Discrete Surfaces

Pierre Alliez

Inria

Outline

- Parametric approximations
- Polygon meshes
- Data structures
- Discrete differential geometry

Parametric Representation

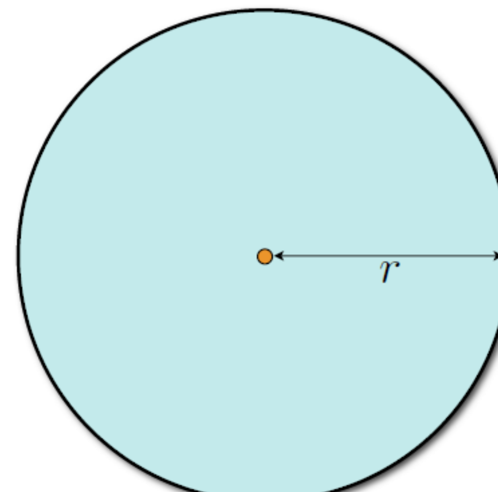
- Surface is the range of a function

$$\mathbf{f} : \Omega \subset \mathbb{R}^2 \to \mathbb{R}^3, \quad \mathcal{S}_\Omega = \mathbf{f}(\Omega)$$

- 2D example: Circle

$$\mathbf{f} : [0, 2\pi] \to \mathbb{R}^2$$

$$\mathbf{f}(t) = \begin{pmatrix} r\cos(t) \\ r\sin(t) \end{pmatrix}$$



Parametric Representation

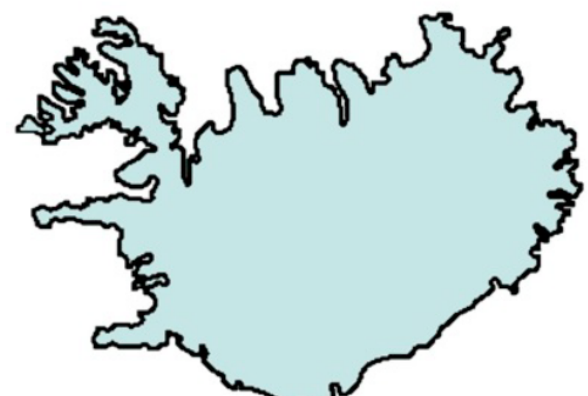
- Surface is the range of a function

$$\mathbf{f} : \Omega \subset \mathbb{R}^2 \rightarrow \mathbb{R}^3, \quad \mathcal{S}_\Omega = \mathbf{f}(\Omega)$$

- 2D example: Island coast line

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

$$\mathbf{f}(t) = \begin{pmatrix} ??? \\ ??? \end{pmatrix}$$



Piecewise Approximation

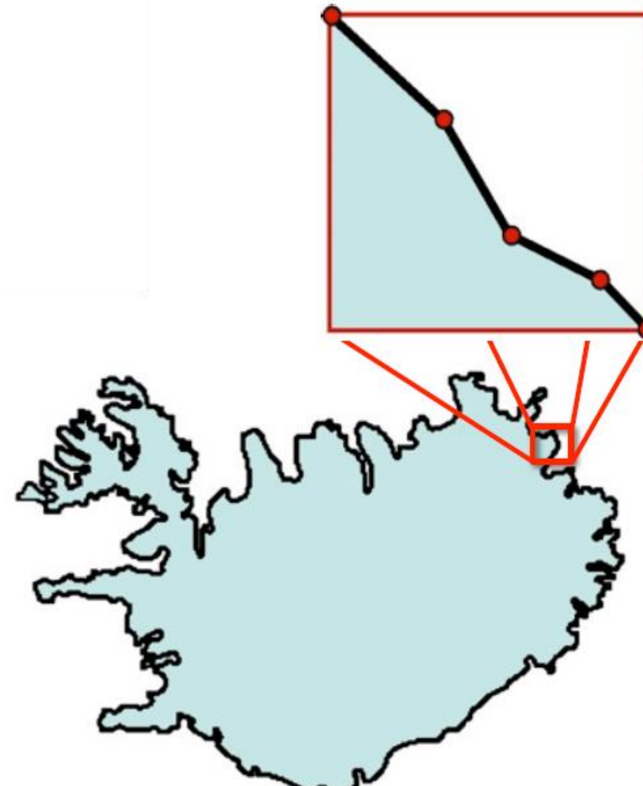
- Surface is the range of a function

$$\mathbf{f} : \Omega \subset \mathbb{R}^2 \to \mathbb{R}^3, \quad \mathcal{S}_\Omega = \mathbf{f}(\Omega)$$

- 2D example: Island coast line

$$\mathbf{f} : [0, 2\pi] \to \mathbb{R}^2$$

$$\mathbf{f}(t) = \begin{pmatrix} ??? \\ ??? \end{pmatrix}$$



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*

$$g(h) \;=\; \sum_{i=0}^{p} \frac{1}{i!}\, g^{(i)}(0)\; h^i \;+\; O\big(h^{p+1}\big)$$

- Error for approximating *g* by polynomial *f*

$$f(t_i) = g(t_i)\,, \quad 0 \le t_0 < \cdots < t_p \le h$$

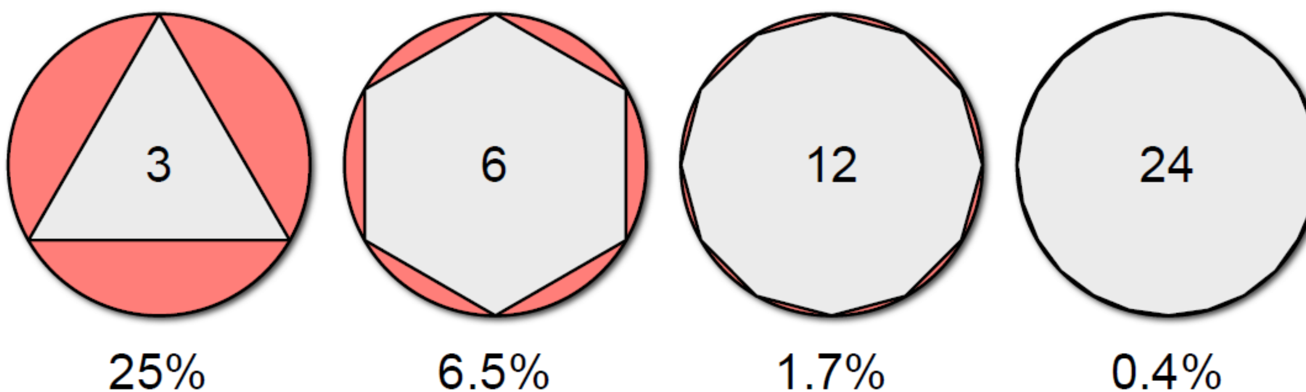
$$|f(t) - g(t)| \;\le\; \frac{1}{(p+1)!}\, \max f^{(p+1)} \prod_{i=0}^{p} (t - t_i) \;=\; O\Big(h^{(p+1)}\Big)$$

