeling.

2 Relevance of Forms for Integration

The evaluation of differential quantities on a discrete space (mesh) is a nontrivial problem. For instance, consider a piecewise-linear 2-dimensional surface embedded in a three-dimensional Euclidean space, *i.e.*, a triangle mesh. Celebrated quantities such as the Gaussian and mean curvatures are delicate to define on it. More precisely, the Gaussian curvature can be easily proven to be zero everywhere *except* on vertices, where it is a Dirac delta function. Likewise, the mean curvature can only be defined in the distributional sense, as a Dirac delta function on edges. However, through local *integrations*, one can easily manipulate these quantities numerically: if a careful choice of non-overlapping regions is made, the delta functions can be properly integrated, rendering the computations relatively simple as shown, for example, in [Meyer et al. 2002; Hildebrandt and Polthier 2004]. Note that the process of integration to suppress discontinuity is, in spirit, equivalent to the idea of weak form used in the Finite Element method.

This idea of integrated value has predated in some cases the equivalent differential statements: for instance, it was long known that the genus of a surface can be calculated through a cell decomposition of the surface via the Euler characteristic. The actual Gauss-Bonnet theorem was, however, derived later on. Now, if one tries to discretize the Gaussian curvature of a piecewise-linear surface in an arbitrary way, it is not likely that its integral over the surface equals the desired Euler characteristic, while its discrete version, defined on vertices (or, more precisely, on the dual of each vertex), naturally preserves this topological invariant.

2.1 From Integration to Differential Forms

Integration is obviously a linear operation, since for any disjoint sets A and B,

$$\int_{A\cup B} = \int_A + \int_B .$$

Moreover, the integration of a smooth function over a subset of measure zero is always zero; for example, an area integral of (a lower dimensional object such as) a curve or a point is equal to zero. Finally, integration is *objective* (*i.e.*, relevant) only if its evaluation is invariant under change of coordinate systems. These three properties combined directly imply that the integrand (*i.e.*, the whole expression after the integral sign) has to be *antisymmetric*. That is, the basic building blocks of any type of integration are **differential forms**. Chances are, the reader is already very well acquainted with forms, maybe without even knowing it.

2.1.1 An Intuitive Definition

A differential form (also denoted as exterior[1] differential form) is, informally, an integrand, *i.e.*, a quantity that can be integrated. It is the dx in $\int dx$ and the $dx\ dy$ in $\iint dx\,dy$. More precisely, consider a smooth function $F(x)$ over an interval in $\mathbb{R}$. Now, define $f(x)$ to be its derivative, that is,

$$f(x) \;=\; \frac{dF}{dx} ,$$

Rewriting this last equation (using slightly abusive notations for simplicity) yields $dF = f(x)dx$, which leads to:

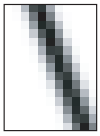
$$\int_a^b dF = \int_a^b f(x)dx = F(b) - F(a). \qquad (1)$$

[1]The word "exterior" is used as the exterior algebra is basically built out of an *outer* product.

This last equation is known as the Newton-Leibnitz formula, or the first fundamental theorem of calculus. The integrand $f(x)dx$ is called a *1-form*, because it can only be integrated over any 1-dimensional (1D) real interval. Similarly, for a function $G(x, y, z)$, we have:

$$dG = \frac{\partial G}{\partial x}dx + \frac{\partial G}{\partial y}dy + \frac{\partial G}{\partial z}dz ,$$

which can be integrated over any 1D curve in $\mathbb{R}^3$, and is also a 1-form. More generally, *a k-form can be described as an entity ready (or designed, if you prefer) to be integrated on a kD (sub)region*. Note that forms are valued zero on (sub)regions that are of higher or lower order dimension than the original space; for example, 4-forms are zero on $\mathbb{R}^3$. These differential forms are extensively used in mathematics, physics and engineering, as we already hinted at the fact in Section 1.4 that most of our measurements of the world are of integral nature: even digital pictures are made out of local area integrals of the incident light over each of the sensors of a camera to provide a set of values at each pixel on the final image (see inset). The importance of this notion of forms in science is also evidenced by the fact that operations like gradient, divergence, and curl can all be expressed in terms of forms only, as well as fundamental theorems like Green's or Stokes.



2.1.2 A Formal Definition

For concreteness, consider the n-dimensional Euclidean space $\mathbb{R}^n$, $n \in \mathbb{N}$ and let $\mathcal{M}$ be an open region $\mathcal{M} \subset \mathbb{R}^n$; $\mathcal{M}$ is also called an *n-manifold*. The vector space $T_x\mathcal{M}$ consists of all the (tangent) vectors at a point $x \in \mathcal{M}$ and can be identified with $\mathbb{R}^n$ itself. A k-form ω^k is a rank-k, anti-symmetric, tensor field over $\mathcal{M}$. That is, at each point $x \in \mathcal{M}$, it is a multi-linear map that takes k tangent vectors as input and returns a real number:

$$\omega^k : T_xM \ldots \times T_xM \longrightarrow \mathbb{R}$$

which *changes sign for odd permutations of the variables* (hence the term antisymmetric). Any k-form naturally induces a k-form on a submanifold, through restriction of the linear map to the domain that is the product of tangent spaces of the submanifold.

Comments on the Notion of Pseudo-forms There is a closely related concept named pseudo-form. Pseudo-forms change sign when we change the orientation of coordinate systems, just like pseudo-vectors. As a result, the integration of a pseudo-form does not change sign when the orientation of the manifold is changed. Unlike k-forms, a pseudo-k-form induces a pseudo-k-form on a submanifold *only* if a transverse direction is given. For example, fluid flux is sometimes called a pseudo-2-form: indeed, given a transverse direction, we know how much flux is going through a piece of surface; it does not depend on the orientation of the surface itself. Vorticity is, however, a true 2-form: given an orientation of the surface, the integration gives us the circulation around that surface boundary induced by the surface orientation. It does *not* depend on the transverse direction of the surface. But if we have an orientation of the ambient space, we can always associate transverse direction with internal orientation of the submanifold. Thus, in our case, we may treat pseudo-forms simply as forms because we can consistently choose a representative from the equivalence class.

2.2 The Differential Structure

Differential forms are the building blocks of a whole calculus. To manipulate these basic blocks, Exterior Calculus defines seven operators:

- $\diamond$ d: the exterior derivative, that extends the notion of the differential of a function to differential forms;
- $\diamond$ $\star$: the Hodge star, that transforms k-forms into $(n\text{-}k)$-forms;
- $\diamond$ $\wedge$: the wedge product, that extends the notion of exterior product to forms;
- $\diamond$ $\sharp$ and $\flat$: the sharp and flat operators, that, given a metric, transform a 1-form into a vector and vice-versa;
- $\diamond$ i_X: the interior product with respect to a vector field X (also called contraction operator), a concept dual to the exterior product;
- $\diamond$ $\mathcal{L}_X$: the Lie derivative with respect to a vector field X, that extends the notion of directional derivative.

In this chapter, we will restrict our discussions to the first three operators, to provide the most basic tools necessary in computational modeling.

2.3 A Taste of Exterior Calculus in $\mathbb{R}^3$

To give the reader a taste of the relative simplicity of Exterior Calculus, we provide a list of equivalences (in the continuous world!) between traditional operations and their Exterior Calculus counterpart in the special case of $\mathbb{R}^3$. We will suppose that we have the usual Euclidean metric. Then, forms are actually quite simple to conceive:

0-form $\Leftrightarrow$ scalar field
1-form $\Leftrightarrow$ vector field
2-form $\Leftrightarrow$ vector field
3-form $\Leftrightarrow$ scalar field

To be clear, we will add a superscript on the forms to indicate their rank. Then applying forms to vector fields amounts to:

1-form: $u^1(v) \Leftrightarrow u \cdot v$.
2-form: $u^2(v,w) \Leftrightarrow u \cdot (v \times w)$.
3-form: $f^3(u,v,w) \Leftrightarrow fu \cdot (v \times w)$.

Furthermore, the usual operations like gradient, curl, divergence and cross product can all be expressed in terms of the basic exterior calculus operators. For example:

$$d^0 f = \nabla f,\ d^1 u = \nabla \times u,\ d^2 u = \nabla \cdot u;$$
$$\star^0 f = f,\ \star^1 u = u,\ \star^2 u = u,\ \star^3 f = f;$$
$$\star^0 d^2 \star^1 u^1 = \nabla \cdot u,\ \star^1 d^1 \star^2 u^2 = \nabla \times u,\ \star^2 d^0 \star^3 f = \nabla f;$$
$$f^0 \wedge u = fu,\ u^1 \wedge v^1 = u \times v,\ u^1 \wedge v^2 = u^2 \wedge v^1 = u \cdot v;$$
$$i_v u^1 = u \cdot v,\ i_v u^2 = u \times v,\ i_v f^3 = fv.$$

Now that we have established the relevance of differential forms even in the most basic vector operations, time has come to turn our attention to make this concept of forms readily usable for computational purposes.

3 Discrete Differential Forms

Finding a discrete counterpart to the notion of differential forms is a delicate matter. If one was to represent differential forms using their coordinate values and approximate the exterior derivative using finite differences, basic theorems such as Stokes theorem would not hold numerically. The main objective of this section is therefore to present a proper discretization of the forms on what is known as simplicial complexes. We will show how this discrete geometric structure, well suited for computational purposes, is designed to preserve all the fundamental differential properties. For simplicity, we restrict the discussion to forms on 2D surfaces or 3D regions embedded in $\mathbb{R}^3$, but the construction is applicable to general manifolds in arbitrary spaces. In fact, the only necessary assumption is that the embedding space must be a vector space, a natural condition in practice.

3.1 Simplicial Complexes and Discrete Manifolds

For the interested reader, the notions we introduce in this section are defined formally in much more details (for the general case of k-dimensional spaces) in references such as [Munkres 1984] or [Hatcher 2004].

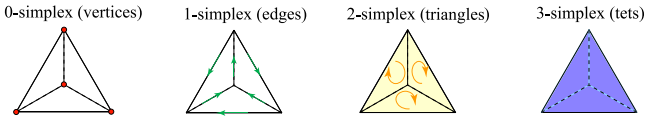


Figure 2: *A 1-simplex is a line segment, the convex hull of two points. A 2-simplex is a triangle,* i.e.*, the convex hull of three distinct points. A 3-simplex is a tetrahedron, as it is the convex hull of four points.*

3.1.1 Notion of Simplex

A k**-simplex** is the generic term to describe the simplest mesh element of dimension k—hence the name. By way of motivation, consider a three-dimensional mesh in space. This mesh is made of a series of adjacent tetrahedra (denoted *tets* for simplicity throughout). The vertices of the tets are said to form a 0-simplex. Similarly, the line segments or edges form a 1-simplex, the triangles or faces form a 2-simplex, and the tets a 3-simplex. Note that we can define these simplices in a top-down manner too: faces (2-simplex) can be thought of as boundaries of tets (3-simplices), edges (1-simplices) as boundaries of faces, and vertices (0-simplices) as boundaries of edges.

The definition of a simplex can be made more abstract as a series of k-tuples (referring to the vertices they are built upon). However, for the type of applications that we are targeting in this chapter, we will often not make any distinction between an abstract simplex and its topological realization (connectivity) or geometrical realization (positions in space) .

Formally, a k-simplex σ_k is the non-degenerate convex hull of k+1 geometrically distinct points $v_0, \dots v_k \in \mathbb{R}^n$ with $n \geq k$. In other words, it is the intersection of all convex sets containing $(v_0, \dots v_k)$; namely:

$$\sigma_k = \{x \in \mathbb{R}^n | x = \sum_{i=0}^{k} \alpha^i \, \mathrm{v}_i \ \text{ with } \ \alpha^i \geq 0 \text{ and } \sum_{i=0}^{k} \alpha^i = 1\}.$$

The entities $v_0, \dots v_k$ are called the *vertices* and k is called the dimension of the k-simplex., which we will denote as:

$$\sigma_k = \{v_0 v_1 ... v_k\} \ .$$

3.1.2 Orientation of a Simplex

Note that all orderings of the $k+1$ vertices of a k-simplex can be divided into two equivalent classes, *i.e.*, two orderings differing by an even permutation. Such a class of orderings is called an *orientation*. In the present work, we always assume that local orientations are *given* for each simplex; that is, each element of the mesh has been given a particular orientation. For example, an edge $\sigma_1 = \{v_0 v_1\}$ in Figure 2 has an arrow indicating its default orientation. If the opposite orientation is needed, we will denote it as $\{v_1 v_0\}$, or, equivalently, by $-\{v_0 v_1\}$. For more details and examples, the reader is referred to [Munkres 1984; Hirani 2003].

3.1.3 Boundary of a Simplex

Any (k-1)-simplex spanned by a subset of $\{v_0, \dots v_k\}$ is called a *(k-1)-face* of σ_k. That is, a *(k-1)-face* is simply a (k-1)-simplex

whose k vertices are all from the k+1 vertices of the k-simplex. The union of the (k-1)-faces is what is called the *boundary* of the k-simplex. One should be careful here: because of the default orientation of the simplices, the formal *signed* sum of the (k-1)-faces defines the boundary of the k-simplex. Therefore, the boundary operator takes a k-simplex and gives the sum of all its (k-1)-faces with 1 or -1 as coefficients depending on whether their respective orientations match or not, see Figure 4.

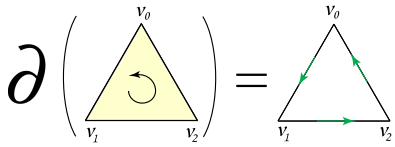


Figure 3: *The boundary operator ∂ applied to a triangle (a 2-simplex) is equal to the signed sum of the edges (*i.e.*, the 1-faces of the 2-simplex).*

To remove possible mistakes in orientation, we can define the *boundary operator* as follows:

$$\partial\{v_0 v_1 ... v_k\} = \sum_{j=0}^{k} (-1)^j \{v_0, ..., \widehat{v}_j, ..., v_k\}, \qquad (2)$$

where $\widehat{v}_j$ indicates that v_j is missing from the sequence, see Figure 3. Clearly, each k-simplex has k+1 (k-1)-faces. For this statement to be valid even for $k = 0$, the empty set $\emptyset$ is usually defined as a (-1)-simplex face of every 0-simplex. The reader is invited to verify this definition on the triangle $\{v_0, v_1, v_2\}$ in Figure 3:

$$\partial\{v_0, v_1, v_2\} = \{v_1, v_2\} - \{v_0, v_2\} + \{v_0, v_1\}$$

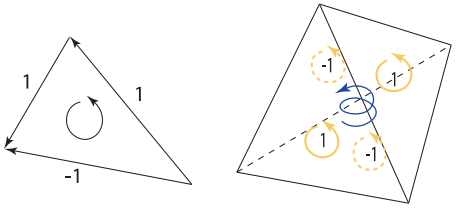
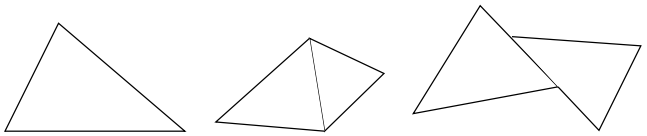


Figure 4: *Boundary operator applied to a triangle (left), and a tetrahedron (right). Orientations of the simplices are indicated with arrows.*

3.1.4 Simplicial Complex

A simplicial complex is a collection $\mathcal{K}$ of simplices, which satisfies the following two simple conditions:

- every face of each simplex in $\mathcal{K}$ is in $\mathcal{K}$;
- the intersection of any two simplices in $\mathcal{K}$ is either empty, or an entire common face.



Simplicial complexes *Not a simplicial complex*

Computer graphics makes heavy use of what is called *realizations* of simplicial complexes. Loosely speaking, a realization of a simplicial complex is an embedding of this complex into the underlying space $\mathbb{R}^n$. Triangle meshes in 2D and tet meshes in 3D are examples of such simplicial complexes (see Figure 1). Notice that polygonal meshes can be easily triangulated, thus can be easily turned into simplicial complexes. One can also use the notion of *cell complex* by allowing the elements of $\mathcal{K}$ to be non-simplicial; we will restrict our explanations to the simpler case of simplicial complexes for simplicity.

3.1.5 Discrete Manifolds

An n-dimensional discrete manifold $\mathcal{M}$ is an n-dimensional simplicial complex that satisfies the following condition: for each simplex, the union of all the incident n-simplices forms an n-dimensional ball (*i.e.*, a disk in 2D, a ball in 3D, *etc*), or half a ball if the simplex is on the boundary. As a consequence, each (n-1)-simplex has exactly two adjacent n-simplices—or only one if it is on a boundary.

Basically, the notion of discrete manifold corresponds to the usual Computer Graphics acceptation of "manifold mesh". For example in 2D, discrete manifolds cannot have isolated edges (also called sticks or hanging edges) or isolated vertices, and each of their edges is adjacent to 2 triangles (except for the boundary; in that case, the edge is adjacent to only one triangle). A surface mesh in 3D cannot have a "fin", *i.e.*, an edge with more than two adjacent triangles. To put it differently, infinitesimally-small, imaginary *inhabitants* of a n-dimensional discrete manifolds would consider themselves living in $\mathbb{R}^n$ as any small neighborhood of this manifold is isomorphic to $\mathbb{R}^n$.

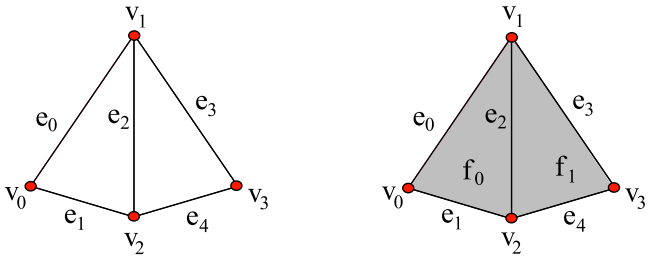


Figure 5: *(a) A simplicial complex consisting of all vertices $\{v_0, v_1, v_2, v_3\}$ and edges $\{e_0, e_1, e_2, e_3, e_4\}$. This simplicial complex is not a discrete manifold because the neighborhoods of the vertices v_1 and v_2 are not 1D balls. (b) If we add the triangles f_0 and f_1 to the simplicial complex, it becomes a 2-manifold with one boundary.*

3.2 Notion of Chains

We have already encountered the notion of chain, without mentioning it. Recall that the boundary operator takes each k-simplex and gives the *signed* sum of all its (k-1)-faces. We say that the boundary of a k-simplex produces a *(k-1)-chain*. The following definition is more precise and general.

3.2.1 Definition

A k-**chain** of an oriented simplicial complex $\mathcal{K}$ is a set of values, one for *each k-simplex* of $\mathcal{K}$. That is, a k-chain c can then be thought of as a linear combination of all the k-simplices in $\mathcal{K}$:

$$c = \sum_{\sigma \in \mathcal{K}} c(\sigma) \cdot \sigma, \qquad (3)$$

where $c(\sigma) \in \mathbb{R}$. We will denote the group of all k-chains as $\mathcal{C}_k$.

3.2.2 Implementation of Chains

Let the set of all k-simplices in $\mathcal{K}$ be denoted $\mathcal{K}^k$, and let its cardinality be denoted as $|\mathcal{K}^k|$. A k-chain can simply be stored as a *vector* (or array) of dimension $|\mathcal{K}^k|$, *i.e.*, one number for each k-simplex $\sigma_k \in \mathcal{K}^k$.

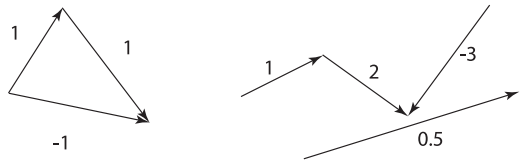
3.2.3 Boundary Operator on Chains

We mentioned that the boundary operator ∂ was returning a particular type of chain, namely, a chain with coefficients equal to either 0,

1, or -1. Therefore, it should not be surprising that we can extend the notion of boundary to act also on k-chains, simply by linearity:

$$\partial \sum_k c_k \sigma_k = \sum_k c_k \partial \sigma_k.$$

That is, from one set of values assigned to all simplices of a complex, one can deduce another set of values derived by weighting the boundaries of each simplex by the original value stored on it. This operation is very natural, and can thus be implemented easily as explained next.



(a) (b)

Figure 6: *(a) An example of* 1*-chain being the boundary of a face (*2*-simplex); (b) a second example of* 1*-chain with 4 non-zero coefficients.*

3.2.4 Implementation of the Boundary Operator

Since the boundary operator is a linear mapping from the space of k-simplices to the space of (k-1)-simplices, it can simply be represented by a *matrix* of dimension $|\mathcal{K}^{k-1}| \times |\mathcal{K}^k|$. The reader can convince herself that this matrix is sparse, as only immediate neighbors are involved in the boundary operator. Similarly, this matrix contains only the values 0, 1, and -1. Notice than in 3D, there are three non-trivial boundary operators ∂_k (∂_1 is the boundary operator on edges, ∂_2 on triangles, ∂_3 on tets). However, the operator needed for a particular operation is obvious from the type of the argument: if the boundary of a tet is needed, the operator ∂_3 is the only one that makes sense to apply; in other words, the boundary of a k-simplex σ_k is found by invoking $\partial_k \sigma_k$. Thanks to this context-dependence, we can simplify the notation and remove the subscript when there is no ambiguity.

3.3 Notion of Cochains

A k-cochain ω is the *dual* of a k-chain, that is to say, ω is a linear mapping that takes k-chains to $\mathbb{R}$. One writes:

$$\begin{array}{rccc} \omega: & \mathcal{C}_k & \rightarrow & \mathbb{R} \\ & c & \rightarrow & \omega(c), \end{array} \tag{4}$$

which reads as: a k-cochain ω operates on a k-chain c to give a scalar in $\mathbb{R}$. Since a chain is a linear combination of simplices, a cochain returns a linear combination of the values of that cochain on each simplex involved.

Clearly, a co-chain also corresponds to one value per simplex (since all the k-simplices form a basis for the vector space $\mathcal{C}_k$, and we only need to know the mapping of vectors in this basis to determine a linear mapping), and hence the notion of duality of chains and co-chains is appropriate. But contrary to a chain, a k-cochain is *evaluated* on each simplex of the dimension k. In other words, a k-cochain can be thought of as a *field* that can be evaluated on each k-simplex of an oriented simplicial complex $\mathcal{K}$.

3.3.1 Implementation of Cochains

The numerical representation of cochains follows from that of chains by duality. Recall that a k-chain can be represented as a vector c_k of length equal to the number of k-simplices in $\mathcal{M}$. Similarly, one may represent ω by a vector ω^k of the same size as c_k.

Now, remember that ω operates on c to give a scalar in $\mathbb{R}$. The linear operation $\omega(c)$ translates into an inner product $\omega^k \cdot c_k$. More specifically, one may continue to think of c_k as a *column vector* so that the $\mathbb{R}$-valued linear mapping ω can be represented by a *row vector* $(\omega^k)^t$, and $\omega(c)$ becomes simply the matrix multiplication of the row vector $(\omega^k)^t$ with the column vector c_k. The evaluation of a cochain is therefore trivial to implement.

3.4 Discrete Forms as Co-Chains

The attentive reader will have noticed by now: *k-cochains are discrete analogs to differential forms*. Indeed, a continuous k-form was defined as a linear mapping from k-dimensional sets to $\mathbb{R}$, as we can only integrate a k-form on a k-(sub)manifold. Note now that a kD set, when one has only a mesh to work with, is simply a *chain*. And a linear mapping from a chain to a real number is what we called a cochain: *a cochain is therefore a natural discrete counterpart of a form*.

For instance a 0-form can be evaluated at each point, a 1-form can be evaluated on each curve, a 2-form can be evaluated on each surface, *etc*. Now if we *restrict* integration to take place only on the k-submanifold which is the sum of the k-simplices in the triangulation, we get a k-cochain; thus k-cochains are a discretization of k-forms. One can further map a continuous k-form to a k-cochain. To do this, first integrate the k-form on each k-simplex and assign the resulting value to that simplex to obtain a k-cochain on the k-simplicial complex. This k-cochain is a discrete representation of the original k-form.

3.4.1 Evaluation of a Form on a Chain

We can now naturally extend the notion of evaluation of a differential form ω on an arbitrary chain simply by linearity:

$$\int_{\sum_i c_i \sigma_i} \omega = \sum_i c_i \int_{\sigma_i} \omega. \tag{5}$$

As mentioned above, the integration of ω on each k-simplex σ_k provides a discretization of ω or, in other words, a mapping from the k-form ω to a k-cochain represented by:

$$\omega[i] = \int_{\sigma_i} \omega.$$

However convenient this chain/cochain standpoint is, in practical applications, one often needs a point-wise value for a k-form or to evaluate the integration on a particular k-submanifold. How do we get these values from a k-cochain? We will cover this issue of *form interpolation* in Section 6.

4 Operations on Chains and Cochains

4.1 Discrete Exterior Derivative

In the present discrete setting where the discrete differential forms are defined as cochains, defining a discrete exterior derivative can be done very elegantly: Stokes' theorem, mentioned early on in Section 2, can be used to *define* the exterior derivative d. Traditionally, this theorem states a vector identity equivalent to the well-known curl, divergence, Green's, and Ostrogradsky's theorems. Written in terms of forms, the identity becomes quite simple: it states that d applied to an arbitrary form ω is evaluated on an arbitrary simplex σ as follows:

$$\int_\sigma d\omega = \int_{\partial\sigma} \omega. \tag{6}$$

You surely recognize the usual property that an integral over a k-dimensional set is turned into a boundary integral (*i.e.*, over a set of dimension k-1). With this simple equation relating the evaluation of $d\omega$ on a simplex σ to the evaluation of ω on the boundary of this simplex, the exterior derivative is *readily defined*: each time you encounter an exterior derivative of a form, replace any evaluation over a simplex σ by a direct evaluation of the form itself over the boundary of σ. Obviously, Stokes' theorem will be enforced by construction!

4.1.1 Coboundary Operator

The operator d is called the *adjoint* of the boundary operator ∂: if we denote the integral sign as a pairing, *i.e.*, with the convention that $\int_\sigma \omega = [\omega, \sigma]$, then applying d on the left hand side of this operator is equivalent to applying ∂ on the right hand: $[d\omega, \sigma] = [\omega, \partial\sigma]$. For this very reason, d is sometimes called the coboundary operator.

Finally, by linearity of integration, we can write a more general expression of Stokes' theorem, now extended to arbitrary chains as follows:

$$\int_{\sum_i c_i\sigma_i} d\omega = \int_{\partial\left(\sum_i c_i\sigma_i\right)} \omega = \int_{\sum_i c_i\partial\sigma_i} \omega = \sum_i c_i \int_{\partial\sigma_i} \omega$$

Consider the example shown in Figure 7. The discrete exterior derivative of the 1-form, defined as numbers on edges, is a 2-form represented by numbers on oriented faces. The orientation of the 1-forms may be opposite to that induced on the edges by the orientation of the faces. In this case, the values on the edges change sign. For instance, the 2-form associated with the d of the 1-forms surrounding the oriented shaded triangle takes the value $\omega = 2 - 1 - 0.75 = 0.25$.

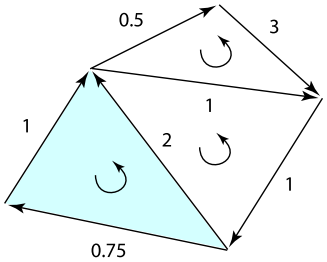


Figure 7: *Given a* 1*-form as numbers on oriented edges, its discrete exterior derivative is a* 2*-form. In particular, this* 2*-form is valued 0.25 on the oriented shaded triangle.*

4.1.2 Implementation of Exterior Derivative

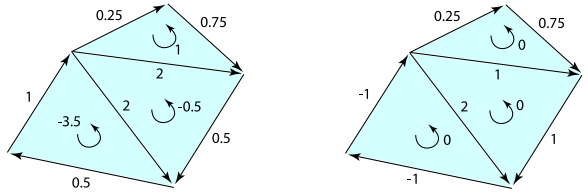
Since we use vectors of dimension $|\mathcal{K}^k|$ to represent a k-cochain, the operator d can be represented by a matrix of dimension $|\mathcal{K}^{k+1}| \times |\mathcal{K}^k|$. Furthermore, this matrix has a trivial expression. Indeed, using the matrix notation introduced earlier, we have:

$$\int_{\partial c} \omega = \omega^t(\partial c) = (\omega^t\partial)c = (\partial^t\omega)^t c = \int_c d\omega.$$

Thus, the matrix d is simply equal to ∂^t. This should not come as a surprise, since we previously discussed that d is simply the adjoint of ∂. Note that care should be used when boundaries are present. However, and without digging too much into the details, it turns out that even for discrete manifolds *with* boundaries, the previous statement is valid. Implementing the exterior derivative while preserving Stokes' theorem is therefore a trivial matter in practice. Notice that just like for the boundary operator, there is actually more than one matrix for the exterior derivative operator: there is one per simplex dimension. But again, the context is sufficient to actually know which matrix is needed. A brute force approach that gets rid of these multiple matrices is to use a notion of super-chain, *i.e.*, a vector storing *all* simplices, ordered from dimension 0 to the dimension of the space: in this case, the exterior derivative can be defined as a single, large sparse matrix that contains these previous matrices as blocks along the diagonal. We will not use this approach, as it makes the exposition less intuitive in general.

4.2 Exact/Closed Forms and Poincaré Lemma

A k-form ω is called exact if there is a (k-1)-form α such that $\omega = d\alpha$, and it is called closed if $d\omega = 0$.



(a) *(b)*

Figure 8: *(a) The 2-form on the oriented shaded triangles defined by the exterior derivative* d *of the 1-form on the oriented edges is called an* exact 2-form*; (b) The 1-form on the oriented edges whose derivative* d *is identically zero is called a* closed 1-form*.*

It is worth noting here that every exact form is closed, as will be seen in Section 4.3. Moreover, it is well-known in the continuous setting that a closed form on a smooth contractible (sub)-manifold is locally exact (to be more accurate: exact over any disc-like region). This result is called the *Poincaré lemma*. The discrete analogue to this lemma can be stated as follows: given a closed k-cochain ω on a star-shaped complex, that is to say, $d\omega = 0$, there exits a (k-1)-cochain α such that $\omega = d\alpha$. For a formal statement and proof of this discrete version, see [Desbrun et al. 2004].

4.3 Introducing the deRham Complex

The boundary of a boundary is the empty set. That is, the boundary operator applied twice to a k-simplex is zero. Indeed, it is easy to verify that $\partial\,\partial\sigma_k = 0$, since each ($k$-2)-simplex will appear exactly twice in this chain with different signs and, hence, cancel out (try it at home!). From the linearity of ∂, one can readily conclude that the property $\partial\,\partial = 0$ is true for all k-chains since the k-simplices form a basis. Similarly, one has that the discrete exterior derivative satisfies $d\ d = \partial^t\partial^t = (\partial\,\partial)^t = 0$, analogously to the exterior derivative of differential forms (notice that this last equality corresponds to the equality of mixed partial derivatives, which in turn is responsible for identities like $\nabla \times \nabla = 0$ and $\nabla \cdot \nabla \times = 0$ in $\mathbb{R}^3$).

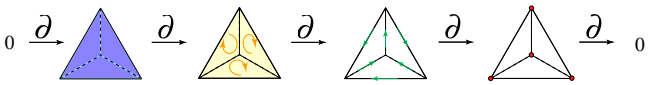


Figure 9: *The chain complex of a tetrahedron with the boundary operator: from the tet, to its triangles, to their edges, and to their vertices.*

4.3.1 Chain Complex

In general, a *chain complex* is a sequence of linear spaces, connected with a linear operator D that satisfies the property $D\,D = 0$. Hence, the boundary operator ∂ (resp., the coboundary operator d) makes the spaces of chains (resp., cochains) into a chain complex, as shown in Figures 9 and 13.

When the spaces involved are the spaces of **differential** forms, and the operator is the exterior derivative d, this chain complex is called

the *deRham complex*. By analogy, the chain complex for the spaces of **discrete** forms and for the coboundary operator is called the discrete deRham complex (or sometimes, the cochain complex).

4.3.2 Examples

Consider the 2D simplicial complex in Figure 10(a) and choose the oriented basis of the i-dimensional simplices ($i = 0$ for vertices, $i = 1$ for edges and $i = 2$ for the face) as suggested by the ordering in the figure.

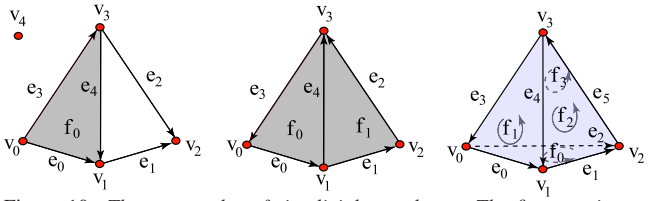


Figure 10: *Three examples of simplicial complexes. The first one is not manifold. The two others are.*

One gets $\partial(f_0) = e_0 - e_4 - e_3$, which can be identified with the vector $(1, 0, 0, -1, -1)$ representing the coefficient in front of each simplex. By repeating similar calculations for all simplices, one can readily conclude that the boundary operator ∂ is given by:

$$\partial_2 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ -1 \\ -1 \end{pmatrix}, \quad \partial_1 = \begin{pmatrix} -1 & 0 & 0 & -1 & 0 \\ 1 & -1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix},$$

That is, the chain complex under the boundary operator ∂ can be written as:

$$0 \longrightarrow \mathcal{C}_2 \xrightarrow{\partial_2} \mathcal{C}_1 \xrightarrow{\partial_1} \mathcal{C}_0 \longrightarrow 0$$

where $\mathcal{C}_i$, $i = 0, 1, 2$, denote the spaces of i-chains.

Consider now the domain to be the mesh shown in Figure 10(b). The exterior derivative operator, or the coboundary operator, can be expressed as:

$$d^0 = \begin{pmatrix} -1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 0 & -1 & 1 \\ 1 & 0 & 0 & -1 \\ 0 & -1 & 0 & 1 \end{pmatrix}, \qquad d^1 = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & -1 \end{pmatrix}.$$

It is worth noting that, since d is adjoint to ∂ by definition, the coboundary operator d induces a cochain complex:

$$0 \longleftarrow \mathcal{C}^2 \xleftarrow{d^1} \mathcal{C}^1 \xleftarrow{d^0} \mathcal{C}^0 \longleftarrow 0$$

where $\mathcal{C}^i$, $i = 0, 1, 2$, denote the spaces of i-cochains.

Finally, suppose the domain is the tetrahedron in Figure 10(c), then the exterior derivative operators are:

$$d^0 = \begin{pmatrix} -1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 \\ -1 & 0 & 1 & 0 \\ 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & -1 & 1 \end{pmatrix}, d^1 = \begin{pmatrix} 1 & 1 & -1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & -1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix}, d^2 = \begin{pmatrix} -1 & 1 & 1 & -1 \end{pmatrix}.$$

4.4 Notion of Homology and Cohomology

Homology is a concept dating back to Poincaré that focuses on studying the topological properties of a space. Loosely speaking, homology does so by counting the number of holes. In our case, since we assume that our space is a simplicial complex (*i.e.*, triangulated), we will only deal with *simplicial homology*, a simpler, more straightforward type of homology that can be seen as a discrete version of the continuous definition (in other words, it is equivalent to the continuous one if the domain is triangulated). As we are about to see, the notion of discrete forms is intimately linked with these topological notions. In fact, we will see that (co)homology is the study of the relationship between *closed* and *exact* (co)chains.

4.4.1 Simplicial Homology

A fundamental problem in topology is that of determining, for two spaces, whether they are topologically equivalent. That is, we wish to know if one space can be morphed into the other without having to puncture it. For instance, a sphere-shaped tet mesh is not topologically equivalent to a torus-shaped tet mesh as one cannot alter the sphere-shaped mesh (*i.e.*, deform, refine, or coarsen it locally) to make it look like a torus.

The key idea of homology is to define *invariants* (*i.e.*, quantities that cannot change by continuous deformation) that characterize topological spaces. The simplest invariant is the number of connected components that a simplicial complex has: obviously, two simplicial complexes with different numbers of pieces cannot be continuously deformed into each other! Roughly speaking, homology groups are an extension of this idea to define more subtle invariants than the number of connected components. In general, one can say that homology is a way to *define* the notion of holes/voids/tunnels/components of an object in any dimension.

Cycles and their Equivalence Classes Generalizing the previous example to other invariants is elegantly done using the notion of *cycles*. A cycle is simply a *closed k-chain*; that is, a linear combination of k-simplices so that the boundary of this chain (see Section 3.2) is the empty set. Any set of vertices is a closed chain; any set of 1D loops are too; etc. Equivalently, a k-cycle is any k-chain that belongs to $\mathrm{Ker}\ \partial_k$, by definition.

On this set of all k-cycles, one can define *equivalence classes*. We will say that a k-cycle is *homologous* to another k-cycle (*i.e.*, in the same equivalence class than the other) when these two chains differ by a boundary of a (k+1)-chain (*i.e.*, by an *exact chain*). Notice that this exact chain is, by definition (see Section 4.2), in the image of ∂_{k+1}, *i.e.*, $\mathrm{Im}\ \partial_{k+1}$. To get a better understanding of this notion of equivalence class, the reader is invited to look at Figure 11: the 1-chains L_1 and L_3 are part of the same equivalence class as their difference is indeed the boundary of a well-defined 2D chain—a rubber-band shape in this case. Notice that as a consequence, L_1 can be deformed into L_3 without having to tear the loop apart. However, L_2 is not of this class, and thus cannot be deformed into L_3; there's no 2-chain that corresponds to their difference.

4.4.2 Homology Groups

Let us now use these definition on the simple case of the 0^{th} homology group $\mathcal{H}_0$.

Homology Group $\mathcal{H}_0$ The boundary of any vertex is $\emptyset$. Thus, any linear combination of vertices is a 0-cycle by definition. Now if two vertices v_0 and v_1 are connected by an edge, $v_1 - v_0$ (*i.e.*, the difference of two cycles) is the boundary of this edge. Thus, by our previous definition, two vertices linked by an edge are *homologous* as their difference *is* the boundary of this edge. By the same reasoning, any two vertices taken from the *same* connected component are, also, homologous, since there exists a chain of edges in between. Consequently, we can pick only one vertex per connected component to form a basis of this homology group. Its dimension, β_0, is therefore simply the number of connected components. The basis elements of that group are called *generators*, since they generate the whole homology group.

Homology Group $\mathcal{H}_1$ Let us proceed similarly for the 1^{st} homology class: we now have to consider 1-cycles (linear combinations of 1D loops). Again, one can easily conceive that there are different *types* of such cycles, and it is therefore possible to separate all possible cycles into different equivalence classes. For instance,

the loop L_1 in Figure 11 is topologically distinct from the curve L_2: one is around a hole and the other is not, so the difference between the two is *not* the boundary of a 2-chain. Conversely, L_1 *is* in the same class as curve L_3 since they differ by one connected area. Thus, in this figure, the 1^{st} homology group is a 1-dimensional group, and L_1 (or L_3, equivalently) is its unique generator. The reader is invited to apply this simple idea on the triangulated torus, to find two loops as generators of $\mathcal{H}_1$.

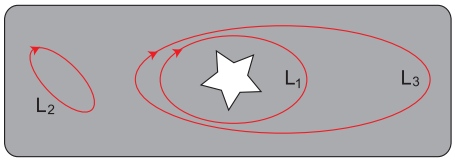


Figure 11: *Example of Homology Classes: the cycles L_1 and L_2 are topologically distinct as one encloses a hole while the other does not; L_1 and L_3 are however in the same equivalence class.*

Formal Definition of Homology Groups We are now ready to generalize this construction to all homology groups. Remember that we have a series of k-chain spaces:

$$C_n \xrightarrow{\partial_n} C_{n-1} \ldots \xrightarrow{\partial_2} C_1 \xrightarrow{\partial_1} C_0$$

with the property that $\partial\ \partial$ is the empty set. This directly implies that the image of C_j is always in the kernel of ∂_{j-1}—such a series is called a *chain complex*. Now, the homology groups $\{\mathcal{H}_k\}_{k=0..n}$ of a chain complex based on ∂ are defined as the following quotient spaces:

$$\mathcal{H}_k = \text{Ker } \partial_k / \text{Im } \partial_{k+1}.$$

The reader is invited to check that this definition is *exactly* what we did for the 0^{th} and 1^{st} homology groups—and it is now valid for any order: indeed, we use the fact that closed chains (belonging to Ker ∂) are homologous *iff* their difference is in Im ∂, and this is exactly what this quotient vector space is.

