Discrete Surfaces

Pierre Alliez
Inria
Outline

• Parametric approximations
• Polygon meshes
• Data structures
• Discrete differential geometry
Parametric Representation

- Surface is the range of a function

\[ f : \Omega \subset \mathbb{R}^2 \rightarrow \mathbb{R}^3, \quad S_\Omega = f(\Omega) \]

- 2D example: Circle

\[ f : [0, 2\pi] \rightarrow \mathbb{R}^2 \]

\[ f(t) = \begin{pmatrix} r \cos(t) \\ r \sin(t) \end{pmatrix} \]
Parametric Representation

• Surface is the range of a function

\[ f : \Omega \subset \mathbb{R}^2 \rightarrow \mathbb{R}^3, \quad S_\Omega = f(\Omega) \]

• 2D example: Island coast line

\[ f : [0, 2\pi] \rightarrow \mathbb{R}^2 \]

\[ f(t) = \left( \begin{array}{c}
??? \\
???
\end{array} \right) \]
Piecewise Approximation

- Surface is the range of a function
  \[ f : \Omega \subset \mathbb{R}^2 \rightarrow \mathbb{R}^3, \quad S_\Omega = f(\Omega) \]
- 2D example: Island coast line
  \[ f : [0, 2\pi] \rightarrow \mathbb{R}^2 \]
  \[ f(t) = \begin{pmatrix} ??? \\ ??? \end{pmatrix} \]
Polynomial Approximation

- Polynomials are computable functions
  \[ f(t) = \sum_{i=0}^{p} c_i t^i = \sum_{i=0}^{p} \tilde{c}_i \phi_i(t) \]

- Taylor expansion up to degree \( p \)
  \[ g(h) = \sum_{i=0}^{p} \frac{1}{i!} g^{(i)}(0) h^i + O(h^{p+1}) \]

- Error for approximating \( g \) by polynomial \( f \)
  \[ f(t_i) = g(t_i), \quad 0 \leq t_0 < \cdots < t_p \leq h \]
  \[ |f(t) - g(t)| \leq \frac{1}{(p+1)!} \max f^{(p+1)} \prod_{i=0}^{p} (t - t_i) = O(h^{p+1}) \]
Polynomial Approximation

• Approximation error is $O(h^{p+1})$

• Improve approximation quality by
  – increasing $p$ ... higher order polynomials
  – decreasing $h$ ... smaller / more segments

• Issues
  – smoothness of the target data ($\max_t f^{(p+1)}(t)$)
  – smoothness conditions between segments
Polygon Meshes

- Polygonal meshes are a good compromise
  - Piecewise linear approximation $\rightarrow$ error is $O(h^2)$
Polygon Meshes

• Polygonal meshes are a good compromise
  – Piecewise linear approximation → error is $O(h^2)$
  – Error inverse proportional to #faces
Polygon Meshes

• Polygonal meshes are a good compromise
  – Piecewise linear approximation $\rightarrow$ error is $O(h^2)$
  – Error inverse proportional to #faces
  – Arbitrary topology surfaces
Polygon Meshes

- Polygonal meshes are a good compromise
  - Piecewise linear approximation $\rightarrow$ error is $O(h^2)$
  - Error inverse proportional to #faces
  - Arbitrary topology surfaces
  - Piecewise smooth surfaces
Polygon Meshes

- Polygonal meshes are a good compromise
  - Piecewise linear approximation $\rightarrow$ error is $O(h^2)$
  - Error inverse proportional to $\#$ faces
  - Arbitrary topology surfaces
  - Piecewise smooth surfaces
  - Curvature adaptive sampling
POLYGON MESHES
Graph Definitions

Graph \{V, E\}
Graph Definitions

Graph \( \{ V, E \} \)
Vertices \( V = \{ A, B, C, \ldots, K \} \)
Graph Definitions

Graph \( \{V, E\} \)

Vertices \( V = \{A, B, C, \ldots, K\} \)

Edges \( E = \{(AB), (AE), (CD), \ldots\} \)
Graph Definitions

Graph \{V, E\}

Vertices \(V = \{A, B, C, \ldots, K\}\)

Edges \(E = \{(AB), (AE), (CD), \ldots\}\)

Faces \(F = \{(ABE), (EBF), (EFIH), \ldots\}\)
Graph Definitions

Vertex degree or valence: number of incident edges.

$$\text{deg}(A) = 4$$
$$\text{deg}(E) = 5$$
**Graph Definitions**

**Connected**: Path of edges connecting every two vertices.
Graph Definitions

**Connected:** Path of edges connecting every two vertices.

**Subgraph:** Graph \(\{V', E'\}\) is a subgraph of graph \(\{V, E\}\) if \(V'\) is a subset of \(V\) and \(E'\) is a subset of \(E\) incident on \(V'\).
Graph Definitions

**Connected**: Path of edges connecting every two vertices.

**Subgraph**: Graph \( \{V', E'\} \) is a subgraph of graph \( \{V, E\} \) if \( V' \) is a subset of \( V \) and \( E' \) is a subset of \( E \) incident on \( V' \).