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

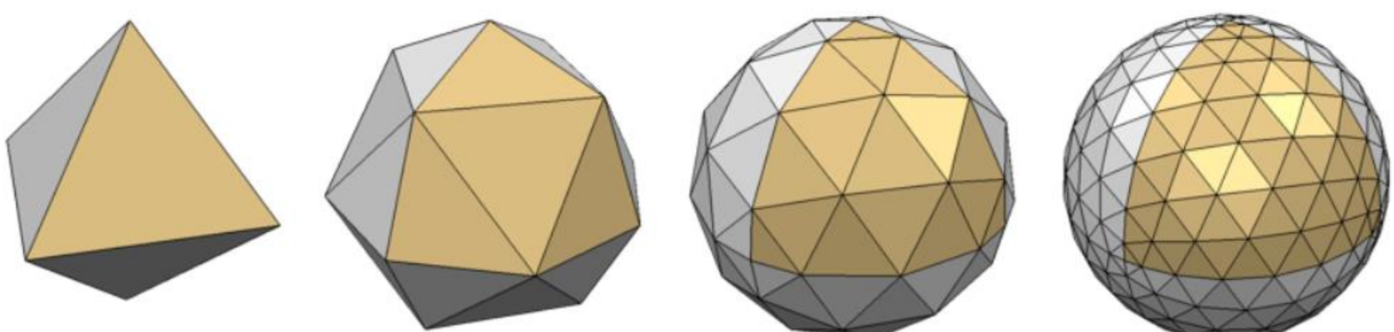
Polygon Meshes

- Polygonal meshes are a good compromise
  - Piecewise linear approximation → error is $O(h^2)$



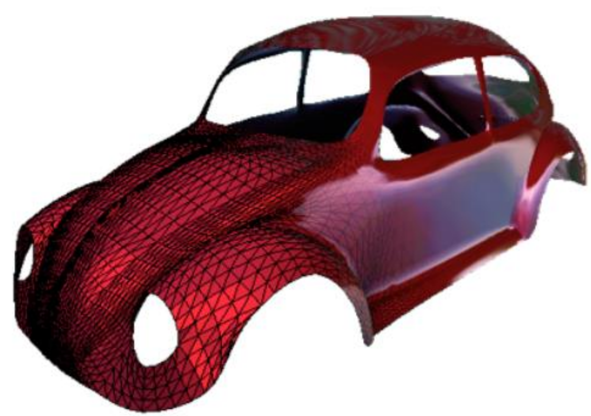
Polygon Meshes

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



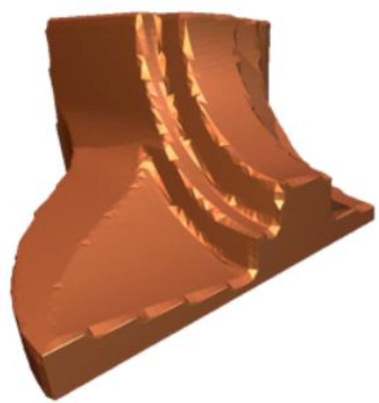
Polygon Meshes

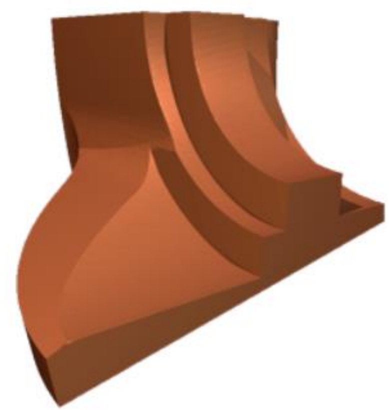
- Polygonal meshes are a good compromise
  - Piecewise linear approximation → error is $O(h^2)$
  - Error inverse proportional to #faces
  - Arbitrary topology surfaces



Polygon Meshes

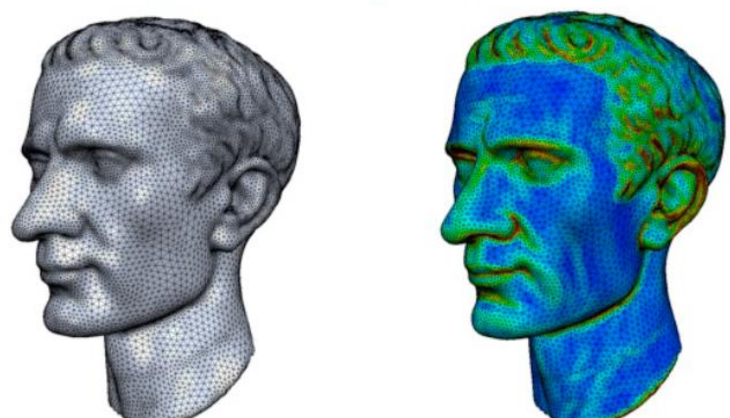
- Polygonal meshes are a good compromise
  - Piecewise linear approximation → error is $O(h^2)$
  - Error inverse proportional to #faces
  - Arbitrary topology surfaces
  - Piecewise smooth surfaces