Example Consider the example in Figure 10(a). Geometrically, $\mathcal{H}_0$ is nontrivial because the simplicial complex σ is disconnected (it is easy to see $\{v_0, v_4\}$ form a basis for $\mathcal{H}_0$), while $\mathcal{H}_1$ is nontrivial since the cycle $(e_1 - e_2 + e_4)$ is not the boundary of any 2-chain of σ ($\{(e_1 - e_2 + e_4)\}$ is indeed a basis for this 1D space $\mathcal{H}_1$).

Link to Betti Numbers The dimension of the k-th cohomology group is called k-th Betti number; $\beta_k = \dim \mathcal{H}_k$. For a 3D simplicial complex embedded in $\mathbb{R}^3$, these numbers have very straightforward meanings. β_0 is the number of connected components, β_1 is the number of tunnels, β_2 is the number of voids, while β_3 is the number of 4D holes, which is 0 in the Euclidean (flat 3D) case. Finally, note that $\sum_{k=0..n}(-1)^k \beta_k$, where β_k is the k-th Betti number, gives us the well-known Euler characteristic.

4.4.3 Cohomology Groups

The definition of homology groups is much more general than what we just reviewed. In fact, the reader can take the formal definition in the previous section, replace all occurrences of chain by cochain, of ∂ by d, and reverse the direction of the operator between spaces (see Section 4.3.2): this will also define equivalence classes. Because cochains are dual of chains, and d is the adjoint of ∂, these equivalence classes define what are actually denoted as *cohomology groups*: the cohomology groups of the deRham complex for the coboundary operator are simply the quotient spaces Ker d/Im d. Finally, note that the homology and cohomology groups are not only dual notions, but they are also isomorphic; therefore, the cardinalities of their bases are equal.

4.4.4 Calculation of the Cohomology Basis

One usual way to calculate a cohomology basis is to calculate a Smith Normal Form to obtain the homology basis first (possibly using progressive meshes [Gu and Yau 2003]), with a worst case complexity of $O(n^3)$, and then find the corresponding cohomology basis derived from this homology basis. We provide an alternative method here with worst case complexity also equal to $O(n^3)$. The advantage of our method is that it directly calculates the cohomology basis.

Our algorithm is a modified version of an algorithm in [Edelsbrunner et al. 2000], although they did not use it for the same purpose[2]. We will use $row\#(.)$ to refer to the row number of the *last non-zero coefficient* in a particular column.

The procedure is as follows:

1. Transform d^k (size $|\mathcal{K}^{k+1}| \times |\mathcal{K}^k|$) in the following manner:

```
// For each column of d^k
for(i = 0; i < |K^k|; i++)
    // Reduce column i
    repeat
        p ← row#(d^k[i])
        find j < i such as p==row#(d^k[j])
        make d^k[i][p] zero by adding to d^k[i] a multiple of d^k[j]
    until j_not_found or column i is all zeros
```

   In the end of this procedure, we get $D^k = d^k\ N^k$, whose non-zero column vectors are linearly independent of each other and with different $row\#(.)$, and N^k is a non-singular upper triangular matrix.

2. Construct $K^k = \{N_i^k \mid D_i^k = 0\}$ (where N_i^k and D_i^k are column vectors of matrices N^k and D^k respectively).
   K^k is a basis for kernel of d^k.

3. Construct $I^k = \{N_i^k | \exists j \text{ such that } i = row\#(D_j^{(k-1)})\}$

4. Construct $P^k = K^k - I^k$
   P^k is a basis of the cohomology.

Short proof of correctness: First, notice that the N_i^k's are all linearly independent because N^k is nonsingular. For any non-zero linear combination of vectors in P^k , $row\#(.)$ of it (say i) equals the max of $row\#(.)$ of vectors with non-zero coefficients. But i is not $row\#(.)$ of any $D_i^{(k-1)}$ (and thus any linear combination of them) by definition of P^k. Therefore, we know that the linear combination is not in the image space of d^{k-1} (since the range of d^{k-1} is the same as D^{k-1}, by construction). Thus, P^k spans a subspace of $\text{Ker}(d^k)/\text{Im}(d^{k-1})$ of dimension $\text{Card}(P^k)$.

One can also prove that I^k is a subset of K^k. Pick such an N_i^k with $i = row\#(D_j^{(k-1)})$. We have: $d^k\ D_j^{(k-1)} = 0$ (since $d^k \circ d^{k-1} = 0$). Now $row\#(\tau \equiv (N^k)^{-1}\ {d^{(k-1)}}_j) = i$ (the inverse of an upper triangular matrix is also an upper triangular matrix). So consequently, $0 = d^k\ {d^{(k-1)}}_j = D^k (N^k)^{-1}\ {d^{(k-1)}}_j = D^k \tau$ means that $D_i^k = 0$ because the columns of D^k are linearly independent or 0. Therefore, $\text{Card}(P^k) = \text{Card}(K^k) - \text{Card}(I^k) = dim(\text{Ker}(d^k)) - dim(\text{Im}(d^{(k-1)}))$, and we conclude that, P^k spans $\text{Ker}(d^k)/\text{Im}(d^{k-1})$ as expected. ∎

[2]Thanks to David Cohen-Steiner for pointing us to the similarities

4.4.5 Example

Consider the 2D simplicial complex in Figure 10(a) again. We will show an example of running the same procedure described above to compute a homology basis. The only difference with the previous algorithm is that we use ∂ instead of d, since we compute the homology basis instead of the cohomology basis.

1. Compute the $D^K = \partial_k N^k$'s and N^k's: D^2 is trivial, as it is the same as ∂_2.

$$D^1 = \begin{pmatrix} -1 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \quad N^1 = \begin{pmatrix} 1 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 & 1 \\ 0 & 0 & 1 & 1 & -1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix},$$

2. Contruct the K's:

$$K^0 = \{\begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}\}$$
$$= \{v_0, v_1, v_2, v_3, v_4\}$$

(N^0 is the identity)

$$K^1 = \{\begin{pmatrix} -1 \\ -1 \\ 1 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ -1 \\ 0 \\ 1 \end{pmatrix}\} = \{(-e_0 - e_1 + e_2 + e_3), (e_1 - e_2 + e_4)\}$$

3. Construct the I's:

$$\begin{aligned} I^0 &= \{v_1\ (1 = row\#(D_0^1)), \\ &\quad v_2\ (2 = row\#(D_1^1)), \\ &\quad v_3\ (3 = row\#(D_2^1))\} \\ I^1 &= \{(e_1 - e_2 + e_4)\ (4 = row\#(D_0^2)\} \end{aligned}$$

4. Consequently, the homology basis is:

$$P^0 = \{v_0, v_1, v_2, v_3, v_4\} - \{v_1, v_2, v_3\} = \{v_0, v_4\}$$
$$P^1 = \{(-e_0 - e_1 + e_2 + e_3)\}$$

This result confirms the basis we gave in the example of Section 4.4.2 (Note that $-(-e_0 - e_1 + e_2 + e_3) - (e_1 - e_2 + e_4) = e_0 - e_4 - e_3 = \partial f_0$, thus $(-e_0 - e_1 + e_2 + e_3)$ spans the same homology space as $(e_1 - e_2 + e_4)$).

4.5 Dual Mesh and its Exterior Derivative

Let us introduce the notion of *dual mesh* of triangulated manifolds, as we will see that it is one of the key components of our discrete calculus. The main idea is to associate to each *primal* k-simplex a *dual* $(n$-$k)$-cell. For example, consider the tetrahedral mesh in Figure 13, we associate a dual 3-cell to each primal vertex (0-simplex), a dual polygon (2-cell) to each primal edge (1-simplex), a dual edge (1-cell) to each primal face (2-simplex), and a dual vertex (0-cell) to the primal tet (3-simplex). By construction, the number of dual $(n$-$k)$-cells is equal to that of primal k-simplices. The collection of dual cells is called a *cell complex*, which need not be a simplicial complex in general.

Yet, this dual complex inherits several properties and operations from the primal simplicial complex. Most important is the notion of *incidence*. For instance, if two primal edges are on the same primal face, then the corresponding dual faces are incident, that is, they share a common dual edge (which is the dual of the primal common face). As a result of this incidence property, one may easily derive a boundary operator on the dual cell complex and, consequently, a discrete exterior derivative! The reader is invited to verify that this exterior derivative on the dual mesh can be simply written as the opposite of a primal one *transposed*:

$$d_{Dual}^{n-k} = (-1)^k (d_{Primal}^{k-1})^t. \tag{7}$$

The added negative sign appears as the orientation on the dual is induced from the primal orientation, and must therefore be properly accounted for. Once again, an implementation can overload the definition of this operator d when used on dual forms using this previous equation. In the remainder of our chapter, we will be using d as a contextual operator to keep the notations a simple as possible. Because we have defined a proper exterior derivative on the dual mesh (still satisfying $d \circ d = 0$), this dual cell complex also carries the structure of a chain complex. The structure on the dual complex may be linked to that of the primal complex using the Hodge star (a metric-dependent operator), as we will discuss in Section 5.

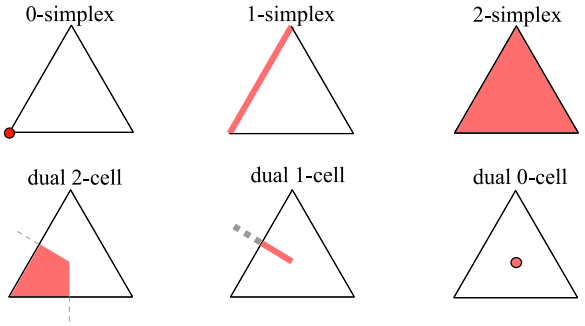


Figure 12: *A 2-dimensional example of primal and dual mesh elements. On the top row, we see the primal mesh (a triangle) with a representative of each simplicial complex being highlighted. The bottom row shows the corresponding circumcentric dual cells (restricted to the triangle).*

4.5.1 Dualization: The $*$ Operator

For simplicity, we use the circumcentric (or Voronoi) duality to construct the dual cell complex. The circumcenter of a k-simplex is defined as the center of the k-circumsphere, which is the unique k-sphere that has all $k+1$ vertices of the k-simplex on its surface. In Figure 12, we show examples of circumcentric dual cells of a 2D mesh. The dual 0-cell associated with the triangular face is the circumcenter of the triangle. The dual 1-cell associated with one of the primal edges is the line segment that joins the circumcenter of the triangle to the circumcenter of that edge, while the dual 2-cell associated with a primal vertex is corner wedge made of the convex hull of the circumcenter of the triangle, the two centers of the adjacent edges, and the vertex itself (see Figure 12, bottom left). Thereafter, we will denote as $*$ the operation of *duality*; that is, a primal simplex σ will have its dual called $*\sigma$ with the orientation induced by the primal orientation and the manifold orientation. For a formal definition, we refer the reader to [Hirani 2003] for instance. It is also worth noting that other notions of duality such as the barycentric duality may be employed. For further details on dual cell (or "block") decompositions , see [Munkres 1984].

4.5.2 Wedge Product

In the continuous setting, the wedge product $\wedge$ is an operation used to construct higher degree forms from lower degree ones; it is the antisymmetric part of the tensor product. For example, let α and β be 1-forms on a subset $\mathcal{R} \subset \mathbb{R}^3$, their wedge product $\alpha \wedge \beta$ is a 2-form on $\mathcal{R}$. In this case, one can relate the wedge product to the cross product of vector fields on $\mathcal{R}$. Indeed, if one considers the vector representations of α and β, the vector proxy to $\alpha \wedge \beta$ is the cross product of the two vectors. Similarly, the wedge product of a

1-form γ with the 2-form $\omega = \alpha \wedge \beta$ is a 3-form $\mu = \gamma \wedge \omega$ (also called volume-form) on $\mathcal{R}$ which is analogous to the scalar triple product of three vectors.

A discrete treatment of the wedge operator can be found in [Hirani 2003]. Here, we only need to introduce the notion of a discrete *primal-dual wedge product*: given a primal k-cochain γ and a dual $(n\text{-}k)$-cochain ω, the discrete wedge product $\gamma \wedge \omega$ is an n-form (or a volume-form). For instance, in the example depicted in the inset, the wedge product of the primal 1-cochain with the dual 1-cochain is a 2-form associated with the diamond region defined by the convex hull of the union between the primal *and* dual edge (see inset).

5 Metric-Dependent Operators on Forms

Notice that up to now, we did *not* assume that a metric was available, *i.e.*, we never required anything to be *measured*. However, such a metric is necessary for many purposes. For instance, simulating the behavior of objects around us requires measurements of various parameters in order to be able to model laws of motion, and compare the numerical results of simulations. Consequently, a certain number of operations on forms can only be defined once a metric is known, as we shall see in this section.

5.1 Notion of Metric and Inner Product

A *metric* is, roughly speaking, a nonnegative function that describes the "distance" between neighboring points of a given space. For example, the Euclidean metric assigns to any two points in the Euclidean space $\mathbb{R}^3$, say $\mathbf{X} = (x_1, x_2, x_3)$ and $\mathbf{Y} = (y_1, y_2, y_3)$, the number:

$$d(\mathbf{X}, \mathbf{Y}) = \|\mathbf{X} - \mathbf{Y}\|^2 = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + (x_3 - y_3)^2}$$

defining the "standard" distance between any two points in $\mathbb{R}^3$. This metric then allows one to measure length, area, and volume. The Euclidean metric can be expressed as the following quadratic form:

$$g^{\text{Euclid}} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Indeed, the reader can readily verify that this matrix g satisfies: $d^2(\mathbf{X}, \mathbf{Y}) = (\mathbf{X} - \mathbf{Y})^t g (\mathbf{X} - \mathbf{Y})$. Notice also that this metric *induces an inner product* of vectors. Indeed, for two vectors $\mathbf{u}$ and $\mathbf{v}$, we can use the matrix g to define:

$$\mathbf{u} \cdot \mathbf{v} = \mathbf{u}^t g \, \mathbf{v}.$$

Once again, the reader is invited to verify that this equality does correspond to the traditional dot product when g is the Euclidean metric. Notice that on a non-flat manifold, subtraction of two points is only possible for points infinitesimally close to each other, thus the metric is actually defined pointwise for the tangent space at each point: it does not have to be constant. Finally, notice that a volume form can be induced from a metric by defining $\mu^n = \sqrt{det(g)} dx^1 \wedge \ldots \wedge dx^n$.

5.2 Discrete Metric

In the discrete setting presented in this paper, we only need to measure length, area, and volume of the simplices and dual cells (note these different notions of sizes depending on dimension will be denoted "intrinsic volumes" for generality). We therefore do not have a full-blown notion of a metric, only a *discrete metric*. Obviously, if one were to use a finer mesh, more information on the metric would be available: having more values of length, area, and volume in a neighborhood provides a better approximation of the real, continuous metric.

5.3 The Differential Hodge Star

Let us go back for a minute to the differential case to explain a new concept. Recall that the metric defines an inner product for vectors. This notion also extends to forms: given a metric, one can define the product of two k-forms $\in \Omega^k(\mathcal{M})$ which will measure, in a way, the projection of one onto the other. A formal definition can be found in [Abraham et al. 1988]. Given this inner product denoted $\langle \ , \ \rangle$, we can introduce an operator $\star$, called the *Hodge star*, that maps a k-form to a complementary $(n\text{-}k)$-form:

$$\star : \ \Omega^k(\mathcal{M}) \to \Omega^{n-k}(\mathcal{M}),$$

and is defined to satisfy the following equality:

$$\alpha \wedge \star\beta = \langle \alpha, \beta \rangle \ \mu^n$$

for any pair of k-forms α and β (recall that μ^n is the volume form induced by the metric g). However, notice that the wedge product is very special here: it is the product of k-form and a $(n\text{-}k)$-form, two complementary forms. This fact will drastically simplify the discrete counterpart of the Hodge star, as we now cover.

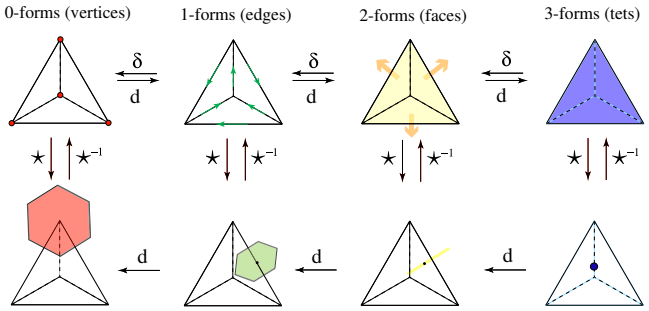


Figure 13: *On the first line, the 'primal' chain complex is depicted and on the second line we see the dual chain complex (i.e., cells, faces, edges and vertices of the Voronoi cells of each vertex of the primal mesh).*

5.4 Discrete Hodge Star

In the discrete setting, the Hodge star becomes easier: we only need to define how to go from a *primal* k-cochain to a *dual* $(n\text{-}k)$-cochain, and vice-versa. By definition of the dual mesh, k-chains and dual $(n\text{-}k)$-chains are represented by vectors of the same dimension. Similarly to the discrete exterior derivative (coboundary) operator, we may use a matrix (this time of size $|\mathcal{K}^k| \times |\mathcal{K}^k|$) to represent the Hodge star. Now the question is: what should the coefficients of this matrix be?

For numerical purposes we want it to be symmetric, positive definite, and sometimes, even diagonal for faster computations. One such diagonal Hodge star can be defined with the diagonal elements as the ratio of intrinsic volumes of a k-simplex and its dual $(n\text{-}k)$-simplex. In other words, we can define the discrete Hodge star through the following simple rule:

$$\frac{1}{|\sigma_k|} \int_{\sigma_k} \omega = \frac{1}{|*\sigma_k|} \int_{*\sigma_k} \star\omega \qquad (8)$$

Therefore, any primal value of a k-form can be easily *transferred* to the dual mesh through proper scaling—and vice-versa; to be precise, we have:

$$\star_k \star_{n-k} = (-1)^{k(n-k)} \mathrm{Id}, \tag{9}$$

which means that $\star$ on the dual mesh is the inverse of the $\star$ on the primal *up to a sign* (the result of antisymmetry of the wedge product, which happens to be positive for any k-form when $n = 3$).

So we must use the inverse of the Hodge star to go from a dual $(n$-$k)$-cochain to a k-cochain. We will, however, use $\star$ to indistinguishably mean either the star or its inverse, as there is no ambiguity once we know whether the operator is applied to a primal or a dual form: this is also a context-dependent operator.

Implementation Based on Eq. (8), the inner product of forms α^k and β^k at the diamond-shaped region formed by each k-simplex and its dual $(n-k)$-simplex is simply the product of the value of α at that k-simplex and value of $\star\beta$ at that dual $(n-k)$-simplex. Therefore, the sum over the whole space gives the following inner product (which involves only linear algebra matrix and vector multiplications)

$$\langle \alpha^k, \beta^k \rangle = \alpha^t \star \beta. \tag{10}$$

where the Hodge star matrix has, as its only non-zero coefficients, the following diagonal terms:

$$(\star_k)_{qq} = |(*\sigma)_q| / |(\sigma_q)|.$$

Notice that this definition of the inner product, when $\alpha = \beta$, induces the definition of the norm of k-forms.

Again, there are three different Hodge stars in $\mathbb{R}^3$, one for each simplex dimension. But as we discussed for all the other operators, the dimension of the form on which this operator is applied disambiguates which star is meant. So we will not encumber our notation with unnecessary indices, and will only use the symbol $\star$ for any of the three stars implied.

The development of an accurate, yet fast to compute, Hodge star is still an active research topic. However, this topic is beyond the scope of the current chapter.

5.5 Discrete Codifferential Operator δ

We already have a linear operator d which maps a k-form to a $k+1$-form, but we do not have a linear operator which maps a k-form to a $(k-1)$-form. Having defined a discrete Hodge star, one can now create such an *adjoint* operator δ for the discrete exterior derivative d. Here, adjoint is meant with respect to the inner product of forms; that is, this operator δ satisfies:

$$\langle d\alpha, \beta \rangle \;=\; \langle \alpha, \delta\beta \rangle \qquad \forall \alpha \in \Omega^{k-1}(M), \beta \in \Omega^k(M)$$

For a smooth, compact manifold without boundary, one can prove that $(-1)^{n(k-1)+1} \star d \star$ satisfies the above condition [Abraham et al. 1988]. Let us try to use the same definition in the discrete setting; *i.e.*, we wish to define the discrete δ applied to k-forms by the relation:

$$\delta \equiv (-1)^{n(k-1)+1} \star d \star, \tag{11}$$

Beware that we use the notation d to mean the context-dependent exterior derivative. If you apply δ to a primal k-form, then the exterior derivative will be applied to a *dual* $(n-k)$*-form*, and thus, Equation 7 should be used. Once this is well understood, it is quite straightforward to verify the following series of equalities:

$$\begin{aligned}
\langle d\alpha, \beta \rangle &\overset{Eq.\,(10)}{=} (d\alpha)^t \star \beta = \alpha^t d^t \star \beta \\
&\overset{Eq.\,(9)\ w/\ k \leftrightarrow k-1}{=} \alpha^t (-1)^{(k-1)(n-(k-1))} \star \star d^t \star \beta \\
&= \alpha^t (-1)^{n(k-1)+1} \star \star (-1)^k d^t \star \beta \\
&\overset{Eq.\,(11)}{=} \langle \alpha, \delta\beta \rangle
\end{aligned}$$

holds on our discrete manifold. So indeed, the discrete d and δ are also adjoint, in a similar fashion in the discrete setting as they were in the continuous sense. For this reason, δ is called the *codifferential operator*.

Implementation of the Codifferential Operator Thanks to this easily-proven adjointness, the implementation of the discrete codifferential operator is a trivial matter: it is simply the product of three matrices, mimicking exactly the differential definition mentioned in Eq. (11).

5.6 Exercise: Laplacian Operator

At this point, the reader is invited to perform a little exercise. Let us first state that the Laplacian Δ of a form is defined as: $\Delta = \delta d + d\delta$. Now, applied to a 0-form, notice that the latter term disappears. Question: in 2D, what *is* the Laplacian of a function f at a vertex i? The answer is actually known: it is the now famous *cotangent formula* [Pinkall and Polthier 1993], since the ratio of primal and dual edge lengths leads to such a trigonometric equality. Applied to a 1-form, however, the expression does have both terms as explicitly given in [Fisher et al. 2007].

6 Interpolation of Discrete Forms

In Section 3.4, we argued that k-cochains are discretizations of k-forms. This representation of discrete forms on chains, although very convenient in many applications, is not sufficient to fulfill certain demands such as obtaining a point-wise value of the k-form. As a remedy, one can use an interpolation of these chains to the rest of space. For simplicity, these interpolation functions can be taken to be linear (by linear, we mean with respect to the coordinates of the vertices).

6.1 Interpolating 0-forms

It is quite obvious how to linearly interpolate discrete 0-forms (as 0-cochains) to the whole space: we can use the usual vertex-based linear interpolation basis, often referred to as the *hat function* in the Finite Element literature. This basis function will be denoted as φ_i for each vertex v_i. By definition, φ_i satisfies:

$$\varphi_i = 1 \quad \text{at } v_i, \qquad \varphi_i = 0 \quad \text{at } v_j \neq v_i$$

while φ_i linearly goes to zero in the one-ring neighborhood of v_i. The reader may be aware that these functions are, within each simplex, *barycentric coordinates*, introduced by Möbius in 1827 as mass points to define a *coordinate-free geometry*.

With these basis functions, one can easily check that if we denote a vertex v_j by σ_j, we have:

$$\int_{v_j} \varphi_{v_i} = \int_{\sigma_j} \varphi_{\sigma_i} = \int_{\sigma_j} \varphi_i = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{if } i \neq j. \end{cases}$$

Therefore, these interpolating functions represent a basis of 0-cochains, that exactly corresponds to the dual of the natural basis of 0-chains.

6.2 Interpolating 1-forms

We would like to be able to extend the previous interpolation technique to 1-forms now. Fortunately, there is an existing method to do just that: the *Whitney* 1-form (used first in [Whitney 1957]) associated with an edge σ_{ij} between v_i and v_j is defined as:

$$\varphi_{\sigma_{ij}} = \varphi_i d\varphi_j - \varphi_j d\varphi_i.$$

A direct computation can verify that:

$$\int_{\sigma_{kl}} \varphi_{\sigma_{ij}} = \begin{cases} 1 & \text{if } i = k \text{ and } j = l, \\ -1 & \text{if } i = l \text{ and } j = k, \\ 0 & \text{otherwise.} \end{cases}$$

Indeed, it is easy to see that the integral is 0 when we are not integrating it on edge e_{ij}, because at least one of the vertices (say, i) is not on the edge, thus, $\varphi_i = 0$ and $d\varphi_i = 0$ on the edge. However, along the edge σ_{ij}, we have $\varphi_i + \varphi_j = 1$, therefore:

$$\int_{\sigma_{ij}} \varphi_{\sigma_{ij}} = \int_{\varphi_i=1}^{\varphi_i=0} (\varphi_i d(1-\varphi_i) - (1-\varphi_i)d\varphi_i) = \int_{\varphi_i=1}^{\varphi_i=0} (-d\varphi_i) = 1.$$

We thus have defined a correct basis for 1-cochains.

6.3 Interpolating with Whitney k-Forms

One can extend these 1-form basis functions to arbitrary k-simplices. In fact, Whitney k-forms are defined similarly:

$$\varphi_{\sigma_{i_0,i_1,\ldots,i_k}} = k! \sum_{j=0\ldots k} (-1)^j \varphi_{i_j} d\varphi_{i_0} \wedge \ldots \wedge \widehat{d\varphi_{i_j}} \ldots \wedge d\varphi_{i_k}$$

where $\widehat{d\varphi_{i_p}}$ means that $d\varphi_{i_p}$ is excluded from the product. Notice how this definition exactly matches the case of vertex and edge bases, and extends easily to higher dimensional simplices.

Remark If a metric is defined (for instance, the Euclidean metric), we can simply identify $d\varphi$ with $\nabla\varphi$ for the real calculation. This corresponds to the notion of *sharp* ($\sharp$), but we will not develop this point other than for pointing out the following remark: the traditional gradient of a linear function f in 2D, known to be constant per triangle, can indeed be re-written à la Whitney:

$$\nabla f = \sum_i f_i \nabla\varphi_i \overset{\varphi_i+\varphi_j+\varphi_k=1}{=} \sum_{i,j,i\neq j} (f_j - f_i)(\varphi_i \nabla\varphi_j - \varphi_j \nabla\varphi_i).$$

The values $(f_j - f_i)$ are the edge values associated with the gradient, *i.e.*, the values of the one-form df.

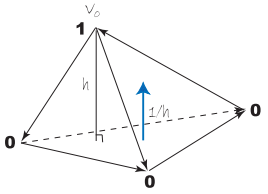


Figure 14: $\nabla\varphi$ *for the vertex on top*

Basis of Forms The integration of the Whitney form φ_{σ_k} associated with the k-simplex σ_k will be 1 on that particular simplex, and 0 on all others. Indeed, it is a simple exercise to see that the integration of φ_{σ_k} is 0 on a different k-simplex, because there is *at least one* vertex of this simplex v_j that does not belong to σ_k, so its hat function φ_j is valued 0 everywhere on σ_k. Since φ_j or $d\varphi_j$ appears in every term, the integral of φ_{σ_k} is 0. To see that the integral is 1 on the simplex itself, we can use Stokes' theorem (as our discrete forms satisfy it exactly on simplices): first, suppose $k < n$, and pick a $k+1$-simplex, such that the k-simplex σ_k is a *face* of it. Since it is 0 on other faces, the integral of the Whitney form is equal to the integral of $d\varphi_{\sigma_k} = (k+1)! d\varphi_{i_0} \wedge \ldots \wedge \varphi_{i_k}$ on the $k+1$-simplex, if we use φ_{i_j} as a local reference frame for the integration, $\int_{\sigma^{k+1}} d\varphi_{i_0} \wedge \ldots \wedge \varphi_{i_k}$ is simply the volume of a standard simplex, which is $\frac{1}{(k+1)!}$, thus the integral is 1. The case when $k = n$ is essentially the same as $k = n-1$.

This means that these Whitney forms are forming a basis of their respective form spaces. In a way, these bases are an extension of the Finite Element bases defined on nodes, or of the Finite Volume elements that are constant per tet.

Note finally that the Whitney forms are not continuous; however, they *are* continuous *along the direction of the k-simplex* (*i.e.*, tangential continuity for 1-forms, and normal continuity for 2-forms); this is the only condition needed to make the integration well defined. In a way, this property is the *least* we can ask them to be. We would lose generality if we were to add any other condition! The interested reader is referred to [Bossavit 1998] for a more thorough discussion on these Whitney bases and their relations to the notion of weak form used in the Finite Element Method.

7 Application to Hodge Decomposition

We now go through a first application of the discrete exterior calculus we have defined up to now. As we will see, the discrete case is often much simpler than its continuous counterpart; yet it captures the same properties.

7.1 Introducing the Hodge Decomposition

It is convenient in some applications to use the Helmholtz-Hodge decomposition theorem to decompose a given continuous vector field or differential form (defined on a smooth manifold $\mathcal{M}$) into components that are mutually orthogonal (in the $\mathcal{L}^2$ sense), and easier to compute (see [Abraham et al. 1988] for details). In fluid mechanics for example, the velocity field is generally decomposed into a part that is the gradient of a potential function and a part that is the curl of a stream vector potential (see Section 8.3 for further details), as the latter one is the incompressible part of the flow. When applied to k-forms, this decomposition is known as the *Hodge decomposition for forms* and can be stated as follows:

> *Given a manifold $\mathcal{M}$ and a k-form ω^k on $\mathcal{M}$ with appropriate boundary conditions, ω^k can be decomposed into the sum of the exterior derivative of a (k-1)-form α^{k-1}, the codifferential of a (k+1)-form β^{k+1}, and a harmonic k-form h^k:*
>
> $$\omega^k = d\alpha^{k-1} + \delta\beta^{k+1} + h^k.$$

Here, we use the term *harmonic* to mean that h^k satisfies the equation $\Delta h^k = 0$, where Δ is the Laplacian operator defined as $\Delta = d\delta + \delta d$. The proof of this theorem is mathematically involved and requires the use of elliptic operator theory and similar tools, as well as a careful study of the boundary conditions to en-

sure uniqueness. The discrete analog that we propose has a very simple and straightforward proof as shown below.

7.2 Discrete Hodge Decomposition

In the discrete setting, the discrete operators such as the exterior derivative and the codifferential can be expressed using matrix representation. This allows one to easily manipulate these operators using tools from linear algebra. In particular, the discrete version of the Hodge decomposition theorem becomes a simple exercise in linear algebra. Note that we will assume a boundaryless domain for simplicity (the generalization to domains with boundary is conceptually as simple).

Theorem 7.1 *Let $\mathcal{K}$ be a discrete manifold and let $\Omega^k(\mathcal{K})$ be the space of discrete Whitney k-forms on $\mathcal{K}$. Consider the linear operator $d^k : \Omega^k(\mathcal{K}) \to \Omega^{k+1}(\mathcal{K})$, such that $d^{k+1} \circ d^k = 0$, and a discrete Hodge star which is represented as a symmetric, positive definite matrix. Furthermore, define the codifferential (the adjoint of the operator d) as done in Section 5.5; namely, let $\delta^{k+1} = (-1)^{n(k-1)+1}(\star^k)^{-1}(d^k)^t\star^{k+1}$. In this case, the following orthogonal decomposition holds for all k:*

$$\Omega^k(\mathcal{K}) = d\Omega^{k-1}(\mathcal{K}) \oplus \delta\Omega^{k+1}(\mathcal{K}) \oplus \mathcal{H}^k(\mathcal{K})$$

where $\oplus$ means orthogonal sum, and $\mathcal{H}^k(\mathcal{K})$ is the space of harmonic k-forms on $\mathcal{K}$, that is, $\mathcal{H}^k(\mathcal{K}) = \{h \mid \Delta^k h = 0\}$.

Proof For notational convenience, we will omit the superscript of the operators when the rank is obvious. We first prove that the three component spaces are orthogonal. Clearly, using the facts that the Laplacian operator Δ is equal to $d\delta$+δd and that d and δ are adjoint operators, one has that $\forall h \in \mathcal{H}^k$:

$$\begin{aligned}\langle \Delta h, h\rangle = 0 \;&\Rightarrow\; \langle d\delta h, h\rangle + \langle \delta dh, h\rangle = \langle dh, dh\rangle + \langle \delta h, \delta h\rangle = 0\\ &\Rightarrow\; dh = 0 \text{ and } \delta h = 0\end{aligned}$$

Also, for all $\alpha \in \Omega^{k-1}(\mathcal{K})$ and $\beta \in \Omega^{k+1}(\mathcal{K})$, one has:

$$\langle d\alpha, \delta\beta\rangle = \langle dd\alpha, \beta\rangle = 0$$

and

$$\langle d\alpha, h\rangle = \langle \alpha, \delta h\rangle = 0 \qquad \langle h, \delta\beta\rangle = \langle dh, \beta\rangle = 0$$

Now, any k-form that is perpendicular to $d\Omega^{k-1}(\mathcal{K})$ and $\delta\Omega^{k+1}(\mathcal{K})$ must be in $\mathcal{H}^k(\mathcal{K})$, because this means $dh = 0$ and $\delta h = 0$, so $\Delta h = d\delta h + \delta dh = 0$.

Alternatively, we can prove that:

$$\Omega^k(\mathcal{K}) = \Delta\Omega^k(\mathcal{K}) \oplus \mathcal{H}^k(\mathcal{K}).$$

By analogy to the previous argument, it is easy to show that $\Delta\Omega^k$ is orthogonal to $\mathcal{H}^k$. Additionally, the dimension of these two spaces sum up to the dimension of Ω^k, which means the decomposition is complete. ∎

Note that the reader can find a similar proof given in Appendix B of [Frankel 2004], where it is used for Kirchhoff's Circuit Laws. There, Frankel does not mention that we can actually use cochains as the discretization of forms, and his operations using a "metric" of cochains can be interpreted as a Hodge star.

Implementation of the Discrete Hodge Decomposition Before we discuss how to numerically implement the discrete Hodge decomposition, we prove a useful result (that has a continuous analog).

Lemma 7.2 *In the discrete setting, one can find exactly one harmonic cochain from each cohomology equivalence class.*

Proof It is can be readily shown that the bases of harmonic cochains and the cohomology groups both have the dimension equal to $dim(\mathrm{Ker}\ d^k) - dim(\mathrm{Im}\ d^{k-1})$. To this end, recall that a cohomology basis is defined as is $\mathrm{Ker}(d^k)/\mathrm{Im}(d^{k-1})$ and has dimension $dim(\mathrm{Ker}\ d^k) - dim(\mathrm{Im}\ d^{k-1})$. Now, in order to see that the space of harmonic cochains has this same dimension, simply note that: $\mathrm{Ker}(d^k) = d\Omega^{k-1} \oplus \mathcal{H}^k$.

Now, the equation $\delta(\omega + df) = 0$ has a solution for each ω in one cohomology equivalence class. We know that the cochains forming different cohomology groups are linearly independent, hence, we conclude that these harmonic cochains span $\mathcal{H}^k$. ∎

By virtue of the above lemma, the implementation of the Hodge decomposition is simply recursive in the rank of the form (*i.e.*, cochain). The case of 0-forms is trivial: fix one vertex to a constant, and solve the Poisson equation for 0-forms. Now suppose that we have a decomposition working for $(k-1)$-forms, and we look for the decomposition of k-forms. Our approach is to get the harmonic component h^k first, so that we only need to solve a Poisson equation for the rest:

$$\Delta\omega^k = f^k - h^k \tag{12}$$

One is left with the problem of finding a basis of harmonic forms. Since we are given a Hodge star operator, we will use it to define the metric on the space of cochains. This metric allows us to define a basis for harmonic k-form (the dimension of this harmonic space is generally small, since it is the k-th Betti number β_k). First, one needs to calculate the cohomology basis $\{P_i\}$ based on the algorithm in Section 4.4.4. Once we have $\{P_i\}$, we solve one special decomposition of (k-1)-forms by first computing the forms f_i satisfying:

$$\Delta f_i = -\delta P_i \tag{13}$$

Now $H^k = P_i + df_i$ gives us the forms in basis for harmonic k-form space. After normalization, we have the basis to calculate the projection $h^k = HH^t f^k$, where we assemble all H^k into a matrix H. This completes the procedure of calculating the decomposition.

A non-singular matrix is often preferable when it comes to solve a linear system efficiently; we can change the Laplacian matrix slightly to make the Poisson equation satisfy this requirement. First, we can get an orthonormal basis for harmonic form space (the dimension is β^k). Now for basis e^j (column vector with j-th element equal to 1, and 0 everywhere else), take the distance of e^j to the harmonic space $|e^j - HH^t e^j|$; notice that this can be done in constant time. Now take out the j-th column and j-th row of Δ if e^j has the smallest distance from harmonic space, and repeat the step for β^k times. We are left with a non-singular matrix, and the solution to the new linear system is a solution to the original Poisson equation.

8 Others Applications

8.1 Form-based Proof of Tutte's Theorem

The notion of forms as convenient, intrinsic substitutes for vector fields has been used to provide a concise proof of the celebrated *Tutte's Embedding Theorem*. This important result in graph theory

states that if one fixes the boundary of a 3-connected graph (*i.e.*, a typical polygonal mesh) to a convex domain in the plane and ensures that every non-boundary vertex is a *strict convex combination* of its neighbors, then one obtains a planar straight-line embedding of the graph. In other words, this embedding procedure will not result in fold-overs. A significantly shorter alternative to the original proof of this theorem was proposed by Gortler, Gotsman, and Thurston [Gortler et al. 2006], using discrete 1-forms on edges. We now present a sketch of their approach, using a formulation more in line with the terms we used in this paper.

A Tutte embedding assigns to each vertex v_i of a graph G some 2D coordinates $\mathbf{X}(v_i) = (x(v_i), y(v_i))$. By definition, each interior vertex v_i satisfies a linear condition on its coordinates of the form: $\mathbf{X}(v_i) = \sum_{v_j \in \mathcal{N}(i)} w_{ij} \mathbf{X}(v_j)$, where $\mathcal{N}(i)$ is the set of 1-ring neighbors of vertex v_i. These coefficients w_{ij} are all non-negative due to the condition of strict convex combination mentioned above. Now, for a given Tutte embedding, one can construct a 0-form $z(v) = \alpha x(v) + \beta y(v)$ for any pair of positive coefficients α and β. Notice that this 0-form satisfies the same convex combination condition: $z(v_i) = \sum_{v_j \in \mathcal{N}(i)} w_{ij} z(v_j)$. As they are non negative, one can identify these coefficients w_{ij} with the diagonal Hodge star of primal 1-forms (see Section 7) defined by a particular metric. Therefore, the relationship $0 = \sum_{v_j \in \mathcal{N}(i)} w_{ij}(z(v_j) - z(v_i))$ is equivalent to: $d \star dz = 0$. There are two immediate conclusions:

- ◇ the 1-form $\omega = dz$ is closed (since it is the exterior derivative of a 0-form), and
- ◇ it is also co-closed since $\delta\omega = (\star d \star) dz = \star(d \star dz) = 0$.

To use the previously defined 1-form ω to prove Tutte's theorem, Gortler *et al.* then invoke the usual definition of index of vector fields, *i.e.*, the number of revolutions that the direction of the vector fields does along any small curve around this vertex. This concept is one of the oldest in Algebraic Topology, initially stated by Poincaré and then developed by Hopf and Morse in the continuous case. Its discrete counterpart was first proposed by Banchoff, and used for instance in [Lazarus and Verroust 1999]. A discrete Poincare-Hopf index theorem also holds, stating that the sum of all indices must be equal to 2 for a genus-0 patch. The final argument uses the link between (co)closed forms and their indices. Indeed, because we found a closed *and* coclosed form ω, it can be easily shown that these two properties induce that the index of each face must be less or equal to zero, as well as the index of each vertex. Because the boundary of the patch is convex, only two vertices on the boundary have index 1. Since all the indices must sum to 2 and each interior index must be less than zero, we can conclude that *each interior index is zero*. Because this argument is valid for every positive pair (α, β), one can easily deduce that each interior face is convex and each vertex is a "wheel"; thus, injectivity can be guaranteed.

This rather elegant proof demonstrates how discrete forms and their obvious links to Algebraic Topology can be quite powerful in a variety of applications. We also point the interested reader to other papers, such as [Mercat 2001; Gu and Yau 2003], for which special discrete Hodge stars are defined to satisfy a discrete definition of conformality: there are also very interesting research on this particular topic, once again using the calculus of exterior forms.