**Connected component**: Maximally connected subgraph.
**Connected**: Path of edges connecting every two vertices.

**Subgraph**: Graph \( \{V', E'\} \) is a subgraph of graph \( \{V, E\} \) if \( V' \) is a subset of \( V \) and \( E' \) is a subset of \( E \) incident on \( V' \).

**Connected component**: Maximally connected subgraph.
**Graph Embedding**

**Embedding:** Graph is embedded in $\mathbb{R}^d$, if each vertex is assigned a position in $\mathbb{R}^d$. 

![Graph Embedding Diagram](image)
Graph Embedding

**Embedding**: Graph is embedded in $\mathbb{R}^d$, if each vertex is assigned a position in $\mathbb{R}^d$. Embedded in $\mathbb{R}^3$
**Triangulation**: Graph where every face is a triangle.

Why...?

- simplifies data structures
- simplifies rendering
- simplifies algorithms
- by definition, triangle is planar
- any polygon can be triangulated
**Topological Genus**

**Genus**: Maximal number of closed simple cutting curves that do not disconnect the graph into multiple components.

(Informally, the number of holes or handles.)

Genus 0  |  Genus 1  |  Genus 2  |  Genus 3
Euler-Poincare Formula

For a closed polygonal mesh of genus $g$, the relation of the number $V$ of vertices, $E$ of edges, and $F$ of faces is given by Euler’s formula

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

Term $2(1-g)$: Euler characteristic
Euler Consequences

• Triangle meshes
  – $F \approx 2V$
  – $E \approx 3V$
  – Average valence = 6

• Quad meshes
  – $F \approx V$
  – $E \approx 2V$
  – Average valence = 4
Two-Manifold Surfaces

- Local neighborhoods are disk-shaped

\[ f(D_\varepsilon[u,v]) = D_\delta[f(u,v)] \]

- Guarantees meaningful neighbor enumeration
  - required by most algorithms

- Non-manifold examples:
DATA STRUCTURES
Mesh Data Structure

How to store geometry & connectivity?

Compact storage

File formats

Efficient algorithms on meshes

Identify time-critical operations

All vertices/edges of a face

All incident vertices/edges/faces of a vertex
Data Structure

What should be stored?

• Geometry: 3D coordinates

• Attributes: normal, color, texture coordinate (per vertex, per face, per edge)

• Connectivity

  What is adjacent to what
Data Structure

What should it support?
• Rendering
• Queries
  – What are the vertices of face #3?
  – Is vertex #6 adjacent to vertex #12?
  – Which faces are adjacent to face #7?
• Modifications
  – Remove/add a vertex/face
  – Vertex split, edge collapse
Data Structure

• How good is it?
  – Time to construct (preprocessing)
  – Time to answer a query
  – Time to perform an operation
  – Space complexity
  – Redundancy
Face Set

- **Face**: 3 positions

<table>
<thead>
<tr>
<th>Triangles</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_{11}$ $y_{11}$ $z_{11}$</td>
</tr>
<tr>
<td>$x_{21}$ $y_{21}$ $z_{21}$</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>$x_{F1}$ $y_{F1}$ $z_{F1}$</td>
</tr>
</tbody>
</table>

\[36 \, \text{B/f} = 72 \, \text{B/v} \]
no connectivity!
Vertices

- \(v_1\) \((x_1; y_1; z_1)\)
- \(v_2\) \((x_2; y_2; z_2)\)
- \(v_3\) \((x_3; y_3; z_3)\)
- \(v_4\) \((x_4; y_4; z_4)\)
- \(v_5\) \((x_5; y_5; z_5)\)
- \(v_6\) \((x_6; y_6; z_6)\)
- \(v_7\) \((x_7; y_7; z_7)\)