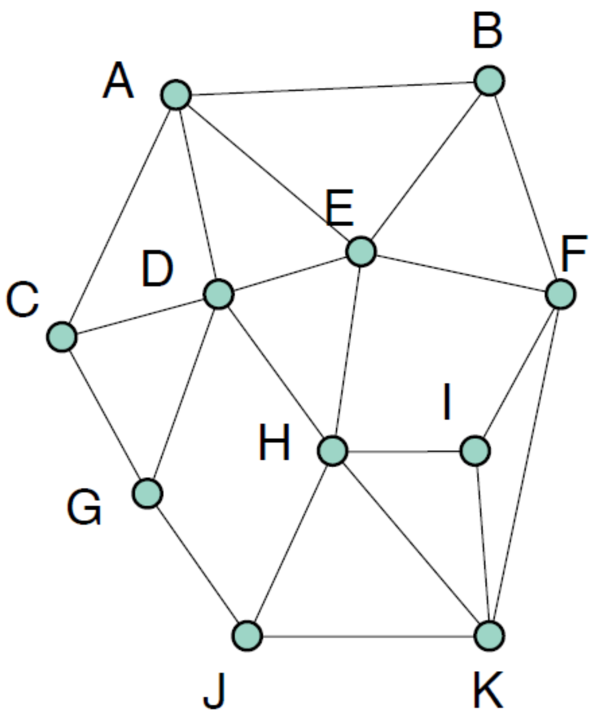
Polygon Meshes

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



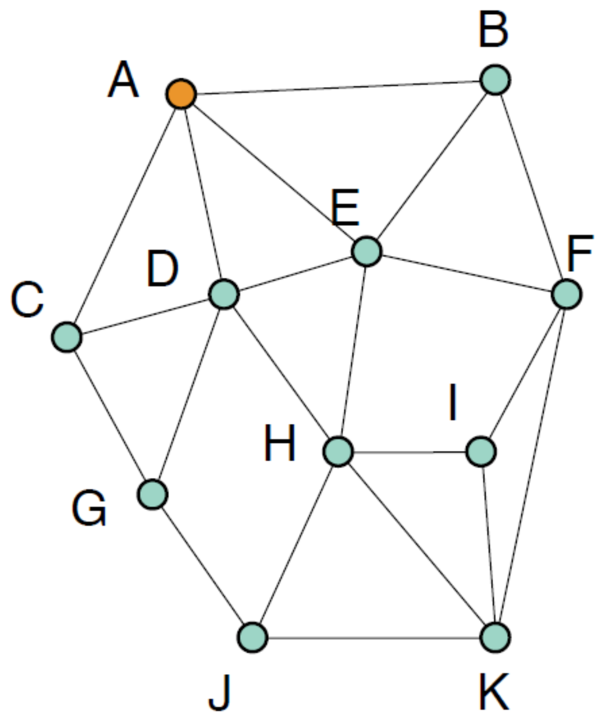
POLYGON MESHES

Graph Definitions



Graph $\{V, E\}$

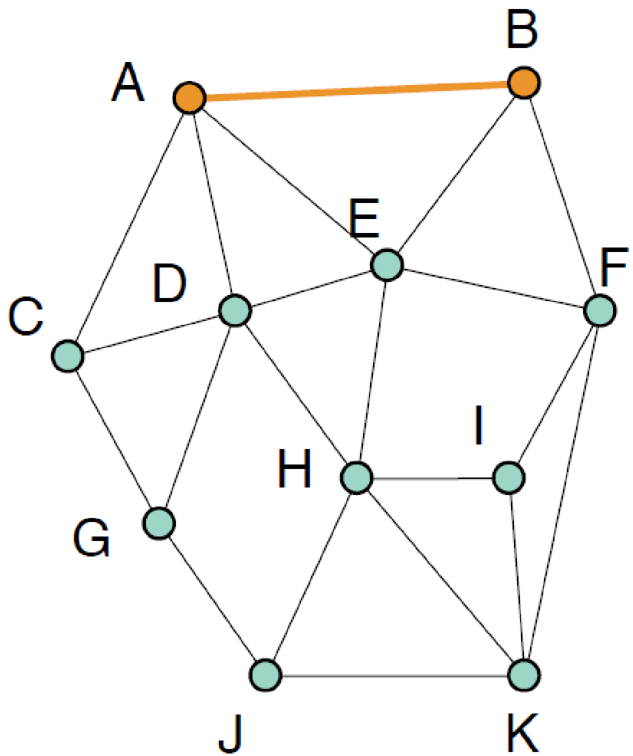
Graph Definitions



Graph $\{V, E\}$

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

Graph Definitions

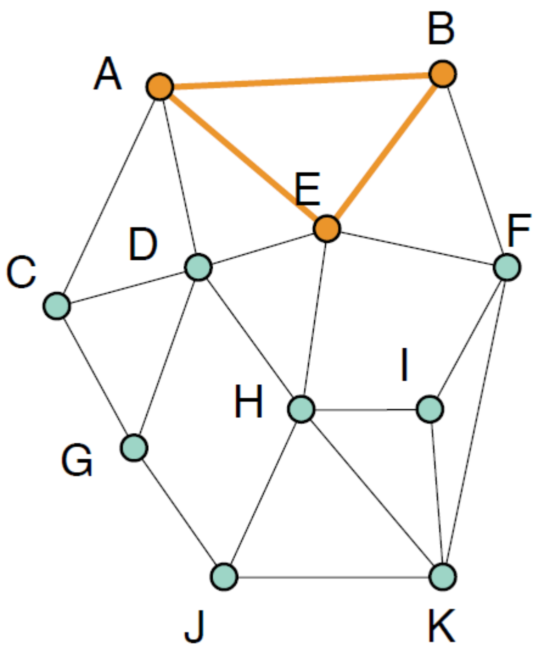


Graph $\{V, E\}$

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

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

Graph Definitions



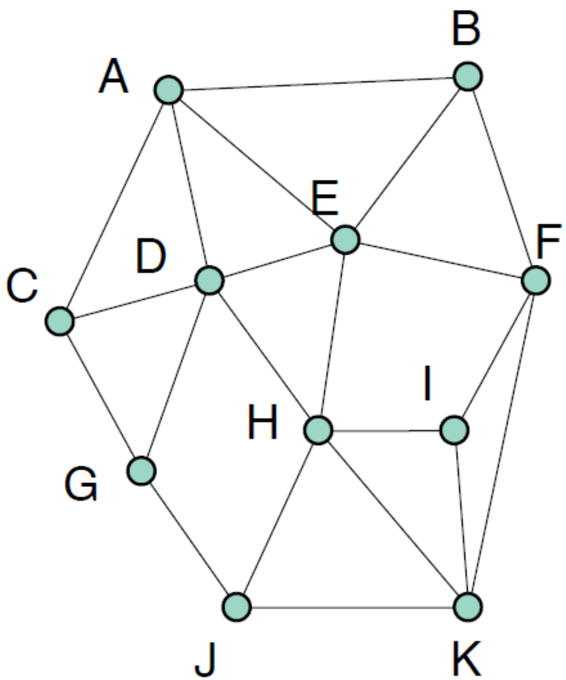
Graph $\{V, E\}$

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

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

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

Graph Definitions

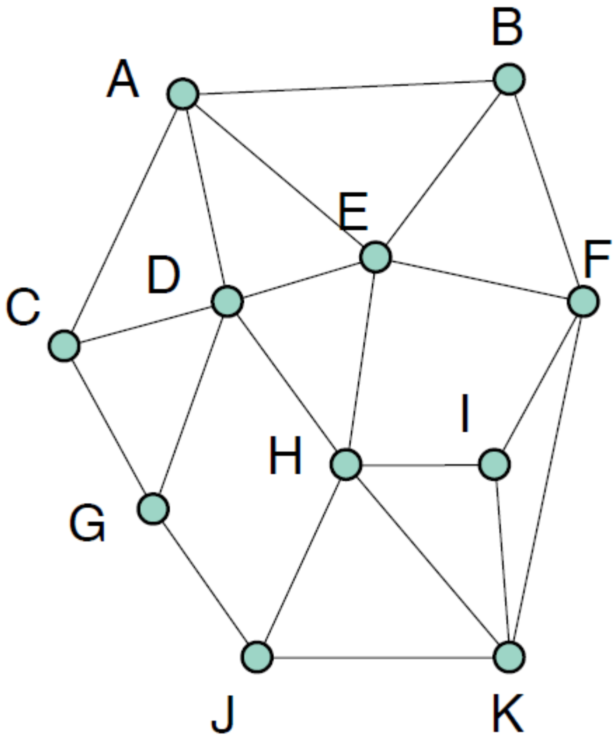


Vertex degree or **valence**:
number of incident edges.