8.2 Electromagnetism with Forms

Electromagnetism can be formulated very elegantly using differential forms. For a detailed exposition of the geometric structure in E&M, we refer the reader to [Bossavit 1998] and [Warnick et al. 1997]. In this approach, the electric field E is represented by a 1-form as the integral of E along a path traced by a test charge q, and is equal to the electromotive force experienced by that charge. The electric displacement L as well as the current density J are represented by 2-forms. The charge distribution ρ is a 3-form. The magnetic field B is represented by a 2-form since it is measured as a flux. whereas the magnetic field intensity H is a 1-form.

With these conventions, Maxwell's equations can be rewritten as follows:

$$\partial_t B + dE = 0, \qquad -\partial_t L + dH = J, \qquad dL = \rho, \tag{14}$$

subject to the *constitutive* equations:

$$L = \epsilon E, \qquad H = \mu B, \tag{15}$$

where ϵ is the permittivity, and μ is the permeability. The constitutive relations (15) are very similar to the Hodge star operator that transforms a k-form to an $(n\text{-}k)$-form. Here, ϵ operates on the electric field E (1-form) to yield the electric displacement L (2-form) while μ transforms the magnetic field B (2-form) into the magnetic field intensity H (1-form). To this end, one may think of both ϵ and μ as Hodge star operators induced from appropriately chosen metrics. Note that the balance laws in (14) are metric-independent.

As the reader can guess, one can readily discretize this representation of the physical quantities $E, L, \ldots$ and the associated system of equations (14-15) using the tools presented in this chapter. The resulting numerical algorithms preserve *exactly* the geometric structure of the system [Bossavit 1998], even in the presence of charge and/or for irregular meshes [Stern et al. 2008].

8.3 Fluids

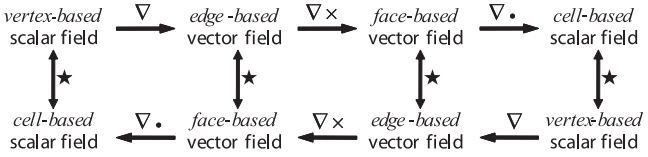
The geometric structure of Fluid Mechanics, specifically Euler's equations for inviscid fluids, has been investigated (see [Marsden and Weinstein 1983] and references therein). In this geometric framework, vorticity is represented as a two-form (an area-form) and Euler's equations can be written as vorticity advection. Roughly speaking, vorticity measures the rotation of a fluid parcel; we say the fluid parcel has vorticity when it spins as it moves along its path. Vorticity advection means that the vorticity (as a two-form) moves dynamically as if it is pushed forward by the fluid flow. The integral of the vorticity on a given bounded domain is equal, by Stokes' theorem, to the circulation around the loop enclosing the domain. This quantity, as the loop is advected by the fluid, is conserved in the absence of external forcing, as well as the total energy of the fluid. Inspired by this geometric viewpoint and in light of the present development of Discrete Exterior Calculus, one can develop a discrete differential approach to fluid mechanics and an integration scheme that satisfy conservation of circulation, see [Elcott et al. 2007] for further details.

8.4 Developments in Geometry Processing

Discrete forms, with their geometric nature we described up to now, lend themselves naturally to geometric applications. From the design of barycentric coordinates over arbitrary polytopes (see, e.g., [Warren et al. 2007]) usable as dual form basis, to the design of smooth, higher-order Whitney forms [Wang et al. 2006], discrete forms have been shown particularly relevant in several geometry processing tasks. Recent applications include point set reconstruction [Alliez et al. 2007], Eulerian treatment of interfaces [Mullen et al. 2007], as well as conformal parameterization and its use for quadrangle meshing [Tong et al. 2006]—note the recent non-linear version that offers a complete notion of conformal equivalence for triangle meshes [Springborn et al. 2008], including a discrete metric.

9 Conclusions

In this chapter, we have provided an introduction to discrete differential forms and explained how they can be extremely useful in computational science. A convenient Discrete Exterior Calculus solely based on values stored on a discrete manifold has been derived. In the common 3D case, this calculus for scalar and vector fields can be summarized by the following schematic graph:



We have also given a discrete version of the Hodge decomposition, useful for a number of computations in various fields. Despite numerous recent developments this geometric approach to computations is still nascent, and many details need to be explored and proven superior to current approaches. In order to work towards this goal, more work needs to be done to further demonstrate that this idea of forms as fundamental readily-discretizable elements of differential equations can be successfully used in various other contexts where predictive power is crucial.

Acknowledgments

The authors wish to first thank Jerrold E. Marsden for his tremendous support along the way. Peter Schröder, Herbert Edelsbrunner, and John M. Sullivan helped us with a thorough proofreading of this document. Bugs were also reported by Henry Adams and F. Fan. We finally wish to acknowledge the support of Alain Bossavit, Anil Hirani, Melvin Leok, David Cohen-Steiner, Sharif Elcott, Pierre Alliez, and Eitan Grinspun.

References

ABRAHAM, R., MARSDEN, J., AND RATIU, T., Eds. 1988. *Manifolds, Tensor Analysis, and Applications*. Applied Mathematical Sciences Vol. 75, Springer.

ALLIEZ, P., COHEN-STEINER, D., TONG, Y., AND DESBRUN, M. 2007. Voronoi-based variational reconstruction of unoriented point sets. In *Symposium on Geometry Processing*, 39–48.

BJÖRNER, A., AND WELKER, V. 1995. The homology of "k-equal" manifolds and related partition lattices. *Advances in Math. 110*, 277–313.

BOBENKO, A., AND SEILER, R., Eds. 1999. *Discrete Integrable Geometry and Physics*. Clarendon Press.

BOCHEV, P. B., AND HYMAN, J. M. 2005. Principles of mimetic discretizations of differential operators. Preprint: LA-UR-05-4244.

BOSSAVIT, A. 1998. *Computational Electromagnetism*. Academic Press, Boston.

BURKE, W. L. 1985. *Applied Differential Geometry*. Cambridge University Press.

CARROLL, S. 2003. *Spacetime and Geometry: An Introduction to General Relativity*. Pearson Education.

CARTAN, É. 1945. *Les Systèmes Differentiels Exterieurs et leurs Applications Géometriques*. Hermann, Paris.

DESBRUN, M., LEOK, M., AND MARSDEN, J. E. 2004. Discrete Poincaré Lemma . *Appl. Num. Math.*.

DIMAKIS, A., AND MÜLLER-HOISSEN, F. 1994. Discrete Differential Calculus, Graphs, Topologies, and Gauge Theory. *Journal of Mathematical Physics 35*, 6703–6735.

DORAN, C., AND LASENBY, A., Eds. 2003. *Geometric Algebra for Physicists*. Cambridge University Press.

EDELSBRUNNER, H., LETSCHER, D., AND ZOMORODIAN, A. 2000. Topological persistence and simplification. In *IEEE Symposium on Foundations of Computer Science*, 454–463.

ELCOTT, S., TONG, Y., KANSO, E., DESBRUN, M., AND SCHRÖDER, P. 2007. Discrete, Circulation-preserving and Stable Simplicial Fluid. *ACM Trans. on Graphics 26(1)* (Jan.).

FISHER, M., SCHRÖDER, P., DESBRUN, M., AND HOPPE, H. 2007. Design of tangent vector fields. *ACM Transactions on Graphics 26*, 3 (July), 56:1–56:9.

FLANDERS, H. 1990. *Differential Forms and Applications to Physical Sciences*. Dover Publications.

FLANDERS, H., Ed. 2001. *Geometric Methods for Computational Electromagnetics*. EMW Publishing, Cambridge Mass.

FORMAN, R. 2003. Bochner's Method for Cell Complexes and Combinatorial Ricci Curvature. *J. of Discrete and Computational Geom. 29*, 3, 323–374.

FRANKEL, T. 2004. *The Geometry of Physics*. Second Edition. Cambridge University Press, United Kingdom.

GORTLER, S., GOTSMAN, C., AND THURSTON, D. 2006. One-Forms on Meshes and Applications to 3D Mesh Parameterization. *Computer Aided Geometric Design 23*, 2, 83–112.

GROSS, P. W., AND KOTIUGA, R. 2004. *Electromagnetic Theory and Computation: A Topological Approach*. Cambridge University Press.

GU, X., AND YAU, S.-T. 2003. Global conformal surface parameterization. In *Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, Eurographics Association, 127–137.

HARRISON, J. 2005. Ravello Lecture Notes on Geometric Calculus – Part I. Tech. rep., UC Berkeley.

HATCHER, A. 2004. *Algebraic Topology*. Cambridge University Press.

HILDEBRANDT, K., AND POLTHIER, K. 2004. Anisotropic filtering of non-linear surface features. In *Computer Graphics Forum*, M.-P. Cani and M. Slater, Eds., vol. 23. Proc. Eurographics 2004.

HIRANI, A. N. 2003. *Discrete Exterior Calculus*. PhD thesis, Caltech.

HYMAN, J. M., AND SHASHKOV, M. 1997. Natural Discretizations for the Divergence, Gradient, and Curl. *International Journal of Computers and Mathematics with Applications 33*.

KANSO, E., ARROYO, M., TONG, Y., YAVARI, A., MARSDEN, J. E., AND DESBRUN, M. 2007. On the Geometric Character of Continuum Mechanics. *Z. angew. Math Phys. 58*.

LAZARUS, F., AND VERROUST, A. 1999. Level Set Diagrams of Polyhedral Objects. In *Proceedings of the 5^{th} ACM Symposium on Solid Modeling and Applications*, 130–140.

LOVELOCK, D., AND RUND, H. 1993. *Tensors, Differential Forms, and Variational Principles*. Dover Publications.

MARSDEN, J. E., AND WEINSTEIN, A. 1983. Coadjoint orbits, vortices and Clebsch variables for incompressible fluids. *Physica D 7*, 305–323.

MARSDEN, J. E., AND WEST, M. 2001. Discrete Mechanics and Variational Integrators. *Acta Numerica 10*, 357–514.

MERCAT, C. 2001. Discrete Riemann Surfaces and the Ising Model. *Commun. Math. Phys. 218*, 1, 177–216.

MEYER, M., DESBRUN, M., SCHRÖDER, P., AND BARR, A. H. 2002. Discrete Differential-Geometry Operators for Triangulated 2-Manifolds. In *Proceedings of VisMath*.

MORITA, S. 2001. *Geometry of Differential Forms*. Translations of Mathematical Monographs, Vol. 201. Am. Math. Soc.

MULLEN, P., MCKENZIE, A., TONG, Y., AND DESBRUN, M. 2007. A variational approach to eulerian geometry processing. *ACM Trans. on Graphics (SIGGRAPH)* (July), 66.

MUNKRES, J. R. 1984. *Elements of Algebraic Topology*. Addison-Wesley, Menlo Park, CA.

PINKALL, U., AND POLTHIER, K. 1993. Computing Discrete Minimal Surfaces. *Experimental Mathematics 2*, 1, 15–36.

RAMASWAMY, V., AND SHAPIRO, V. 2004. Combinatorial Laws for Physically Meaningful Design. *J. of Computing and Information Science in Engineering 4*, 1, 3–10.

SCHREIBER, U. 2003. On Superstrings in Schrödinger Representation. *Preprint*.

SHARPE, R. W. 1997. *Differential Geometry:Cartan's Generalization of Klein's Erlangen Programm*. Springer-Verlag, NY.

SPRINGBORN, B., SCHRÖDER, P., AND PINKALL, U. 2008. Conformal equivalence of triangle meshes. *ACM Trans. Graph. 27*, 3, 1–11.

STERN, A., TONG, Y., DESBRUN, M., AND MARSDEN, J. E. 2008. Variational integrators for maxwell's equations with sources. *PIERS 4*, 7, 711–715.

TONG, Y., ALLIEZ, P., COHEN-STEINER, D., AND DESBRUN, M. 2006. Designing quadrangulations with discrete harmonic forms. In *ACM/EG Symposium on Geometry Processing*, 201–210.

WANG, K., WEIWEI, TONG, Y., DESBRUN, M., AND SCHRÖDER, P. 2006. Edge subdivision schemes and the construction of smooth vector fields. *ACM Trans. on Graphics (SIGGRAPH) 25*, 3 (July), 1041–1048.

WARNICK, K. F., SELFRIDGE, R. H., AND ARNOLD, D. V. 1997. Teaching Electromagnetic Field Theory Using Differential Forms. *IEEE Trans. on Education 40*, 1, 53–68.

WARREN, J., SCHAEFER, S., HIRANI, A., AND DESBRUN, M. 2007. Barycentric coordinates for convex sets. *Advances in Computational Mathematics 27*, 3, 319–338.

WHITNEY, H. 1957. *Geometric Integration Theory*. Princeton Press, Princeton.

ZAPATRIN, R. 1996. Polyhedral Representations of Discrete Differential Manifolds. *Preprint*.

Further Reading

Despite a large number of theoretical books, we are aware of only a few books with a truly "applied flavor" in line with this chapter. For applications based on this exterior calculus or other geometric algebras, see [Bossavit 1998; Flanders 2001; Bobenko and Seiler 1999; Doran and Lasenby 2003; Gross and Kotiuga 2004; Ramaswamy and Shapiro 2004]. The reader interested in the application of differential forms to E&M is further referred to [Warnick et al. 1997], for applications in fluid mechanics see [Marsden and Weinstein 1983], and in elasticity see [Kanso et al. 2007] and [Frankel 2004]. The reader is also invited to check out current developments of variants of DEC, for instance, in [Dimakis and Müller-Hoissen 1994; Schreiber 2003; Zapatrin 1996; Harrison 2005], as well as the nice, thorough review from Bochev and Hyman [Bochev and Hyman 2005].

Finally, the interested reader can find additional material on the following websites:

Graphics and Applied Geometry at Caltech:
http://multires.caltech.edu/pubs/
http://www.geometry.caltech.edu/
Computational E&M (Alain Bossavit):
http://www.lgep.supelec.fr/mse/perso/ab/bossavit.html
Discrete Vector Fields and Combinatorial Topology (R. Forman):
http://math.rice.edu/∼forman/
Discrete Mechanics at Caltech (Jerrold E. Marsden):
http://www.cds.caltech.edu/∼marsden/

Chapter 8: Building Your Own DEC at Home

Sharif Elcott
Caltech

Peter Schröder
Caltech

1 Overview

The methods of Discrete Exterior Calculus (DEC) have given birth to many new algorithms applicable to areas such as fluid simulation, deformable body simulation, and others. Despite the (possibly intimidating) mathematical theory that went into deriving these algorithms, in the end they lead to simple, elegant, and straightforward implementations. However, readers interested in implementing them should note that the algorithms presume the existence of a suitable simplicial complex data structure. Such a data structure needs to support local traversal of elements, adjacency information for all dimensions of simplices, a notion of a *dual mesh*, and all simplices must be *oriented*. Unfortunately, most publicly available tetrahedral mesh libraries provide only *unoriented* representations with little more than vertex-tet adjacency information (while we need vertex-edge, edge-triangle, edge-tet, *etc.*). For those eager to implement and build on the algorithms presented in this course without having to worry about these details, we provide an implementation of a DEC-friendly tetrahedral mesh data structure in C++. This chapter documents the ideas behind the implementation.

1.1 Motivation

Extending a classic pointer-based mesh data structure to 3D is unwieldy, error-prone, and difficult to debug. We instead take a more abstract set-oriented view in the design of our data structure, by turning to the formal definition of an abstract simplicial complex. This gives our implementation the following desirable properties:

- We treat the mesh as a graph and perform all of our operations combinatorially.
- There is no cumbersome pointer-hopping typical of most mesh data structures.
- The design easily generalizes to arbitrary dimension.
- The final result is very compact and simple to implement.

In effect we are taking advantage of the fact that during assembly of all the necessary structures one can use high level, abstract data structures. That way formal definitions can be turned into code almost verbatim. While these data structures (*e.g.*, sets and maps) may not be the most efficient for *computation*, an approach which uses them during assembly is far less error prone. Once everything has been assembled it can be turned easily into more efficient packed representations (*e.g.*, compressed row storage format sparse matrices) with their more favorable performance during the actual computations which occur, *e.g.*, in physical simulation.

1.2 Outline

We will begin with a few definitions in Section 2, and see how these translate into our tuple-based representation in Section 3. The boundary operator, described in Section 4, facilitates mesh traversal and implements the discrete exterior derivative. We show how everything is put together in Section 5. Finally, we discuss our implementation of the DEC operators in Section 6.

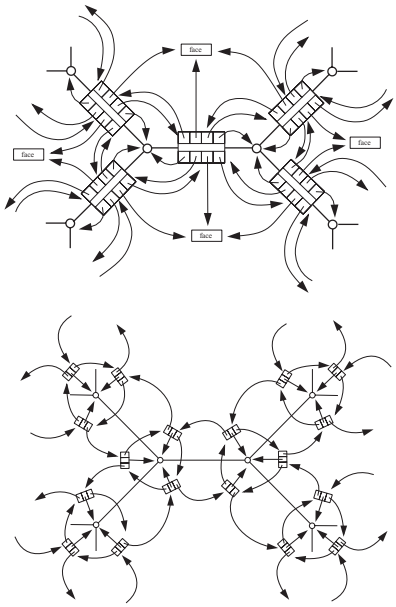


Figure 1: *Some typical examples of 2D mesh representations (from [Joy et al. 2002]; used with permission). Such pointer-based data structures become quite difficult to manage once they are extended to 3D.*

2 Definitions

We begin by recalling the basic definitions of the objects we are dealing with. The focus here is on the rigorous mathematical definitions in a form which then readily translates into high level algorithms. The underlying concepts are simply what we all know informally as *meshes* in either two (triangle) or three (tet) dimensions.

Simplices A *simplex* is a general term for an element of the mesh, identified by its dimension. 0-simplices are vertices, 1-simplices are edges, 2-simplices are triangles, and 3-simplices are tetrahedra.

Abstract Simplicial Complex This structure encodes all the relationships between vertices, edges, triangles, and tets. Since we are only dealing with combinatorics here the atomic element out of which everything is built are the integers $0 \leq i < n$ referencing the underlying vertices. For now they do not yet have point positions in space. Formally, an abstract simplicial complex is a *set of subsets* of the integers $0 \leq i < n$, such that if a subset is contained in the complex then so are all its subsets. For example, a 3D complex is a collection of tetrahedra (4-tuples), triangles (3-tuples), edges (2-tuples), and vertices (singletons), such that if a tetrahedron is present in the complex then so must be its triangles, edges, and

vertices. All our simplicial complexes will be proper three or two manifolds, possibly with boundary and may be of arbitrary topology (*e.g.*, containing voids and tunnels).

Manifold The DEC operators that we build on are defined only on meshes which represent manifolds. Practically speaking this means that in a 3D simplicial complex all triangles must have two incident tets only (for a boundary triangle there is only one incident tet). Every edge must have a set of tets incident on it which form a single "ring" which is either open (at the boundary) or closed (in the interior). Finally for vertices it must be true that all incident tets form a topological sphere (or hemisphere at the boundary). These properties should be asserted upon reading the input. For example, for triangles which bound tets one must assert that each such triangle occurs in at most two tets. For an edge the "ring" property of incident tets can be checked as follows. Start with one incident tet and jump across a shared triangle to the next tet incident on the edge. If this walk leads back to the original tet *and* all tets incident on the edge can thusly be visited, the edge passes the test. (For boundary edges such a walk starts at one boundary tet and ends at another.) The test for vertices is more complex. Consider all tets incident on the given vertex. Using the tet/tet adjacency across shared triangles one can build the adjacency graph of all such tets. This graph must be a topological sphere (or hemisphere if the vertex is on the boundary).

Since we need everything to be properly oriented we will only allow *orientable* manifolds (*i.e.*, no Möbius strips or Klein bottles).

Regularity To make life easier on ourselves we also require the simplicial complex to be *strongly regular*. This means that simplices must not have identifications on their boundaries. For example, edges are not allowed to begin and end in the same vertex. Similarly, the edges bounding a triangle must not be identified nor do we allow edges or triangles bounding a tet to be identified. In practice this is rarely an issue since the underlying geometry would need to be quite contorted for this to occur. Strictly speaking though such identifications are possible in more general, abstract settings without violating the manifold property.

Embedding It is often useful to distinguish between the *topology* (neighbor relationships) and the *geometry* (point positions) of the mesh. A great deal of the operations performed on our mesh can be carried out using only topological information, *i.e.*, without regard to the embedding. The embedding of the complex is given by a map $p : [0,n) \mapsto (x,y,z) \in \mathbb{R}^3$ on the vertices (which is extended piecewise linearly to the interior of all simplices). For example, when we visualize a mesh as being composed of piecewise linear triangles (for 2D meshes) or piecewise linear tets, we are dealing with the geometry. Most of the algorithms we describe below do not need to make reference to this embedding. When implementing these algorithms it is useful to only think in terms of combinatorics. There is only one stage where we care about the geometry: the computation of metric dependent quantities needed in the definition of the Hodge star.

3 Simplex Representation

Ignoring orientations for a moment, each k-simplex is represented as a $(k+1)$-tuple identifying the vertices that bound the simplex. In this view a tet is simply a 4-tuple of integers, a triangle is a 3-tuple of integers, an edge is a 2-tuple, and a vertex is a singleton. Note that all permutations of a given tuple refer to the same simplex. For example, (i,j,k) and (j,i,k) are different *aliases* for the same triangle. In order to remove ambiguities, we must designate one *representative* alias as the representation of the simplex in our data structures. We do this by using the *sorted* permutation of the tuple. Thus each simplex (tuple) is stored in our data structures as its canonical (sorted) representative. Then if we, for example, need to check whether two simplices are in fact the same we only need to compare their representatives element by element.

All this information is stored in lists we designate **V**, **E**, **F**, and **T**. They contain one representative for every vertex, edge, triangle, and tet, respectively, in the mesh.

3.1 Forms

The objects of computation in an algorithm using DEC are forms. Formally, a differential k-form is a quantity that can be integrated over a k dimensional domain. For example, consider the expression $\int f(x)dx$ (x being a scalar). The integrand $f(x)dx$ is called a 1-*form*, because it can be integrated over any 1-dimensional interval. Similarly, the dA in $\int\int dA$ would be a 2-form.

Discrete differential forms are dealt with by storing the results of the integrals themselves, instead of the integrands. That is, discrete k-forms associate one value with each k-simplex, representing the integral of the form over that simplex. With this representation we can recover the integral over any k-dimensional chain (the union of some number of k-simplices) by summing the value on each simplex (using the linearity of the integral).

Since all we have to do is to associate one value with each simplex, for our purposes forms are simply vectors of real numbers where the size of the vector is determined by the number of simplices of the appropriate dimension. 0-forms are vectors of size $|\mathbf{V}|$, 1-forms are vectors of size $|\mathbf{E}|$, 2-forms are vectors of size $|\mathbf{F}|$, and 3-forms are vectors of size $|\mathbf{T}|$. Such a vector representation requires that we assign an index to each simplex. We use the position of a simplex in its respective list (**V**, **E**, **F**, or **T**) as its index into the form vectors.

3.2 Orientation

Because the vectors of values we store represent integrals of the associated k-form over the underlying simplices, we must keep track of orientation. For example, reversing the bounds of integration on $\int_a^b f(x)dx$ flips the sign of the resulting value. To manage this we need an *intrinsic orientation* for each simplex. It is with respect to this orientation that the values stored in the form vectors receive the appropriate sign. For example, suppose we have a 1-form f with value f_{ij} assigned to edge $e = (i,j)$; that is, the real number f_{ij} is the integral of the 1-form f over the line segment (p_i, p_j). If we query the value of this form on the edge (j,i) we should get $-f_{ij}$.

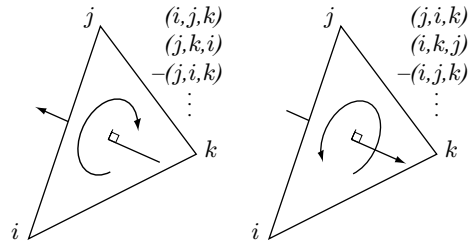


Figure 2: *All permutations of a triple (i,j,k) refer to the same triangle, and the sign of the permutation determines the orientation.*

Hence every tuple must be given a sign indicating whether it agrees $(+)$ or disagrees $(-)$ with the intrinsic orientation of the simplex. Given a set of integers representing a simplex, there are two equivalence classes of orderings of the given tuple: the even and odd permutations of the integers in question. These two equivalence classes correspond to the two possible orientations of the simplex (see Fig. 2).

Note that assigning a sign to any one alias (*i.e.*, the representative) implicitly assigns a sign to all other aliases. Let us assume for a moment that the sign of all representatives is known. Then the sign S of an arbitrary tuple t, with representative r, is

$$\mathbf{S}(t) = \begin{cases} \mathbf{S}(r) & \text{if } t \text{ is in the same equivalence class as } r \\ -\mathbf{S}(r) & \text{if } t \text{ is in the opposite equivalence class.} \end{cases}$$

More formally, let P be the permutation that permutes t into r (*i.e.*, $r = P(t)$). Then

$$\mathbf{S}(t) = \mathbf{S}(P)\mathbf{S}(P(t)).$$

(Here $\mathbf{S}(P)$ denotes the sign of the permutation P with $+1$ for even and -1 for odd permutations.)

All that remains, then, is to choose an intrinsic orientation for each simplex and set the sign of the representative alias accordingly. In general the assignment of orientations is arbitrary, as long as it is consistent. For all subsimplices we choose the representative to be positively oriented, so that the right-hand-side of the above expression reduces to $\mathbf{S}(P)$. For top-level simplices (tets in 3D, triangles in 2D), we use the convention that a positive volume corresponds to a positively oriented simplex. We therefore require a volume form which, together with an assignment of points to vertices, will allow us to orient all tets. Recall that a volume form accepts three (for 3D; two for 2D) vectors and returns either a positive or negative number (assuming the vectors are linearly independent). So the sign of a 4-tuple is:

$$\mathbf{S}(i_0,i_1,i_2,i_3) = \mathbf{S}(\mathrm{Vol}(p_{i_1} - p_{i_0}, p_{i_2} - p_{i_0}, p_{i_3} - p_{i_0})).$$

4 The Boundary Operator

The *faces* of a k-simplex are the $(k-1)$-simplices that are incident on it, *i.e.*, the subset of one lower dimension. Every k-simplex has $k+1$ faces. Each face corresponds to removing one integer from the tuple, and the relative orientation of the face is $(-1)^i$ where i is the index of the integer that was removed. To clarify:

- The faces of a tet $+(t_0,t_1,t_2,t_3)$ are $-(t_0,t_1,t_2)$, $+(t_0,t_1,t_3)$, $-(t_0,t_2,t_3)$, and $+(t_1,t_2,t_3)$.
- The faces of a triangle $+(f_0,f_1,f_2)$ are $+(f_0,f_1)$, $-(f_0,f_2)$, and $+(f_1,f_2)$.
- The faces of an edge $+(e_0,e_1)$ are $-(e_0)$ and $+(e_1)$.

We can now define the boundary operator ∂ which maps simplices to their their faces. Given the set of tets $\mathbf{T}$ we define $\partial^3 : \mathbf{T} \to \mathbf{F}^4$ as

$$\begin{aligned} \partial^3(+(i_0,i_1,i_2,i_3)) \quad = \quad & \{-(i_0,i_1,i_2),+(i_0,i_1,i_3), \\ & -(i_0,i_2,i_3),+(i_1,i_2,i_3)\}. \end{aligned}$$

Similarly for $\partial^2 : \mathbf{F} \to \mathbf{E}^3$ (which maps each triangle to its three edges) and $\partial^1 : \mathbf{E} \to \mathbf{V}^2$ (which maps each edge to its two vertices).

We represent these operators as sparse adjacency matrices (or, equivalently, signed adjacency lists), containing elements of type $+1$ and -1 only. So ∂^3 is implemented as a matrix of size $|\mathbf{F}| \times |\mathbf{T}|$ with 4 non-zero elements per column, ∂^2 an $|\mathbf{E}| \times |\mathbf{F}|$ matrix with 3 non-zero elements per column, and ∂^1 a $|\mathbf{V}| \times |\mathbf{E}|$ matrix with 2 non-zero elements per column (one $+1$ and one -1). The transposes of these matrices are known as the *coboundary* operators, and they map simplices to their *cofaces*—neighbor simplices of one higher dimension. For example, $(\partial^2)^T$ maps an edge to the "pinwheel" of triangles incident on that edge.

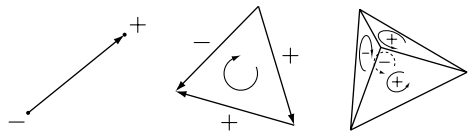


Figure 3: *The boundary operator identifies the faces of a simplex as well as their* relative *orientations. In this illustration, arrows indicate intrinsic orientations and signs indicate the relative orientation of a face to a parent.*

These matrices allow us to iterate over the faces or cofaces of any simplex, by walking down the columns or across the rows, respectively. In order to traverse neighbors that are more than one dimension removed (*i.e.*, the tets adjacent to an edge or the faces adjacent to a vertex) we simply concatenate the appropriate matrices, but without the signs. (If we kept the signs in the matrix multiplication any such consecutive product would simply return the zero matrix reflecting the fact that the boundary of a boundary is always empty.)

5 Construction

Although we still need a few auxiliary wrapper and iterator data structures to provide an interface to the mesh elements, the simplex lists and boundary matrices contain the entirety of the topological data of the mesh. All that remains, then, is to fill in this data.

We read in our mesh as a list of (x,y,z) vertex positions and a list of 4-tuples specifying the tets. Reading the mesh in this format eliminates the possibility of many non-manifold scenarios; for example, there cannot be an isolated edge that does not belong to a tet. We assume that all integers in the range $[0,n)$ appear at least once in the tet list (this eliminates isolated vertices), and no integer outside of this range is present.

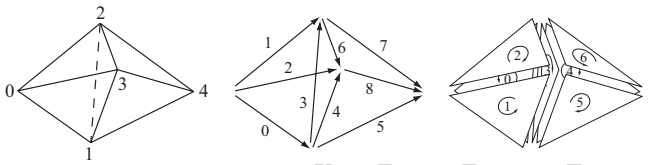
Once $\mathbf{T}$ is read in, building $\mathbf{E}$ and $\mathbf{F}$ is trivial; for each tuple in $\mathbf{T}$, append all subsets of size 2 and 3 to $\mathbf{E}$ and $\mathbf{F}$ respectively. We must be sure to avoid duplicates, either by using a unique associative container, or by sorting the list afterward and removing duplicates. Then the boundary operator matrices are constructed as follows:

```
for each simplex s
    construct a tuple for each face f of s
        as described in Section 4
    determine the index i of f by locating
        its representative
    set the entry of the appropriate matrix
        at row i, column s to S(f)
```

Figure 4 shows a complete example of a mesh and its associated data structure.

6 DEC Operators

Now we discuss the implementation of the two most commonly used DEC operators: the exterior derivative and the Hodge star. As we will see, in the end these also amount to nothing more than sparse matrices that can be applied to our form vectors.



$$\begin{array}{cccc} \mathbf{V} & \mathbf{E} & \mathbf{F} & \mathbf{T} \\ [0] & [0,1] & [0,1,2] & [0,1,2,3] \\ [1] & [0,2] & [0,1,3] & [1,2,3,4] \\ [2] & [0,3] & [0,2,3] & \\ [3] & [1,2] & [1,2,3] & \\ [4] & [1,3] & [1,2,4] & \\ & [1,4] & [1,3,4] & \\ & [2,3] & [2,3,4] & \\ & [2,4] & & \\ & [3,4] & & \end{array}$$

$$\partial^3 = \begin{pmatrix} -1 & \\ 1 & \\ -1 & \\ 1 & -1 \\ & 1 \\ & -1 \\ & 1 \end{pmatrix}$$

$$\partial^2 = \begin{pmatrix} 1 & 1 & & & & & \\ -1 & & 1 & & & & \\ & -1 & -1 & & & & \\ 1 & & & 1 & 1 & & \\ & 1 & & -1 & & 1 & \\ & & & & -1 & -1 & \\ & & 1 & 1 & & & 1 \\ & & & & 1 & & -1 \\ & & & & & 1 & 1 \end{pmatrix}$$

$$\partial^1 = \begin{pmatrix} -1 & -1 & -1 & & & & & & \\ 1 & & & -1 & -1 & -1 & & & \\ & 1 & & 1 & & & -1 & -1 & \\ & & 1 & & 1 & & 1 & & -1 \\ & & & & & 1 & & 1 & 1 \end{pmatrix}$$

Figure 4: *A simple mesh and all associated data structures.*

6.1 Exterior Derivative

As we have seen earlier in the course, the discrete exterior derivative is defined using Stokes' theorem such that

$$\int_\sigma d\omega = \int_{\partial\sigma} \omega$$

where ω is a k-form, and σ is a $(k+1)$-simplex. In words, this equation states that the evaluation of $d\omega$ on a simplex is equal to the evaluation of ω on the boundary of that simplex.

Let us try to understand this theorem with a few examples. Consider a 0-form f, *i.e.*, a function giving values at vertices. With that, df is a 1-form which can be integrated along an edge (say with endpoints denoted a and b) and Stokes' theorem states the well known fact

$$\int_{[a,b]} df = f(b) - f(a).$$

The right hand side is simply the evaluation of the 0-form f on the boundary of the edge (*i.e.*, its endpoints), with appropriate signs indicating the orientation of the edge.

What about triangles? If f is a 1-form (one value per edge), then df is a 2-form that can be evaluated on a triangle abc as

$$\begin{aligned} \int_{\Delta abc} df &= \int_{\partial(\Delta abc)} f \\ &= \int_{[a,b]} f + \int_{[b,c]} f + \int_{[c,a]} f \\ &= f_{ab} + f_{bc} + f_{ca} \end{aligned}$$

using the subscript notation from Section 3.2. Again, the right hand side is simply the evaluation of the 1-form f on the boundary of the triangle—its three edges.

We can restate the general form of the theorem for our discrete forms as

$$d\omega_\sigma = \sum_{s \in \partial\sigma} \omega_s$$

Written this way, it is easy to see that this can be implemented as the multiplication of a form vector by the coboundary matrix ∂^T.

6.2 The Dual Mesh and the Hodge Star

Every complex has a dual. The dual of a simplicial complex is a *cell complex* where primal k-simplices correspond to dual $(n-k)$-cells. So in our case there are $|\mathbf{V}|$ dual polyhedra, $|\mathbf{E}|$ dual polygons, $|\mathbf{F}|$ dual edges, and $|\mathbf{T}|$ dual vertices, corresponding to primal vertices, edges, triangles, and tetrahedra, respectively (see Fig. 5). Note that, since every dual cell is co-located with a primal simplex and the cardinality is the same, in the code there is no explicit representation for the dual mesh. Where appropriate, dual cells are queried through the corresponding primal simplex index.

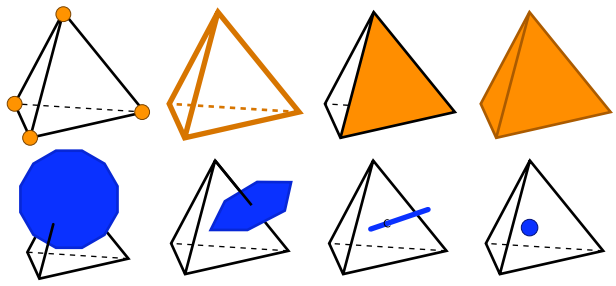


Figure 5: *There is one dual polyhedron for every primal vertex, one dual polygon for every primal edge, one dual edge for every primal triangle, and one dual vertex for every primal tetrahedron.*

The operator that transforms a primal k-form into a dual $(n-k)$-form is known as the *Hodge star*. There are many different kinds of Hodge stars, the simplest of which is the *diagonal Hodge star*.

We again attempt to motivate the definition with some intuition. When transferring a quantity from a primal simplex to a dual cell, the quantities must "agree" somehow. Since these are integral values, simply setting the value on the dual to be equal to the value on the primal does not make sense, as the domain of integration is unrelated. Instead, we require that the *integral density* be equal. So, if ω denotes the evaluation of a form on a primal k-simplex σ, then $\star\omega$ is the value on the dual $(n-k)$-cell $\tilde{\sigma}$ such that

$$\frac{\omega}{\mathrm{Vol}(\sigma)} = \frac{\star\omega}{\mathrm{Vol}(\tilde{\sigma})}$$

allowing us to define $\star$ as

$$\star = \frac{\mathrm{Vol}(dual)}{\mathrm{Vol}(primal)}.$$

In effect the diagonal Hodge star requires that the averages of the integrand over the respective domains agree.

This is represented as a diagonal matrix so that, again, application of the operator becomes a simple matrix-vector multiplication. Note that when transforming quantities from the dual to the primal, the inverse of this matrix is used. Since the matrix is diagonal we only store the diagonal entries. There are as many of these as there are simplices of the appropriate dimension. Consequently the diagonal Hodge star can be represented with vectors of length $|\mathbf{V}|$, $|\mathbf{E}|$, $|\mathbf{F}|$, and $|\mathbf{T}|$ respectively.

6.2.1 Calculating Dual Volumes

So far the entire implementation has been in terms of the combinatorics of the mesh, but when constructing the Hodge star we must finally introduce the geometry. After all, the purpose of the Hodge star is to capture the metric. The volumes of the primal simplices are straightforward: 1 for vertices, length for edges, area for triangles, and volume for tetrahedra. The dual volumes are similarly defined, but in order to avoid constructing the graph of the dual mesh explicitly, we calculate the dual volumes as follows.

If we use the circumcentric realization of the dual mesh (*i.e.*, dual vertices are at the circumcenters of the associated tets), we can exploit the following facts when calculating the dual volumes.[1]

- A dual edge (dual of a primal triangle t) is linear, is normal to t, and is collinear with the circumcenter of t (though the line segment need not necessarily pass through t).
- A dual polygon (dual of a primal edge e) is planar, is orthogonal to e, and is coplanar with the center of e (though it need not intersect e).
- A dual cell (dual of a primal vertex v) is the convex intersection of the half-spaces defined by the perpendicular bisectors of the edges incident on v.

Just as with primal vertices, the volume of a dual vertex is defined to be 1. For the others, we can conceptually decompose each cell into pieces bounded by lower dimensional cells, and sum the volumes of the pieces. For example, a dual polyhedron can be seen as the union of some number of pyramids, where the base of each pyramid is a dual polygon and the apex is the primal vertex. Similarly, a dual polygon can be seen as a union of triangles with dual edges at the bases, and dual edges can be seen as a union of (two) line segments with dual vertices at the bases. The following pseudocode illustrates how the volumes are calculated.

```
vec3 C( Simplex s );  // gives the circumcenter of s

// Initialize all dual volumes to 0.

// Dual edges
for each primal triangle f
    for each primal tet t_f incident on f
        b ← t_f.dualVolume  // 1
        h ← ||C(f) − C(t_f)||
        f.dualVolume ← f.dualVolume + (1/1) bh

// Dual polygons
for each primal edge e
    for each primal triangle f_e incident on e
        b ← f_e.dualVolume
        h ← ||C(e) − C(f_e)||
        e.dualVolume ← e.dualVolume + (1/2) bh

// Dual polyhedra
for each primal vertex v
    for each primal edge e_v incident on v
        b ← e_v.dualVolume
        h ← ||C(v) − C(e_v)||
        v.dualVolume ← v.dualVolume + (1/3) bh
```

Note that, even when dealing with the geometry of the mesh, this part of the implementation still generalizes trivially to arbitrary dimension.

[1] Circumcentric duals may only be used if the mesh satisfies the Delaunay criterion. If it does not, a barycentric dual mesh may be used. However, care must be taken if a barycentric dual mesh is used, as dual edges are no longer straight lines (they are piecewise linear), dual faces are no longer planar, and dual cells are no longer necessarily convex.

7 Summary

All the machinery discussed above can be summarized as follows:

- k-forms as well as the Hodge star are represented as vectors of length $|\mathbf{V}|$, $|\mathbf{E}|$, $|\mathbf{F}|$, and $|\mathbf{T}|$;
- the discrete exterior derivative is represented as (transposes of) sparse adjacency matrices containing only entries of the form $+1$ and -1 (and many zeros); the adjacency matrices are of dimension $|\mathbf{V}| \times |\mathbf{E}|$ (boundary of edges), $|\mathbf{E}| \times |\mathbf{F}|$ (boundary of triangles), and $|\mathbf{F}| \times |\mathbf{T}|$ (boundary of tets).

In computations these matrices then play the role of operators such as grad, curl, and div and can be composed to construct operators such as the Laplacian (and many others).

While the initial setup of these matrices is best accomplished with associative containers, their final form can be realized with standard sparse matrix representations. Examples include a compressed row storage format, a vector of linked lists (one linked list for each row), or a two dimensional linked list (in effect, storing the matrix and its transpose simultaneously) allowing fast traversal of either rows or columns. The associative containers store integer tuples together with orientation signs. For these we suggest the use of sorted integer tuples (the canonical representatives of each simplex). Appropriate comparison operators needed by the container data structures simply perform lexicographic comparisons.