Connectivity

- \(f_1\) \((v_1; v_3; v_2)\)
- \(f_2\) \((v_4; v_3; v_1)\)
- \(f_3\) \((v_4; v_1; v_5)\)
- \(f_4\) \((v_1; v_6; v_5)\)
- \(f_5\) \((v_6; v_1; v_7)\)
- \(f_6\) \((v_2; v_7; v_1)\)
- \(f_7\) \((...\))
Shared Vertices

- Indexed Face List
  - Vertex: position
  - Face: vertex indices

\[
\begin{array}{ccc}
\text{Vertices} & \text{Triangles} \\
 \begin{array}{ccc}
 x_1 & y_1 & z_1 \\
 \cdots \\
 x_v & y_v & z_v \\
 \cdots \\
 \end{array} & \begin{array}{ccc}
 i_{11} & i_{12} & i_{13} \\
 \cdots \\
 \cdots \\
 i_{F_1} & i_{F_2} & i_{F_3} \\
 \cdots \\
 \end{array} \\
\end{array}
\]

12 \ B/v + 12 \ B/f = 36 \ B/v

no neighborhood info
Face-based Connectivity

- **Vertex:**
  - position
  - 1 face

- **Face:**
  - 3 vertices
  - 3 face neighbors

64 B/v
no edges!
Edge-Based Connectivity

- **Vertex**
  - position
  - 1 edge

- **Edge**
  - 2 vertices
  - 2 faces
  - 4 edges

- **Face**
  - 1 edge

120 B/v
edge orientation?
Halfedge-Based Connectivity

- **Vertex**
  - position
  - 1 halfedge

- **Halfedge**
  - 1 vertex
  - 1 face
  - 1, 2, or 3 halfedges

- **Face**
  - 1 halfedge

96 to 144 B/v
no case distinctions during traversal
Example: One-ring Traversal

1. Start at vertex
2. Outgoing halfedge
3. Opposite halfedge
4. Next halfedge
5. Opposite
6. Next
7. ...
DISCRETE DIFFERENTIAL GEOMETRY
Discrete Curvatures

How to discretize curvatures on a mesh?
– Zero curvature within triangles
– Infinite curvature at edges / vertices
– Pointwise definition does not make sense
Discrete Curvatures

Approximate differential properties at point $x$ as average over local neighborhood $A(x)$

- $x$ is a mesh vertex
- $A(x)$ within one-ring neighborhood
Discrete Curvatures

Approximate differential properties at point $\mathbf{x}$ as average over local neighborhood $A(\mathbf{x})$

$$K(v) \approx \frac{1}{A(v)} \int_{A(v)} K(\mathbf{x}) \, dA$$
Discrete Curvatures

Which curvatures to discretize?
– Discretize Laplace-Beltrami operator
– Laplace-Beltrami gives us mean curvature $H$
– Discretize Gaussian curvature $K$
– From $H$ and $K$ we can compute $\kappa_1$ and $\kappa_2$
Laplace Operator

\[ \Delta f = \text{div} \nabla f = \sum_i \frac{\partial^2 f}{\partial x_i^2} \]
Laplace Operator

Let $f$ be a function on a manifold $S$. The Laplace-Beltrami operator is given by

$$\Delta_S f = \text{div}_S \nabla_S f$$

- $\Delta_S f$ is the Laplace-Beltrami operator.
- $\text{div}_S$ is the divergence operator on manifold $S$.
- $\nabla_S f$ is the gradient of $f$ on manifold $S$. 
Laplace Operator

\[ \Delta_S x = \text{div}_S \nabla_S x = -2Hn \]

- Laplace-Beltrami
- Gradient operator
- Mean curvature
- Coordinate function
- Divergence operator
- Surface normal
Laplace Operator on Meshes?

- Extend finite differences to meshes?
  - What weights per vertex / edge?

1D grid

2D grid

2D/3D mesh
Uniform Laplace

- Uniform discretization

\[ \Delta_{\text{uni}} f(v_i) := \frac{1}{|N_1(v_i)|} \sum_{v_j \in N_1(v_i)} (f(v_j) - f(v_i)) \]

- Properties
  - depends only on connectivity
  - simple and efficient
  - bad approximation for irregular triangulations
Uniform Laplace

• Uniform discretization

\[ \Delta_{\text{uni}} x_i := \frac{1}{|N_1(v_i)|} \sum_{v_j \in N_1(v_i)} (x_j - x_i) \approx -2Hn \]