deg(A) = 4

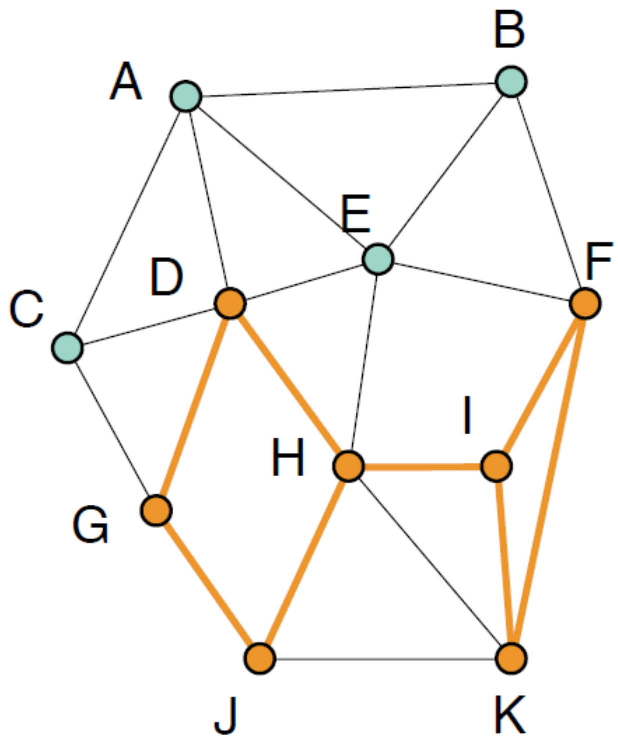
deg(E) = 5

Graph Definitions



Connected: Path of edges connecting every two vertices.

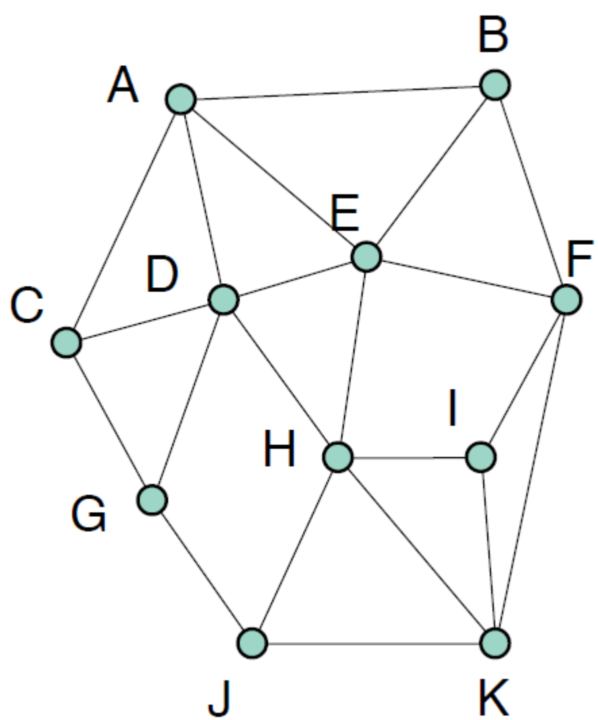
Graph Definitions



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

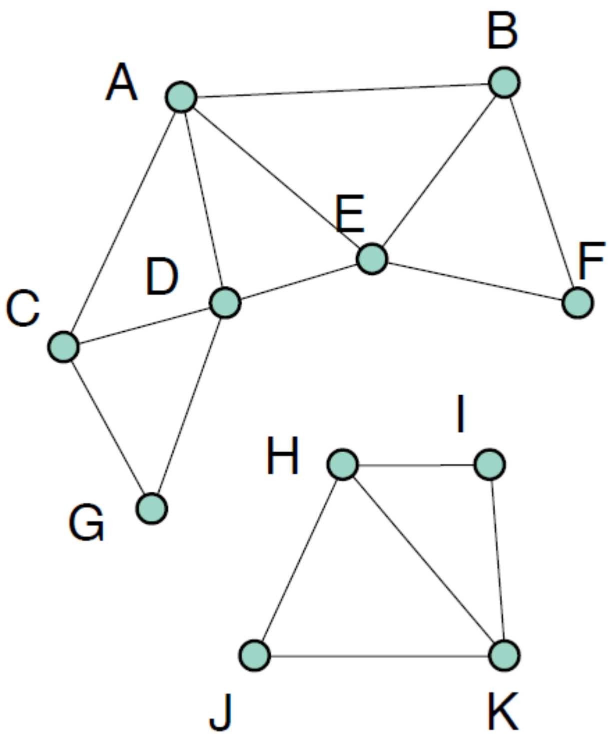


Connected: Path of edges connecting every two vertices.

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.

Graph Definitions



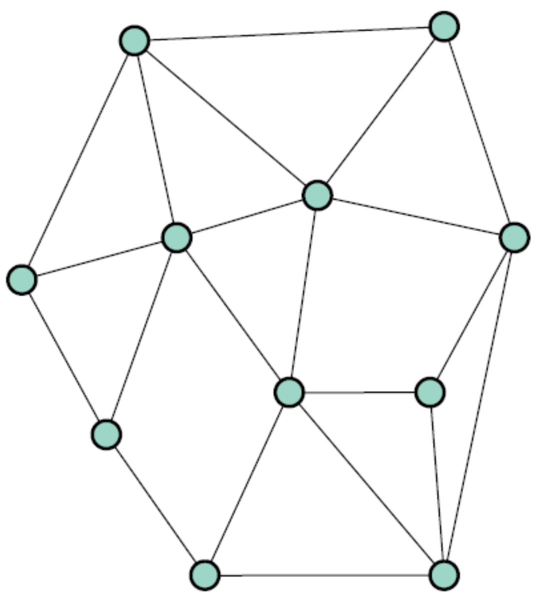
Connected: Path of edges connecting every two vertices.

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'.

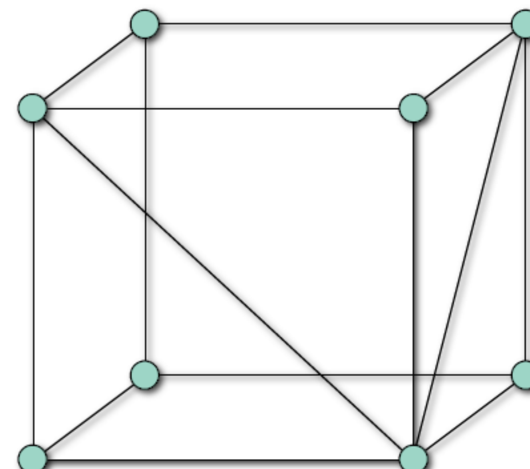
Connected component: Maximally connected subgraph.