And that's all there is to it!

Acknowledgments This work was supported in part through a James Irvine Fellowship to the first author, NSF (DMS-0220905, DMS-0138458, ACI-0219979), DOE (W-7405-ENG-48/B341492), nVidia, the Center for Integrated Multiscale Modeling and Simulation, Alias, and Pixar.

References

JOY, K. I., LEGAKIS, J., AND MACCRACKEN, R. 2002. *Data Structures for Multiresolution Representation of Unstructured Meshes*. Springer-Verlag, Heidelberg, Germany.

Chapter 9:
Stable, Circulation-Preserving, Simplicial Fluids

Sharif Elcott, Yiying Tong*, Eva Kanso†, Peter Schröder, Mathieu Desbrun
Caltech

Figure 1: *Discrete Fluids: we present a novel geometric integration scheme for fluids applicable to tetrahedral meshes of arbitrary domains. Aside from resolving the boundaries precisely, our approach also provides an accurate treatment of vorticity through a discrete preservation of Kelvin's circulation theorem. Here, a hot smoke cloud rises inside a bunny shaped domain of* 7*K vertices (*32*K tetrahedra—equivalent complexity of a* 19^3 *regular grid), significantly reducing the computational cost of the simulation for such an intricate boundary compared to regular grid-based techniques (*0.47*s/frame on a Pentium 4, 3GHz).*

Abstract

Visual quality, low computational cost, and numerical stability are foremost goals in computer animation. An important ingredient in achieving these goals is the conservation of fundamental motion invariants. For example, rigid and deformable body simulation benefits greatly from conservation of linear and angular momenta. In the case of fluids, however, none of the current techniques focuses on conserving invariants, and consequently, they often introduce a visually disturbing numerical diffusion of *vorticity*. Visually just as important is the resolution of complex simulation domains. Doing so with regular (even if adaptive) grid techniques can be computationally delicate. In this paper, we propose a novel technique for the simulation of fluid flows. It is designed to respect the defining differential properties, *i.e.*, the *conservation of circulation* along arbitrary loops as they are transported by the flow. Consequently, our method offers several new and desirable properties: arbitrary simplicial meshes (triangles in 2D, tetrahedra in 3D) can be used to define the fluid domain; the computations involved in the update procedure are efficient due to discrete operators with small support; and it preserves *discrete circulation* avoiding numerical diffusion of vorticity.

1 Introduction

Conservation of motion invariants at the discrete computational level, *e.g.*, linear and angular momenta in solid mechanics, is now widely recognized as being an important ingredient in the construction of numerically stable simulations with a high degree of visual realism [Marsden and West 2001]. Much of the progress in this direction has been enabled by a deeper understanding of the underlying geometric structures and how they can be preserved as we go from continuous models to discrete computational realizations. So far, advances of this type have yet to deeply impact fluid flow simulations. Current methods in fluid simulation are rarely able to conserve *defining physical properties*. Consider, for example, the need in many methods to continually project the numerically updated velocity field onto the set of divergence free velocity fields; or the need to continually reinject vorticity lost due to numerical dissipation as a simulation progresses. In this paper we introduce a novel, geometry-based algorithm for fluid simulation which, among other desirable properties, exactly preserves vorticity at a discrete level.

1.1 Previous Work

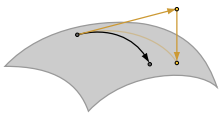
Fluid Mechanics has been studied extensively in the scientific community both mathematically and computationally. The physical behavior of incompressible fluids is usually modeled by the Navier Stokes (NS) equations for viscous fluids and by the Euler equations for inviscid (non-viscous) fluids. Numerical approaches in computational fluid dynamics typically discretize the governing equations through Finite Volumes (FV), Finite Elements (FE) or Finite Difference (FD) methods. We will not attempt to review the many methods proposed (an excellent survey can be found in [Langtangen et al. 2002]) and instead focus on approaches used for fluids in computer graphics. Some of the first fluid simulation techniques were based on vortex blobs [Yaeger et al. 1986], vortex particles [Gamito et al. 1995] and Finite Differences [Foster and Metaxas 1997]. To circumvent the ill-conditioning of the latter approach for large time steps and achieve unconditional stability, Jos Stam [1999; 2001] introduced to the graphics community the *method of characteristics* for fluid advection and the Helmholtz-Hodge decomposition [Chorin and Marsden 1979] to ensure the divergence-free nature of the fluid motion. The resulting algorithm, called *Stable Fluids*, is an extremely successful semi-Lagrangian approach based on a regular grid Eulerian space discretization, that led to many refinements and extensions which have contributed to the enhanced visual impact of fluid animations. Among others, these include the use of staggered grids and monotonic cubic interpolation [Fedkiw et al. 2001]; improvements in the handling of interfaces [Foster and Fedkiw 2001; Guendelman et al. 2005]; extensions to curved surfaces [Stam 2003; Shi and Yu 2004] and visco-elastic objects [Goktekin et al. 2004]; as well as goal oriented control of fluid motion [Treuille et al. 2003; McNamara et al. 2004; Pighin et al. 2004; Shi and Yu 2005].

*Now at Michigan State University.
†Now at the University of Southern California.

1.2 Shortcomings of Stable Fluids

However, the Stable Fluids technique is not without drawbacks. Chief among them is the difficulty of accommodating complex domain boundaries with regular grids, as addressed recently with hybrid meshes [Feldman et al. 2005]. Local adaptivity [Losasso et al. 2004] can greatly help, but the associated octree structures require significant overhead. Note that regular partitions of space (adaptive or not) can suffer from preferred direction sampling, leading to visual artifacts similar to aliasing in rendering.

Additionally, the different variants of the original Stable Fluids algorithm [Stam 1999] are all based on a class of discretization approaches known in Computational Fluid Dynamics as fractional step methods. In order to numerically solve the Euler equations over a time step, they proceed in two stages. They first update the velocity field assuming the fluid is inviscid and disregard the divergence-free constraint. Then, the resulting velocity is projected onto the closest divergence-free flow (in the $\mathscr{L}^2$ sense) through an exact Helmholtz-Hodge decomposition. Despite the simplicity of this fractional integration, one of its consequences is excessive numerical diffusion: advecting velocity before reprojecting onto a divergence-free field creates significant energy loss [Chang et al. 2002]. While this shortcoming is not a major issue in graphics as the visual impact of such a loss is not always significant, another consequence of the fractional integration is an *excessive diffusion of vorticity*. This last issue affects the motion significantly, since the presence of vortices in liquids and volutes in smoke is one of the most important visual cues of our perception of fluidity. Vorticity confinement [Steinhoff and Underhill 1994; Fedkiw et al. 2001] was designed to counteract this diffusion through local reinjection of vorticity. Unfortunately, it is hard to control how much can safely be added back without affecting stability or plausibility of the results. Adding vortex particles can further reduce this loss [Selle et al. 2005], but adds computational cost to the Stable Fluids method. One can understand this seemingly inevitable numerical diffusion through the following geometric argument: the solutions of Euler equations are *geodesic* (*i.e.*, shortest) paths *on* the manifold of all possible divergence-free flows; thus, advecting the fluid *out* of the manifold (to simulate a time integration over a small time step) is not a proper substitute to this intrinsic constrained minimization, even if the post re-projection is, in itself, exact. For a more detailed numerical analysis of this flaw, see [Chang et al. 2002].



1.3 Contributions

In this paper we show that a careful setup of *discrete differential quantities*, designed to respect structural relationships such as vector calculus identities, leads in a direct manner to a numerical simulation method which respects the defining *geometric structure* of the fluid equations. A key ingredient in this approach is the location of physical quantities on the appropriate geometric structures (*i.e.*, vertices, edges, faces, or cells). This greatly simplifies the implementation as all variables are *intrinsic*. It also ensures that the approach works for curved manifolds without any changes. With the help of a discrete calculus on simplicial complexes we construct a novel integration scheme which employs intrinsically divergence-free variables. Thus, our time integration method removes the need to enforce the usual divergence-free constraint: it preserves circulation (and consequently vorticity) *by construction* while being simple and numerically efficient, achieving high visual quality even for large time steps. Although our approach shares the same algorithmic structure as Stable Fluids based methods (use of backward path tracing and sparse linear systems), it fundamentally differs from previous techniques on the following points:

- **our technique is based on a classical *vorticity* formulation of the Navier-Stokes and Euler equations**; unlike most vorticity-based methods in CG, our approach is *Eulerian* as we work only with a fixed mesh and *not* a Lagrangian representation involving vorticity particles (or similar devices);
- we adopt an unconditionally-stable, semi-Lagrangian backward advection strategy as in Stable Fluids; **however, in contrast to Stable Fluids, we do not point-sample velocity, but rather compute *integrals of vorticity***; this simple change removes the need to enforce incompressibility of the velocity field through projection; note that this convenient recourse to a vorticity-based formulation has been proposed by multiple authors (see, for instance, [Weißmann 2006; Angelidis et al. 2006; Park and Kim 2005; Angelidis et al. 2005; E and Liu 1996b; E and Liu 1996a]).
- **our strategy exactly preserves circulation along discrete loops in the mesh**; capturing this geometric property of fluid dynamics guarantees that vorticity does not get dissipated as is typically the case in Stable Fluids; consequently no vorticity confinement (or other post processes) are required to maintain this important element of visual realism;
- **our method has multiple advantages from an implementation point of view**: it handles arbitrary meshes (regular grids, hybrid meshes [Feldman et al. 2005], and even cell complexes) with non-trivial topology; the operators involved have very small support leading to very sparse linear systems; all quantities are stored intrinsically (scalars on edges and faces) without reference to global or local coordinate frames (unlike recent work on Finite Volume fluid simulation [Wendt et al. 2005]). These novel features are achieved with a computational cost similar to previous Stable Fluids approaches.

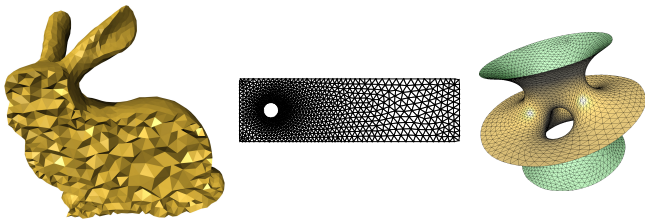


Figure 2: *Domain Mesh: our fluid simulator uses a simplicial mesh to discretize the equations of motion; (left) the domain mesh (shown as a cutaway view) used in Fig. 1; (middle) a coarser version of the flat 2D mesh used in Fig. 7; (right) the curved triangle mesh used in Fig. 11.*

The organization of this paper is as follows. In Section 2, we motivate our approach through a brief overview of the theory and computational algorithms for Fluid Mechanics. We present a novel discretization of fluid mechanics and a circulation-preserving integration algorithm in Section 2.2. The computational machinery required by our approach is developed in Section 3, while the algorithmic details are given in Section 4. Several numerical examples are shown and discussed in Section 5.

2 Of Motion and Discrete Flows

Before going into the details of our algorithm we discuss the underlying fluid equations with their relevant properties and how these can be captured over discretized domains.

2.1 Geometry of Fluid Motion

Euler Fluids Consider an inviscid, incompressible, and homogeneous fluid on a domain $\mathscr{D}$ in 2 or 3D. The *Euler equations*, governing the motion of this fluid (with no external forces for now, and

slip conditions on boundaries), can be written as:

$$\begin{aligned} \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} &= -\nabla p \,, \\ div(\mathbf{u}) &= 0 \,, \quad \mathbf{u} \parallel \partial \mathscr{D} \,. \end{aligned} \tag{1}$$

We assume unit density ($\rho = 1$) and use $\mathbf{u}$ to denote the fluid velocity, p the pressure, and $\partial\mathscr{D}$ the boundary of the fluid region $\mathscr{D}$. The pressure term in Eq. (1) can be dropped easily by rewriting the Euler equations in terms of *vorticity*. Recall that traditional vector calculus defines vorticity as the curl of the velocity field, $\boldsymbol{\omega} = \nabla \times \mathbf{u}$. Taking the curl ($\nabla\times$) of Eq.(1), we obtain

$$\begin{aligned} \frac{\partial \boldsymbol{\omega}}{\partial t} + \mathscr{L}_{\mathbf{u}} \boldsymbol{\omega} &= \mathbf{0} \,, \\ \boldsymbol{\omega} = \nabla \times \mathbf{u} \,, \quad div(\mathbf{u}) &= 0 \,, \quad \mathbf{u} \parallel \partial \mathscr{D} \,. \end{aligned} \tag{2}$$

where $\mathscr{L}_{\mathbf{u}}\boldsymbol{\omega}$ is the Lie derivative, equal in our case to $\mathbf{u}\cdot\nabla\boldsymbol{\omega} - \boldsymbol{\omega}\cdot\nabla\mathbf{u}$. In this form, this vorticity-based equation states that *vorticity is simply advected along the fluid flow*.[1] Note that Equation (2) is equivalent to the more familiar $\frac{D\boldsymbol{\omega}}{Dt} = \boldsymbol{\omega}\cdot\nabla\mathbf{u}$, and therefore already includes the vortex stretching term appearing in *Lagrangian* approaches. Roughly speaking, vorticity measures the local spin of a fluid parcel. Therefore, vorticity advection means that this local spin moves with the flow.

Since the integral of vorticity on a given bounded surface equals (by Stokes' theorem) the *circulation* around the bounding loop of the surface, one can explain the geometric nature of an ideal fluid flow in particularly simple terms: *the circulation around any closed loop $\mathscr{C}$ is conserved throughout the motion of this loop in the fluid*. This key result is known as Kelvin's circulation theorem, and is usually written as:

$$\Gamma(t) = \oint_{\mathscr{C}(t)} \mathbf{u} \cdot \mathbf{dl} = \text{constant} \,, \tag{3}$$

where $\Gamma(t)$ is the circulation of the velocity on the loop $\mathscr{C}$ at time t as it gets advected in the fluid. This will be the key to our time integration algorithm.

Viscous Fluids In contrast to ideal fluids, incompressible *viscous* fluids generate very different fluid behaviors. They can be modelled by the *Navier-Stokes* equations (compare with Eq. (1)):

$$\begin{aligned} \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} &= -\nabla p + \nu \Delta \mathbf{u} \,, \\ div(\mathbf{u}) &= 0 \,, \quad \mathbf{u}|_{\partial\mathscr{D}} = \mathbf{0} \,. \end{aligned} \tag{4}$$

where Δ denotes the Laplace operator, and ν is the *kinematic viscosity*. Note that different types of boundary conditions can be added depending on the chosen model. Despite the apparent similarity between these two models for fluid flows, the added diffusion term dampens the motion, resulting in a slow decay of circulation. This diffusion also implies that the velocity of a viscous fluid at the boundary of a domain must be null, whereas an inviscid fluid could have a non-zero tangential component on the boundary. Here again, one can avoid the pressure term by taking the curl of the equations (compare with Eq. (2)):

$$\begin{aligned} \frac{\partial \boldsymbol{\omega}}{\partial t} + \mathscr{L}_{\mathbf{u}} \boldsymbol{\omega} &= \nu \Delta \boldsymbol{\omega} \,, \\ \boldsymbol{\omega} = \nabla \times \mathbf{u} \,, \quad div(\mathbf{u}) &= 0 \,, \quad \mathbf{u}|_{\partial\mathscr{D}} = \mathbf{0} \,. \end{aligned} \tag{5}$$

[1] Note that this is a more canonical characterization of the motion than the velocity-based one: the latter was also an advection but under the additional *constraint* of keeping the velocity field divergence-free, which is the reason for the gradient of pressure.

2.2 Discrete Setup

For a discrete time and space numerical simulation of Eqs. (2) and (5) we need a discretized geometry of the domain (given as a simplicial mesh for instance), appropriate discrete analogs of velocity $\mathbf{u}$ and vorticity fields $\boldsymbol{\omega}$, along with the operators which act on them. We will present our choices before describing the geometry-based time integration approach.

2.2.1 Space Discretization

We discretize the spatial domain in which the flow takes place using a locally oriented simplicial complex, *i.e.*, either a tet mesh for 3D domains or a triangle mesh for 2D domains, and refer to this discrete domain as $\mathscr{M}$ (see Figure 2). The domain may have non-trivial topology, *e.g.*, it can contain tunnels and voids (3D) or holes (2D), but is assumed to be compact. To ensure good numerical properties in the subsequent simulation we require the simplices of $\mathscr{M}$ to be well shaped (aspect ratios bounded away from zero). This assumption is quite common since many numerical error estimates depend heavily on the element quality. We use meshes generated with the method of [Alliez et al. 2005]. Collectively we refer to the sets of vertices, edges, triangles, and tets as V, E, F, and T.

We will also need a *dual mesh*. It associates with each original simplex (vertex, edge, triangle, tet, respectively) its dual (dual cell, dual face, dual edge, and dual vertex, respectively—see Fig. 3). The geometric realization of the dual mesh uses tet circumcenters as dual vertices and Voronoi cells as dual cells; dual edges are line segments connecting dual vertices across shared tet faces and dual faces are the faces of the Voronoi cells. Notice that storing values on primal simplices or on their associated dual cells is conceptually equivalent, since both sets have the same cardinality. We will see in Section 3 that corresponding primal and dual quantities are related through a simple (diagonal) linear operator.

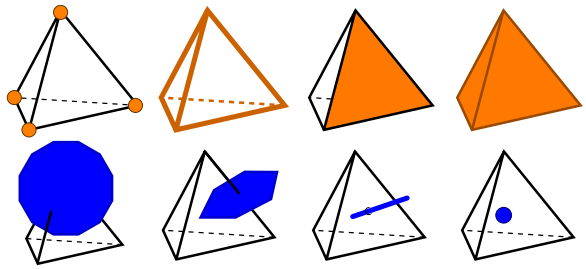


Figure 3: *Primal and Dual Cells: the simplices of our mesh are vertices, edges, triangles and tets (top); their circumcentric duals are dual cells, dual faces, dual edges and dual vertices (bottom).*

2.2.2 Intrinsic Discretization of Physical Quantities

In order to faithfully capture the geometric structure of fluid mechanics on the discrete mesh, we define the usual physical quantities, such as velocity and vorticity, through *integral values over the elements of the mesh $\mathscr{M}$*. This is the sharpest departure from traditional numerical techniques in CFD: we not only use values at nodes and tets (as in FEM and FVM), but also allow association (and storage) of field values at *any* appropriate simplex. Depending on whether a given quantity is defined pointwise, or as a line, area or volume density, the corresponding discrete representation will "live" at the associated 0, 1, 2, and 3 dimensional mesh elements. With this in mind we use a simple discretization of the physical quantities of fluid mechanics based on fluxes associated to

faces, reminiscent of Finite Volume methods (see also [Nicolaides and Wu 1997] for a similar setup for Div-Curl equations).

Velocity as Discrete Flux To encode a coordinate free (intrinsic) representation of velocity on the mesh we use *flux*, *i.e.*, the mass of fluid transported across a given surface area per unit time. Note that this makes flux an *integrated*, not pointwise, quantity. On the discrete mesh, fluxes are associated with the triangles of the tet mesh. Thus fluid velocity $\mathbf{u}$ is treated as a 2-form and represented as a vector U of scalar values on faces (size $|F|$). This coordinate-free point of view, also used in [Feldman et al. 2005], is reminiscent of the staggered grid method used in [Fedkiw et al. 2001] and other non-collocated grid techniques (see [Goktekin et al. 2004]). In the staggered grid approach one does not store the x, y, z components of a vector at nodes but rather associates them with the corresponding grid faces. We may therefore think of the idea of storing fluxes on the triangles of our tet mesh as a way of *extending* the idea of staggered grids to the more general simplical mesh setting. This was previously exploited in [Bossavit and Kettunen 1999] in the context of E&M computations. It also makes the usual no-transfer boundary conditions easy to encode: boundary faces experience no flux across them. Encoding this boundary condition for velocity vectors stored at vertices is far more cumbersome.

Divergence as Net Flux on Tets Given the incompressibility of the fluid, the velocity field must be divergence-free ($\nabla \cdot \mathbf{u} = 0$). In the discrete setting, the integral of the divergence over a tet becomes particularly simple. According to the generalized Stokes' theorem this integral equals the sum of the fluxes on all four faces: the discrete notion of divergence is therefore simply the net flux for each tet. Divergence can therefore be stored as a 3-form, *i.e.*, as a value associated to each tet (a vector of cardinality $|T|$). The notion of incompressibility becomes particularly intuitive: whatever gets in each tet must be compensated by whatever comes out (see Fig. 4).

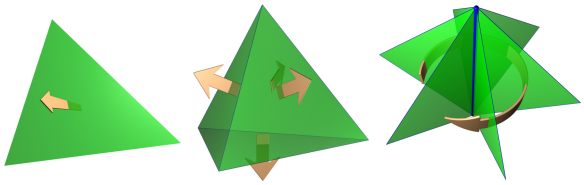


Figure 4: *Discrete Physical Quantities: in our geometric discretization, fluid flux lives on faces (left), divergence lives on tets (middle), and vorticity lives on edges (right).*

Vorticity as Flux Spin Finally we need to define vorticity on the mesh. To see the physical intuition behind our definition, consider an edge in the mesh. It has a number of faces incident on it, akin to a paddle wheel (see Figure 4, right). The flux on each face contributes a torque to the edge. The sum of all these, when going around an edge, is the net torque that would "spin" the edge. We can thus give a physical definition of vorticity as a weighted sum of fluxes on all faces incident to a given edge. This quantity is now associated with primal edges—or, equivalently, dual faces—and is thus represented by a vector Ω of size $|E|$. Note that we will talk about discrete vorticity from now on to mean "area integral of the vorticity vector field", or equivalently, "integral of the vorticity seen as a 2-form".

In Section 3, we will see that these physical intuitions can be formally derived from simple algebraic relationships.

2.3 Geometric Integration of Fluid Motion

Since we are using the vorticity formulation of the fluid equations (Eqs. (2) or (5)) the time integration algorithm must update the discrete vorticity variables which are stored on each primal edge. We have seen that the fluid equations state that vorticity is advected by the velocity field. The fundamental idea of our geometric integration algorithm is thus to ensure that Kelvin's theorem holds in the discrete setting: the circulation around any loop in the fluid remains constant as the loop is advected. This can be achieved by backtracking loops: for any given loop at the current time, determine its backtracked image in the velocity field ("where did it come from?") and compute the circulation around the backtracked loop. This value is then assigned as the circulation around the original loop at the present time, *i.e.*, circulation is properly advected *by construction* (see Figure 5 for a depiction of this loop advection idea).

Since we store vorticity on primal edges, a natural choice for these loops are the bounding loops of the dual faces associated to each primal edge (see Figure 3). Notice that these loops are polylines formed by sequences of dual vertices around a given primal edge. Consequently an efficient implementation of this idea requires only that we backtrack *dual vertices* in the velocity field (the same way primal vertices are backtracked in the traditional Stable Fluids method). Once these positions are known *all* backtracked dual loops associated to primal edges are known. These Voronoi loops can indeed generate any discrete, dual loop: the sum of adjacent loops is a larger, outer loop as the interior edges cancel out due to opposite orientation as sketched in Fig. 5(right). The evaluation of circulation around these backtracked loops will be quite straightforward. Invoking Stokes' theorem, the integral of vorticity over a dual face equals the circulation around its boundary. With this observation we can achieve our goal of updating vorticities and, *by design*, ensured a discrete version of Kelvin's theorem. The algorithmic details of this simple geometric approach to time integration of the equations of motion for fluids will be given in Section 4.

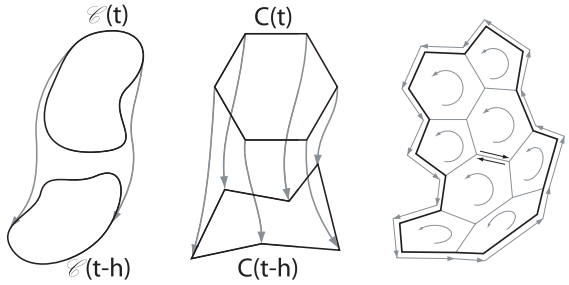


Figure 5: *Kelvin's Theorem: (left) in the continuous setting, the circulation on any loop being advected by the flow is constant. (middle) our discrete integration scheme enforces this property on each Voronoi loop, (right) thus on any discrete loop.*

3 Computational Machinery

Now that we have described our choices of spatial and physical discretizations, along with a way to use them to integrate the fluid's motion, we must manipulate the numerics involved in a principled manner to guarantee proper physical behavior. In this section, we point out that the intuitive definition of our physical quantities living at the different simplices of a mesh can be made precise through the definition of a *discrete differential structure*. Consequently, the basic operators to go from fluxes to the divergence, curl, or Laplacian of the velocity field can be *formally* defined through the use of discrete differential forms. We will mostly focus on presenting the practical implementation of the few operators we need; more importantly, we will show that this implementation reduces to *simple linear algebra* with very sparse matrices.

3.1 Discrete Differential Structure

Integrals and Forms In Section 2.2, we have opted for manipulating the physical quantities in the form of a line, surface, and volume integral computed directly on our meshed domain to render the setup entirely intrinsic, *i.e.*, with no need for vector fields

to be stored with respect to arbitrary coordinate frames. Such an integral represents the evaluation of a mathematical entity called a *differential form*. In the continuous three-dimensional setting, a 0-form is simply a function on that 3D space. A 1-form, or line-form, is a quantity that can be *evaluated through integration over a curve*. Thus a 1-form can be thought of as a proxy for a vector field, and its integral over a curve becomes the circulation of this vector field. A 2-form, or area-form, is to be *integrated over a surface*, that is, it can be viewed as a proxy for a vector perpendicular to that surface (and its evaluation becomes the flux of that vector field through the surface); finally, a 3-form, or volume-form, is to be *integrated over a volume* and can be viewed as a proxy for a function. Classic calculus and vector calculus can then be substituted with a special calculus involving only differential forms, called *exterior calculus*—the basis of Hodge theory and modern differential geometry (for a comprehensive discussion, see, for example, [Abraham et al. 1988]).

Discrete Forms and Their Representation However, in our framework, the continuous domain is replaced (or approximated) by a mesh, the only structure we can work with. Therefore, the integrated physical values we store on the mesh corresponds to *discrete differential k-forms* [Desbrun et al. 2006]. A discrete differential k-form, $k = 0, 1, 2$, or 3, is the evaluation (*i.e.*, the integral) of the differential k-form on all k-cells, or k-simplices. In practice, discrete k-forms can simply be considered as *vectors of numbers* associated to the simplices they live on: 0-forms live on vertices, and are expressed as a vector of length $|V|$; correspondingly, 1-forms live on edges (length $|E|$), 2-forms live on faces (length $|F|$), and 3-forms live on tetrahedra (length $|T|$). Dual forms, *i.e.*, forms that we will evaluate on the dual mesh, are treated similarly. The reader should now realize that in our discretization of physical quantities, the notion of flux that we described is thus a primal 2-form (integrated over faces), while its vorticity is a dual 2-form (integrated over dual faces), and its divergence becomes a primal 3-form (integrated over tetrahedra).

Discrete Differential Calculus on Simplicial Meshes These discrete forms can now be used to build the tools of calculus through Discrete Exterior Calculus (DEC), a coherent calculus mimicking the continuous setting that only uses discrete combinatorial and geometric operations [Munkres 1984; Hirani 2003]. At the core of its construction is the definition of a discrete d operator (analog of the continuous exterior derivative), and a discrete Hodge star, which will allow us to move values from the primal mesh to the dual mesh and vice-versa. For a more comprehensive introduction to DEC and the use of discrete differential forms, we refer the interested reader to [Desbrun et al. 2006] and [Bochev and Hyman 2006].

3.2 Two Basic Operators

The computations involved in our approach only require the definition of two basic operators: one is the exterior derivative d, necessary to compute derivatives, like gradients, divergences, or curls; the other is the Hodge star, to transfer values from primal simplices to dual simplices.

Exterior Derivative d Given an oriented mesh, we implement our first operator by simply assembling the *incidence matrices* of the mesh. These will act on the vectors of our discrete forms and implement the discrete exterior derivative operator d as explained in more details in Appendix A. For our 3D implementation, there are three sparse matrices involved, which contain only entries of type 0, $+1$, and -1. Care is required in assembling these incidence matrices, as the orientation must be taken into account in a consistent manner [Elcott and Schröder 2006]. The first one is d_0, the transpose of the incidence matrix of vertices and edges ($|V|$ rows and $|E|$ columns). Each row of the transpose contains a single $+1$ and -1 for the end points of the given edge (and zero otherwise). The sign is determined from the orientation of the edge. Similarly, the second matrix d_1 is the transpose of the incidence matrix of edges and faces ($|E|$ rows and $|F|$ columns), with appropriate $+1$ and -1 entries according to the orientation of edges as one moves around a face. More generally d_k is the transpose of the incidence matrix of k-cells on $k+1$-cells. A simple debugging sanity check (necessary but not sufficient) is to compute consecutive products: d_0 followed by d_1 must be a matrix of zeros; d_1 followed by d_2 must similarly give a zero matrix. This reflects the fact that the boundary of any boundary is the empty set. It also corresponds to the calculus property that curl of grad is zero as is divergence of curl (see [Desbrun et al. 2006]).

Hodge Star The second operator we need will allow us to transfer quantities back and forth between the primal and dual mesh. We can project a primal k-form to a conceptually-equivalent dual $(3-k)$-form with the *Hodge star*. We will denote $\star_0$ (resp., $\star_1, \star_2, \star_3$) the Hodge star taking a 0-form (resp., 1-form, 2-form, and 3-form) to a dual 3-form (resp., dual 2-form, dual 1-form, dual 0-form). In this paper we will use what is known as the *diagonal Hodge star* [Bossavit 1998] as it offers an acceptable approximation at very little computational cost. This operator simply scales whatever quantity that is stored on mesh cells by the volumes of the corresponding dual and primal cells: let $\mathrm{vol}(.)$ denote the volume of a cell (*i.e.*, 1 for vertices, length for edges, area for triangles, and volume for tetrahedra), then

$$(\star_k)_{ii} = \mathrm{vol}(\tilde{\sigma}_i) / \mathrm{vol}(\sigma_i)$$

where σ_i is any primal k-simplex, and $\tilde{\sigma}_i$ is its dual (all other terms, off the diagonal, are zero). These linear operators, describing the local metric, are diagonal and can be stored as vectors. Conveniently, the inverse matrices going from dual to primal quantities are trivial to compute for this diagonal Hodge star.

Overloading Operators Note that both the d_k and the $\star_k$ operators are *typed*: the subscript k is *implicitly determined* by the cardinality of the argument. For example, the velocity field $\mathbf{u}$ is a 2-form stored as a vector U of cardinality $|F|$. Consequently the expression dU implies use of the $|T| \times |F|$-sized matrix d_2. In the implementation this is accomplished with operator overloading (in the sense of `C++`). We will take advantage of this in the paper from now on and drop the dimension subscripts.

3.3 Offline Matrix Setup

With these overloads of d and $\star$ in place, we can now set up the only two matrices (C and L) that will be used during simulation. They respectively represent the discrete analogs of the curl and Laplace operators [Desbrun et al. 2006].

Curl Since we store fluxes on faces and gather them in a vector U, the *circulation* of the vector field $\mathbf{u}$ can be derived as values on dual edges through $\star U$. *Vorticity*, typically thought of as a 2-form in fluid mechanics [Marsden and Wenstein 1983], is easily computed by then summing this circulation along the dual edges that form the boundary of a dual face. In other words, $\boldsymbol{\omega} = \nabla \times \mathbf{u}$ becomes, in terms of our discrete operators, simply $\Omega = d^T \star U$. We therefore create a matrix $C = d^T \star$ offline, *i.e.*, the composition of an incidence matrix with a diagonal matrix.

Laplacian The last matrix we need to define (to handle viscous fluids) is the discrete Laplacian. The discrete analog of $\Delta\boldsymbol{\phi} = (\nabla\nabla\cdot - \nabla\times\nabla\times)\boldsymbol{\phi} = \boldsymbol{\omega}$ is simply $(\star d \star^{-1} d^T \star + d^T \star d)\,\Phi = \Omega$ as explained in Appendix B. This last matrix, a direct composition of incidence and diagonal matrices, is precomputed and stored as L for later use.

4 Implementation

To facilitate a direct implementation of our integration scheme, we provide pseudocode in Figure 6. A series of implementation details follow, providing comments on specific steps and how these relate to the machinery developed in earlier sections.

```
//Load mesh and build incidence matrices
C ← d^T ⋆
L ← ⋆d⋆^{-1} d^T ⋆ + d^T ⋆ d

//Time stepping h
loop
    //Advect Vorticities
    for each dual vertex (tet circumcenter) c_i
        ĉ_i ← PathTraceBackwards(c_i);
        v_i ← InterpolateVelocityField(ĉ_i);
    for each dual face f
        Ω_f ← 0
        for each dual edge (i, j) on the boundary of f
            Ω_f ← Ω_f + ½ (v_i + v_j) · (ĉ_i − ĉ_j);

    //Add forces
    Ω ← Ω + h C F

    //Add diffusion for Navier-Stokes
    Ψ ← SolveCG( (⋆ − ν h L)Ψ = Ω );
    Ω ← ⋆ Ψ

    //Convert vorticities back to fluxes
    Φ ← SolveCG( L Φ = Ω );
    U ← dΦ;
```

Figure 6: *Pseudocode of our fluid motion integrator (*SolveCG *calls a linear solver, typically based on the Conjugate Gradient method).*

4.1 Converting Vorticities back to Fluxes

After we update the vorticity on each dual Voronoi face of the domain through semi-Lagrangian backtracing and resampling via integration, the resulting vorticity field needs to be converted back to a velocity field that the next time step will utilize to backtrack vorticity once again. Since the vorticities Ω are related to the fluxes U through $\Omega = d^T \star U$, we can directly solve for the fluxes via a Poisson equation as explained in detail in Appendix B. The linear system involved being sparse and symmetric, a Conjugate Gradient is recommended for efficiency. Note that a thresholding of the vorticity values stored on each dual face can be also performed during this step to offer a simple way to control any excessive amount of vorticity in the flow (due to exaggerated user-added external forces, for instance).

4.2 External Body Forces

The use of external body forces, like buoyancy, gravity, or stirring, is common practice to create interesting motions. Incorporating external forces into Eq. (4) is straightforward, resulting in:

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} \;=\; -\nabla p + \nu \Delta \mathbf{u} + \mathbf{f}\,.$$

Again, taking the curl of this equation allows us to recast this equation in terms of vorticity:

$$\frac{\partial \boldsymbol{\omega}}{\partial t} + \mathscr{L}_{\mathbf{u}} \boldsymbol{\omega} \;=\; \nu \Delta \boldsymbol{\omega} + \nabla \times \mathbf{f}\,. \tag{6}$$

Thus, we note that an external force influences the vorticity only through the force's curl (the $\nabla \cdot \mathbf{f}$ term is compensated for by the pressure term keeping the fluid divergence-free). Thus, if we express our forces through the vector F of their resulting fluxes in each face, we can directly add the forces to the domain by incrementing Ω by the circulation of F over the time step h, *i.e.*:

$$\Omega \leftarrow \Omega + h\, C\, F.$$

4.3 Adding Diffusion

If we desire to simulate a viscous fluid, we must add the diffusion term present in Eq. (5). Note that previous methods were sometimes omitting this term because their numerical dissipation was already creating (uncontrolled) diffusion. In our case, however, this diffusion needs to be properly handled if viscosity is desired. This is easily done through an unconditionally-stable implicit integration as done in Stable Fluids (*i.e.*, we also use a fractional step approach). Using the discrete Laplacian (given in Appendix B, Eq. (8)) and the current vorticity Ω, we simply solve for the diffused vorticity Ω' using the following linear system:

$$(\star - \nu h L)\ \star^{-1} \Omega' = \Omega.$$

4.4 Interpolation of Velocity

In order to perform the backtracking of dual vertices we must first define a velocity field over the entire domain using the data we have on primal faces (fluxes). This can be done by computing a unique velocity vector for each dual vertex and then using barycentric interpolation of these vectors over each dual Voronoi cell [Warren et al. 2007], defining a continuous velocity field over the entire domain. This velocity field can be used to backtrack dual vertices as well as transport particles or dyes (*e.g.*, for visualization purposes) with standard methods.

To see that such a vector, one for each dual vertex, is well defined consider the following argument. The flux on a face corresponds under duality (via the Hodge star) to a circulation along the dual edge of this face. Now, there is a linear relation between fluxes per tetrahedra due to the incompressibility condition (fluxes must sum to zero). This translates directly to a linear condition on the four circulations at each tetrahedra too. Thus, there is a unique vector (with three components) at the dual vertex whose projection along the dual edges is consistent with the observed circulations.

Relation to k-form Basis Functions The standard method to interpolate k-form data in a piecewise linear fashion over simplicial complexes is based on Whitney forms [Bossavit 1998]. In the case of primal 2-forms (fluxes) this results in a piecewise constant (per tetrahedra) velocity field. Our argument above, using a Voronoi cell based generalized barycentric interpolation of dual 1-forms (circulation), in fact *extends* the Whitney form machinery to the dual setting. This is a novel contribution, recently exploited in [Klingner et al. 2006], which may be useful as well in other computational applications of discrete forms. We note that the generalized barycentric coordinates have linear accuracy [Warren et al. 2007], an important requirement in many settings.

4.5 Handling Boundaries

The algorithm as described above does not constrain the boundaries, thus achieving "open" boundary conditions. No-transfer boundary conditions are easily imposed by setting the fluxes through the boundary triangles to zero. Non-zero flux boundary conditions (*i.e.*, forced fluxes through the boundary as in the case of Fig. 7) are more subtle. First, remark that all these boundary fluxes *must* sum to zero; otherwise, we would have little chance of getting a divergence-free fluid in the domain. Since the total divergence is zero, there exists a harmonic velocity field satisfying exactly these conditions. This is a consequence of the Helmholtz-Hodge decomposition theorem with normal boundary conditions [Chorin

and Marsden 1979]. Thus, this harmonic part **h** can be computed *once and for all* through a Poisson equation using the same setup as described in Appendix B. This precomputed velocity field allows us to deal very elegantly with these boundary conditions: we simply perform the same algorithm as we described by setting all boundary conditions to zero (with the exception of backtracking which takes the precomputed velocity into account), and *reinject* the harmonic part at the end of each time step (*i.e.*, add **h** to the current velocity field).

Viscous Fluids near Boundaries The Voronoi cells at the boundaries are slightly different from the usual, interior ones, since boundary vertices do not have a full 1-ring of tetrahedra. In the case of NS equations, this has no significant consequence: we set the velocity on the boundary to zero, resulting also in a zero circulation on the dual edges on the boundary. The rest of the algorithm can be used as is.

Inviscid Fluids near Boundaries For Euler equations, however, the tangential velocity at the boundary is not explicitly stored anywhere. Consequently, the boundary Voronoi faces need an additional variable to remedy this lack of information. We store in these dual faces the current integral vorticity. From this additional information given at time $t = 0$, we can *deduce* at each later time step the missing circulation on the boundary: since the circulation over the inside dual edges is known, and since the total integral must sum to the vorticity (Stokes' theorem), a simple substraction is all that is needed to update this missing circulation.

4.6 Handling Arbitrary Topology

Although the problem of arbitrary domain topology (*e.g.*, when the first Betti number is not zero) is rarely discussed in Computer Animation, it is important nonetheless. In the absence of external forces, the circulation along each loop (of winding number 1) around a tunnel is constant in time. So we can precalculate a *constant* harmonic field based on the initial circulation around each tunnel, and simply *add it* to the current velocity field for advection purposes. This is achieved in our implementation by first computing the cohomology basis (see [Desbrun et al. 2006; Tong et al. 2006]) of the fluid domain, then projecting the initial velocity field on it to compute the harmonic part; this purely harmonic part of the velocity field is then systematically added back to the velocity field computed by our algorithm. This procedure serves two purposes: first, we now automatically enforce the discrete equivalent of Kelvin's theorem on *any* (shrinkable or non-shrinkable) loop; second, arbitrary topologies are accurately and efficiently handled.

If external forces **f** are added, the purely harmonic part of the velocity field may change. We thus need to project these external forces onto the comology bases once again to extract the harmonic part: this part of the forces (times dt) is then added to the purely harmonic vector field, to be reinjected to the velocity field at each time step as usual.