• Properties
  – depends only on connectivity
  – simple and efficient
  – bad approximation for irregular triangulations
    • can give non-zero $H$ for planar meshes
    • tangential drift for mesh smoothing

Curvature flow
Barycentric Cells

Connect edge midpoints and triangle barycenters

– Simple to compute
– Area: 1/3 of triangle areas
– Slightly wrong for obtuse triangles
Mixed Cells

Connect edge midpoints and
– Circumcenters for non-obtuse triangles
– Midpoint of opposite edge for obtuse triangles
– Better approximation, more complex to compute.
“Cotan” Laplace

- Piecewise linear functions

\[ f(u) = f_i B_i(u) + f_j B_j(u) + f_k B_k(u) \]

function value at vertex

linear basis function

\[ B_i(x), B_j(x), B_k(x) \]
Reminder: Triangle barycentric coordinates

\[ b_1 = \frac{A_1}{A} \quad b_2 = \frac{A_2}{A} \quad b_3 = \frac{A_3}{A} \]

\[ A = A_1 + A_2 + A_3 \]

A. F. Möbius
[1790–1868]
“Cotan” Laplace

• Piecewise linear functions

\[ f(\mathbf{u}) = f_i B_i(\mathbf{u}) + f_j B_j(\mathbf{u}) + f_k B_k(\mathbf{u}) \]

– Gradient

\[ \nabla f(\mathbf{u}) = f_i \nabla B_i(\mathbf{u}) + f_j \nabla B_j(\mathbf{u}) + f_k \nabla B_k(\mathbf{u}) \]
Cotan Laplace

- Piecewise linear functions

\[ f(\mathbf{u}) = f_i B_i(\mathbf{u}) + f_j B_j(\mathbf{u}) + f_k B_k(\mathbf{u}) \]

- Gradient

\[ \nabla f(\mathbf{u}) = f_i \nabla B_i(\mathbf{u}) + f_j \nabla B_j(\mathbf{u}) + f_k \nabla B_k(\mathbf{u}) \]

\[ \nabla B_i(\mathbf{u}) = \frac{(x_k - x_j) \perp}{2 \, A_T} \]
Cotan Laplace

• Divergence Theorem

$$\int_{A_i} \text{div} \mathbf{F}(u) \, dA = \int_{\partial A_i} \mathbf{F}(u) \cdot \mathbf{n}(u) \, ds$$

– Applied to Laplacian

$$\int_{A_i} \Delta f(u) \, dA = \int_{A_i} \text{div} \nabla f(u) \, dA = \int_{\partial A_i} \nabla f(u) \cdot \mathbf{n}(u) \, ds$$
Discrete Laplace-Beltrami

- Cotangent discretization

\[ \Delta_S f(v) := \frac{1}{2A(v)} \sum_{v_i \in \mathcal{N}_1(v)} (\cot \alpha_i + \cot \beta_i) (f(v_i) - f(v)) \]

- Problems
  - weights can become negative (when?)
  - depends on triangulation

- Still the most widely used discretization
Discrete Curvatures

- Mean curvature (absolute value)
  \[ H = \frac{1}{2} ||\Delta_s x|| \]
- Gaussian curvature
  \[ K = (2\pi - \sum_j \theta_j)/A \]
- Principal curvatures
  \[ \kappa_1 = H + \sqrt{H^2 - K} \quad \kappa_2 = H - \sqrt{H^2 - K} \]
PRINCIPAL CURVATURES
Principal Curvatures

Principal curvature directions  Lines of curvatures
Principal Curvature Directions

We can define curvatures at an edge $e$ in terms of the angle $\beta(e)$ between curve segments*:

– The min/max curvature is 0, with principal curvature direction along $e$.

– The max/min curvature is equal to the dihedral angle ($\beta(e)=\angle n_1 n_2$), with principal curvature direction along $n_e e$.

* [Cohen-Steiner et al. ‘03]
Principal Curvature Directions

This allows us to define a 3x3 curvature tensor along the edge $e$ as the symmetric matrix with eigenvalue $\beta(e)$ in the direction across $e$ and eigenvalues of 0 in perpendicular directions:

$$
\mathbf{C}(p \in e) = \beta(e) \frac{1}{\|n_e \times e\|^2} (n_e \times e)(n_e \times e)^t
$$
Principal Curvature Directions

This, in turn, allows us to define the curvature tensor around a vertex \( v \), average over a neighborhood \( B \) around \( v \):

\[
C(v) = \frac{1}{|B|} \sum_e |B \cap e| \beta(e) \frac{1}{\|n_e \times e\|^2} (n_e \times e)(n_e \times e)^t
\]
Principal Curvature Directions

\[ C(v) = \frac{1}{|B|} \sum_{e} |B \cap e\beta(e) \frac{1}{\|n_e \times e\|^2} (n_e \times e)(n_e \times e)^t \]

Computing the eigen-decomposition of the curvature tensor we get an estimate of:

– The normal: The eigenvector with smallest absolute eigenvalue.