Graph Embedding

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



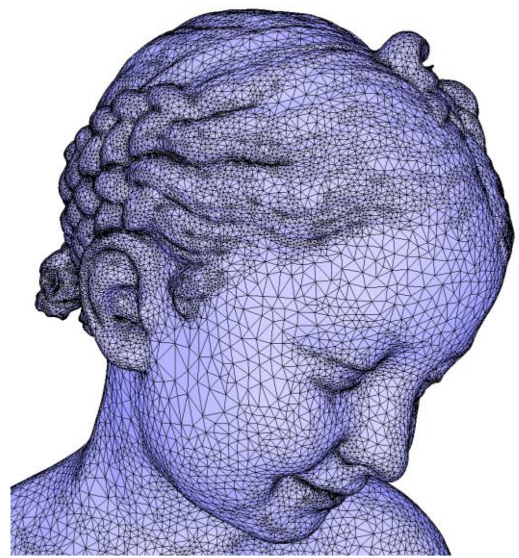
Embedded in $\mathbf{R}^2$



Embedded in $\mathbf{R}^3$

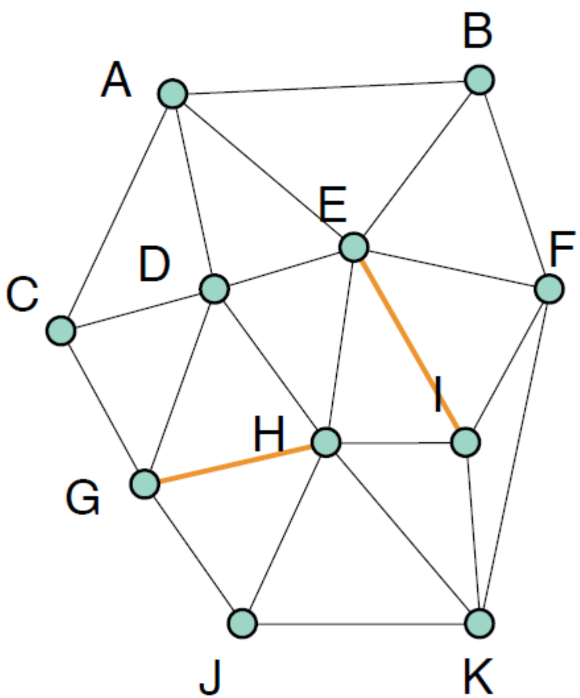
Graph Embedding

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



Embedded in $\mathbf{R}^3$

Triangulations



Triangulation: Graph where every face is a triangle.

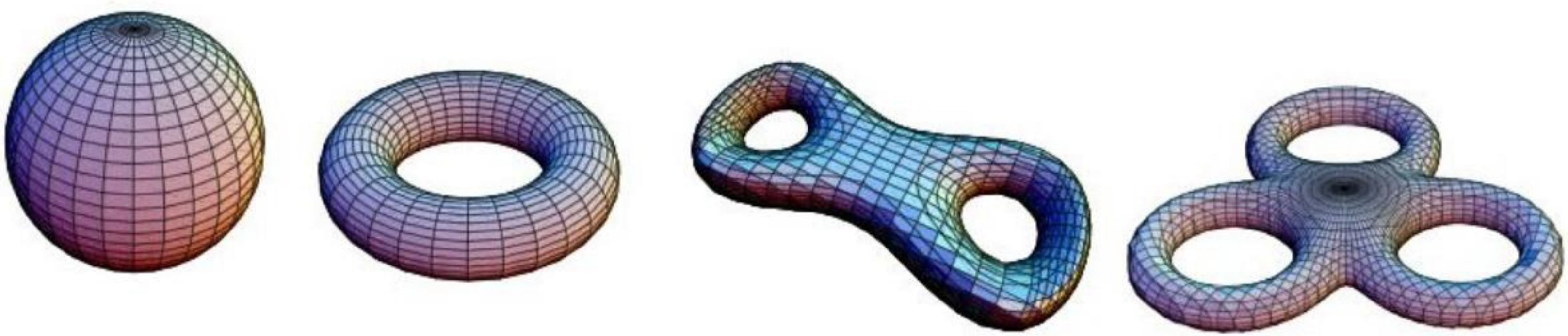
Why...?

- 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

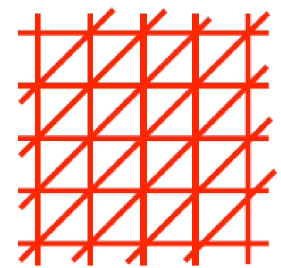
For a closed polygonal mesh of genus g, the relation of the number V of vertices, E of edges, and F of faces is given by *Euler's formula*

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

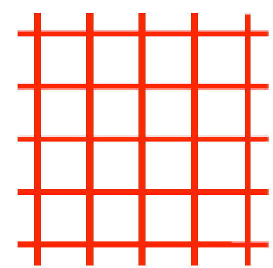
Term $2(1-g)$: Euler characteristic

Euler Consequences

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



- Quad meshes
  - $F \approx V$
  - $E \approx 2V$
  - Average valence = 4

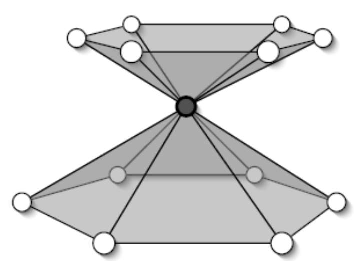
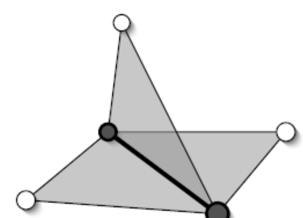
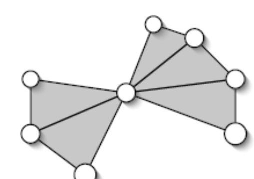


Two-Manifold Surfaces

- Local neighborhoods are disk-shaped

$$\mathbf{f}(D_\varepsilon[u,v]) = D_\delta[\mathbf{f}(u,v)]$$

- 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

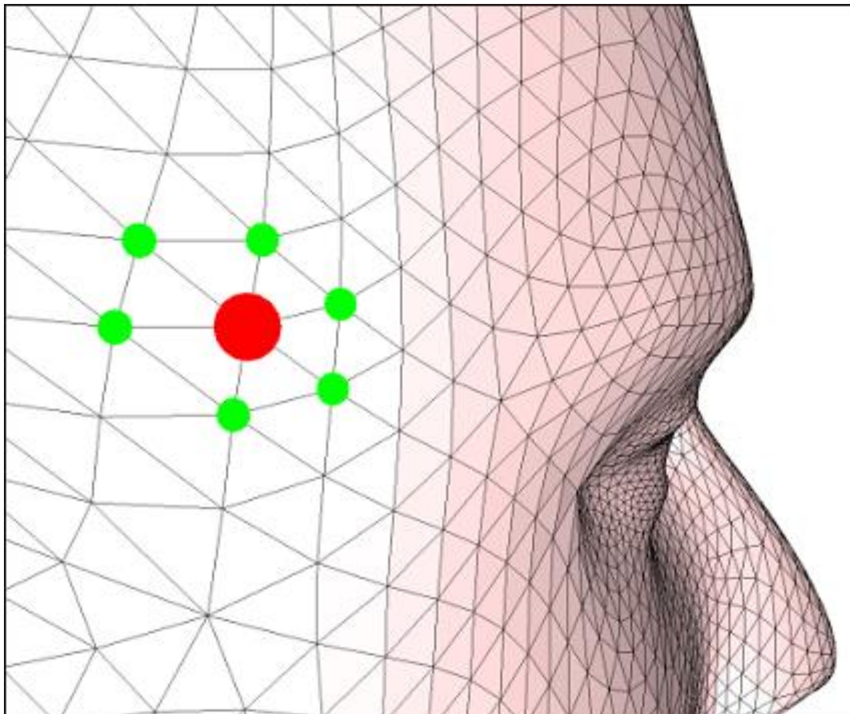
Face Set

- Face:
  - 3 positions

Triangles		
x_{11} y_{11} z_{11}	x_{12} y_{12} z_{12}	x_{13} y_{13} z_{13}
x_{21} y_{21} z_{21}	x_{22} y_{22} z_{22}	x_{23} y_{23} z_{23}
...	...	...
x_{F1} y_{F1} z_{F1}	x_{F2} y_{F2} z_{F2}	x_{F3} y_{F3} z_{F3}