5 Results and Discussion

We have tested our method on some of the usual "obstacle courses" in CFD. We start with the widely studied example of a flow past a disk (see Fig. 7). If initiated with zero vorticity, it is well known that in the case of an inviscid fluid, the flow remains irrotational for all times. By construction, our method does respect this physical behavior since circulation is preserved for the Euler equations. We then increase the viscosity of the fluid incrementally, and observe the formation of a vortex wake behind the obstacle, in agreement with physical experiments. As evidenced by the vorticity plots, vortices are shed from the boundary layer as a result of the adherence of the fluid to the obstacle, thanks to our proper treatment of the boundary conditions. A second test, now with a rotating obstacle in a fluid flow at high Reynolds number (Re$= 15,000$), shows good numerical agreement with high-resolution simulations obtained in [Shiels and Leonard 2001] using millions of vortex particles.

The behavior of vortex interactions observed in existing experimental results was also compared to numerical results based on our novel model and those obtained from the semi-Lagrangian advection method. It is known from theory that two like-signed vortices with a finite vorticity core will merge when their distance of separation is smaller than some critical value. This behavior is captured by the experimental data and shown in the first series of snapshots of Fig. 9. As the next row of snapshots indicates, the numerical results that our model generated present striking similarities to the experimental data. In the last row, we see that a traditional semi-Lagrangian advection followed by re-projection misses most of the fine structures of this phenomenon. This can be attributed to the loss of total integral vorticity as evidenced in the graph on the side; in comparison our technique preserves this integral exactly.

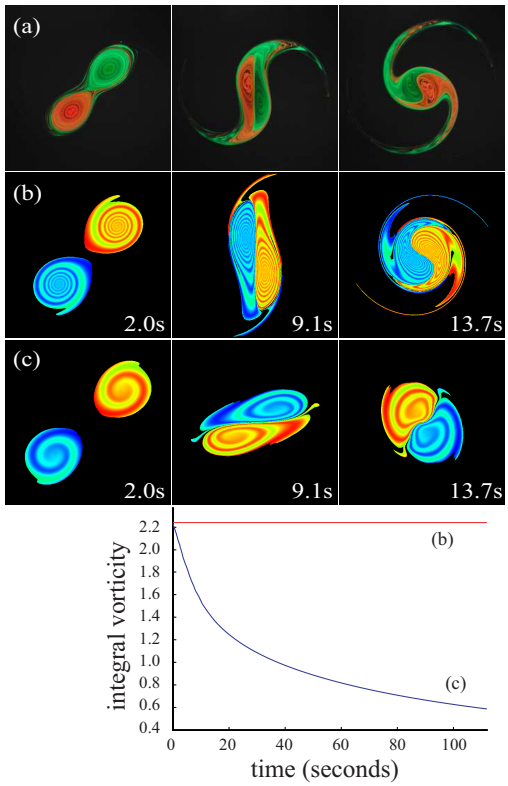


Figure 9: *Two Merging Vortices: (left) discrete fluid simulations are compared with a real life experiment (courtesy of Dr. Trieling, Eindhoven University; see* `http://www.fluid.tue.nl/WDY/vort/index.html`*) where two vortices (colored in red and green) merge slowly due to their interaction (a); while our method faithfully captures the merging phenomenon (b), a traditional semi-lagrangian scheme does not capture the correct motion because of vorticity damping (even with smaller time steps, because the more bactracking steps, the larger the energy loss) (c). (right) the integral vorticity of both simulation techniques are shown on a graph.*

We have also considered the flow on curved surfaces in 3D with complex topology, as depicted in Fig. 11. We were able to easily extend our implementation of two-dimensional flows to this curved case thanks to the intrinsic nature of our approach.

We also consider a smoke cloud surrounded by air, filling the body of a bunny as an example of flow in a domain with complex bound-

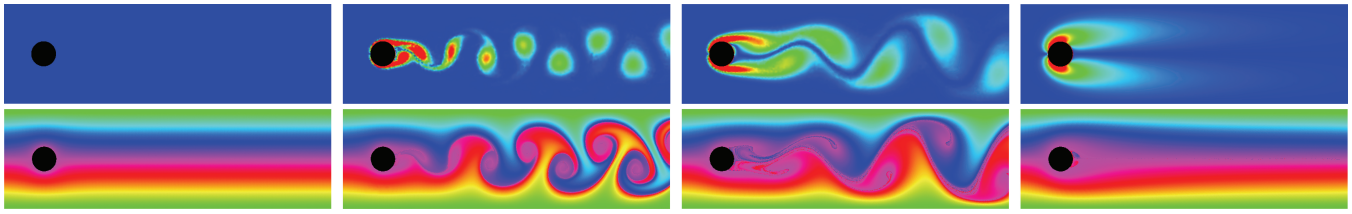


Figure 7: *Obstacle Course 1: in the usual experiment of a flow (going left to right) passing around a disk, the viscosity as well as the velocity can significantly affect the flow appearance; (top) our simulation results for increasing viscosity and same left-border boundary flux; the vorticity magnitude (shown in false colors) is shown; (bottom) same frames, where color dye has been passively advected with the flow for visualization purposes. Notice the usual irrotational flow (leftmost) for zero viscosity, and the von Karman vortex street when viscosity is introduced.*

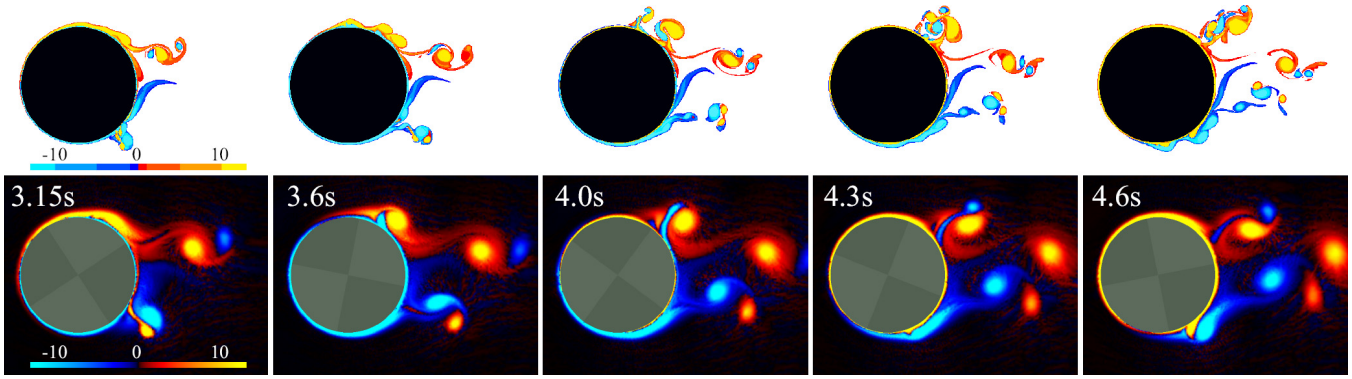


Figure 8: *Obstacle Course 2: Comparison with a very-high resolution viscous vortex particle method for the flow over an oscillating cylinder at high Reynolds number (Re= 15,000). The top row of images are color vorticity plots from [Shiels and Leonard 2001] (used with permission), while the bottom row shows our results at the exact same times.*

ary. The buoyancy drives the air flow which, in turn, advects the smoke cloud in the three-dimensional domain bounded by the bunny mesh as shown in Fig. 1. Note that this domain is made out of $7K$ vertices, which is the equivalent complexity (in terms of the number of degrees of freedom) of a regular grid of size 19^3. Consequently, it took *less than half a second per frame* to compute, exemplifying the advantage of using tet meshes to resolve fine boundaries since the equivalent low-resolution regular grid would not nearly be able to capture the complex geometry of the bunny mesh.

In the last simulation, we use a snow globe with a bunny inside (see Fig. 10). We emulate a flow due to an initial spin of the globe using a swirl described as a vorticity field. The snow particles are (passively) transported by the flow as they fall down under the effect of gravity. Here again, each frame is generated in less than half a second.

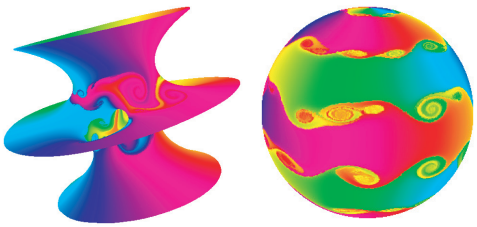


Figure 11: *Weather System on Planet Funky: the intrinsic nature of the variables used in our algorithm makes it amenable to the simulation of flows on arbitrary curved surfaces.*

6 Conclusion

In this paper, we have introduced a novel theoretical approach to fluid dynamics, along with a practical implementation and various simulation results. We have carefully discretized the physics of flows to respect the most fundamental geometric structures that characterize their behavior. Among the several specific benefits that we demonstrated, the most important is the circulation preservation property of the integration scheme, as evidenced by our numerical examples. The discrete quantities we used are intrinsic, allowing us to go to curved manifolds with no additional complication. Finally, the machinery employed in our approach can be used on any simplicial complex. However, the same methodology also applies directly to more general spatial partitions, and in particular, to regular grids and hybrid meshes [Feldman et al. 2005]—rendering our approach widely applicable to existing fluid simulators.

For future work, a rigorous comparison of the current method with standard approaches should be undertaken. Using Bjerknes' circulation theorem for compressible flows may also be an interesting avenue. Additionally, we limited ourselves to the investigation of our scheme without focusing on the separate issue of order of accuracy. Coming up with an integration scheme that is higher-order accurate will be the object of further investigation, as it requires a better (denser) Hodge star; in particular, it could reduce the diffusion of vorticity introduced during the vorticity backtracking step. Note also that our geometric approach bears interesting similarities with the work of Chang *et al.* [2002], in which they propose a purely algebraic approach to remedy the shortcomings of the traditional fractional step approach; using their numerical analysis on our approach may provide a simple way to study the accuracy of our scheme. Finally, mixing our method with the Lagrangian treatment of vortex rings as in [Weißmann 2006] could allow for intuitive control of the fluid motion.

Acknowledgments The authors wish to thank Jerrold E. Marsden and Anthony Leonard for their enthusiastic help and support. This work was partially supported by the NSF (DMS-0453145, CCF-0503786, CCF-0528101, ACI-0219979, CCR-

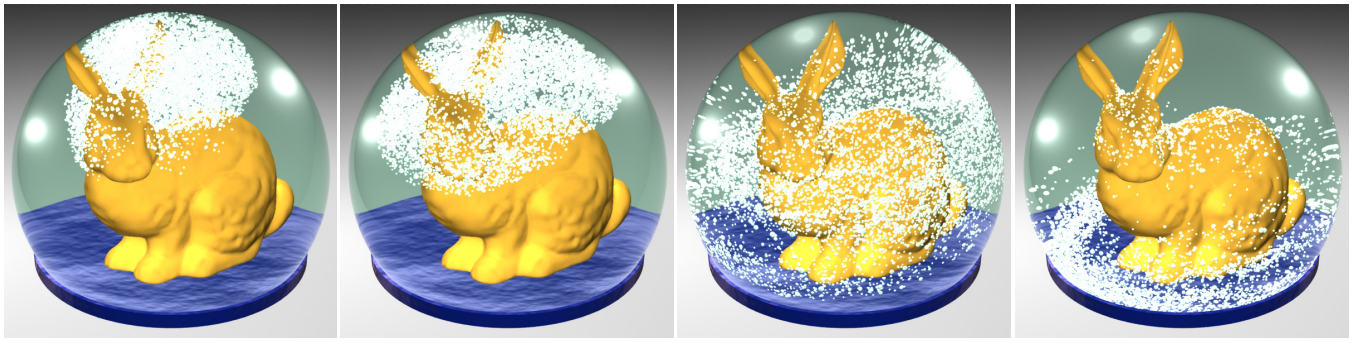


Figure 10: *Bunny Snow Globe: the snow in the globe is advected by the inner fluid, initially stirred by a vortex to simulate a spin of the globe.*

0133983, CMMI-0757106), the DOE (DE-FG02-04ER25657, W-7405-ENG-48/B341492), the Center for Integrative Multiscale Modeling and Simulation at Caltech, the Okawa Foundation, the Irvine Foundation, the Center for the Mathematics of Information, Autodesk, and Pixar Animation Studios.

References

ABRAHAM, R., MARSDEN, J., AND RATIU, T., Eds. 1988. *Manifolds, Tensor Analysis, and Applications*, vol. 75 of *Applied Mathematical Sciences*. Springer.

ALLIEZ, P., COHEN-STEINER, D., YVINEC, M., AND DESBRUN, M. 2005. Variational tetrahedral meshing. *ACM Transactions on Graphics 24*, 3, 617–625.

ANGELIDIS, A., NEYRET, F., SCHPOK, J., DWYER, W. T., AND EBERT, D. S. 2005. Modeling and animating gases with simulation features. In *ACM/Eurographics Symposium on Computer Animation*, 97–106.

ANGELIDIS, A., NEYRET, F., SINGH, K., AND NOWROUZEZAHRAI, D. 2006. A controllable, fast and stable basis for vortex based smoke simulation. In *ACM/EG Symposium on Computer Animation*, 25–32.

BOCHEV, P. B., AND HYMAN, J. M. 2006. Principles of mimetic discretizations of differential operators. *Compatible Spatial Discretization Series /IMA Vol. 142 in Mathematics and Applications*, 89–120.

BOSSAVIT, A., AND KETTUNEN, L. 1999. Yee-like schemes on a tetrahedral mesh. *Int. J. Num. Modelling: Electr. Networks, Dev. and Fields 12* (July), 129–142.

BOSSAVIT, A. 1998. *Computational Electromagnetism*. Academic Press, Boston.

CHANG, W., GIRALDO, F., AND PEROT, B. 2002. Analysis of an exact fractional step method. *Journal of Computational Physics 180*, 3 (Nov.), 183–199.

CHORIN, A., AND MARSDEN, J. 1979. *A Mathematical Introduction to Fluid Mechanics*, 3rd edition ed. Springer-Verlag.

DESBRUN, M., KANSO, E., AND TONG, Y. 2006. Discrete differential forms for computational sciences. In *Discrete Differential Geometry*, E. Grinspun, P. Schröder, and M. Desbrun, Eds., Course Notes. ACM SIGGRAPH.

E, W., AND LIU, J.-G. 1996. Finite difference schemes for incompressible flows in vorticity formulations. 181–195.

E, W., AND LIU, J.-G. 1996. Vorticity boundary condition and related issues for finite difference schemes. *J. Comput. Phys. 124*, 2, 368–382.

ELCOTT, S., AND SCHRÖDER, P. 2006. Building your own dec at home. In *Discrete Differential Geometry*, E. Grinspun, P. Schröder, and M. Desbrun, Eds., Course Notes. ACM SIGGRAPH.

FEDKIW, R., STAM, J., AND JENSEN, H. W. 2001. Visual simulation of smoke. In *Proceedings of ACM SIGGRAPH*, Computer Graphics Proceedings, Annual Conference Series, 15–22.

FELDMAN, B. E., O'BRIEN, J. F., AND KLINGNER, B. M. 2005. Animating gases with hybrid meshes. *ACM Transactions on Graphics 24*, 3, 904–909.

FOSTER, N., AND FEDKIW, R. 2001. Practical animation of liquids. In *Proceedings of ACM SIGGRAPH*, Computer Graphics Proceedings, Annual Conference Series, 23–30.

FOSTER, N., AND METAXAS, D. 1997. Modeling the motion of a hot, turbulent gas. In *Proceedings of SIGGRAPH*, Computer Graphics Proceedings, Annual Conference Series, 181–188.

GAMITO, M. N., LOPES, P. F., AND GOMES, M. R. 1995. Two-dimensional simulation of gaseous phenomena using vortex particles. In *Proceedings of the 6th Eurographics Workshop on Computer Animation and Simulation*, 3–15.

GOKTEKIN, T. G., BARGTEIL, A. W., AND O'BRIEN, J. F. 2004. A method for animating viscoelastic fluids. *ACM Transactions on Graphics 23*, 3 (Aug.), 463–468.

GUENDELMAN, E., SELLE, A., LOSASSO, F., AND FEDKIW, R. 2005. Coupling water and smoke to thin deformable and rigid shell. *ACM Transactions on Graphics 24*, 3, 973–981.

HIRANI, A. 2003. *Discrete Exterior Calculus*. PhD thesis, California Institute of Technology.

KLINGNER, B. M., FELDMAN, B. E., CHENTANEZ, N., AND O'BRIEN, J. F. 2006. Fluid animation with dynamic meshes. *ACM Transactions on Graphics (SIGGRAPH)* (Aug.).

LANGTANGEN, H.-P., MARDAL, K.-A., AND WINTER, R. 2002. Numerical methods for incompressible viscous flow. *Advances in Water Resources 25*, 8-12 (Aug-Dec), 1125–1146.

LOSASSO, F., GIBOU, F., AND FEDKIW, R. 2004. Simulating Water and Smoke with an Octree Data Structure. *ACM Transactions on Graphics 23*, 3 (Aug.), 457–462.

MARSDEN, J. E., AND WENSTEIN, A. 1983. Coadjoint orbits, vortices and Clebsch variables for incompressible fluids. *Physica D 7*, 305–323.

MARSDEN, J. E., AND WEST, M. 2001. Discrete mechanics and variational integrators. *Acta Numerica 10*, 357–515.

MCNAMARA, A., TREUILLE, A., POPOVIC, Z., AND STAM, J. 2004. Fluid control using the adjoint method. *ACM Transactions on Graphics 23*, 3 (Aug.), 449–456.

MUNKRES, J. R. 1984. *Elements of Algebraic Topology*. Addison-Wesley.

NICOLAIDES, R. A., AND WU, X. 1997. Covolume solutions of three-dimensional div-curl equations. 2195–2203.

PARK, S. I., AND KIM, M. J. 2005. Vortex Fluid for Gaseous Phenomena. In *ACM/EG Symposium on Computer Animation*, 261–270.

PIGHIN, F., COHEN, J. M., AND SHAH, M. 2004. Modeling and Editing Flows using Advected Radial Basis Functions. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 223–232.

SELLE, A., RASMUSSEN, N., AND FEDKIW, R. 2005. A vortex particle method for smoke, water and explosions. *ACM Transactions on Graphics 24*, 3 (Aug.), 910–914.

SHI, L., AND YU, Y. 2004. Inviscid and Incompressible Fluid Simulation on Triangle Meshes. *Journal of Computer Animation and Virtual Worlds 15*, 3-4 (June), 173–181.

SHI, L., AND YU, Y. 2005. Controllable smoke animation with guiding objects. *ACM Transactions on Graphics 24*, 1, 140–164.

SHIELS, D., AND LEONARD, A. 2001. Investigation of a drag reduction on a circular cylinder in rotary oscillation. *J. of Fluid Mech. 431* (Mar.), 297–322.

STAM, J. 1999. Stable fluids. In *Proceedings of ACM SIGGRAPH*, Computer Graphics Proceedings, Annual Conference Series, 121–128.

STAM, J. 2001. A simple fluid solver based on the fft. *Journal of Graphics Tools 6*, 2, 43–52.

STAM, J. 2003. Flows on surfaces of arbitrary topology. *ACM Transactions on Graphics 22*, 3 (July), 724–731.

STEINHOFF, J., AND UNDERHILL, D. 1994. Modification of the euler equations for *Vorticity Confinement*: Applications to the computation of interacting vortex rings. *Physics of Fluids 6*, 8 (Aug.), 2738–2744.

TONG, Y., LOMBEYDA, S., HIRANI, A. N., AND DESBRUN, M. 2003. Discrete multiscale vector field decomposition. *ACM Transactions on Graphics 22*, 3, 445–452.

TONG, Y., ALLIEZ, P., COHEN-STEINER, D., AND DESBRUN, M. 2006. Designing Quadrangulations with Discrete Harmonic Forms. In *ACM/EG Symposium on Geometry Processing*, 201–210.

TREUILLE, A., MCNAMARA, A., POPOVIĆ, Z., AND STAM, J. 2003. Keyframe control of smoke simulations. *ACM Transactions on Graphics 22*, 3 (July), 716–723.

WARREN, J., SCHAEFER, S., HIRANI, A., AND DESBRUN, M. 2007. Barycentric coordinates for convex sets. *Advances in Computational Mathematics (to appear)*.

WEISSMANN, S. 2006. *Real Time Simulation of Fluid Flow*. Master's thesis, Technische Universität Berlin.

WENDT, J., BAXTER, W., OGUZ, I., AND LIN, M. 2005. Finite Volume Flow Simulations on Arbitrary Domains. Tech. Rep. TR05-019, UNC-CH Computer Science, Sept.

YAEGER, L., UPSON, C., AND MYERS, R. 1986. Combining physical and visual simulation - creation of the planet jupiter for the film 2010. *Computer Graphics (Proceedings of ACM SIGGRAPH) 20*, 4, 85–93.

A Discrete Exterior Derivative

A thorough explanation of the discrete exterior derivative of discrete forms is out of the scope of this paper, and we refer the reader to existing tutorials [Desbrun et al. 2006; Bochev and Hyman 2006]. However, the reader should be aware that this operator is simply implemented via the use of *incidence matrices*. Indeed, a key ingredient to defining the discrete version of the exterior derivative d is Stokes' theorem:

$$\int_\sigma d\alpha = \int_{\partial\sigma} \alpha,$$

where σ denotes a $(k+1)$-cell and α is a k-form. Stokes' theorem states that the integral of $d\alpha$ (a $(k+1)$-form) over a $(k+1)$-cell equals the integral of the k-form α over the *boundary* of the $(k+1)$-cell (*i.e.*, a sum of k-cells). Stokes' theorem can thus be used as a way to *define* the d operator in terms of the boundary operator ∂. Or, said differently, once we have the boundary operator, the operator d follows immediately if we wish Stokes' theorem to hold on the simplicial complex.

To use a very simple example, consider a 0-form f, *i.e.*, a function giving values at vertices. With that, df is a 1-form which can be integrated along an edge (say with end points denoted a and b) and Stokes' theorem states the well known fact:

$$\int_{[a,b]} df = f(b) - f(a).$$

The right hand side is simply the evaluation of the 0-form f on the boundary of the edge, *i.e.*, its endpoints (with appropriate signs indicating the orientation of the edge). Actually, one can define a hierarchy of these operators that mimic the operators given in the continuous setting (up to an application of the Hodge star) by the gradient (∇), curl ($\nabla\times$), and divergence ($\nabla\cdot$), namely,

- d_0: maps 0-forms to 1-forms and corresponds to the **Gradient**;
- d_1: maps 1-forms (values on edges) to 2-forms (values on faces). The value on a given face is simply the sum (by linearity of the integral) of the 1-form values on the boundary (edges) of the face with the signs chosen according to the local orientation. d_1 corresponds to the **Curl**;
- d_2: maps 2-forms to 3-forms and corresponds to the **Divergence**.

Since the boundary of any mesh element can be directly read from the incidence matrices of the mesh, the exterior derivative is trivial to implement once the mesh is known as it depends only on its connectivity [Elcott and Schröder 2006].

B Recovery of Velocity

We have seen in Section 4 how the vorticity ω can be derived directly from the set of all face fluxes as $d^T \star U = \omega$. However, during the simulation, we will also need to recover flux *from* vorticity. For this we employ the Helmholtz-Hodge decomposition theorem, stating that any vector field $\mathbf{u}$ can be decomposed into three components (given appropriate boundary conditions)

$$\mathbf{u} = \nabla \times \boldsymbol{\phi} + \nabla\psi + \mathbf{h}.$$

When represented in terms of discrete forms this reads as:

$$U = d\Phi + \star^{-1} d^T \star \Psi + H \tag{7}$$

For the case of incompressible fluids (*i.e.*, with zero divergence), two of the three components are sufficient to describe the velocity field: the curl of a vector potential and a harmonic field. To see this apply d to both sides of Eq. 7:

$$dU = 0 = dd\Phi + d\star^{-1} d^T \star \Psi + dH.$$

Since $dd = 0$ and d of a harmonic form always vanishes, we find that $d\star^{-1} d^T \star \Psi = 0$ to begin with. Since Ψ is a 3-form $d\star^{-1} d^T \star \Psi = \Delta\Psi = 0$, *i.e.*, Ψ is harmonic which implies in particular that $\star^{-1} d^T \star \Psi = 0$, proving our claim that

$$U = d\Phi + H.$$

If the topology of the domain is trivial, we can furthermore ignore the harmonic part H (we discuss a full treatment of arbitrary topology in Section 4.6), leaving us with $U = d\Phi$.

Since our algorithm computes an updated Ω which is related to U as $d^T \star U$, we need to find a solution to

$$\Omega = d^T \star d\Phi,$$

where Ω is the known quantity, and $d\Phi$ the unknown. Unfortunately the kernel of $d^T \star d$ is not empty so we can not determine Φ directly from this equation. To pick a unique solution for Φ, we require additionally that $d^T \star \Phi = 0$. By doing so we pick a *particular* solution from the kernel of d^T. But if $d^T \star \Phi = 0$ then certainly $(\star d \star^{-1} d^T \star)\Phi = 0$ and we can add it to our equation for Ω arriving at

$$\Omega = (d^T \star d + \star d \star^{-1} d^T \star)\Phi. \tag{8}$$

This latter equation is simply a Poisson equation for Φ since

$$\star^{-1}\Omega = \Delta\Phi$$

which has a unique solution. Let $U = d\Phi$, and we have recovered U as desired.

The fact that Eq. 8 is indeed a Poisson problem follows from the definition of the Laplacian Δ in *differential calculus* as $\star^{-1} d^T \star d + d \star^{-1} d^T \star$. In the language of vector calculus this translates to $\Delta\boldsymbol{\phi} = (\nabla\nabla\cdot - \nabla\times\nabla\times)\boldsymbol{\phi} = \nabla\times\mathbf{u}$. Notice that the left-side matrix (that we will denote L) is *symmetric and sparse*, thus ideally suited for fast numerical solvers. Our linear operators (and, in particular, the discrete Laplacian) differ from another discrete Poisson setup on simplicial complexes proposed in [Tong et al. 2003]: the ones we use have smaller support, which results in sparser and better conditioned linear systems [Bossavit 1998]—an attractive feature in the context of numerical simulation.

Chapter 10:
Conformal Equivalence of Triangle Meshes

Boris Springborn
TU Berlin

Peter Schröder
Caltech

Ulrich Pinkall
TU Berlin

Abstract

We present a new algorithm for conformal mesh parameterization. It is based on a precise notion of *discrete conformal equivalence* for triangle meshes which mimics the notion of conformal equivalence for smooth surfaces. The problem of finding a flat mesh that is discretely conformally equivalent to a given mesh can be solved efficiently by minimizing a convex energy function, whose Hessian turns out to be the well known cot-Laplace operator. This method can also be used to map a surface mesh to a parameter domain which is flat except for isolated cone singularities, and we show how these can be placed automatically in order to reduce the distortion of the parameterization. We present the salient features of the theory and elaborate the algorithms with a number of examples.

Keywords: Discrete Differential Geometry; conformal parameterization; conformal equivalence; discrete Riemannian metric; cone singularities; texture mapping

1 Introduction

In this paper we present a definition for discrete conformal equivalence of triangle meshes and apply it to the problem of conformal mesh parameterization. Our approach arises from two basic premises.

First, the discrete setting should parallel the smooth setting. There the problem of finding a conformal parameterization for a *smooth* surface in space is equivalent to finding a flat metric on the surface that is conformally equivalent to the metric induced by the embedding. Hence, we cast the parameterization problem as one of finding *conformally equivalent flat metrics for a given metric*.

Second, we treat the parameterization problem for *triangle meshes* in the spirit of *Discrete Differential Geometry* [Bobenko and Suris 2005]: instead of viewing discretization as a means of making the smooth problem amenable to numerical methods, we seek to develop on the discrete level a geometric theory that is as rich as the analogous theory for the smooth problem. *The aim is to discretize the whole theory, not just the equations*.

Instead of asking for an *approximation* of the smooth problem, we are thus guided by questions like: What corresponds to a Riemannian metric and Gaussian curvature in an analogous theory for triangle meshes? When should two triangle meshes be considered conformally equivalent? We answer the first question in the obvious way: the edge lengths and the angle defects at vertices. The answer that we give to the second question is also straightforward and intuitive. The striking point is that once we have thus fixed the fundamental definitions of the discrete theory, there is an efficient algorithm to solve the discrete conformal parameterization problem.

Contributions In this paper we present the first algorithm for mesh parameterization which is based on a definition of *discrete conformal equivalence between triangle meshes* which is satisfactory in the sense that (a) it depends only on the geometry of the meshes and (b) defines an equivalence relation. Meshes in a conformal equivalence class are characterized by length scale factors associated to the vertices and by conserved quantities: the length cross ratios. This discrete notion of conformality comes with a comprehensive theory carrying many of the hallmarks of the smooth setting. Most importantly for graphics applications, we present effective computational procedures based on minimizing a convex energy. The triangulated surfaces may be of arbitrary topology with or without boundary and no a priori cutting is required to deal with higher genus, for example. Flexible boundary conditions support a range of options from full control over boundary curvature to full control over length distortion, including isometrically mapped boundaries. The theory and algorithms also admit cone singularities, which may be placed by the user or automatically to reduce length distortion and flatten higher genus meshes. Last but not least, our discretely conformal parameterizations admit a piecewise projective interpolation scheme for texture coordinates, which yields improved interpolation at no additional cost.

Figure 1: *Customers who visited our Tattoo parlor, where repeating patterns can be applied seamlessly and length distortion is kept low through cone singularities. (Insets show texture tiles.)*

2 Conformal Equivalence

Our definitions are based on a fairly direct "translation" of classic notions to the discrete setting of meshes. In smooth differential geometry two Riemannian metrics g and $\tilde{g}$ on a differentiable 2-manifold M are said to be *conformally equivalent* if

$$\tilde{g} = e^{2u} g \tag{1}$$

for some smooth function $u : M \to \mathbb{R}$, which gives the logarithm of length change between g and $\tilde{g}$.

In the discrete setting M becomes an abstract surface triangulation consisting of vertices, edges, and triangles, $M = (V, E, T)$. We do not restrict its topology or whether it possesses a boundary. While the combinatorics of a mesh are encoded in such an abstract triangulation, its intrinsic geometry is encoded in the edge lengths. Thus we define:

Definition 2.1. A *discrete metric* on M is a function l on the set of edges E, assigning to each edge e_{ij} a positive number l_{ij} so that the triangle inequalities are satisfied for all triangles $t_{ijk} \in T$.

The smooth function u becomes a function on the set of vertices and we define discrete conformal equivalence as:

Definition 2.2. Two discrete metrics l and $\tilde{l}$ on M are *(discretely) conformally equivalent* if, for some assignment of numbers u_i to the vertices v_i, the metrics are related by

$$\tilde{l}_{ij} = e^{(u_i+u_j)/2} l_{ij}. \tag{2}$$

This notion of discrete conformality also appears in [Luo 2004].

It will turn out to be convenient to use the *logarithmic lengths* $\lambda_{ij} := 2 \log l_{ij}$, turning Eq. (2) into the linear relation

$$\tilde{\lambda}_{ij} = \lambda_{ij} + u_i + u_j. \tag{3}$$

Note that this notion of discrete conformal equivalence is indeed an equivalence relation (*i.e.*, it is reflexive, symmetric, and transitive) on the set of discrete metrics on M, and also on the set of meshes. We consider two *meshes* as discretely conformally equivalent if they have the same abstract triangulation and equivalent edge lengths according to Eq. (2).

The primary motivation for this definition is of course the obvious analogy with the smooth setting. Another is that it behaves correctly under Möbius transformations of space: applying a Möbius transformation to the vertices of a mesh, the resulting mesh is discretely conformally equivalent to the untransformed mesh. Möbius transformations—compositions of inversions in spheres—are the only (smooth) conformal transformations of space. Thus, in particular, two meshes whose vertices are related by a smooth conformal transformation of space are also discretely conformally equivalent. (To verify this claim, note that the distance between two points is related to the distance between their image points under a Möbius transformation f by $\|f(x) - f(y)\| = \rho(x)\rho(y)\|x - y\|$ for a real valued function ρ. This is obvious for translations, rotations and scaling, and a straightforward calculation shows that it is true also for inversions in a sphere.) As a practical consequence, this Möbius invariance of discrete conformality enables us to map meshes not only to the plane but also to the sphere.

Discrete conformal equivalence can also be characterized in terms of conserved quantities, namely the length cross ratios:

Definition 2.3. Given a discrete metric l, we associate with each interior edge e_{ij} (between t_{ijk} and t_{jim}) the *length cross ratio*

$$\mathfrak{c}_{ij} := l_{im}/l_{mj} \cdot l_{jk}/l_{ki}. \tag{4}$$

Proposition. *Two meshes are discretely conformally equivalent if and only if their length cross ratios are the same.*

Proof. In one direction, the implication is obvious: if the meshes are discretely conformally equivalent, then their length cross ratios are equal because the scale factors $e^{u/2}$ cancel. To see the converse, consider two discrete metrics l and $\tilde{l}$. We have to show that if their length cross ratios $\mathfrak{c}$ and $\tilde{\mathfrak{c}}$ are equal, then we can associate numbers u_i to the vertices v_i satisfying Eqs. (2) (one equation for each edge). In general, this system of equations for the u_i is overdetermined. For each triangle t_{ijk} the three equations corresponding to its edges already determine values for u at its vertices:

$$e^{u_i} = \tilde{l}_{ij}/l_{ij} \cdot \tilde{l}_{ki}/l_{ki} \cdot l_{jk}/\tilde{l}_{jk}. \tag{5}$$

Considering the neighboring triangle t_{jim} one obtains the same value for e^{u_i} precisely if $\mathfrak{c}_{ij} = \tilde{\mathfrak{c}}_{ij}$. Hence, if l and $\tilde{l}$ have identical length cross ratios, values for u at the vertices are consistently determined by Eqs. (2). □

The discrete theory we have set up now informs our consequent approach. Given a mesh M and a discrete metric l on it, we consider the equivalence class of conformally equivalent metrics. In this class we look for a metric $\tilde{l}$ which is flat. Only in a subsequent step (Section 3.3) do we construct vertex coordinates which realize this flat metric. To find the desired flat metric we need the relation between length and curvature.

2.1 From Curvatures to Lengths

Given a triangle $t_{ijk} \in T$ and lengths l_{ij}, l_{jk}, l_{ki} satisfying the triangle inequality, the angle α^i_{jk} at vertex v_i opposite edge e_{jk} can be recovered by using the cosine formula or the half-angle formula:

$$\alpha^i_{jk} = 2 \tan^{-1} \sqrt{\frac{(l_{ij}+l_{jk}-l_{ki})(l_{jk}+l_{ki}-l_{ij})}{(l_{ki}+l_{ij}-l_{jk})(l_{jk}+l_{ki}+l_{ij})}} \tag{6}$$

(and similarly for $\tilde{\alpha}$ as a function of $\tilde{l}$). Given the angles, curvatures follow as the excess angle sums at vertices. The angle sum at a vertex v_i is denoted by Θ_i:

$$\Theta_i = \sum_{t_{ijk} \ni v_i} \alpha^i_{jk}$$

(and similarly for $\widetilde{\Theta}$). The curvature at an interior vertex is $K_i = 2\pi - \Theta_i$ and the boundary curvature at a boundary vertex is $\kappa_i = \pi - \Theta_i$ (and similarly for $\widetilde{K}$, $\tilde{\kappa}$).

Our goal now is to find a new metric, which is conformally equivalent to a given one, and which possesses prescribed curvatures. Thus, we have to solve the following problem:

Problem. *Given*, for each vertex, a desired angle sum $\widehat{\Theta}_i$, *find* u_i such that the new metric $\tilde{l}$ has angle sums $\widetilde{\Theta}_i = \widehat{\Theta}_i$.

For instance, if we prescribe $\widetilde{\Theta}_i = 2\pi$ at interior vertices, *i.e.*, $\widetilde{K}_i = 0$, a solution to our problem will give a conformally equivalent flat metric with prescribed boundary curvatures $\tilde{\kappa}_i$.

The following conditions on the prescribed angle sums are necessary for the existence of a solution. They must satisfy $0 < \widehat{\Theta} < \pi \cdot |\{t_{ijk} \ni v_i\}|$ because the triangle angles are between 0 and π, and they must be compatible with the Gauss-Bonnet formula

$$\sum_{v_i \in M \setminus \partial M} K_i + \sum_{v_i \in \partial M} \kappa_i = 2\pi\chi(M),$$

where $\chi(M) = |T| - |E| + |V|$ is the Euler characteristic of M.

In view of Eqs. (2) and (6), the problem is equivalent to solving a set of $n := |V|$ equations in n variables u_i. Since the $\widetilde{\Theta}_i$ automatically satisfy one linear relation (Gauss-Bonnet) and the angles are scale invariant, we really have $n - 1$ conditions for $n - 1$ essential degrees of freedom. We fix the scaling ambiguity by requiring $\sum_{v_i} u_i = 0$.

Luckily, if a solution to this set of non-linear equations exists, it can be found as the *unique minimizer of a convex energy* in u

$$E(u) = \sum_{t_{ijk} \in T} \left(f(\tilde{\lambda}_{ij}, \tilde{\lambda}_{jk}, \tilde{\lambda}_{ki}) - \tfrac{\pi}{2}(u_i + u_j + u_k) \right) + \tfrac{1}{2} \sum_{v_i \in V} \widehat{\Theta}_i u_i \quad (7)$$

where

$$f(\tilde{\lambda}_{ij}, \tilde{\lambda}_{jk}, \tilde{\lambda}_{ki}) =$$
$$\tfrac{1}{2} \left(\tilde{\alpha}^i_{jk} \tilde{\lambda}_{jk} + \tilde{\alpha}^j_{ki} \tilde{\lambda}_{ki} + \tilde{\alpha}^k_{ij} \tilde{\lambda}_{ij} \right) + \text{Л}(\tilde{\alpha}^i_{jk}) + \text{Л}(\tilde{\alpha}^j_{ki}) + \text{Л}(\tilde{\alpha}^k_{ij}), \quad (8)$$

with $\tilde{\lambda}$ as in Eq. (3); Л(.) denoting Milnor's Lobachevsky function (see Appendix A); and both $\tilde{\lambda}$ and $\tilde{\alpha}$ depending on u.

Indeed, the partial derivatives of this energy are

$$\partial_{u_i} E = \tfrac{1}{2} \left(\widehat{\Theta}_i - \textstyle\sum_{t_{ijk} \ni v_i} \tilde{\alpha}^i_{jk} \right), \quad (9)$$

so that $\operatorname{grad} E(u) = 0$ *iff* u solves the problem. The Hessian of the energy turns out to be one half of the well-known cot-Laplace operator:

$$(\operatorname{Hess} E \cdot \delta u)_i = \tfrac{1}{2} (\Delta \delta u)_i = \tfrac{1}{4} \textstyle\sum_{e_{ij} \ni v_i} w_{ij} (\delta u_i - \delta u_j), \quad (10)$$

with $w_{ij} = \cot \tilde{\alpha}^k_{ij} + \cot \tilde{\alpha}^l_{ij}$ for interior edges and only one cot term for boundary edges. (We adopt the sign convention for the Laplace operator which renders it positive semi-definite. See Appendix B and C for proofs of Eqs. (9) and (10).)

The Hessian is therefore positive semi-definite with only the constant functions in its null-space. This corresponds to the fact that the energy $E(u)$ is scale invariant, *i.e.*, its value does not change if the same number is added to all u_i.

2.2 Relation to Other Approaches

There are many algorithms for mesh parameterization in the literature and we will not attempt a comprehensive review here (the interested reader is referred to [Floater and Hormann 2005] and [Sheffer et al. 2006]). Instead we focus on approaches which are based on *discretized* or *discrete* notions of harmonicity and conformality.

Discrete Harmonicity Discrete versions of the theory of harmonic and analytic functions were developed as early as the 1940s and '50s [Duffin 1956], based on simple difference equations analogous to Laplace's equation and the Cauchy-Riemann equations. Indeed, the discovery of the cot-Laplace operator heralds from this time [Duffin 1959], and these ideas still inform contemporary notions of discrete conformality and harmonicity that are based on linear conditions on the vertex coordinates. Examples of theory and applications include [Pinkall and Polthier 1993; Mercat 2001; Desbrun et al. 2002; Lévy et al. 2002; Gu and Yau 2003; Tong et al. 2007].

The linear theory of discrete harmonicity is interesting and rich, attractive computationally, and enormously useful in applications. But the implied notion of a discretely conformal map as a pair of conjugate discretely harmonic functions is deficient because the inverse of such a map, and the composition of two such maps, are no longer discretely conformal. Thus, this notion of a discretely conformal map does not define a notion of discrete conformal equivalence.