– The principal directions and values: The other two eigenvectors and their associated eigenvalues.
Principal Curvature Directions

$$C^{(v)} = \frac{1}{|B|} \sum_{e \cap e} \beta(e) \frac{1}{\|n_e \times e\|^2} (n_e \times e)(n_e \times e)^t$$

**Note:**
When the two principal directions have the same principal curvature values, the principal directions are not well defined.
Principal Curvature Directions

\[ C(v) = \frac{1}{|B|} \sum_{e} |B \cap e| \beta(e) \frac{1}{\|n_e \times e\|^2} (n_e \times e)(n_e \times e)^t \]

**Note:**
When the two principal directions have the same principal curvature values, the principal directions are not well defined.
Such points are called **umbilical** points.
Umbilic Points
Principal Direction Fields

Linear singularities

**Trisector**

**Wedge**

**Umbilic**
(spherical point)
2D tensor
proportional to identity

Topology of tensor fields.
[Delmarcelle & Hesselink 94]
[Tricoche 02]
What are the principal curvature lines?

Assuming that we are away from the umbilical points, we can define two vector fields:

1. $v_{\text{min}}$: Aligns with the min. curvature
2. $v_{\text{max}}$: Aligns with the max. curvature
Principal Curvature Lines

What are the principal curvature lines?

Assuming that we are away from the umbilical points, we can define two vector fields:

1. $v_{\text{min}}$: Aligns with the min. curvature
2. $v_{\text{max}}$: Aligns with the max. curvature

Given a starting $p$, solve the diff. eq.:

$$\gamma_{\text{min/max}}'(t) = v_{\text{min/max}}(\gamma(t)) \quad \text{s.t.} \quad \gamma(0) = p$$
Principal Curvature Lines

How far should we integrate?

We should integrate the min/max curves until they are within a prescribed density:

1. Accuracy of the remesh
2. Local curvature
Principal Curvature Lines

Q: If the user wants the remeshed surface to be within a distance of $\varepsilon$ from the original surface, how far should the minimal/maximal curvature lines be from each other?
Principal Curvature Lines

A: Consider the surface between two lines of minimal/maximal curvature:
Principal Curvature Lines

A: Consider the surface between two lines of minimal/maximal curvature:
The curve between them will follow the maximal/minimal curvature direction.
Principal Curvature Lines

A: Consider the surface between two lines of minimal/maximal curvature:
The curve between them will follow the maximal/minimal curvature direction.
The curve will be, roughly, a circular arc with radius equal to one over the maximal/minimal curvature.
Principal Curvature Lines

Looking at this in cross section, we choose the distance $d$ between the curves so that the distance to the surface is below a threshold $\varepsilon$. 

![Diagram of principal curvature lines with distance $d$ and threshold $\varepsilon$.]
Principal Curvature Lines

Looking at this in cross section, we choose the distance $d$ between the curves so that the distance to the surface is below a threshold $\varepsilon$.

Denoting the distance by $\varepsilon$ we get:

$$
\left( \frac{d}{2} \right)^2 + \left( \frac{1}{|\kappa|} - \varepsilon \right)^2 = \left( \frac{1}{|\kappa|} \right)^2
$$
Principal Curvature Lines

Looking at this in cross section, we choose the distance $d$ between the curves so that the distance to the surface is below a threshold $\varepsilon$.

Denoting the distance by $\varepsilon$ we get:

\[
\left(\frac{d}{2}\right)^2 + \left(\frac{1}{|\kappa|} - \varepsilon\right)^2 = \left(\frac{1}{|\kappa|}\right)^2
\]

\[
d = 2\sqrt{\varepsilon\left(\frac{2}{|\kappa|} - \varepsilon\right)}
\]
Principal Curvature Lines
Literature

Taubin: *A signal processing approach to fair surface design*, SIGGRAPH 1996.
– Alexa, Wardetzky: *Discrete Laplacians on General Polygonal Meshes*, SIGGRAPH 2011