36 B/f = 72 B/v
no connectivity!

Shared Vertices



Vertices	Connectivity
v1 (x1;y1;z1)	f1 (v1;v3;v2)
v2 (x2;y2;z2)	f2 (v4;v3;v1)
v3 (x3;y3;z3)	f3 (v4;v1;v5)
v4 (x4;y4;z4)	f4 (v1;v6;v5)
v5 (x5;y5;z5)	f5 (v6;v1;v7)
v6 (x6;y6;z6)	f6 (v2;v7;v1)
v7 (x7;y7;z7)	f7 (...)

Shared Vertices

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

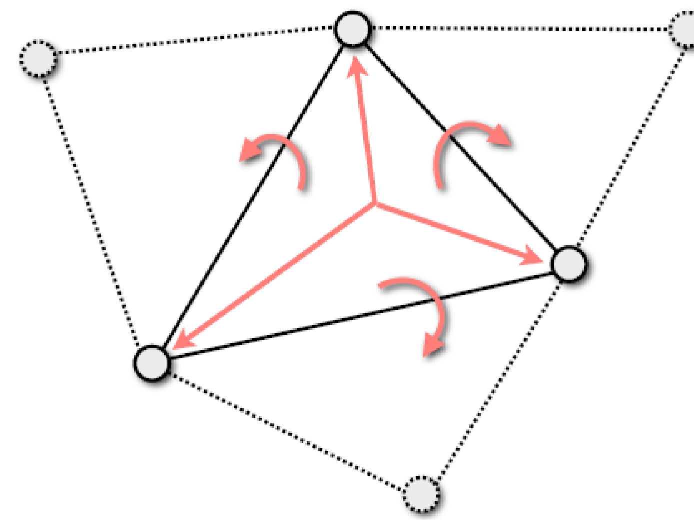
Vertices
x_1 y_1 z_1
...
x_V y_V z_V

Triangles
i_{11} i_{12} i_{13}
...
...
...
...
i_{F1} i_{F2} i_{F3}

12 B/v + 12 B/f = 36 B/v
no neighborhood info

Face-based Connectivity

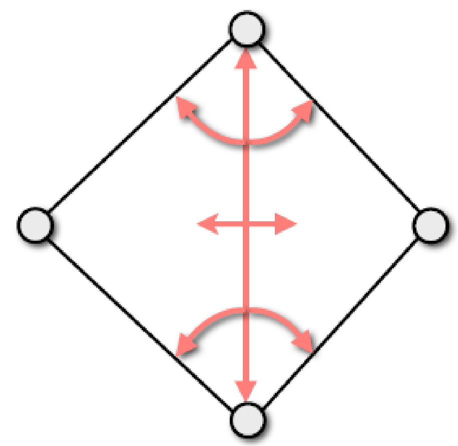
- Vertex:
  - position
  - 1 face
- Face:
  - 3 vertices
  - 3 face neighbors



64 B/v
no edges!

Edge-Based Connectivity

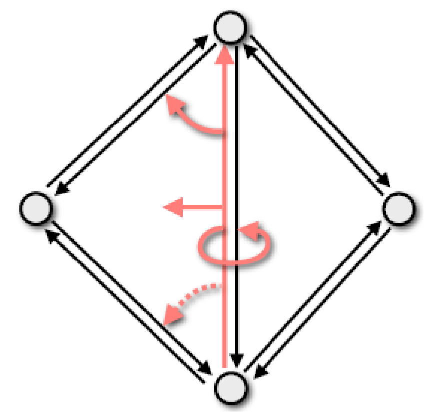
- Vertex
  - position
  - 1 edge
- Edge
  - 2 vertices
  - 2 faces
  - 4 edges
- Face
  - 1 edge



120 B/v
edge orientation?

Halfedge-Based Connectivity

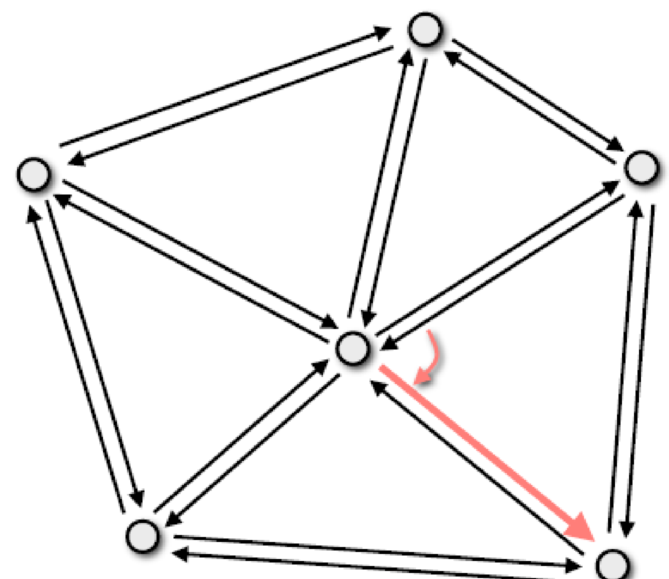
- Vertex
  - position
  - 1 halfedge
- Halfedge
  - 1 vertex
  - 1 face
  - 1, 2, or 3 halfedges
- Face
  - 1 halfedge



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

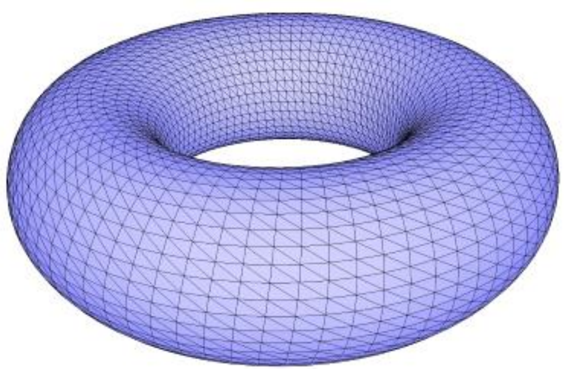
Discrete Curvatures

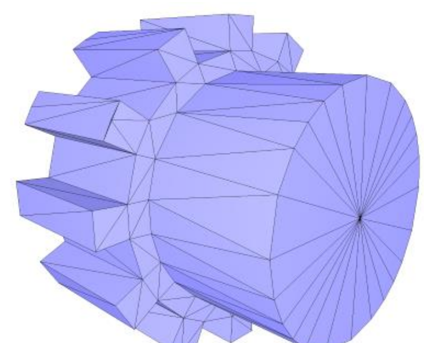
How to discretize curvatures on a mesh?