Circle Patterns Going back to an unpublished idea of Thurston (see [Stephenson 2003] for an eyewitness account), discrete conformality can be approached through circle patterns. Here meshes are considered together with a system of circles associated to faces [Kharevych et al. 2006] or vertices [Bowers and Hurdal 2003; Stephenson 2005; Jin et al. 2007; Yang et al. 2008]. Two such meshes are considered conformally equivalent, if the intersection angles of the circles—or inversive distances for non-intersecting circles—are equal in both meshes.

These methods require the solution of non-linear equations, using a convex variational principle, a specially tailored relaxation procedure, or a flow.

This approach to discrete conformality is also not fully satisfactory. In the case of Kharevych *et al.*, the intersection angles determine the curvature of the mesh. Consequently, they must change when the mesh is flattened. For the other circle methods, the notion of discrete conformality either depends on an arbitrary choice of circles, or it is defined only if the edge-lengths satisfy some further conditions, or both.

Curvature Flow Consider u as a function of time and evolving under the negative gradient of E

$$\partial_t u = -2 \operatorname{grad} E = (\widehat{K} - \widetilde{K}).$$

(For simplicity, assume that M has no boundary.) We may then think of $u^* = \operatorname{argmin}_u E(u)$—which we compute using Newton's method—as the steady state solution of this curvature flow with given target curvatures. Using $\partial_t \widetilde{K} = \partial_u \widetilde{K} \cdot \partial_t u$ together with Eq. (10) we also find that the evolution of the curvature is governed by the equation

$$\partial_t \widetilde{K} = \widetilde{\Delta}(\widehat{K} - \widetilde{K})$$

with $\widetilde{\Delta}$ denoting the (positive semi-definite) cot-Laplace operator of the discrete metric induced by $u(t)$.

In the case where $\widehat{K} = 0$, this flow was considered by Luo [2004] as a discrete version of Yamabe flow. He proves that the flow is variational, but he gives no formula for the energy and does not seem to be aware that the cot-Laplace operator appears in the evolution equation for $\widetilde{K}$. A different discrete curvature flow forms the foundation of the approach of Jin *et al.* [2007] who use Chow and Luo's definition of discrete Ricci flow [2003]. (For 2-dimensional Riemannian manifolds, Yamabe flow is the same as Ricci flow.) Our explicitly variational formulation provides a new approach to discrete Ricci flow for surfaces.

Angle Based Flattening Smooth conformal maps preserve angles exactly. Enforcing this property as-best-as-possible in the discrete setting forms the foundation for Angle Based Flattening (ABF) [Sheffer et al. 2005]. The resulting optimization problem for the angles of the flat mesh is non-linear with non-convex non-linear constraints. It requires sophisticated numerical methods [Sheffer et al. 2005] (for some recent progress see [Zayer et al. 2007]), and is not solved to high enough accuracy to actually produce a flat metric. Thus a further approximation step is necessary to produce the final coordinate functions.

Metric Scaling There is one very recent approach that is closely related to ours [Ben-Chen et al. 2008]. Just as we do, they first compute a metric for the image mesh and only then a set of vertex positions. They start with the following well known relation. Given a smooth 2-manifold and two metrics g, $\tilde{g}$—related through the conformal factor e^{2u} as in Eq. (1)—the curvatures K, $\widetilde{K}$ satisfy:

$$e^{2u} \widetilde{K} = K + \Delta u, \quad (11)$$

where Δ is the—positive semi-definite—Laplace-Beltrami operator. The metric $\tilde{g}$ is flat ($\widetilde{K} = 0$), if u solves the Poisson equation

$$\Delta u = -K. \quad (12)$$

To flatten a mesh, Ben-Chen *et al.* proceed by solving the discretized Poisson equation with Δ the cot-Laplace operator in the

original metric and K_i the angle defect at a vertex. This is precisely the first Newton step when minimizing our energy $E(u)$. (Their definition of length change induced by the u_i is different from Eq. (2) but this is immaterial to the present argument.) As Ben-Chen *et al.* point out, this does not in general yield a flat metric. They address this by solving a cot-Laplace layout problem in the new metric to find flat vertex positions which approximate the desired metric. In contrast we solve for a (numerically) flat metric and require no further approximation to recover the actual vertex positions.

In their method, target curvatures at the boundary as well as for cone singularities are computed based on a diffusion problem and its associated Green's functions—one each per boundary vertex and interior cone vertex. Incidentally, this computational burden could be removed by using our natural boundary conditions and free cone angles (Sections 4.2 and 5.1).

2.2.1 Discussion

In contrast to all other approaches, ours is based on a proper notion of discrete conformal equivalence for triangle meshes. This notion depends on the meshes alone without any arbitrary choices, and it is adhered to not approximately but precisely.

We have an explicit expression for the energy—as do Kharevych *et al.* but not Jin *et al.*—which facilitates the use of standard, globally convergent, Newton trust region methods. We can also accommodate cone singularities—as introduced by Kharevych *et al.* and recently demonstrated for discrete harmonic approaches [Tong et al. 2007; Kälberer et al. 2007]—and do not require a priori cutting for higher genus surfaces—as is done in ABF. Boundary conditions include constraints on angles or lengths and—novel among all approaches—we support *isometrically* mapped boundaries. Incidentally, the latter also "pick out" the unique least distorted mapping in a given equivalence class (more on these aspects in Section 4).

Comparing our method with ABF illustrates how choosing the "right" discrete notions can lead to concrete practical benefits. ABF preserves triangle angles as much as possible. Our approach preserves the *discrete* conformal structure of the mesh *precisely*. As it happens, this leads to a mathematically simpler optimization problem.

After this discussion of related approaches in computer graphics, we want to acknowledge at least briefly some mathematical work we have built upon. Troyanov [1986] treated the problem of finding a conformally equivalent flat metric with prescribed cone singularities in the smooth setting and proved existence and uniqueness of the solution. The construction of our energy relied heavily on previous work on variational principles for circle patterns [Colin de Verdière 1991; Rivin 1994; Leibon 2002; Bobenko and Springborn 2004]. There is a fruitful connection between all of these circle pattern energies and our energy $E(u)$ on the one side and hyperbolic geometry on the other side. A reader familiar with hyperbolic geometry will realize that the appearance of Milnor's Lobachevsky function is a hint that our energy $E(u)$ has something to do with the volumes of ideal hyperbolic tetrahedra, and she may recognize the logarithmic edge lengths λ_{ij} and length cross ratios $\mathfrak{c}_{ij}$ as Penner coordinates and Thurston-Fock shear coordinates for the Teichmüller spaces $\mathcal{T}_{g,n}$ of genus g Riemann surfaces with n punctures. In fact, minimizing $E(u)$ also solves the problem of finding polyhedral realizations for complete hyperbolic surfaces. But this side of the story is beyond the scope of the present paper.

3 Discretely Conformal Parameterization

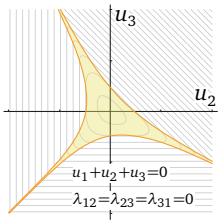
Now that the theoretical foundations are laid down we proceed with a discussion of our basic conformal parameterization method.

3.1 Convex Optimization

So far we have a convex energy, but we do not yet have a convex optimization problem since the domain, *i.e.*, the set of u resulting in new edge lengths satisfying the triangle inequalities, is not convex. The inset figure below illustrates the legal range (yellow) of u values for an example triangle. We get around this difficulty by extending the domain of $E(u)$ to all of $\mathbb{R}^n$:

$$\text{if} \quad \tilde{l}_{jk} \not\leq \tilde{l}_{ki} + \tilde{l}_{ij} \quad \text{then} \quad \tilde{\alpha}^i_{jk} = \pi \quad \text{and} \quad \tilde{\alpha}^j_{ki} = \tilde{\alpha}^k_{ij} = 0.$$

Now Eqs. (7) and (8) define $E(u)$ for all values of u. This simple way of extending $E(u)$ is C^1. The gradient of the extended energy is still given by Eq. (9), and the Hessian is given by Eq. (10), where, however, the terms $\cot\tilde{\alpha}^k_{ij}$ in the equation for the weights must be replaced by 0 whenever $\tilde{\alpha}^k_{ij} = 0$ or π. The figure on the right indicates the iso-contours of the extended energy.



We now have a standard unconstrained convex optimization problem with explicit formulæ for the target function, its gradient, and Hessian. The Hessian is positive semi-definite with only the constant vector in its null-space, reflecting the fact that $E(u)$ remains invariant if a constant vector is added to u. To find the argmin of the extended energy we use the globally convergent Newton-Steihaug trust region method [Steihaug 1983] as implemented in PETSc/TAO [Balay et al. 2007; Benson et al. 2007].

3.2 Violation of the Triangle Inequality

When minimizing the extended energy it is possible that the global minimum is achieved for a u^* giving $\tilde{l}$ which violate the triangle inequality. Figure 2 shows a sampling of such cases which are representative of our experience. In *all* the cases we have observed we found that the triangulation near the bad edge was highly degenerate. In almost all cases this was due to a triangle with one angle close to π. In others, multiple triangles were "folded over" one another (see the second and third example in the top row and first example in the bottom row of Figure 2). Unless the mesh has regions full of such degenerate situations, the problem is simply fixed by flipping or alternatively subdividing the edges opposite the straight angles. Our worst example was the Gargoyle which required flips for 55 out of 74964 edges. (Edge flipping was also considered by Luo [2004], albeit during his curvature flow.)

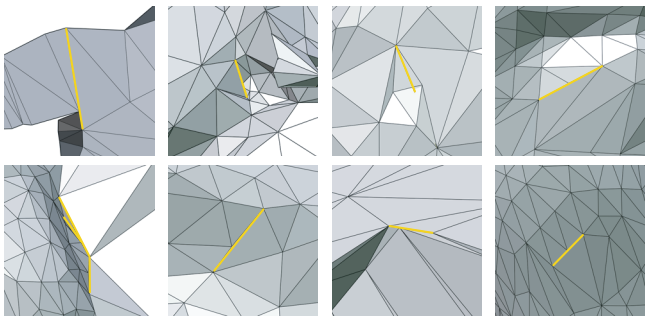


Figure 2: *A sampling of "ripped" edges. In all cases the local geometry was near singular and edge flipping (alternatively, edge subdivision) remedied the trouble.*

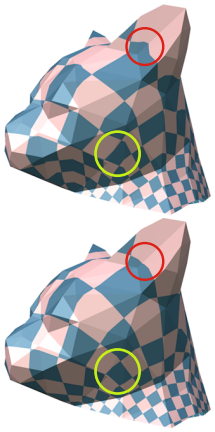
3.3 Layout

With $u^* = \operatorname{argmin}_u E(u)$ we have new lengths $\tilde{l}$ and angles $\tilde{\alpha}$, but not yet a new mesh. For texture mapping we need vertex positions which requires a layout procedure. Suppose M is a topological disk and there are no cone singularities in the interior. We traverse the dual graph of M in a breadth first manner, laying out opposing vertices each time an edge is crossed. The orientation of each edge is determined by *summation* of corner angles along the traversal. With this layout procedure meshes with 100s of thousands of vertices and length ratios as high as 10^6 can be handled with high precision. Typical examples start with metric data flat to within $\|\operatorname{grad} E(u^*)\|_2 < \epsilon = 10^{-14} \ldots 10^{-12}$ and yield *worst* relative length errors bounded by 10ϵ to 1000ϵ.

Arbitrary topology meshes and those with cone singularities are reduced to the topological disk case. To resolve handles we compute a *system of loops* [Erickson and Whittlesey 2005]. Cutting the loops gives a topological disk, possibly with cones in its interior. To resolve the cones, we trace a path from each cone back to the base vertex of the system of loops using the Dijkstra tree from the first phase. (Any boundaries, including the one due to the system of loops, are treated as if each were a single cone vertex from the point of view of spanning tree construction.) Cutting this cone spanning tree yields a topological disk with no cones in the interior and it may be laid out as above. We have made no efforts to move the paths to inconspicuous locations, though methods such as Seamster [Sheffer and Hart 2002] could be employed. Note that vertices along the cut path will have multiple positions in the layout.

3.4 Piecewise Projective Interpolation

After vertex positions for the image mesh have been found, it is usually necessary, *e.g.*, in texture mapping, to extend the map from the vertices to the whole mesh surface through interpolation. Typically this is done piecewise linearly. For conformally equivalent meshes there is another possibility unique to them.



Given a domain and range triangle, there is always a unique projective transformation mapping one onto the other *and* the circumcircle of one onto the circumcircle of the other. In general, these projective mappings do not fit together continuously across edges. A unique exception are the discretely conformally equivalent meshes: This circumcircle preserving piecewise projective interpolation is continuous across edges *iff* the meshes are discretely conformally equivalent. This follows from the proposition in Appendix D, which also shows that in practice it is rather easy to take advantage of this (and we do so in all our texture visualizations): If (x_i, y_i, z_i) and (s_i, t_i) are the vertex coordinates for the original and image mesh respectively, the correct projective interpolation is achieved by performing linear interpolation on the homogeneous coordinates $(x_i, y_i, z_i, 1)$ and $e^{-u_i}(s_i, t_i, 1)$. (Note the scale factor e^{-u_i} which is applied to the homogeneous coordinates of the image mesh.) Since conventional graphics cards support linear interpolation between homogeneous coordinates for texturing anyway, the implementation of the piecewise projective interpolation scheme involves nothing more than using these properly scaled homogeneous coordinates. The figure above shows a comparison between linear interpolation (top) and projective interpolation (bottom) with some notable differences highlighted.

4 Boundary Conditions

Both as a practical matter and as a key to achieving functionality relevant in applications, boundary conditions of different types can be employed. We review these in turn, using practical examples to motivate them.

4.1 Fixed Boundary Curvature

The Problem posed in Section 2.1 requires us to prescribe angle sums $\widehat{\Theta}_i$ for all vertices, *i.e.*, target curvatures $\widetilde{K}_i$ at interior vertices—mostly 0 unless we want a cone singularity—and boundary curvatures $\tilde{\kappa}_i$ for boundary vertices. For example, to achieve a rectangular layout, which has obvious advantages for texture packing, one sets $\widetilde{\Theta}_i = \pi$ for all boundary vertices but four, which receive $\widetilde{\Theta}_i = \pi/2$ (Figure 3, left; see also the Stamp part in Figure 14).

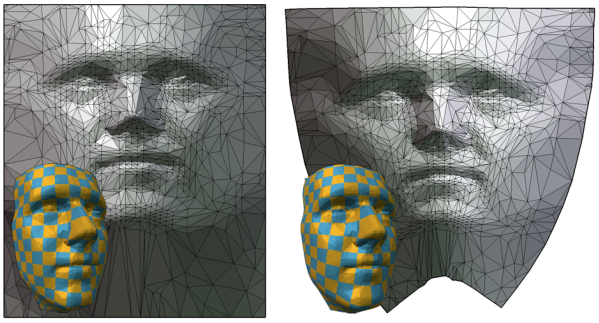


Figure 3: *Fixing curvatures on the boundary can be useful for achieving a particular boundary shape. In contrast, natural boundaries do not constrain the boundary curvature at all (right) and yield the least distorted mapping.*

4.2 Free Boundary Curvature

Alternatively, we may prescribe angle sums $\widehat{\Theta}$ at some vertices and fix u at others. In this case the unique solution can be found by considering the fixed u_i as constants and minimizing $E(u)$ as a function of the remaining variables. Since the value assigned to $\widehat{\Theta}$ at vertices with fixed u is clearly irrelevant for the minimization we may leave it undefined at these vertices. With some u_i fixed, the problem is not scale-invariant anymore and the reduced Hessian becomes positive definite.

An extreme example is to fix u for all boundary vertices. This gives complete control over the length distortion along the boundary while leaving the boundary curvature free.

Natural Boundary Conditions One choice is to set $u_i = 0$ for all boundary vertices. In that case all boundary edges *retain their original length*, *i.e.*, the boundary is mapped *isometrically*. Figure 3 (right) shows a mesh flattened with $\widetilde{\Theta}_i = 2\pi$ for all interior vertices and $u_i = 0$ for all boundary vertices. We call these boundary conditions "natural" because of the following remarkable fact from the smooth theory: Among all flat metrics $\tilde{g}$ that are conformally equivalent to a given metric g, one with *least distortion* is obtained by setting $u = 0$ on the boundary (where the distortion is measured by the Dirichlet energy of u; see Appendix E).

4.3 Topological Spheres

Meshes with sphere topology are sufficiently common that specific parameterization algorithms for them have been designed. Like Kharevych *et al.* [2006], we exploit the Möbius invariance of discrete conformality to map a mesh to the unit sphere. First some vertex is sent to infinity through inversion. Removing its vertex star we fix $u_i = 0$ for the vertices of its link, and $\widehat{\Theta}_i = 2\pi$

for all other vertices. The resulting planar layout is projected stereographically to the sphere and the knocked out vertex re-inserted at the north pole (Figure 4). Subsequent Möbius normalizations can be applied to ensure, for example, that the center of mass of all vertices is at the origin [Springborn 2005]—a notion of "equidistribution" of points. A simple calculation shows that the resulting mesh is discretely conformally equivalent, even at the knocked out vertex.

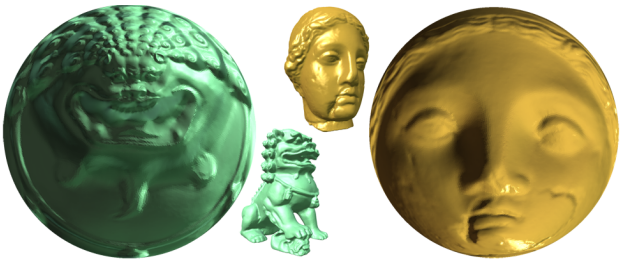


Figure 4: *Sphere parameterizations of Dragon and Hygeia visualized as spheres with the original normals.*

5 Cone Singularities

In practice, a significant issue with conformal maps is their at times severe length distortion. Gu *et al.* [2002], for example, added cuts into the worst distortion area, repeating as necessary. To avoid cutting and the associated discontinuities or complex boundary conditions, Gu and Yau [2003] used topological puncturing and double cover constructions.

The most general and flexible approach to date though was introduced by Kharevych *et al.* [2006] and is based on the selective introduction of isolated vertices which are not required to have zero Gaussian curvature. At these *cone vertices*, the local metric is that of a Euclidean cone with some cone angle $\widetilde{\Theta}_i \neq 2\pi$. This was used both to accommodate higher genus surfaces and reduce length distortion. Placement relied exclusively on human geometric intuition and trial and error. More recently cone singularities with cone angles restricted to positive integer multiples of $\pi/2$ were used in rectangular surface patching [Ray et al. 2006] and in discrete harmonic methods [Tong et al. 2007; Kälberer et al. 2007] for the construction of union-of-quad parameterizations.

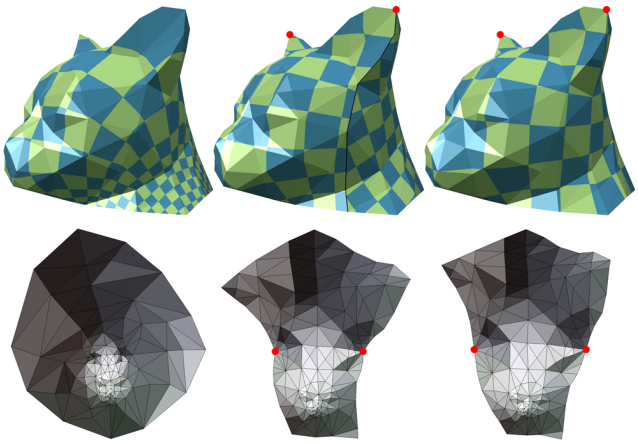


Figure 5: *The effect of cone singularities. Beginning with natural boundary conditions (left), two free cones are added automatically (middle) to reduce distortion (Section 5.1). Depending on the texture, seams may appear. Choosing a suitable cone angle (here π) can remove this problem (right). All three flat metrics are in the same equivalence class as the original metric.*

Aiming to align iso-parameter lines (roughly) with principal curvature directions, these approaches all placed cone singularities at numerically estimated umbilic points.

Our method supports free and fixed cone angles (Figure 5) and we are interested in placing cone singularities in a manner which reduces length distortion, *i.e.*, reduces the variation of u, and to do so automatically.

5.1 Automatic Cone Singularities

Where should the zero curvature assumption be relaxed so as to reduce distortion? Considering Eq. (12) for the smooth setting, we see that a local change in K leads to a localized change in u, since adding a Dirac to the right hand side has the effect of adding a Green's function to the solution u. Thus, we choose the vertex with the worst distortion, $v_i = \operatorname{argmax}_{v_i} |u_i|$, as the location for a cone singularity. (If a boundary is present we assume natural boundary conditions, otherwise we set $\sum u_i = 0$.) Instead of prescribing a cone angle, we leave it free, setting $u_i = 0$ (Section 4.2). Since a cone singularity may be seen as the limiting case of a very small hole, this makes sense in view of the minimal distortion property of natural boundary conditions. Our strategy for the *placement* of the cone singularities is thus very similar to the one of Ben-Chen *et al.* [2008], but we do not have to do any extra work to determine the cone angles.

A possible greedy approach starts with some minimizer u of E and selects the vertex with the largest magnitude u_i, setting it to free cone status, and repeating. Figure 6 shows a sequence of such greedy free placements and their impact on u.

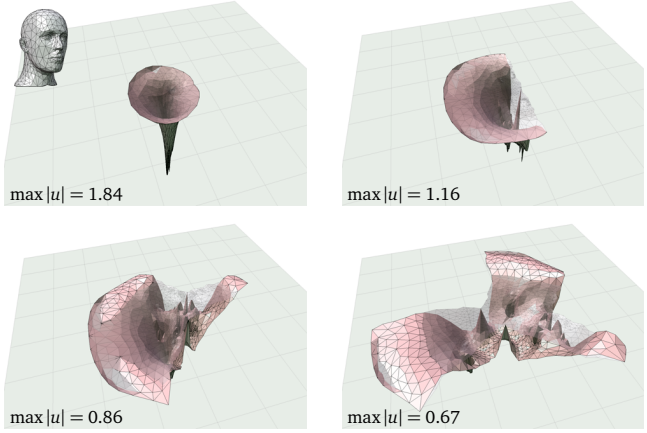


Figure 6: *Visualization of u as a graph over the domain for the mannequin head. Using natural boundaries, zero to three greedy free cones are placed, significantly reducing the magnitude of u in each step.*

If the mesh is a closed surface of genus $\neq 1$ we cannot—due to the Gauss-Bonnet relation—get an initial solution for u without prescribing initial curvatures at vertices. Instead of concentrating all of the required total curvature at some vertex like Ben-Chen *et al.*, we distribute the required total curvature over the whole mesh such that each vertex receives an equal share initially. This is done for the initial solve only. As free cones are placed, the remaining vertices revert to the desired zero curvature setting.

Because the greedy procedure cannot undo earlier decisions, it comes as no surprise that its results can be improved (Figure 7). Four free cone singularities were placed in a greedy manner resulting in two cones on top of the head (front and back), at the left ear and on the nose (left). Starting from this initial placement of the four cones, several sweeps—three in this case—of a nonlinear Gauss-Seidel solver result in the placement on the right.

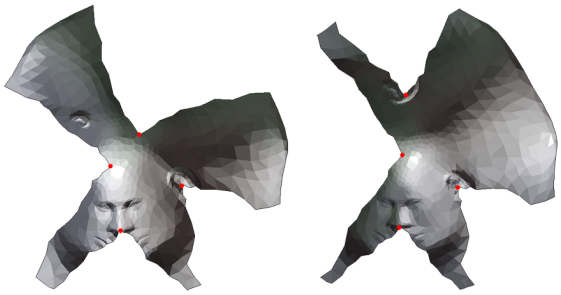


Figure 7: *Comparison of the greedy strategy (left) with concurrent cone optimization (right) for four cones.*

Cones are now placed far more symmetrically: at the top of the head, on each ear and on the nose. Typically it takes two or three rounds for the cone positions to "settle." To save effort during the Gauss-Seidel stage, we do not solve for u to high accuracy (only one Newton step is used). The final u-function is solved to high precision. Figure 8 shows another where this procedure was applied with a total of 18 cones.

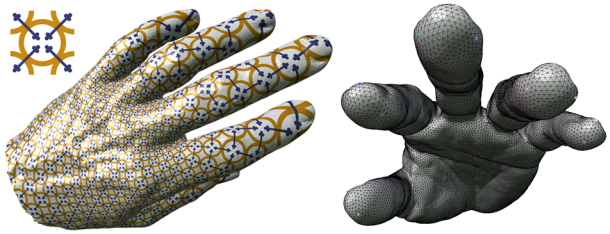


Figure 8: *Olivier's hand with 18 optimized free cones (distributed over the finger tips and inbetween the fingers).*

6 Texture Rectification

When mapping unstructured textures, minimizing distortion in the parameterization is an important criterion. Using textures with symmetries, *e.g.*, checker boards, or when using parameterizations for purposes of resampling, additional constraints on cone placement and cone angles are called for. Quadrilateral patching in particular calls for cones with $\widehat{\Theta}_i = k\pi/2$, $0 < k \neq 4$. A simple example of this is seen in Figure 5, where two free cones were quantized to a fixed angle of π each. Layout followed by a suitable rigid motion and global scale places the two cones at $(0,0)$ and $(1,0)$ in the parameter plane. A seamless checker board texture results.

But quantizing the cone angles to integer multiples of $\pi/2$ is generally not enough if there are more than two cone vertices. Figure 9 illustrates such an example. Four cones of π each were placed on the Venus mesh—a topological sphere. The resulting layout—regardless of the particulars of the cone spanning tree—*must* arrange the cone images as the corners of a parallogram, but generally not as the corners of a rectangle with integer side lengths. In the case of the Venus example a simple shear and scale is enough to produce a seamless checker board texture. (Afterward the metric is no longer conformally equivalent.)

Let us consider this problem in greater generality. Assume M is a topological disk or sphere (the reason for this restriction will be explained below) and any cones multiples of $\pi/2$. For a topological disk, any free cones—resulting from our automatic procedure (Section 5.1)—are rounded to the nearest positive multiple of $\pi/2$. In the case of a closed surface, the quantized values may violate the Gauss-Bonnet relation, for example, the sum of quantized cone values may be too large. In that case the cone angle that was rounded up the most has its quantized value decreased (unless it is $\pi/2$ already). Repeating this procedure if

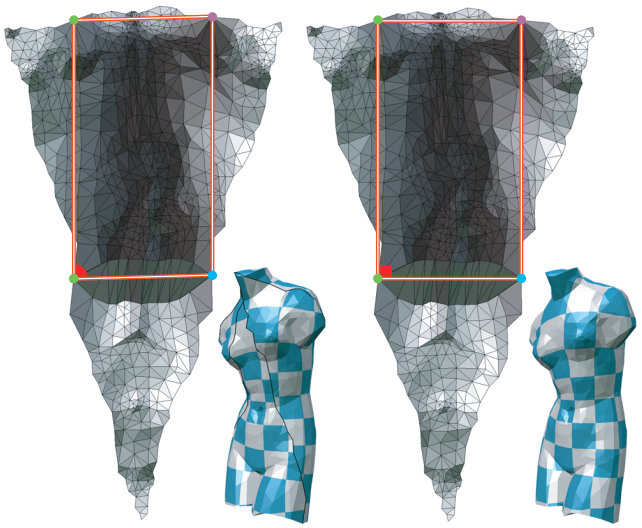


Figure 9: *A layout for the Venus with four cones of π each. These map to corners of a parallelogram (left side). Using a checker board texture leads to artifacts (inset). These can be removed by rectifying the layout with an additional transformation (right side).*

necessary leads to a set of quantized values which minimize the maximum error between orginal and quantized values while satisfying the Gauss-Bonnet relation. (An analogous procedure can be applied if the initial sum is too small.) Using these now fixed cones the corresponding conformally equivalent metric is solved for and laid out (Section 3.3). It defines the closed *meta-polygon* $P = \{p_i \in \mathbb{C} \,|\, i = 0, \ldots, m-1\}$ given by the vertex positions—treated as complex numbers—of m *special vertices* encountered in a single, ordered traversal of the layout boundary. Special vertices are (1) cone vertices, (2) interior vertices with more than two cut edges incident on them, and (3) boundary vertices with a cut edge incident on them.

Figure 10 shows a meta-polygon for the Lion dataset. Five free cones were placed with the simultaneous optimization procedure, quantized to the nearest multiple of $\pi/2$ ($3\pi/2$ for all five cones in this case), and the corresponding conformally equivalent metric laid out.

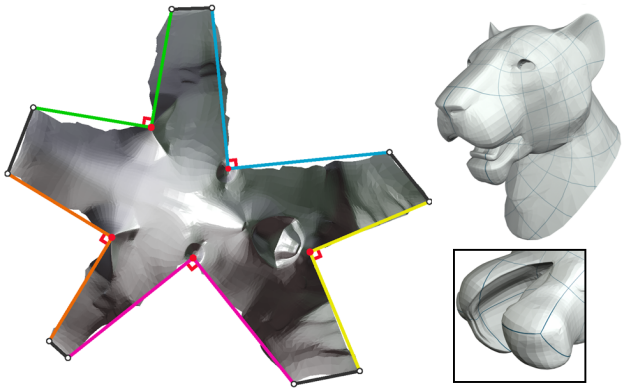


Figure 10: *Meta-polygon for the Lion with quantized cones (solid red) before rectification. After rectification, which forces all cone points to integer locations, the texture tiles the model seamlessly (inset shows $3\pi/2$ cone on chin underside).*

Meta-edges correspond to paths of edges along the boundary of the layout, joining two consecutive special vertices. Such a path is either along the boundary of the original mesh (black edges), or one side of a cut (colored edges), with the other side (matching

color) corresponding to a different meta-edge. The two sides of a cut path fit together: a Euclidean motion moves one to the other, and the same motion aligns the corresponding meta-edges. As Figure 10 suggests, the rotation angles of these motions are multiples of $\pi/2$. This is true in general due to the quantized cone angles, but only if the original mesh is simply connected—hence the topological restriction to disks and spheres.

Suppose for a moment that the special vertices of type (1), the cone vertices, have integer coordinates. Then a checkerboard pattern with these vertices in the centers of the squares would fit seamlessly on the mesh, because the Euclidean motions aligning corresponding meta-edges would be symmetries of the pattern.

In reality, the coordinates are not integer. (Figure 11 shows an example with eight optimized and quantized cones, which are not on integers; note the apparent "rip" in the texture on the far end of the right wing.) How close they are to being integer, and hence how much distortion is needed to make them so, depends on rotation and scaling: the larger the scale, the finer the integer grid is relative to the layout size. Suppose such a rotated and scaled layout is given. The idea now is to deform the meta-polygon so that all cone vertices become *complex integers*, $\mathbb{Z}^2 := \mathbb{Z} + I\mathbb{Z} \in \mathbb{C}$, and adapt the layout accordingly. The result will no longer be exactly discretely conformal, but only approximately so: Seamless alignment can *only* be achieved by relaxing the conformality requirement.

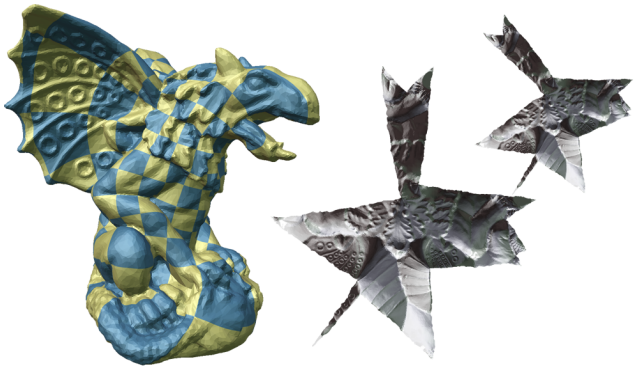


Figure 11: *The Gargoyle—a topological sphere—with eight optimized and subsequently quantized cones. On the top right the optimized cone domain and in the center the result of quantizing the cones. Since the cones do not have integer locations the checkerboard texture has seams.*

6.1 Deforming the Meta-Polygon

When deforming the meta-polygon, associated pairs of meta-edges must maintain their alignment through Euclidean motions with rotations being multiples of $\pi/2$. This requirement is best treated by considering the meta-edge differences $p_{i+1} - p_i =: d_i \in \mathbb{C}$ (all index arithmetic is understood modulo m). Meta-edges due to cutting come in pairs (i, j) and we have

$$d_i = R_{ij} d_j \quad \text{with} \quad R_{ij} \in \{+1, +I, -1, -I\}.$$

The perturbed edge differences $\bar{d}$ must maintain these relations and keep the polygon closed: $\sum \bar{d}_i = 0$. This can be written as

$$A\bar{d} = 0,$$

where A is a matrix with m columns and one more row than there are pairs of corresponding meta-edges. Let N_A be a matrix of (column) basis vectors spanning the null space of A, chosen so that all entries of N_A are complex integers.

Now assume the vertices of the meta-polygon are numbered so that p_0 is a cone vertex and translate it to the origin. Then all cone vertices have integer coordinates if $\sum_{i=0}^{k-1} \bar{d}_i \in \mathbb{Z}^2$ for all k such that p_k is a cone vertex. This can be written as

$$B\bar{d} \in (\mathbb{Z}^2)^l$$

where B is a matrix with m columns and l rows (one less than there are cone vertices). Finally, the columns of BN_A—all entries are complex integers—span a complex subspace which has non-empty intersection with the lattice $(\mathbb{Z}^2)^l$. From this subset lattice we wish to select the nearest (in some norm) lattice point to the cone positions in the given (non-integer) layout. This is an instance of the classic *closest lattice vector* (CLV) problem. Optimal solutions to this problem are computationally hard. For the examples in this paper we proceeded by computing the row rank of BN_A, fixed as many $(l-1)$ of the cones to the nearest point in $\mathbb{Z}^2$ and solved for the remaining (l^{th}) cone location.

Given a solution $\bar{d}$ the implied deformation must now be interpolated across the layout. We do this by performing a cot-Laplace layout as explained in the next section.

6.2 Generalized Laplace Layout

In Section 3.3 we described our breadth first layout procedure to turn metric data into coordinate functions. Another way to turn metric data into coordinate functions is through the use of the cot-Laplace operator.

For a flat metric the cot-Laplace operator—as a map $\mathbb{C}^n \to \mathbb{C}^n$. $(n = |V|)$ and using natural boundary conditions [Desbrun et al. 2002; Lévy et al. 2002]—has a two dimensional (complex) null space consisting of the constant and linear functions on the vertices. The layout procedure produces the unique—up to scale and a rigid motion—null space vector orthogonal to the constant vector. We now turn this prescription around: Given a cot-Laplace operator due to a flat metric, produce the null space vector with zero mean. Because of the presence of cone singularities this cot-Laplace operator must incorporate the edge identifications via rigid motions encoded in the meta-polygon. This is done in the same manner as [Tong et al. 2007]. (Their method accommodates general rotations with no problem including the fact that only a single complex variable is needed for each vertex.) Cone singularities enter as Dirichlet data with positions. The cot-Laplace operator now has only *one* remaining vector in its null space: the original layout (this is true even for general cone angles).

Figure 12: *Rectified domain for Max Planck dataset and visualization of corresponding iso-lines on the mesh.*

We have run comparisons between breadth first and Laplace layout, and found the results to be of comparable accuracy even for very large layouts. Solving large cot-Laplace systems to high accuracy is in general computationally far more demanding than a simple breadth first traversal, of course.

If the meta-polygon is deformed and we use the *new* cone locations as Dirichlet data the cot-Laplace operator will generally not have a null space anymore and we return the vector of least Dirichlet energy. Figure 12 shows the rectified domain and corresponding iso-lines on the model of Max Planck; see also Figure 10 for the rectified Lion head, and Figure 13 for the rectified Torso parameterization.

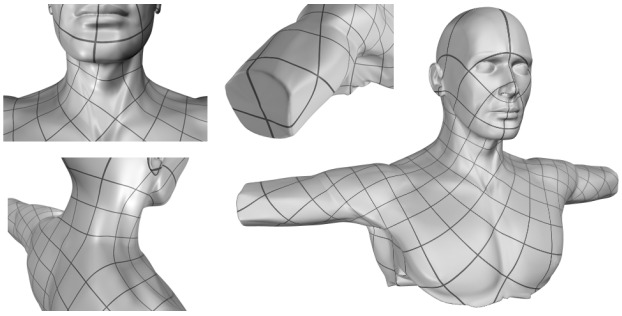


Figure 13: *Visualization of the parameterization on the Torso with 14 optimized, quantized, and subsequently rectified cones (each ear, each nostril, either side of back of neck, on Adam's apple, three on each arm stump and one cone on the torso bottom.*

7 Numerical Results

We have applied our techniques to a number of meshes and report a representative subset of the results here. Mesh sizes ranged from a few hundred vertices to over 200,000 (Julius dataset in Figure 14). In all cases we achieved gradient magnitude residuals in the $10^{-14}\ldots10^{-12}$ range. None of our layouts—either breadth first or the cot-Laplace operator for rectified layouts—have any flipped triangles. The maximum, relative length layout error was always bounded by 10 to 1000 times the gradient residual norm. The Newton trust region solver typically requires around 10 iterations, each iteration having a cost proportional to the assembly and solve phases of a single cot-Laplace problem. The number of Newton steps is thus the relevant figure of merit in terms of overall runtime. (Absolute performance numbers are meaningless since individual solver implementations and processor differences can easily account for factors of 10 and more in observed wall clock times. With this proviso, we note that PETSc/TAO running on a 1.1GHz Pentium M requires 162s for Julius and 230s for the Torso.) The case of the Gargoyle is exceedingly atypical in our experience. This particular mesh is very degenerate in many places and the trust region solver took very small steps. As a quantitative distortion measure we used the area weighted mean of quasi-conformality [Kharevych et al. 2006] (denoted $\|qc\|_1$ in the table), *i.e.*, the ratio of larger to smaller singular value of the surface to parameter plane tangent map. A value of one indicates no angle distortion at all. While it is difficult to compare these numbers across papers ours appear generally somewhat worse than what was reported by Ben-Chen *et al.* and Kharevych *et al.*, for example. Given that we effectively enforce more structure than Ben-Chen and co-workers, and do not allow any approximation of the original data as Kharevych and co-workers do, this is perhaps not surprising. The last column in the table indicates the boundary conditions (free, rectangle, sphere) or the number of optimized (o) free cones, whether they were subsequently quantized (q), and/or rectified (r).

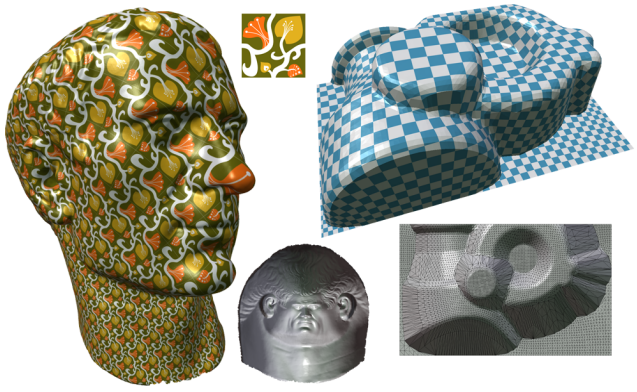


Figure 14: *Further examples of a free boundary (Julius) and use of boundary curvatures to enforce a rectangular domain (Stamp).*

Model	$\|V\|$	Iter.	$\|u\|_{\max}$	$\|qc\|_1$	cones
Face	1042	16	1.164	1.0702	rectangle
Stamp	4100	6	0.431	1.0541	free
Stamp	4100	6	0.742	1.0625	rectangle
Lion	8356	6	1.089	1.0282	5 (oqr)
Olivier	24795	6	1.188	1.0087	18 (o)
Max Planck	25445	11	0.908	1.0097	4 (oqr)
Gargoyle	24990	56	1.849	1.0509	8 (oq)
Hygeia	140654	5	0.650	1.0120	sphere
Torso	142348	12	1.957	1.0048	4 (o)
Torso	142348	21	1.091	1.0036	14 (oqr)
Dragon	152803	5	3.759	1.0277	sphere
Julius	209083	5	1.173	1.0027	free

8 Conclusion

We have shown that our simple notion of discrete conformal equivalence leads to attractive algorithms for mesh flattening, including the automatic determination of cone singularities to reduce distortion and deal with higher genus surfaces, as well as a new piecewise projective interpolation scheme. In Section 6 we have taken first steps to tackle the problem of mapping a symmetric pattern seamlessly onto an arbitrary surface. We have discussed a way to deal with cone singularities, but how to achieve texture rectification for surfaces of higher genus remains an open question. Here we see a promising new direction for further research.

Acknowledgment This work was supported in part by NSF (CCF-0528101 and CCF-0635112), DOE (W-7405-ENG-48/B341492), the Caltech Center for Mathematics of Information, DFG Research Center Matheon, the Alexander von Humboldt Stiftung, and Autodesk. The authors are gratefully indebted to Alexander Bobenko for inspiring discussions. Special thanks to Cici Koenig, Andreas Fabri, Pierre Alliez, and Mathieu Desbrun.

References

Balay, S., Buschelman, K., Eijkhout, V., Gropp, W. D., Kaushik, D., Knepley, M. G., McInnes, L. C., Smith, B. F., and Zhang, H. 2007. PETSc Users Manual. Tech. Rep. ANL-95/11 (Revision 2.3.3), Argonne National Laboratory. http://www.mcs.anl.gov/petsc/.

Ben-Chen, M., Gotsman, C., and Bunin, G. 2008. Conformal Flattening by Curvature Prescription and Metric Scaling. *Comp. Graph. Forum 27*, 2, 449–458.

Benson, S., McInnes, L. C., Moré, J., Munson, T., and Sarich, J. 2007. TAO User Manual. Tech. Rep. ANL/MCS-TM-242 (Revision 1.9), Argonne National Laboratory. http://www.mcs.anl.gov/tao.

Bobenko, A. I., and Springborn, B. A. 2004. Variational Principles for Circle Patterns and Koebe's Theorem. *Trans. Amer. Math. Soc. 356*, 2, 659–689.

Bobenko, A. I., and Suris, Y. B., 2005. Discrete Differential Geometry. Consistency as Integrability. Preprint arXiv:math/

0504358v1. To appear in *Graduate Studies in Mathematics* of the AMS.

Bowers, P. L., and Hurdal, M. K. 2003. Planar Conformal Mappings of Piecewise Flat Surfaces. In *Vis. and Math. III*. Springer, 3–34.

Chow, B., and Luo, F. 2003. Combinatorial Ricci Flows on Surfaces. *J. Diff. Geom. 63*, 1, 97–129.

Colin de Verdière, Y. 1991. Un Principe Variationnel pour les Empilements de Cercles. *Invent. Math. 104*, 655–669.

Desbrun, M., Meyer, M., and Alliez, P. 2002. Intrinsic Parameterizations of Surface Meshes. *Comp. Graph. Forum 21*, 3, 209–218.

Duffin, R. J. 1956. Basic Properties of Discrete Analytic Functions. *Duke Math. J. 23*, 335–363.

Duffin, R. 1959. Distributed and Lumped Networks. *J. Math. Mech.* [continued as *Indiana Univ. Math. J.*] *8*, 793–826.

Erickson, J., and Whittlesey, K. 2005. Greedy Optimal Homotopy and Homology Generators. In *Proc. ACM/SIAM Symp. on Disc. Alg.*, SIAM, 1038–1046.

Floater, M. S., and Hormann, K. 2005. Surface Parameterization: a Tutorial and Survey. In *Advances in Multiresolution for Geometric Modelling*, Mathematics and Visualization. Springer, 157–186.

Gu, X., and Yau, S.-T. 2003. Global Conformal Surface Parameterization. In *Proc. Symp. Geom. Proc.*, Eurographics, 127–137.

Gu, X., Gortler, S. J., and Hoppe, H. 2002. Geometry Images. *ACM Trans. Graph. 21*, 3, 355–361.

Jin, M., Kim, J., and Gu, X. D. 2007. Discrete Surface Ricci Flow: Theory and Applications. In *Mathematics of Surfaces 2007*, R. Martin, M. Sabin, and J. Winkler, Eds., Vol. 4647 of *Lecture Notes in Computer Science*. Springer, 209–232.

Kälberer, F., Nieser, M., and Polthier, K. 2007. QuadCover—Surface Parameterization using Branched Coverings. *Comp. Graph. Forum 26*, 3, 375–384.

Kharevych, L., Springborn, B., and Schröder, P. 2006. Discrete Conformal Mappings via Circle Patterns. *ACM Trans. Graph. 25*, 2, 412–438.

Leibon, G. 2002. Characterizing the Delaunay Decompositions of Compact Hyperbolic Surfaces. *Geom. Topol. 6*, 361–391.

Lévy, B., Petitjean, S., Ray, N., and Maillot, J. 2002. Least Squares Conformal Maps for Automatic Texture Atlas Generation. *ACM Trans. Graph. 21*, 3, 362–371.

Lewin, L. 1981. *Polylogarithms and Associated Functions*. North Holland.

Luo, F. 2004. Combinatorial Yamabe Flow on Surfaces. *Commun. Contemp. Math. 6*, 765–780.

Macleod, A. J. 1996. Algorithm 757: MISCFUN, a Software Package to Compute Uncommon Special Functions. *ACM Trans. Math. Softw. 22*, 3, 288–301.

Mercat, C. 2001. Discrete Riemann Surfaces and the Ising Model. *Comm. in Math. Physics 218*, 1, 177–216.

Milnor, J. 1982. Hyperbolic Geometry: The First 150 Years. *Bul. Amer. Math. Soc. 6*, 1, 9–24.

Pinkall, U., and Polthier, K. 1993. Computing Discrete Minimal Surfaces and Their Conjugates. *Experiment. Math. 2*, 1, 15–36.

Ray, N., Li, W. C., Lévy, B., Sheffer, A., and Alliez, P. 2006. Periodic Global Parameterization. *ACM Trans. Graph. 25*, 4, 1460–1485.

Rivin, I. 1994. Euclidean Structures on Simplicial Surfaces and Hyperbolic Volume. *Ann. of Math. (2) 139*, 553–580.

Sheffer, A., and Hart, J. C. 2002. Seamster: Inconspicuous Low-Distortion Texture Seam Layout. In *Proc. IEEE Vis.*, IEEE Comp. Soc., 291–298.

Sheffer, A., Lévy, B., Mogilnitsky, M., and Bogomyakov, A. 2005. ABF++: Fast and Robust Angle Based Flattening. *ACM Trans. Graph. 24*, 2, 311–330.

Sheffer, A., Praun, E., and Rose, K. 2006. Mesh Parameterization Methods and their Applications. *Found. Trends Comput. Graph. Vis. 2*, 2, 105–171.

Springborn, B. 2005. A Unique Representation of Polyhedral Types. Centering via Möbius Transformations. *Math. Z. 249*, 513–517.

Steihaug, T. 1983. The Conjugate Gradient Method and Trust Regions in Large Scale Optimization. *SIAM J. Numer. Anal. 20*, 3, 626–637.

Stephenson, K. 2003. Circle Packing: A Mathematical Tale. *Notices Amer. Math. Soc. 50*, 11, 1376–1388.

Stephenson, K. 2005. *Introduction to Circle Packing*. Cambridge University Press.

Tong, Y., Alliez, P., Cohen-Steiner, D., and Desbrun, M. 2007. Designing Quadrangulations with Discrete Harmonic Forms. In *Proc. Symp. Geom. Proc.*, Eurographics, 201–210.

Troyanov, M. 1986. Les Surfaces Euclidiennes à Singularités Coniques. *Enseign. Math. (2) 32*, 79–94.

Yang, Y., Kim, J., Luo, F., Hu, S., and Gu, D. 2008. Optimal Surface Parameterization Using Inverse Curvature Map. *IEEE Trans. Vis. Comp. Graph.* (to appear).

Zayer, R., Lévy, B., and Seidel, H.-P. 2007. Linear Angle Based Parameterization. In *Proc. Symp. Geom. Proc.*, Eurographics, 135–141.

A Milnor's Lobachevsky Function

Milnor's Lobachevsky function [Milnor 1982] is defined by

$$\Pi(x) = -\int_0^x \log|2\sin t|\,dt.$$

Up to scale it agrees with *Clausen's integral* $\mathrm{Cl}_2(x) = 2\Pi(x/2)$ [Lewin 1981]. The function $\Pi(x)$ is π-periodic, continuous, and odd. It is smooth except at integer multiples of π where its graph has a vertical tangent. For our purposes note that

$$\Pi'(x) = -\log|2\sin x| \qquad \text{and} \qquad \Pi''(x) = -\cot x.$$

(For us the absolute value signs are irrelevant because we only consider $\Pi(\alpha)$ for $0 \le \alpha \le \pi$.) Numerical evaluation can be performed very efficiently and with high accuracy [Macleod 1996].

B Gradient of E

To show Eq. (9) for the partial derivatives of the energy E, we show first that

$$\partial_{\tilde\lambda_{jk}} f(\tilde\lambda_{ij}, \tilde\lambda_{jk}, \tilde\lambda_{ki}) = \tfrac{1}{2}\tilde\alpha^i_{jk}. \tag{13}$$

From Eq. (8) one obtains

$$\begin{aligned}
&\partial_{\tilde\lambda_{jk}} f(\tilde\lambda_{ij}, \tilde\lambda_{jk}, \tilde\lambda_{ki}) = \tfrac{1}{2}\tilde\alpha^i_{jk} + \left(\tfrac{1}{2}\tilde\lambda_{jk} - \log(2\sin\tilde\alpha^i_{jk})\right)\partial_{\tilde\lambda_{jk}}\tilde\alpha^i_{jk} \\
&+ \left(\tfrac{1}{2}\tilde\lambda_{ki} - \log(2\sin\tilde\alpha^j_{ki})\right)\partial_{\tilde\lambda_{jk}}\tilde\alpha^j_{ki} + \left(\tfrac{1}{2}\tilde\lambda_{ij} - \log(2\sin\tilde\alpha^k_{ij})\right)\partial_{\tilde\lambda_{jk}}\tilde\alpha^k_{ij} \\
&= \tfrac{1}{2}\tilde\alpha^i_{jk} + \left(\log\tfrac{\tilde l_{jk}}{2\sin\tilde\alpha^i_{jk}}\right)\partial_{\tilde\lambda_{jk}}\tilde\alpha^i_{jk} + \left(\log\tfrac{\tilde l_{ki}}{2\sin\tilde\alpha^j_{ki}}\right)\partial_{\tilde\lambda_{jk}}\tilde\alpha^j_{ki} + \left(\log\tfrac{\tilde l_{ij}}{2\sin\tilde\alpha^k_{ij}}\right)\partial_{\tilde\lambda_{jk}}\tilde\alpha^k_{ij}.
\end{aligned}$$

Since $\tilde l_{jk}/(2\sin\tilde\alpha^i_{jk}) = \tilde l_{ki}/(2\sin\tilde\alpha^j_{ki}) = \tilde l_{ij}/(2\sin\tilde\alpha^k_{ij}) = \tilde R$ (the circumcircle radius for a triangle with sides $\tilde l_{ij}$, $\tilde l_{jk}$, and $\tilde l_{ki}$) and

$\partial_{\tilde\lambda_{jk}}(\tilde\alpha^i_{jk}+\tilde\alpha^j_{ki}+\tilde\alpha^k_{ij})=0$ (the angle sum is π), Eq. (13) follows. One finds also (in view of Eq. (3)) $\partial_{u_i} f(\tilde\lambda_{ij},\tilde\lambda_{jk},\tilde\lambda_{ki}) = (\partial_{\tilde\lambda_{ij}} + \partial_{\tilde\lambda_{ki}}) f(\tilde\lambda_{ij},\tilde\lambda_{jk},\tilde\lambda_{ki}) = (\tilde\alpha^k_{ij}+\tilde\alpha^j_{ki})/2$, and hence

$$\partial_{u_i}\big(f(\tilde\lambda_{ij},\tilde\lambda_{jk},\tilde\lambda_{ki}) - \tfrac{\pi}{2}(u_i+u_j+u_k)\big) = -\tfrac{1}{2}\tilde\alpha^i_{jk}.$$

Using this, it is straightforward to derive Eq. (9) from Eq. (7).

C Hessian of E

To show Eq. (10) for the Hessian of E, note that $(\mathrm{Hess}\, E\cdot\delta u)_i$ is the derivative of the ith component of $\mathrm{grad}\, E$ in the direction δu. Since

$$(\mathrm{grad}\, E)_i = \tfrac{1}{2}\widehat\Theta_i - \tfrac{1}{2}\sum_{t_{ijk}\ni v_i}\tilde\alpha^i_{jk},$$

and $\widehat\Theta_i$ is constant, we are done once we have shown that the first-order change in $\tilde\alpha^i_{jk}$ is

$$\delta\tilde\alpha^i_{jk} = \tfrac{1}{2}\cot\tilde\alpha^j_{ki}(\delta u_k-\delta u_i) + \tfrac{1}{2}\cot\tilde\alpha^k_{ij}(\delta u_j-\delta u_i). \qquad (14)$$

To derive this relation between $\delta\tilde\alpha$ and δu, we start with the sine theorem: $\tilde l_{ij}/\tilde l_{ki} = \sin\tilde\alpha^k_{ij}/\sin\tilde\alpha^j_{ki}$. (Alternatively one could start with the cosine theorem, compare [Ben-Chen et al. 2008], App. A.) Take the logarithm on both sides to get

$$(\tilde\lambda_{ij}-\tilde\lambda_{ki})/2 = \log\sin\tilde\alpha^k_{ij} - \log\sin\tilde\alpha^j_{ki}$$

and, for the first-order changes,

$$(\delta\tilde\lambda_{ij}-\delta\tilde\lambda_{ki})/2 = \cot\tilde\alpha^k_{ij}\,\delta\tilde\alpha^k_{ij} - \cot\tilde\alpha^j_{ki}\,\delta\tilde\alpha^j_{ki}.$$

Since $\delta\tilde\lambda_{ij}=\delta u_i+\delta u_j$ and $\delta\tilde\lambda_{ki}=\delta u_k+\delta u_i$, one obtains

$$(\delta u_j-\delta u_k)/2 = \cot\tilde\alpha^k_{ij}\,\delta\tilde\alpha^k_{ij} - \cot\tilde\alpha^j_{ki}\,\delta\tilde\alpha^j_{ki},$$

and two other such equations through cyclic permutation of ijk. These three linear equations for $\delta\tilde\alpha^k_{ij}$, $\delta\tilde\alpha^i_{jk}$, $\delta\tilde\alpha^j_{ki}$, are linearly dependent (they sum to zero). Two of them together with $\delta\tilde\alpha^i_{jk}+\delta\tilde\alpha^j_{ki}+\delta\tilde\alpha^k_{ij}=0$ form a linear system of equations which determines $\delta\tilde\alpha$:

$$A\cdot\begin{pmatrix}\delta\tilde\alpha^i_{jk}\\ \delta\tilde\alpha^j_{ki}\\ \delta\tilde\alpha^k_{ij}\end{pmatrix} = \tfrac{1}{2}\begin{pmatrix}\delta u_j-\delta u_i\\ \delta u_i-\delta u_k\\ 0\end{pmatrix}, \qquad A=\begin{pmatrix}\cot\tilde\alpha^i_{jk} & -\cot\tilde\alpha^j_{ki} & 0\\ -\cot\tilde\alpha^i_{jk} & 0 & \cot\tilde\alpha^k_{ij}\\ 1 & 1 & 1\end{pmatrix}.$$

Using the addition theorems for sine and cosine one shows $\det A=-1$. Applying Cramer's rule then yields

$$\delta\tilde\alpha^i_{jk} = -\det\begin{pmatrix}\frac{1}{2}(\delta u_j-\delta u_i) & -\cot\tilde\alpha^j_{ki} & 0\\ \frac{1}{2}(\delta u_i-\delta u_k) & 0 & \cot\tilde\alpha^k_{ij}\\ 0 & 1 & 1\end{pmatrix}$$

which is Eq. (14).

D Circumcircle Preserving Projective Maps

Consider two triangles $\triangle$, $\tilde\triangle$ in the plane with respective Euclidean vertex coordinates $p_i=(x_i,y_i)$ and $\tilde p_i=(\tilde x_i,\tilde y_i)$, and let $l_{ij}=\|p_i-p_j\|$ and $\tilde l_{ij}=\|\tilde p_i-\tilde p_j\|$ be the edge lengths. Define u_i by (5) so that (2) holds. In homogeneous coordinates, p_i and $\tilde p_i$ are represented by the vectors $w_i=(x_i,y_i,1)$ and $\tilde w_i=(\tilde x_i,\tilde y_i,1)$. The projective transformations mapping $\triangle$ to $\tilde\triangle$ come from the linear transformations $f:\mathbb{R}^3\to\mathbb{R}^3$ of homogeneous coordinates with $f(w_i)=a_i\tilde w_i$ where $a_i\in\mathbb{R}_{>0}$.

Proposition. *The projective transformation corresponding to such a linear transformation f maps the circumcircle of $\triangle$ to the circumcircle of $\tilde\triangle$ if and only if a_1, a_2, a_3 are chosen proportional to e^{-u_1}, e^{-u_2}, e^{-u_3}.*

Proof. In homogeneous coordinates $w=(x,y,z)$, the circumcircle of $\triangle$ is described by the equation $q(w)=0$ where $q(w)$ is the quadratic form $q(w)=x^2+y^2+2cxz+2dyz+ez^2$ with c, d, e determined by the conditions $q(w_i)=0$. Similarly, let $\tilde q(\tilde w)$ be the quadratic form describing the circumcircle of $\tilde\triangle$. The circumcircle of $\triangle$ is mapped to the cirumcircle of $\tilde\triangle$ if $q(w)$ is up to some constant factor μ identical to $\tilde q(f(w))$, *i.e.* $q(w)=\mu\cdot\tilde q(f(w))$, or equivalently, if the corresponding symmetric bilinear forms $b(w,w')$, $\tilde b(\tilde w,\tilde w')$ satisfy $b(w,w')=\mu\cdot\tilde b(f(w),f(w'))$. This is the case *iff* $b(w_i,w_j)=\mu\cdot\tilde b(a_i\tilde w_i,a_j\tilde w_j)$, because the w_i form a basis for $\mathbb{R}^3$. But since

$$l_{ij}^2 = q(w_i-w_j) = q(w_i)-2b(w_i,w_j)+q(w_j) = -2b(w_i,w_j)$$

and similarly $\tilde l_{ij}^2=-2\tilde b(w_i,w_j)$, this is equivalent to $l_{ij}^2=\mu\,a_ia_j\tilde l_{ij}^2$, and, using (2), to $a_i=\mu^{-1/2}e^{-u_i}$. □

E Minimal Distortion

Let M be a smooth connected oriented 2-manifold with boundary, equipped with a Riemannian metric g. If $\tilde g$ is a conformally equivalent metric as in Eq. (1), then the Gaussian curvatures $K,\widetilde K$ of g and $\tilde g$ are related by Eq. (11). Thus, the metric $\tilde g$ is flat if u is a solution of the Poisson equation (12).

Now how can one measure the distortion caused by a conformal change of metric? If u is constant, the new metric differs from the old one only by a global change of scale, which we do not consider as distortion. A reasonable measure for the distortion is therefore the *Dirichlet energy* of u

$$D(u) = \tfrac{1}{2}\int_M du\wedge *du = \tfrac{1}{2}\int_M g(\mathrm{grad}\,u,\mathrm{grad}\,u)\,dA$$

which measures "how much u changes."

Theorem. *Among all conformally equivalent flat metrics $\tilde g$, the ones with least distortion are obtained if u is a solution for the Poisson equation* (12) *with $u|_{\partial M}=const$.*

Note that this measure of distortion is symmetric: Interchanging g and $\tilde g$ does not change the distortion. Note also that different choices for the constant boundary value change the solution $\tilde g$ only by a global scale factor, so one might as well choose $u|_{\partial M}=0$.

To prove the theorem, we proceed as usual in the calculus of variations. Suppose $\tilde g$ is flat and consider a variation of $\tilde g$ within the space of conformally equivalent flat metrics, *i.e.*, a variation $\dot u$ of u with $\Delta\dot u=0$. Then the variation of the Dirichlet energy is

$$\dot D = \int_M g(\mathrm{grad}\,u,\mathrm{grad}\,\dot u)\,dA = \int_{\partial M} u\cdot g(\mathrm{grad}\,\dot u,N)\,ds,$$

where N is the outward pointing unit normal vector field on the boundary. As $\dot u$ ranges over all smooth harmonic functions, $g(\mathrm{grad}\,\dot u,N)$ ranges over all smooth functions $h:\partial M\to\mathbb{R}$ satisfying

$$\int_{\partial M} h\,ds = 0. \qquad (15)$$

Indeed, precisely for all h satisfying (15), there is a harmonic $\dot u$ with $g(\mathrm{grad}\,\dot u,N)=h$ on ∂M: This is just the Neumann boundary value problem for Laplace's equation.

So $\tilde g$ is a critical point of the Dirichlet energy under variations within the space of conformally equivalent flat metrics *iff*

$$\forall h \text{ satisfying Eq. (15)}: \int_{\partial M} u\cdot h\,ds = 0.$$

This is clearly the case if $u|_{\partial M}$ is constant. To complete the argument, suppose that $u|_{\partial M}$ is not constant, so $u(p_1)>u(p_2)$ for some $p_1,p_2\in\partial M$. Then, by continuity, there are neighborhoods U_1,U_2 of p_1 and p_2 such that $u(q_1)>u(q_2)$ for all $q_1\in U_1$ and $q_2\in U_2$. Finally, consider a bump function h which is positive only inside U_1, negative only inside U_2, and zero everywhere else, and which satisfies (15) to see that u cannot be critical.

Eurographics Symposium on Geometry Processing (2007)
Alexander Belyaev, Michael Garland (Editors)

Chapter 11: Discrete Laplace operators: No free lunch

Max Wardetzky[1] Saurabh Mathur[2] Felix Kälberer[3] Eitan Grinspun[2]

[1]Now at University of Göttingen [2]Columbia University [3]Freie Universität Berlin

Abstract

Discrete Laplace operators are ubiquitous in applications spanning geometric modeling to simulation. For robustness and efficiency, many applications require discrete operators that retain key structural properties inherent to the continuous setting. Building on the smooth setting, we present a set of natural properties for discrete Laplace operators for triangular surface meshes. We prove an important theoretical limitation: discrete Laplacians cannot satisfy all natural properties; retroactively, this explains the diversity of existing discrete Laplace operators. Finally, we present a family of operators that includes and extends well-known and widely-used operators.

1. Introduction

Discrete Laplace operators on triangular surface meshes span the entire spectrum of geometry processing applications, including mesh filtering, parameterization, pose transfer, segmentation, reconstruction, re-meshing, compression, simulation, and interpolation via barycentric coordinates [Tau00, Zha04, FH05, Sor05].

In applications one often requires certain structural properties of discrete Laplacians—such as symmetry, sparsity, linear precision, positivity, and convergence—requirements that are motivated by an attempt to keep properties of the continuous case, leading to a large and diverse pool of discrete versions. What is missing is a characterization of this vast pool by means of a unified *conceptual treatment*.

As a step toward such a unified treatment, we describe a set of natural properties for discrete Laplace operators on triangular surface meshes (§2). Building on a century-old theorem by Maxwell and Cremona [Max64, Cre90], we prove an important theoretical limitation: not all the natural properties can be satisfied simultaneously, *i.e.*, a 'perfect' discrete Laplacian does not exist (§3). This result imposes a taxonomy on all discrete Laplacians, by considering those properties that they *fail* to respect. Retroactively, this explains the diversity of existing Laplacians proposed in the literature, as different applications are bound to choose different operators. We complement this analysis with a framework for constructing sparse symmetric discrete Laplacians (§4).

1.1. Properties of smooth Laplacians

Consider a smooth surface S, possibly with boundary, equipped with a Riemannian metric, *i.e.*, an intrinsic notion of distance. Let the intrinsic L^2 inner product of functions u and v on S be denoted by $(u,v)_{L^2} = \int_S uv\, dA$, and let $\Delta = -\operatorname{div}\operatorname{grad}$ denote the intrinsic smooth Laplace-Beltrami operator [Ros97]. We list salient properties of this operator:

(NULL) $\Delta u = 0$ whenever u is constant.

(SYM) Symmetry: $(\Delta u, v)_{L^2} = (u, \Delta v)_{L^2}$ whenever u and v are sufficiently smooth and vanish along the boundary of S.

(LOC) Local support: for any pair $p \neq q$ of points, $\Delta u(p)$ is independent of $u(q)$. Altering the function *value* at a distant point will not affect the *action* of the Laplacian locally.

(LIN) Linear precision: $\Delta u = 0$ whenever S is part of the Euclidean plane, and $u = ax + by + c$ is a linear function on the plane.

(MAX) Maximum principle: harmonic functions (those for which $\Delta u = 0$ in the interior of S) have no local maxima (or minima) at interior points.

(PSD) Positive semi-definiteness: the *Dirichlet energy*, $E_D(u) = \int_S \|\operatorname{grad} u\|^2\, dA$, is non-negative. By our choice of sign for Δ, we obtain $E_D(u) = (\Delta u, u)_{L^2} \geq 0$ whenever u is sufficiently smooth and vanishes along the boundary of S.

In applications, one often requires a *discrete* Laplacian having properties corresponding to (some subset of) the properties listed above.

2. Discrete Laplacians

Discrete Laplacians defined Consider a triangular surface mesh Γ, with vertex set V, edge set E, and face set F. We define a *discrete Laplace operator* on Γ by its linear action on vertex-based functions,

$$(\mathrm{L}u)_i = \sum_j \omega_{ij}(u_i - u_j)\ , \tag{1}$$

where i and j refer to vertex labels. Note that (1) automatically implies that L satisfies (NULL). Vice-versa, *any* linear operator on function values at vertices, $(\mathrm{L}u)_i = \sum_j l_{ij}u_j$, which vanishes on constants, satisfies $0 = \sum_j l_{ij}$, and can hence be written as in (1) by setting $\omega_{ij} = -l_{ij}$. The properties of L are encoded by the coefficient matrix, (ω_{ij}).

Desired properties for discrete Laplacians We describe a set of natural properties for discrete Laplacians. Each property is primarily motivated by a core structural property of the smooth Laplacian, but where possible we attempt to provide additional geometric and physical intuition.

SYMMETRY (SYM): $\omega_{ij} = \omega_{ji}$. Motivation: Real symmetric matrices exhibit real eigenvalues and orthogonal eigenvectors.

LOCALITY (LOC): Weights are associated to mesh edges (1-ring support), so that $\omega_{ij} = 0$ if i and j do not share an edge in Γ. Changing the *function value* u_j will not alter the Laplacian's *action* $(\mathrm{L}u)_i$, if i and j do not share an edge. Motivation: Smooth Laplacians govern diffusion processes via $u_t = -\Delta u$. When discretized via random walks on graphs, (ω_{ij}) are transition probabilities along *edges* of Γ.

LINEAR PRECISION (LIN): $(\mathrm{L}u)_i = 0$ at each interior vertex whenever Γ is straight-line embedded into the plane and u is a *linear* function on the plane, point-sampled at the vertices of Γ. This is *equivalent* to requiring that

$$0 = (\mathrm{L}\mathbf{x})_i = \sum_j \omega_{ij}(\mathbf{x}_i - \mathbf{x}_j) \tag{2}$$

for all interior vertex labels i, where $\mathbf{x} \in \mathbb{R}^{2|V|}$ denotes the vector of positions of the $|V|$ vertices of Γ in the plane[†]. Motivation: In graphics applications, (2) is desirable for (*i*) de-noising, where one expects to remove normal noise only but not to introduce tangential vertex drift [DMSB99], (*ii*) parameterization, where one expects planar regions to remain invariant under parameterization [FH05], and (*iii*) plate bending energies, which must vanish for flat configurations [WBH*07].

POSITIVE WEIGHTS (POS): $\omega_{ij} \geq 0$ whenever $i \neq j$. Additionally we require that for each vertex i there exists at least one vertex j such that $\omega_{ij} > 0$. Motivation: (*i*) (POS) is a sufficient condition for a *discrete maximum principle* (recall (MAX) from the smooth case). (*ii*) Physically, in diffusion problems corresponding to $u_t = -\Delta u$, (POS) assures that flow travels from regions of higher to regions of lower potential, not vice-versa. (*iii*) (POS) establishes a connection to *barycentric coordinates* by setting

$$\lambda_{ij} = \frac{\omega_{ij}}{\sum_{j\neq i}\omega_{ij}} \quad \text{so that} \quad \sum_{j\neq i}\lambda_{ij} = 1\ .$$

Indeed, u is discrete harmonic ($(\mathrm{L}u)_i = 0$ at all interior vertices) if and only if u_i is a convex combination of its neighbors ($u_i = \sum_{j\neq i}\lambda_{ij}u_j$). (*iv*) The combination (LOC)+(LIN)+(POS) is related to Tutte's embedding theorem for planar graphs [Tut63, GGT06]: positive weights associated to edges yield a straight-line embedding of an abstract planar graph. For fixed boundary vertices, this embedding is unique, and it satisfies (LIN) by construction.

POSITIVE SEMI-DEFINITENESS (PSD): L is symmetric positive semi-definite with respect to the standard inner product and has a one-dimensional kernel. Motivation: The *non-negative* discrete Dirichlet energy is given by $E_D(u) = \sum_{i,j}\omega_{ij}(u_i - u_j)^2$. Note that (SYM) and (POS) imply (PSD), but (PSD) does not imply (POS).

CONVERGENCE (CON): $\mathrm{L}_n \to \Delta$, in the sense that solutions to the discrete Dirichlet problem, involving L_n, converge to the solution of the smooth Dirichlet problem, involving Δ, under appropriate refinement conditions and in appropriate norms [HPW06]. Motivation: (CON) is indispensable when seeking to approximate solutions to PDEs.

Examples We briefly survey several Laplacians used in computer graphics. Purely *combinatorial* Laplacians [Zha04], such as the umbrella operator ($\omega_{ij} = 1$ iff vertex i and j share edge) and the Tutte Laplacian, ($\omega_{ij} = 1/d_i$, where d_i denotes the valence of vertex i) fail to be geometric, *i.e.*, they violate (LIN). Floater's *mean value weights* and the Wachspress coordinates are widely used for mesh parameterization [FH05], but violate (SYM) and (CON). The ubiquitous *cotan weights* [PP93] and their variants, commonly used for mesh de-noising, violate (POS) on general meshes.

To resolve cotan's violation of (POS), [BS05] uses the *intrinsic Delaunay* triangulation of the polyhedral surface, at the cost of violating (LOC). One could alter the definition of (LOC) so that it refers to the intrinsic Delaunay triangulation instead of the input mesh, Γ (in general these two triangulations have differing edges). Even so, an extended notion of locality would be violated: there is no universal (input-independent) integer k, such that the Delaunay edges incident to i can be computed from the knowledge *only* of a k-neighborhood of i in Γ. We refer to §3.3 for further discussion, and summarize the situation:

[†] The equivalence follows from observing that (2) implies that L vanishes on two linear functions, the $x-$ and $y-$coordinates. Since L vanishes on constants by definition, it follows that it vanishes on all linear functions.

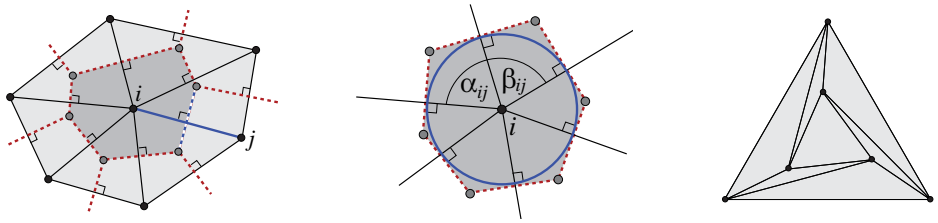


Figure 1: Left*: Primal graph (solid lines) and orthogonal dual (dashed lines), with edge e_{ij} and its dual highlighted. The dark shaded region defines the dual cell, $\star i$.* Middle*: Mean value weights correspond to dual edges tangent to the unit circle around the center vertex.* Right*: The projection of the Schönhardt polytope is not regular, so it does not allow for a discrete Laplacian satisfying* (SYM)+(LOC)+(LIN)+(POS)*.*

	SYM	LOC	LIN	POS	PSD	CON
MEAN VALUE	○	●	●	●	○	○
INTRINSIC DEL	●	○	●	●	●	?
COMBINATORIAL	●	●	○	●	●	○
COTAN	●	●	●	○	●	●

Observe that none of the Laplacians considered in graphics fulfill *all* desired properties. Even more: none of them satisfy the first four properties. This is not a coincidence:

3. No free lunch

Main result Not all meshes admit Laplacians satisfying properties (SYM), (LOC), (LIN), and (POS) simultaneously.

We prove our main result by interpreting a theorem known to Maxwell and Cremona [Max64, Cre90]. Our contribution is to relate their classical result to the study of discrete Laplacians (and barycentric coordinates) in graphics. While the technical tools used here are not new, we use them in developing the *central obstruction* to the existence of 'perfect' discrete Laplacians.

As a first step of deriving this obstruction (§3.1), we establish a correspondence between properties (SYM)+(LOC)+(LIN) and *orthogonal (reciprocal) dual graphs*, based on the Maxwell-Cremona theorem. In a second step (§3.2), we show that orthogonal duals which additionally satisfy (POS) correspond to *regular triangulations*. Since not every mesh is regular, it follows that general meshes do not admit Laplacians that satisfy (SYM)+(LOC)+(LIN)+(POS).

3.1. Geometric Laplacians and orthogonal dual graphs

Maxwell-Cremona view One may view the weights, ω_{ij}, as stresses on a planar framework (with $\omega_{ij} > 0$ corresponding to pulling stresses and $\omega_{ij} < 0$ for pushing stresses). Then (2) is the Euler-Lagrange equation of the equilibrium state of the framework when all boundary vertices are held fixed. The Maxwell-Cremona theorem states that the framework is in equilibrium *if and only if* there exists a orthogonal (reciprocal) dual framework.

Orthogonal duals Consider a planar graph, Γ, *embedded into the plane* with *straight* edges that do not cross. An *orthogonal dual* is a realization of the dual graph, $\Gamma^* = (V^*, E^*, F^*) = (F, E, V)$, in the plane, with straight edges *orthogonal* to primal edges (viewed as vectors in the plane)[‡], see Figure 1-*left*.

To relate orthogonal duals to our properties, first consider a Laplacian on Γ that satisfies (SYM)+(LOC)+(LIN). For each primal edge $\mathbf{e}_{ij}$ of Γ, viewed as a vector in the plane, we can define a corresponding *dual* edge by

$$\star\mathbf{e}_{ij} = \mathrm{R}^{90}(\omega_{ij}\mathbf{e}_{ij}) \,,$$

where R^{90} denotes rotation by 90 degrees in the plane. In general, dual edges do not necessarily form closed cycles when moving around an interior primal vertex, *i.e.*, in general, $\sum_j \star\mathbf{e}_{ij} \neq 0$. However, in our case, it is straightforward to check that (2) provides exactly the requisite cycle condition. Therefore, we obtain a realization of the dual graph in the plane whose edges are *orthogonal* to primal edges (viewed as vectors in the plane). Observe that the (straight) edges of Γ^* are allowed to cross because we allow for negative (primal) weights.

Vice versa, consider a pair (Γ, Γ^*) of a primal graph and a corresponding orthogonal dual, both embedded into the plane with straight edges. We obtain weights per primal edge via

$$\omega_{ij} := \frac{|\star\mathbf{e}_{ij}|}{|\mathbf{e}_{ij}|} \,. \tag{3}$$

Here, $|\mathbf{e}_{ij}|$ denotes the usual Euclidean length, and $|\star\mathbf{e}_{ij}|$ denotes the *signed* Euclidean length of the dual edge. The sign is obtained as follows. The dual edge, $\star\mathbf{e}_{ij}$, connects two dual vertices $\star f_1$ and $\star f_2$, corresponding to the primal faces f_1 and f_2. The sign of $|\star\mathbf{e}_{ij}|$ is *positive* if along the direction of the ray from $\star f_1$ through $\star f_2$, the primal face f_1

‡ Our definition of *orthogonal duals* is different from the one of [Aur87] who considers what we call *positive* orthogonal duals here.

lies before f_2. The sign is negative otherwise. With this sign convention, one readily checks that (3) implies (2). We therefore obtain a Laplacian satisfying (SYM)+(LOC)+(LIN).

Examples Discrete Laplacians derived from orthogonal duals on arbitrary (including non-planar) triangular surfaces were recently introduced in [Gli05], however, without noting the *equivalence* to (SYM)+(LOC)+(LIN) in the planar case. A prominent example of orthogonal duals are the cotan weights [PP93], which (as noted in [DHLM05]) arise from assigning dual vertices to *circumcenters* of primal triangles.

If we drop (SYM) from the previous discussion, we still obtain an orthogonal *dual face per primal vertex*, although these dual faces no longer fit into a consistent dual graph. When the dual edges all have positive length, we obtain an operator satisfying (LOC)+(LIN)+(POS) but not (SYM). [FHK06] explored a subspace of this case: a one-parameter family of *linear precision barycentric coordinates*, including mean value and Wachspress coordinates (see Figure 1-*middle*). [LBS06] showed that each member of this family corresponds to a specific choice of orthogonal dual face per primal vertex.

3.2. Positive Laplacians and regular triangulations

We now show the central obstruction: A triangulation of the plane allows for discrete Laplacians which satisfy (SYM)+(LOC)+(LIN)+(POS) if and only if the triangulation is regular.

While there are various equivalent definitions of *regularity* [Ede01], the above obstruction immediately follows when combining the previous discussion with an observation of Aurenhammer [Aur87]: a straight-line triangulation of the plane is regular if and only if it allows for a *positive* orthogonal dual, *i.e.*, a dual with positive weights, ω_{ij}. Unfortunately, an arbitrary input mesh, Γ, is not guaranteed to be regular, see Figure 1-*right*. This completes the proof of our main result: there are no 'perfect' discrete Laplacians for general meshes.

3.3. Discussion

Extended notion of locality To encompass additional possibilities for discrete operators, one could consider extending (LOC) from 1-rings to k-rings for some fixed $k > 1$, *i.e.*, where ω_{ij} is allowed to be non-zero if i and j are no more than k edges apart. Such an extension would accommodate, *e.g.*, methods using higher-order basis functions. The Laplacians provided in [Xu04], based on Loop subdivision bases, use $k = 2$, but they break (SYM) and (POS). We conjecture, but do not prove, that extending (LOC) to $k > 1$ does not remove the fundamental obstruction to a perfect Laplacian.

Regularity-restoring approaches Motivated by [BS05], one could attempt to circumvent the central obstruction to perfect Laplacians by considering an algorithm that first modifies the input (Γ) mesh combinatorics to ensure regularity. One might then modify the definition of (LOC) to refer to the intrinsic triangulation rather than Γ. We discuss this possibility and conjecture that this route violates another notion of locality of the Laplacian, which we call (LOC2): the existence of a universal (mesh-independent) integer k such that the weights ω_{ij} can be computed from the k-neighborhood of i in the original triangulation Γ.

As in the planar picture, one can turn any (non-flat) triangular mesh into a regular one without changing its intrinsic structure by *intrinsic edge flips* [Gli05, FSBS]. After regularity has been restored via intrinsic edge flips, one could redefine (LOC) with respect to the intrinsic triangulation, rather than Γ, to obtain Laplacians satisfying (SYM)+(LOC)+(LIN)+(POS). Unfortunately, for the specific case of an intrinsic Delaunay re-triangulation of Γ, we observed in §2 that (LOC2) would still be violated.

We conjecture that any approach that intrinsically restores regularity must violate (LOC2). Our belief stems from the link between regularity and *weighted* Delaunay triangulation [Ede01]: given a weighted Delaunay triangulation, when a vertex (arbitrarily far away from a given vertex i) is moved, the restoration of the weighted-Delaunay invariants can require re-tessellation or re-assignment of weights locally around i.

3.4. Taxonomy of the literature

In hindsight, our result explains the diversity of discrete Laplacians considered in graphics, each application choosing the subset of properties closest tailored to their needs: dropping (LOC) yields intrinsic (weighted) Delaunay (or meshless) Laplacians, dropping (SYM) gives rise to barycentric coordinates, dropping (LIN) yields combinatorial Laplacians, and dropping (POS) gives rise to cotan weights and their generalization (3).

4. General construction for discrete Laplacians

In this final section, we offer a framework for constructing discrete Laplacians using adjoint operators and L^2 inner products. We show that (SYM) and (LOC) arise from choosing *diagonal* inner products, (LOC2) holds if inner products depend only on local k-neighborhoods of Γ, (POS) corresponds to inner products with *positive* entries, (PSD) arises from *positive semi-definite* inner products, and (LIN) corresponds to a *geometric* choice.