– Zero curvature within triangles

– Infinite curvature at edges / vertices

– Pointwise definition does not make sense



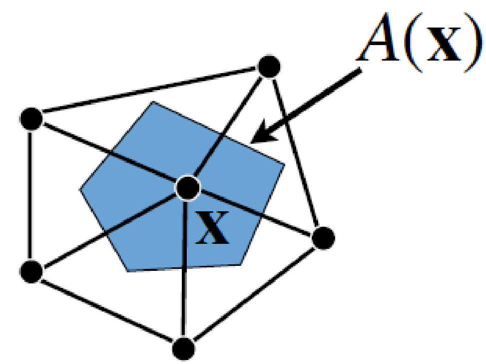


Discrete Curvatures

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

– **x** is a mesh vertex

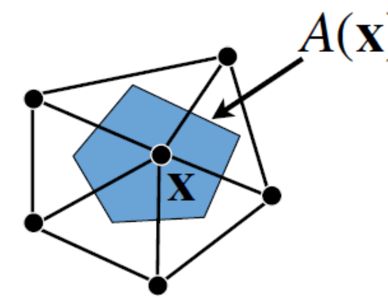
– $A(\mathbf{x})$ within one-ring neighborhood



Discrete Curvatures

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

$$K(v) \approx \frac{1}{A(v)} \int_{A(v)} K(\mathbf{x})\, \mathrm{d}A$$



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 κ1 and κ2

Laplace Operator

Laplace operator

gradient operator

2nd partial derivatives

$$\Delta f = \mathrm{div} \nabla f = \sum_i \frac{\partial^2 f}{\partial x_i^2}$$

function in Euclidean space

divergence operator

Cartesian coordinates

Laplace Operator

Laplace-
Beltrami

gradient
operator

$$\Delta_{\mathcal{S}} f = \mathrm{div}_{\mathcal{S}}\, \nabla_{\mathcal{S}} f$$

function on
manifold $\mathcal{S}$

divergence
operator

Laplace Operator

Laplace-Beltrami

gradient operator

mean curvature

$$\Delta_{\mathcal{S}} \mathbf{x} = \mathrm{div}_{\mathcal{S}} \nabla_{\mathcal{S}} \mathbf{x} = -2H\mathbf{n}$$

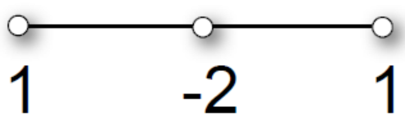
coordinate function

divergence operator

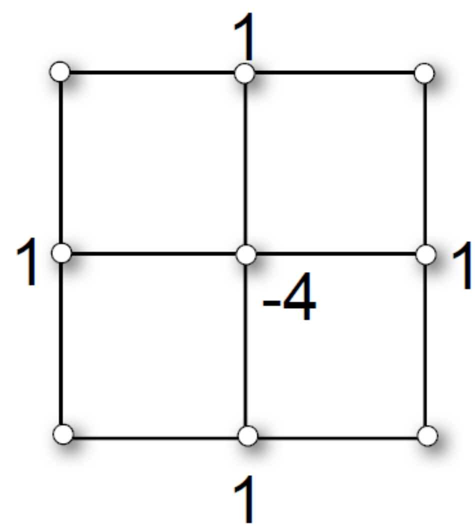
surface normal

Laplace Operator on Meshes?

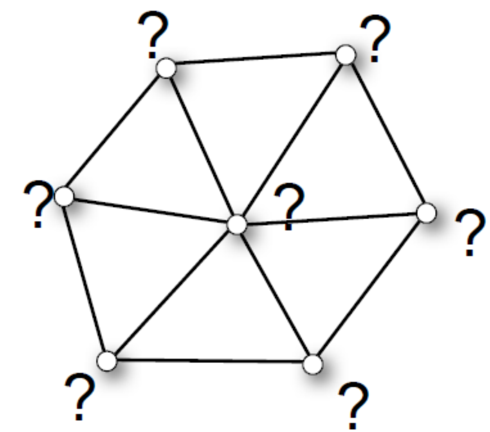
- Extend finite differences to meshes?
  - What weights per vertex / edge?



1D grid



2D grid

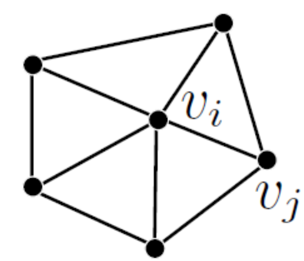


2D/3D mesh

Uniform Laplace

- Uniform discretization

$$\Delta_{\text{uni}} f(v_i) := \frac{1}{|\mathcal{N}_1(v_i)|} \sum_{v_j \in \mathcal{N}_1(v_i)} (f(v_j) - f(v_i))$$



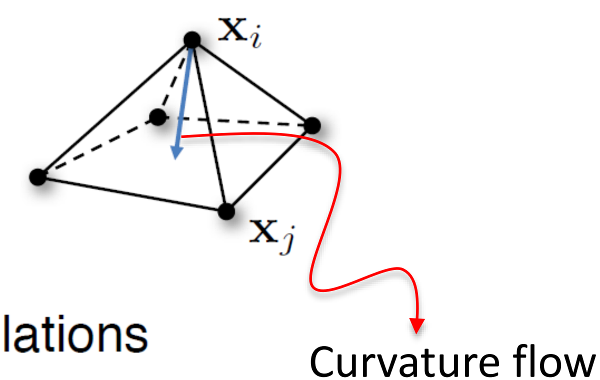
- Properties
  - depends only on connectivity
  - simple and efficient
  - bad approximation for irregular triangulations

Uniform Laplace

- Uniform discretization

$$\Delta_{\mathrm{uni}} \mathbf{x}_i \;:=\; \frac{1}{|\mathcal{N}_1(v_i)|} \sum_{v_j \in \mathcal{N}_1(v_i)} (\mathbf{x}_j - \mathbf{x}_i) \;\approx\; -2H\mathbf{n}$$