Construction It is known from the continuous setting that the Laplacian on functions can be written as $\Delta = \delta \mathrm{d} u$, where d denotes the usual *metric-free* derivative taking 0-forms (functions) to 1-forms, and δ is the *adjoint operator*, taking 1-forms to 0-forms. Using L^2 inner products, δ is defined by the identity $(\mathrm{d}u, \alpha)_{L^2_1} = (u, \delta\alpha)_{L^2_0}$, where u is a function and α is a 1-form. Notice that d is defined independent of any metric, whereas δ cannot be defined without a metric. For

the Laplacian we obtain

$$(\Delta u, v)_{L_0^2} = (\delta \mathrm{d} u, v)_{L_0^2} = (\mathrm{d}u, \mathrm{d}v)_{L_1^2} \,. \tag{4}$$

In the discrete case, we identify 0-forms with values at vertices, and 1-forms with values at edges. The metric-independent derivative, d, taking 0-forms to 1-forms is

$$(\mathrm{d}u)(e_{ij}) = u_j - u_i \,.$$

It remains to define the adjoint operator, δ. As before, its definition is metric-dependent. Denoting edge lengths by $|e|$, we obtain L^2 inner products for 0-forms and 1-forms by summing over all vertex pairs (j, j'), respectively all edge pairs (e, e'):

$$(u,v)_{L_0^2} = \sum_{j,j'} m_{jj'} u_j v_{j'} \quad \text{and} \quad (\alpha,\beta)_{L_1^2} = \sum_{e,e'} l_{ee'} \frac{\alpha(e)}{|e|} \frac{\beta(e')}{|e'|} \,.$$

Notice that the square matrix $(m_{jj'})$ is vertex-based, while the square matrix $(l_{ee'})$ is edge-based. In the specific case of *diagonal* matrices, we can treat $(l_{ee'})$ as vertex-based by setting $l_{ij} := l_{e_{ij}e_{ij}}$. From (4) we obtain

$$(\mathrm{L}u)_i := (\Delta u, 1_i)_{L_0^2} = m_{ii}(\Delta u)_i = \sum_j \frac{l_{ij}}{|e_{ij}|^2} \left(u_i - u_j\right) \,, \tag{5}$$

where 1_i is the discrete Dirac delta function, which has unit value at vertex i and vanishes on all others. Observe that by appropriate choice of inner products, l_{ij}, we *recover all discrete Laplacians* (1) which satisfy (LOC) and (SYM).

Properties Observe that (LOC) and (SYM) are satisfied automatically in (5), (LOC2) holds if l_{ij} can be computed from local mesh information, (POS) is equivalent to $l_{ij} \geq 0$, and (PSD) is equivalent to $(\mathrm{d}u, \mathrm{d}u)_{L_1^2} \geq 0$ with equality only if u is constant. Finally, (LIN) corresponds to geometric inner products. To see this, recall from §3.1 that (LIN) corresponds to orthogonal duals. The geometric view is obtained by setting $m_{ii} = |\star i|$ (*area* of the dual cell), and $l_{ij} = |\star \mathbf{e}_{ij}||\mathbf{e}_{ij}|$ (where $|\star \mathbf{e}_{ij}|$ is signed length), exactly reproducing the weights of (3).

As a concluding remark we note that our inner product view generalizes the approach of [DHLM05], which constructs δ and Δ from a discrete Hodge star, based on circumcentric duals. Indeed, while it is straightforward to generalize the Hodge star framework of [DHLM05] from circumcentric to arbitrary orthogonal duals, it is not obvious whether this approach generalizes to Laplacians not arising from a dual mesh. In contrast, our inner product view is entirely primal-based, with the use of a dual mesh restricted to a special (geometric) case.

Acknowledgments We would like to thank Herbert Edelsbrunner and Jonathan Shewchuk for their insight on weighted-Delaunay triangulations, David Glickenstein for sharing his expertise on orthogonal duals, and the reviewers and Miklós Bergou for valuable feedback. This work was supported in part by the DFG Research Center MATHEON in Berlin and the NSF (MSPA Award No. IIS-05-28402, CSR Award No. CNS-06-14770, CAREER Award No. CCF-06-43268).

References

[Aur87] AURENHAMMER F.: A criterion for the affine equivalence of cell complexes and convex polyhedra. *Discrete Comp. Geom. 2* (1987), 49–64.

[BS05] BOBENKO A. I., SPRINGBORN B. A.: A discrete Laplace-Beltrami operator for simplicial surfaces. `arXiv:math.DG/0503219`.

[Cre90] CREMONA L.: *Graphical statics (Transl. of Le figure reciproche nelle statica graphica, Milano, 1872).* Oxford University Press, 1890.

[DHLM05] DESBRUN M., HIRANI A., LEOK M., MARSDEN J. E.: Discrete exterior calculus. `arXiv:math.DG/0508341`.

[DMSB99] DESBRUN M., MEYER M., SCHRÖDER P., BARR A. H.: Implicit fairing of irregular meshes using diffusion and curvature flow. *SIGGRAPH* (1999), 317–324.

[Ede01] EDELSBRUNNER H.: *Geometry and topology for mesh generation*. Cambridge University Press, 2001.

[FH05] FLOATER M. S., HORMANN K.: Surface Parameterization: A Tutorial and Survey. In *Advances in Multiresolution for Geometric Modeling*. 2005, pp. 157–186.

[FHK06] FLOATER M. S., HORMANN K., KÓS G.: A general construction of barycentric coordinates over convex polygons. *Adv. Comp. Math. 24*, 1–4 (2006), 311–331.

[FSBS] FISHER M., SPRINGBORN B., BOBENKO A. I., SCHRÖDER P.: An algorithm for the construction of intrinsic Delaunay triangulations with applications to digital geometry processing. *Computing*. To appear.

[GGT06] GORTLER S. J., GOTSMAN C., THURTSON D.: Discrete one-forms on meshes and applications to 3D mesh parameterization. *Computer Aided Geometric Design 23*, 2 (2006), 83–112.

[Gli05] GLICKENSTEIN D.: Geometric triangulations and discrete Laplacians on manifolds. `arxiv:math.MG/0508188`.

[HPW06] HILDEBRANDT K., POLTHIER K., WARDETZKY M.: On the convergence of metric and geometric properties of polyhedral surfaces. *Geometricae Dedicata 123* (2006), 89–112.

[LBS06] LANGER T., BELYAEV A., SEIDEL H.-P.: Spherical barycentric coordinates. In *Siggraph/Eurographics Sympos. Geom. Processing* (2006), pp. 81–88.

[Max64] MAXWELL J. C.: On reciprocal figures and diagrams of forces. *Phil. Mag. 27* (1864), 250–261.

[PP93] PINKALL U., POLTHIER K.: Computing discrete minimal surfaces and their conjugates. *Experim. Math. 2* (1993), 15–36.

[Ros97] ROSENBERG S.: *The Laplacian on a Riemannian manifold*. No. 31 in Student Texts. London Math. Soc., 1997.

[Sor05] SORKINE O.: Laplacian mesh processing. *Eurographics STAR - State of The Art Report* (2005), 53–70.

[Tau00] TAUBIN G.: Geometric signal processing on polygonal meshes. *Eurographics STAR - State of The Art Report* (2000).

[Tut63] TUTTE W. T.: How to draw a graph. *Proc. London Math. Soc. 13*, 3 (1963), 743–767.

[WBH*07] WARDETZKY M., BERGOU M., HARMON D., ZORIN D., GRINSPUN E.: Discrete Quadratic Curvature Energies. *Computer Aided Geometric Design* (2007). To appear.

[Xu04] XU G.: Discrete Laplace-Beltrami operators and their convergence. *Computer Aided Geometric Design 21*, 8 (2004), 767–784.

[Zha04] ZHANG H.: Discrete combinatorial Laplacian operators for digital geometry processing. In *Proc. SIAM Conf. Geom. Design and Comp.* (2004), pp. 575–592.

Chapter 12: Discrete Geometric Mechanics for Variational Time Integrators

Ari Stern Mathieu Desbrun

Caltech

Abstract

In this chapter, we present a *geometric*—instead of a traditional numerical-analytic—approach to the problem of time integration. Geometry at its most abstract is the study of symmetries and their associated invariants. Variational approaches based on such notions are commonly used in geometric modeling and discrete differential geometry. Here we will treat mechanics in a similar way. Indeed, the very essence of a mechanical system is characterized by its *symmetries* and *invariants*. Thus preserving these symmetries and invariants (*e.g.*, certain momenta) into the discrete computational setting is of paramount importance if one wants discrete time integration to properly capture the underlying continuous motion. Motivated by the well-known variational and geometric nature of most dynamical systems, we review the use of *discrete variational principles* as a way to derive robust, and accurate time integrators.

1 Introduction

Prediction is difficult, especially of the future.

—Mark Twain

Time Evolution of Dynamical Systems Time evolving phenomena such as the swinging of a clock pendulum, the bouncing of a soft ball on the floor, or even biological systems and stock market indicators are often modeled (*i.e.*, studied and understood) as dynamical systems. Mathematical models of the evolution in time of these systems generally involve systems of differential equations. *Solving* a physical system means figuring out how to move the system forward in time from a set of initial conditions, allowing the computation of, for instance, the trajectory of the soft ball (*i.e.*, its position as a function of time) thrown onto the floor. Although this example can easily be solved analytically, direct solutions of the differential equations governing a system are generally hard or impossible—we need to resort to *numerical techniques* to find a discrete temporal description of a motion. Consequently, there has been a significant amount of research in applied mathematics on how to deal with some of the most useful systems of equations, leading to a plethora of numerical schemes with various properties, orders of accuracy, and levels of complexity of implementation (see [Press et al. 1992] for a general overview).

Accurate vs. Qualitative Integrators While it is unavoidable to make approximations in numerical algorithms (*i.e.*, to differ from the continuous equivalent), the matter becomes whether the numerics can provide satisfactory results. The notion of satisfactory is, however, objective-dependent. If simulation is used for the design of a plane wing through a series of tests over a wide range of situations, *qualitative* reproduction of the wing behavior may be preferable over absolute numerical *accuracy*. If, however, simulation is used to find the proper launch parameters for a satellite to be put at a particular orbit, accurate results are crucial. This apparent mismatch in objectives has been, historically, aggravated by the cultural gap existing between applied and theoretical communities. We will show that in fact, one does not have to ask for *either* predictability *or* accuracy: simple methods exist that guarantee *good statistical predictability* by respecting the geometric properties of the exact flow of the differential equations, while being *also* easily rendered arbitrarily accurate.

Animation, or Simulation? In Computer Animation, time integrators are crucial computational tools at the core of most physics-based animation techniques. Animating a rigid body for instance uses the principles of classical mechanics, involving second order differential equations. In their most rudimentary form, these principles express the *relationship between forces acting on the body and its acceleration* given by *Newton's laws of motion*. From these equations of motion, classical time integrators (such as fourth-order Runge-Kutta, implicit Euler, and more recently the Newmark scheme) have been methods of choice in practice [Parent 2001; Hauth et al. 2003] to result in motions with good visual behavior—arguably, the top priority in graphics. Nonetheless, allowing the equations of motion to be slightly violated is commonly used to better control the resulting animation [Barzel et al. 1996], as long as it still looks visually plausible. In other words, local accuracy can be tinkered with just as long as the motion is still "globally" right.

Goals In this chapter, we provide an introduction to geometric mechanics, first from a continuous, then from a discrete point of view. Departing sharply from traditional numerical-analytic expositions, we point out how respecting the geometry of mechanics is not only natural, but it provides simple and powerful foundations for the design of robust time integrators. In particular, we will introduce the notion of *variational integrators* as a class of solvers specifically designed to preserve this underlying physical structure, even for large time steps that would produce overdamped or diverging results with more traditional methods.

2 Geometric Approach to Mechanics

Dynamics as a Variational Problem Considering mechanics from a variational point of view goes back to Euler, Lagrange and Hamilton. The form of the variational principle most important for continuous mechanics is due to Hamilton, and is often called *Hamilton's principle* or the *least action principle*: it states that a dynamical system always finds an optimal course from one position to another—or, as P.L. Moreau de Maupertuis put it, "Nature is thrifty in all its actions". A more formal definition will be presented in Section 4.1, but one consequence is that we can recast the traditional way of thinking about an object accelerating in response to applied forces into a geometric viewpoint. There, the path followed by the object has *optimal geometric properties*—analog to the notion of geodesics on curved surfaces. This point of view is equivalent to Newton's laws in the context of classical mechanics,

but is broad enough to encompass areas ranging to E&M and quantum mechanics.

Discrete Structure-Preserving Integrators Geometric integrators are a class of numerical time-stepping methods that exploit this geometric structure of mechanical systems [Hairer et al. 2002]. Of particular interest within this class, *variational integrators* [Marsden and West 2001] discretize the variational formulation of mechanics we mentioned above, providing a solution for most ordinary and partial differential equations that arise in mechanics. While the idea of discretizing variational formulations of mechanics is standard for elliptic problems using Galerkin Finite Element methods for instance, only recently has it been used to derive variational time-stepping algorithms for mechanical systems. This approach allows the construction of integrators with any order of accuracy [West 2003; Lew 2003], and can handle constraints as well as external forcing. Results have been shown to be equal or superior to all other types of integrators for simulations of a large range of physical phenomena [Kane et al. 2000], making this discrete-geometric framework both versatile and powerful.

Of particular interest in computer animation, the simplest variational integrator can be implemented by taking two consecutive positions $q_0 = q(t_0)$ and $q_1 = q(t_0+dt)$ of the system to compute the next position $q_2 = q(t_0 + 2dt)$. Repeating this process calculates an entire discrete (in time) trajectory. In this chapter, we describe the foundations necessary to derive such variational schemes based on geometric arguments.

3 A Motivating Example: The Pendulum

Before we delve into the details of what variational integrators are, let us first look at a simple example to exemplify how slight variations in the design of time integrators can result in widely different behaviors.

3.1 Setup and Conventions

Consider a simple pendulum of mass m and length L, swinging under the influence of the gravitational acceleration g. Let $q(t)$ represents the pendulum's angle with the vertical at time t. As this angle is the only degree of freedom for this simple example, we can express the equations of motion for this system based solely on q and its derivatives:

$$\ddot{q} = -\frac{g}{L}\sin q, \tag{1}$$

where we use the "dot" notation to represent derivatives with respect to time, *i.e.*:

$$\dot{q} := \frac{dq}{dt}, \qquad \text{and} \qquad \ddot{q} := \frac{d^2q}{dt^2}.$$

We can rewrite this equation as a system of two *coupled* first-order equations in the variables q and v:

$$\dot{q} = v \tag{2}$$

$$\dot{v} = -\frac{g}{L}\sin q \tag{3}$$

If the initial conditions $q(0)$ and $\dot{q}(0)$ are given, then we could theoretically solve this differential equation for q. Assume for a moment that we don't have access to the analytical solution to this problem (in fact, as in many cases, no such solution is known). We can only hope to *approximate* the solution using an integrator. To achieve this goal, we first discretize the problem. That is, we break up time into N equal steps of length h, so that we no longer have a continuous notion of time, but have instead a discrete set of times $t_k = kh$. Then, finding an approximation to the differential equation on our new discrete time domain is tantamount to solving for the values of the angles at the various times, *i.e.*, finding the values $q_k = q(t_k)$ for $k = 1, \ldots, N$.

Given this setup, how can we compute the q_k's? There are actually many choices, and the important point to realize is, not all of them perform equally well.

3.2 Three Numerical Schemes

Assuming that the time step h is small enough with respect to all other derivatives of q, we could leverage the well-known Taylor expansion:

$$q(t+h) = q(t) + h\dot{q}(t) + \mathcal{O}(h)\,.$$

Using this first order approximation, one can easily derive the following, straightforward update rules by applying Taylor expansion to both q and v:

$$\begin{cases} q_{k+1} = q_k + h\, v_k \\ v_{k+1} = v_k - h\, \dfrac{g}{L}\sin q_k \end{cases}$$

Given the previous values q_k, v_k, this method gives us an explicit formula to compute the next values in time q_{k+1}, v_{k+1}; this specific time integrator is called the ***explicit Euler method***. Repeating this procedure by setting $k := k + 1$ provides a way to compute the whole motion.

Alternatively, we could change the time integration procedure by evaluating the right hand sides of the former rules at the *next* time step, through:

$$\begin{cases} q_{k+1} = q_k + h\, v_{k+1} \\ v_{k+1} = v_k - h\, \dfrac{g}{L}\sin q_{k+1} \end{cases}$$

This method is no longer explicit, but *implicit*: one needs to use a (non-linear) solver to find the pair q_{k+1}, v_{k+1} that satisfy these equations, given the current values q_k and v_k. This time integrator is traditionally called the ***implicit Euler method***.

Finally, one could use a seemingly strange mix of the two, by first updating v_{k+1} explicitly, then q_{k+1} using the new value v_{k+1} (thus, still explicitly):

$$\begin{cases} v_{k+1} = v_k - h\, \dfrac{g}{L}\sin q_k \\ q_{k+1} = q_k + h\, v_{k+1} \end{cases}$$

Notice that the difference with the first scheme is rather minimal. However, this particular time integrator is known as the ***symplectic Euler method***.

These three methods are called *finite difference methods*, since they approximate the left-hand side derivatives of Eqs. (2-3) by taking the difference between consecutive values. Notice in particular that, while the implicit method is more computationally expensive, the two others involve the exact same amount of operations. Thus, their behavior should not be very different, right?

3.3 Comparing Integrators

Numerical tests of these three integrators reveal obvious differences in practice (to avoid going too much into sordid details of numerical analysis, we will stick to a fixed time step $h = 0.01$ for all

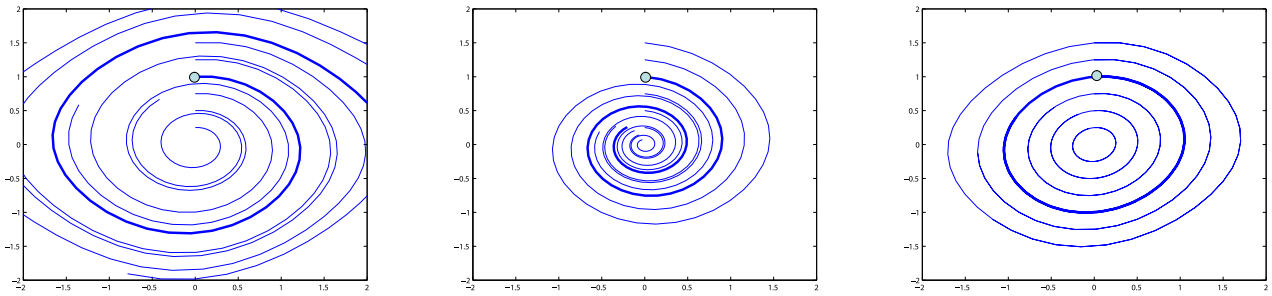


Figure 1: *Three integrators in phase space (q, p): (left) explicit, (middle) implicit, (right) symplectic. Six initial conditions are shown, with their respective trajectories; only the symplectic integrator captures the periodic nature of the pendulum. The bold trajectories correspond to the exact same initial condition.*

experiments). First, one quickly realizes that the explicit Euler suffers from stability problems: the motion of the pendulum *amplifies* over time! An obvious consequence is that the pendulum's energy increases over time, rather than being conserved. Thus, in practice, the solution often "blows up" and becomes unstable as time progresses—not a great quality for a time integrator. Fortunately, the implicit Euler *is* stable: the amplitude of the pendulum's oscillations actually *decreases* over time, avoiding any chance of numerical divergence (see Fig. 2). However, this stability comes at a cost: the pendulum *loses* energy, causing the pendulum to slow down towards a stop, even if our original equations do not include any damping forces. Effectively, we resolved the stability issue through the introduction of ***numerical dissipation***—but we induced the opposite problem instead. The symplectic method, on the other hand, both is stable *and* oscillates with constant amplitudes. This is obviously a superior method for physical simulation, given that no additional numerical operations were needed to get the correct qualitative behavior!

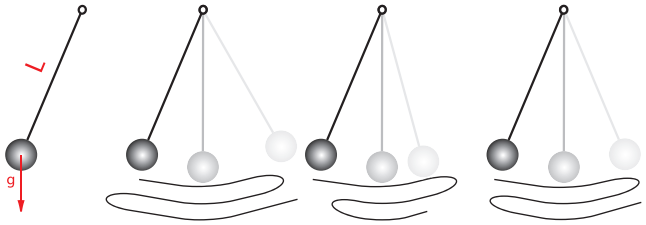


Figure 2: *The pendulum: for the equation of motion of a pendulum of length L and unit mass in a gravitation field g (left), our three integrators behave very differently: while the explicit Euler integrator exhibits amplifying oscillations, the implicit one dampens the motion, while the symplectic integrator perfectly captures the periodic nature of the pendulum.*

Now, if we are only solving for the position of the pendulum only at one particular time, it does not really matter which method we use: taking small enough time steps will guarantee arbitrarily good accuracy. However, if we wish our time integrator to be globally predictive, the least we can ask for is to get a pendulum that actually keeps on swinging. Even a simple animation of a grandfather clock or a child on a swing would look unrealistic if it seemed to gain or lose amplitude inexplicably. In other words, the behavior of energy over time is of key importance. But how do we know that an integrator will have these good properties ahead of time? Can we construct them for an arbitrary physical system? The answer, as we shall see, comes from the world of geometric mechanics and a concept called *symplecticity*.

4 Geometric Mechanics

In the familiar Newtonian view of mechanics, we begin by adding up the forces F on a body and writing the equations of motion using the famous second law,

$$F = ma, \tag{4}$$

where a represents the acceleration of the body. With geometric mechanics, however, we consider mechanics from a variational point of view. In this section, we review the basic foundations of *Lagrangian mechanics*, one of the two main flavors of geometric mechanics (we will only point to some connections with *Hamiltonian mechanics*).

4.1 Lagrangian Mechanics

Consider a finite-dimensional dynamical system parameterized by the *state variable* q, i.e., the vector containing all degrees of freedom of the system. In mechanics, a function of a position q and a velocity $\dot{q}$ called the Lagrangian function L is defined as the kinetic energy K (usually, only function of the velocity) minus the potential energy U of the system (usually, only function of the state variable):

$$L(q, \dot{q}) = K(\dot{q}) - U(q).$$

Variational Principle The *action functional* is then introduced as the integral of L along a path $q(t)$ for time $t \in [0, T]$:

$$S(q) = \int_0^T L(q, \dot{q})\, dt.$$

With this definition, the main result of Lagrangian dynamics, *Hamilton's principle*, can be expressed quite simply: this variational principle states that *the correct path of motion of a dynamical system is such that its action has a stationary value*, *i.e.*, the integral along the correct path has the same value to within first-order infinitesimal perturbations. As an "integral principle" this description encompasses the entire motion of a system between two fixed times (0 and T in our setup). In more ways than one, this principle is very similar to a statement on the *geometry* of the path $q(t)$: the action can be seen as the analog of a measure of "curvature", and the path is such that this curvature is extremized (*i.e.*, minimized or maximized).

Euler-Lagrange Equations How do we determine which path optimizes the action, then? The method is similar to optimizing an ordinary function. For example, given a function $f(x)$, we know

that its critical points exist where the derivative $\nabla f(x) = 0$. Since q is a path, we cannot simply take a "derivative" with respect to q; instead, we take something called a ***variation***. A variation of the path q is written δq, and can be thought of as an infinitesimal perturbation to the path at each point, with the important property that the perturbation is null at the endpoints of the path. Computing variations of the action induced by variations δq of the path $q(t)$ results in:

$$\delta S(q) = \delta \int_0^T L(q(t), \dot{q}(t))\, dt = \int_0^T \left[\frac{\partial L}{\partial q} \cdot \delta q + \frac{\partial L}{\partial \dot{q}} \cdot \delta \dot{q} \right] dt$$
$$= \int_0^T \left[\frac{\partial L}{\partial q} - \frac{d}{dt} \left(\frac{\partial L}{\partial \dot{q}} \right) \right] \delta q \, dt \; + \; \left[\frac{\partial L}{\partial \dot{q}} \cdot \delta q \right]_0^T ,$$

where integration by parts is used in the last equality. When the endpoints of $q(t)$ are held fixed with respect to all variations $\delta q(t)$ (*i.e.*, $\delta q(0) = \delta q(T) = 0$), the rightmost term in the above equation vanishes. Therefore, the condition of stationary action for arbitrary variations δq with fixed endpoints stated in Hamilton's principle directly indicates that the remaining integrand in the previous equation must be zero for all time t, yielding what is known as the *Euler-Lagrange equations*:

$$\frac{\partial L}{\partial q} - \frac{d}{dt} \left(\frac{\partial L}{\partial \dot{q}} \right) \; = \; 0. \tag{5}$$

For a given Lagrangian, this formula will give the *equations of motion* of the system.

Forced Systems To account for non-conservative forces or dissipation F, the least action principle is modified as follows:

$$\delta \int_0^T L(q(t), \dot{q}(t))\, dt + \int_0^T F(q(t), \dot{q}(t)) \cdot \delta q \, dt = 0 \, .$$

This is known as the *Lagrange-d'Alembert principle*.

Lagrangian vs. Hamiltonian Mechanics. Hamiltonian mechanics provides an alternative formulation, which is closely related to the Lagrangian. The reader may consult any book on mechanics for the relationships between the two descriptions. We simply point out here (as it will be useful later) that in the Hamiltonian formulation, the dynamics are described in *phase space*, i.e, the current state of a dynamical system is given as a pair (q, p), where q is the state variable, while p is the momentum, defined by $p = \partial L / \partial \dot{q}$.

4.2 Example

Let make the previous definitions more concrete by detailing a particularly simple example. Given a particle with mass M in a gravitational field, *i.e.*, in a potential field $V = Mg \cdot q$, the Lagrangian is written:

$$L(q, \dot{q}) = \frac{1}{2} \dot{q}^T M \dot{q} - Mg \cdot q.$$

Taking the variation of the action, one gets:

$$\delta \int_a^b \left(\frac{1}{2} \dot{q}^T M \dot{q} - Mg \; q \right) dt = \int_a^b \left(M\dot{q} \cdot \delta \dot{q} - Mg \cdot \delta q \right) dt.$$

Next, we integrate the $\delta \dot{q}$ term by parts; the boundary terms disappear, since $\delta q = 0$ at the endpoints.

$$= \int_a^b (-M\ddot{q} - Mg) \cdot \delta q \, dt = 0.$$

Since the integral equals 0 for any variation δq, the first term inside the integral must equal 0. Therefore, the Euler-Lagrange equations become:

$$M\ddot{q} = -Mg,$$

which are precisely the Newtonian equation of motion $F = ma$.

4.3 Symmetries and Invariants

Finally, we arrive at a crucial question: why exactly do physical systems conserve certain quantities? If we can answer this question *and* mimic the continuous dynamics in our discrete implementations, only then can we hope to get good numerical properties for our time integrators. This question is partially answered by ***Noether's theorem***, an extremely powerful theorem in physics which states that each symmetry of a system leads to a physical invariant (*i.e.*, a conserved quantity). For example, take the dynamics of an elastic object in the void. The Lagrangian can easily be shown to be translation invariant: translating all the mass particles of the elastic object would not change the value of the Lagrangian. Similarly, the Lagrangian is rotation-invariant as moving all the particles of the object by a global rotation has no reason to affect the Lagrangian either. This means that the system has a *translational and rotational symmetry*. Noether's theorem then states that the *linear and angular momenta* are preserved. These symmetries, if respected in the discrete setting, will provide equivalent discrete invariants in time integrators! In fact, we will see that these invariants can be preserved in time integrators at no extra computational cost by simply respecting the geometric, variational nature of dynamics.

4.4 Phase Space and Symplecticity

To visualize a dynamical system, we often plot its trajectories in ***phase space***. In its simplest version as in the one-dimensional pendulum case, it is in fact a phase plane where one axis represents the position q and the other axis represents either velocity $\dot{q}$ or, more usually, momentum $p = m\dot{q}$. Note that for higher dimensional systems, there is an additional axis corresponding to each additional position component q^i and its corresponding velocity $\dot{q}^i$ (or momentum p_i). The graphs that result from plotting the trajectories in phase space are called ***phase portraits***.

Going back to our motivating example of the pendulum, we can now more clearly see the qualities/flaws of the time integrators by looking at their respective phase portraits in Fig. 1. While the pendulum's phase portrait has a characteristic structure of nested, energy-preserving orbits (since the oscillations are periodic), this was not true for the two first discrete approximations: the trajectories of explicit Euler spiraled outwards (dramatically increasing magnitude of oscillations, thus energy), while those of implicit Euler spiraled inwards. Why did some of the phase portraits look better than others? How can we preserve the closedness of the orbits without making the time integrator more complicated?

One of the key features of Lagrangian flows (*i.e.*, motions) is that they are ***symplectic***. Formally, this means that the flow preserves the canonical two-form $\Omega = dq^i \wedge dp_i$. In the two-dimensional phase plane, this directly implies that *the area of any phase-space region is preserved under the flow* (see Liouville's theorem in classical mechanics). For example, let us take a given region of initial conditions in phase-space. If we advance all these states simultaneously, the regions deforms under the flow in a way that preserves the original area as shown in Fig. 3 a cat-head shaped region: this phenomenon is called *symplecticity*. However, as seen on this same figure, explicit and implicit Euler both fail the test of symplecticity. Because orbits spiral outward under explicit Euler, a region

will *expand*, and its area will increase. Conversely, implicit Euler decreases the area inside the evolving region. Preserving this property of the flow in phase space for our time integrators (that is, having them be *symplectic in a discrete sense*) is key to ensure globally correct behavior!

5 Discrete Geometric Mechanics

Having quickly reviewed classical Lagrangian mechanics in the continuous domain, we now explain how this geometric view of mechanics can elegantly be mimicked in the discrete setting.

5.1 General Idea

The driving idea behind discrete geometric mechanics is to leverage the variational nature of mechanics and to preserve this variational structure in the discrete setting. In fact, very few integrators have a variational nature: the explicit and implicit Euler methods discussed above are not variational, and not surprisingly, they both exhibited poor global behavior in the case of the pendulum. Instead of simply approximating the equations of motion to first (or higher) order as we did before, one can directly *discretize the variational principle* behind them. That is, if one designs a discrete equivalent of the Lagrangian, then discrete equations of motion can be easily derived from it by paralleling the derivations followed in continuous case. In essence, good numerical methods will come from discrete analogs to the Euler-Lagrange equations—equations that truly derive from a variational principle.

5.2 Discrete Lagrangian Dynamics

Setup The main idea is to discretize the least action principle directly rather than discretizing (5). To this end, a path $q(t)$ for $t \in [0,T]$ is replaced by a *discrete path* $q : \{t_0 = 0, t_1, \ldots, t_k, \ldots, t_N = T\}$ where $k, N \in \mathbb{N}$. Here, q_k is viewed as an approximation to $q(t_k)$.

Discrete Lagrangian The Lagrangian L is approximated on each time interval $[t_k, t_{k+1}]$ by a *discrete Lagrangian*[1] $L_d(q_k, q_{k+1}, h)$, with h being the time interval between two samples $h = t_{k+1} - t_k$ (chosen here to be constant for simplicity):

$$L_d(q_k, q_{k+1}) \approx \int_{t_k}^{t_{k+1}} L(q, \dot{q})\, dt.$$

Now, the right-hand side integral can be approximated through a one-point quadrature, i.e., by the length of the interval times the value of the integrand evaluated somewhere between q_k and q_{k+1} and with $\dot{q}$ replaced by $(q_{k+1} - q_k)/h$:

$$L_d(q_k, q_{k+1}, h) = h\, L\left((1-\alpha)q_k + \alpha q_{k+1}, \frac{q_{k+1} - q_k}{h}\right) \quad (6)$$

where $\alpha \in [0,1]$. For $\alpha = 1/2$, the quadrature is second-order accurate, while any other value leads to linear accuracy.

[1] This term could also be called an action, as it is a time integral of a Lagrangian; however, just like the term "discrete curvature" in CG refers to a small local integral of a continuous curvature, we prefer this naming convention.

Discrete Stationary Action Principle Given the discrete Lagrangian, the *discrete action functional* becomes simply a sum:

$$S_d := S_d(\{q_i\}_{i=0..N}) = \sum_{k=0}^{N-1} L_d(q_k, q_{k+1}) \approx \int_a^b L(q, \dot{q})\, dt = S(q).$$

Taking fixed-endpoint variations of this discrete action S_d, we obtain:

$$\delta S_d = \sum_{k=0}^{N-1} \Big[D_1 L_d(q_k, q_{k+1}) \cdot \delta q_k + D_2 L_d(q_k, q_{k+1}) \cdot \delta q_{k+1} \Big],$$

where $D_1 L$ (resp., $D_2 L$) denotes the partial derivative with respect to the first (resp., second) arguments of L. Reindexing the rightmost terms, and using the fixed endpoint condition $\delta q_0 = \delta q_N = 0$, one gets:

$$\delta S_d = \sum_{k=1}^{N-1} \Big[D_1 L_d(q_k, q_{k+1}) + D_2 L_d(q_{k-1}, q_k) \Big] \cdot \delta q_k.$$

Setting this variation equal to 0 and noting that each δq_k is arbitrary, we arrive at the ***discrete Euler-Lagrange (DEL) equations***

$$D_1 L_d(q_k, q_{k+1}) + D_2 L_d(q_{k-1}, q_k) = 0. \quad (7)$$

Notice that this condition only involves three consecutive positions. Therefore, for two given successive positions q_k and q_{k+1}, Eq. (7) *defines* q_{k+2}. That is, these equations of motion are actually the algorithm for an integrator! And since the DEL equations derive from the extremization of a discrete action, such an algorithm enforces the variational aspect of the motion numerically.

Link to Previous Numerical Schemes Let us go back to the pendulum case. For this system, the Lagrangian (kinetic energy minus potential energy) is:

$$L(q, \dot{q}) = \frac{1}{2} L^2 \dot{q}^2 + gL\, \cos(q).$$

First, the user can convince her/himself that the Euler-Lagrange equation is indeed, Eq. (1) through a simple derivation. Second, it is also a simple (yet, interesting) exercise to verify that the symplectic Euler integrator used earlier results from the DEL equations just described, for the particular choice of $\alpha = 0$ in the quadrature rule defined in Eq. 6.

5.3 Update Rule in Phase Space

In mechanics, the initial conditions are typically specified as a position and a velocity or momentum rather than two positions, therefore it is beneficial to write (7) in a position-momentum form [West 2003]. To this end, define the momentum at time t_k to be:

$$p_k := D_2 L_d(q_{k-1}, q_k) = -D_1 L_d(q_k, q_{k+1})$$

where the second equality holds due to (7). The position-momentum form of the variational integrator discussed above is then given by:

$$p_k = -D_1 L_d(q_k, q_{k+1}), \qquad p_{k+1} = D_2 L_d(q_k, q_{k+1}). \quad (8)$$

For (q_k, p_k) known, (8)(left) is an (often implicit) equation whose solution gives q_{k+1}. q_{k+1} is then substituted in (8)(right) to find p_{k+1}. This provides an update rule in phase space.

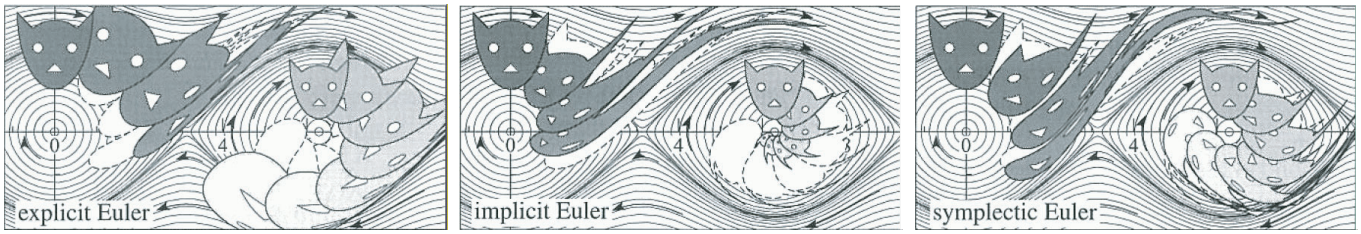


Figure 3: *Symplecticity [reproduced from [Hairer et al. 2002]]: while a continuous Lagrangian system is symplectic (that is to say, in this simple case, an area in phase space evolves along the flow without changing its area), discrete time integrators rarely share this property. From our three time integrators compared in Section 3, only the last one is symplectic. In the background, the reader will recognize the shape of the orbits obtained in Fig. 1(right).*

5.4 Adding Dissipation

In case of forcing and/or dissipation, the discrete action can be modified by adding the non-conservative force term and using the discrete Lagrange-d'Alembert principle [Marsden and West 2001]:

$$\delta S_d + \sum_{k=0}^{N} \left(F_d^-(q_k, q_{k+1}) \cdot \delta q_k + F_d^+(q_k, q_{k+1}) \cdot \delta q_{k+1} \right) = 0.$$

where $F_d^-(q_k, q_{k+1})$ and $F_d^+(q_k, q_{k+1})$ are discrete external forces acting respectively on the right of q_k and on the left of q_{k+1}. In other words, $F_d^-(q_k, q_{k+1}) \cdot \delta q_k + F_d^+(q_k, q_{k+1}) \cdot \delta q_{k+1}$ can be seen as a two-point quadrature of the continuous forcing term $\int_{t_k}^{t_{k+1}} F \cdot \delta q \, dt$. The *forced discrete Euler-Lagrange equations* can be expressed in a convenient, position-momentum form as follows:

$$p_k = -D_1 L_d(q_k, q_{k+1}) - F_d^-(q_k, q_{k+1}),$$
$$p_{k+1} = D_2 L_d(q_k, q_{k+1}) + F_d^+(q_k, q_{k+1}).$$

This variational treatment of energy decay, despite its simplicity, has also been proven superior to the usual time integration schemes that often add numerical viscosity to get stability [West 2003].

5.5 Last Words

Variational integrators often perform better than their non-variational counterparts because they preserve the *underlying geometry* of the physical system. This has two important consequences. First, the integrators are guaranteed to be symplectic, which in practice will result in excellent energy behavior, rather than perpetual damping or blowing up. Second, they are also guaranteed to preserve discrete momenta of the system (via a discrete version of Noether's theorem). As a consequence, simulations and animations using these integrators usually have great physical and visual fidelity with low computational cost, even for dissipative systems (see [Kharevych et al. 2006] for a discussion on damping in animation). To build upon this short introduction, the reader is invited to investigate recent developments in variational integrators, such as Lie group integrators, Hamilton-Pontryagin integrators, asynchronous variational updates (where timesteps are different for each mesh element), and stochastic variational integrators.

Caveat: The reader may be misled into thinking that explicit variational schemes does not require the typical Courant-Friedrichs-Levy (CFL) condition (or equivalent) on the time step size. This is, of course, untrue: the same usual theoretical limitations of explicit schemes are still valid for symplectic explicit schemes. However, we can easily design symplectic implicit schemes that do not share this particular limitation, generally allowing for much larger time steps. Finally, we can make them of arbitrarily higher order by simply improving the quadrature rule used to convert the continuous Lagrangian into a discrete Lagrangian.

Aknowledgements The authors are most grateful to Yiying Tong, Jerrold E. Marsden, Eva Kanso, Lily Kharevych, and Patrick Mullen for their constant help.

References

BARZEL, R., HUGHES, J., AND WOOD, D. N. 1996. Plausible motion simulation for computer graphics animation. In *EG Workshop on Computer Animation and Simulation*, 183–197.

HAIRER, E., LUBICH, C., AND WANNER, G. 2002. *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations*. Springer.

HAUTH, M., ETZMUSS, O., AND STRASSER, W. 2003. Analysis of Numerical Methods for the Simulation of Deformable Models. *The Visual Computer 19*, 7-8, 581–600.

KANE, C., MARSDEN, J. E., ORTIZ, M., AND WEST, M. 2000. Variational Integrators and the Newmark Algorithm for Conservative and Dissipative Mechanical Systems. *Int. J. Numer. Methods Engrg. 49*, 1295–1325.

KHAREVYCH, L., WEIWEI, TONG, Y., KANSO, E., MARSDEN, J. E., SCHRÖDER, P., AND DESBRUN, M. 2006. Geometric, variational integrators for computer animation. In *ACM/EG Symposium on Computer animation*, 43–51.

LEW, A. 2003. *Variational Time Integrators in Computational Solid Mechanics*. Phd thesis, California Institute of Technology.

MARSDEN, J. E., AND WEST, M. 2001. Discrete Mechanics and Variational Integrators. *Acta Numerica*, 357–515.

PARENT, R. 2001. *Computer Animation: Algorithms and Techniques*. Series in Computer Graphics. Morgan Kaufmann.

PRESS, W. H., FLANNERY, B. P., TEUKOLSKY, S. A., AND VETTERLING, W. T. 1992. *Numerical Recipes in C: The Art of Scientific Computing*, 2nd ed. Cambridge University Press.

WEST, M. 2003. *Variational Integrators*. PhD thesis, California Institute of Technology.