- 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



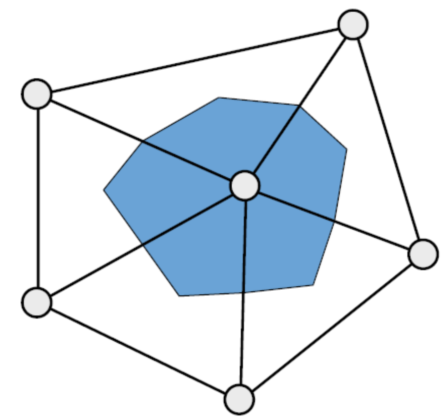
Barycentric Cells

Connect edge midpoints and triangle barycenters

– Simple to compute

– Area: 1/3 of triangle areas

– Slightly wrong for obtuse triangles



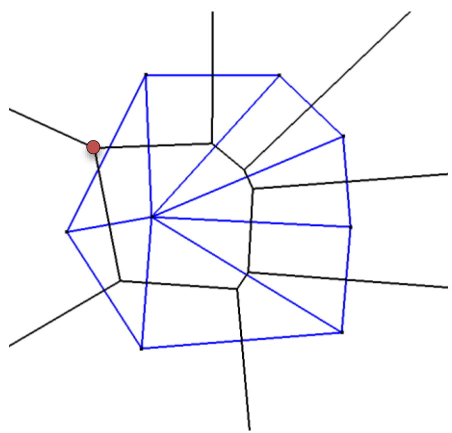
Mixed Cells

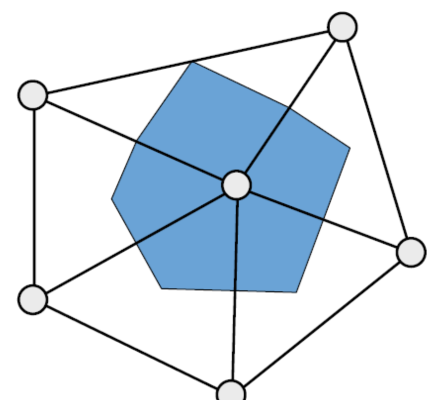
Connect edge midpoints and

– Circumcenters for non-obtuse triangles

– Midpoint of opposite edge for obtuse triangles

– Better approximation, more complex to compute.





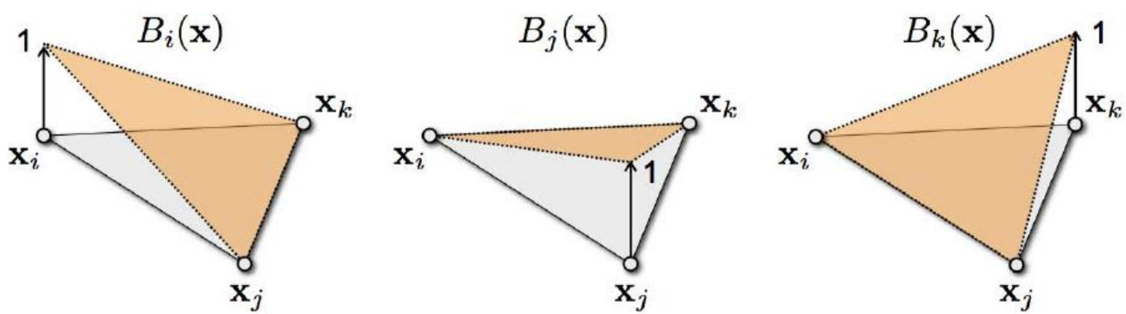
"Cotan" Laplace

- Piecewise linear functions

function value at vertex

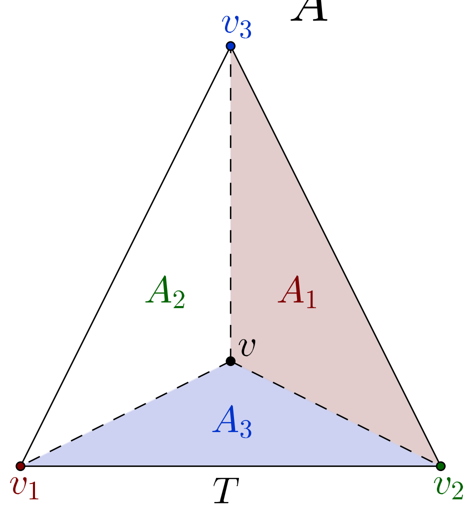
$$f(\mathbf{u}) \;=\; f_i B_i(\mathbf{u}) + f_j B_j(\mathbf{u}) + f_k B_k(\mathbf{u})$$

linear basis function



Reminder: Triangle barycentric coordinates

$$b_3 = \frac{A_3}{A}$$



$$b_1 = \frac{A_1}{A} \qquad b_2 = \frac{A_2}{A}$$

$$A = A_1 + A_2 + A_3$$

A. F. Möbius
[1790−1868]

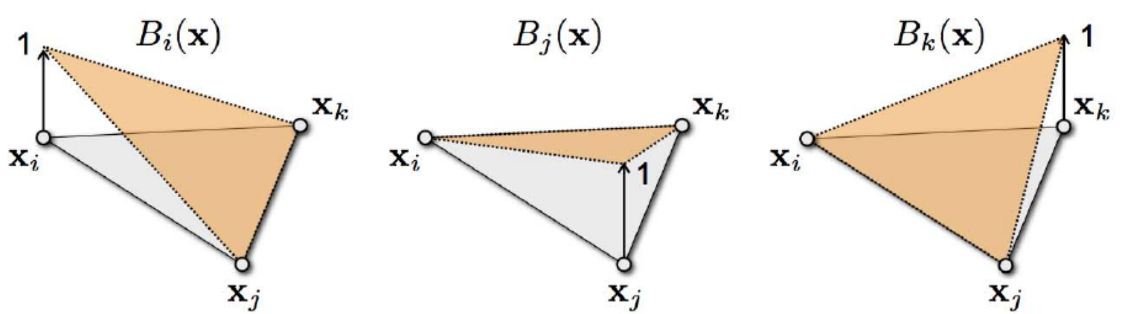
"Cotan" Laplace

- 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

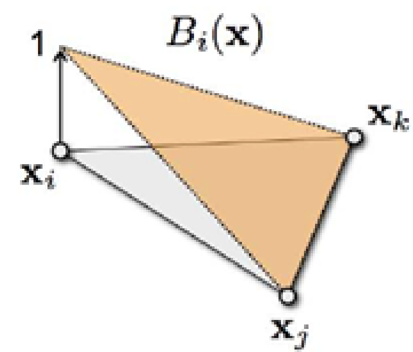
- Piecewise linear functions

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

  - Gradient

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

$$\nabla B_i(\mathbf{u}) \;=\; \frac{(\mathbf{x}_k - \mathbf{x}_j)^{\perp}}{2\, A_T}$$



Cotan Laplace

- Divergence Theorem

$$\int_{A_i} \operatorname{div} \mathbf{F}(\mathbf{u}) \mathrm{d}A = \int_{\partial A_i} \mathbf{F}(\mathbf{u}) \cdot \mathbf{n}(\mathbf{u}) \, \mathrm{d}s$$

  - Applied to Laplacian

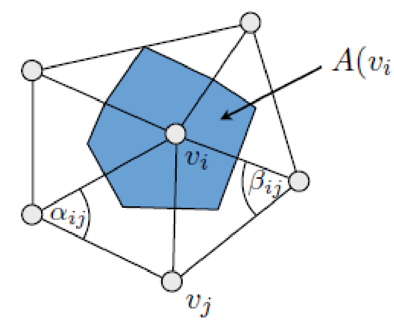
$$\int_{A_i} \Delta f(\mathbf{u}) \, \mathrm{d}A = \int_{A_i} \operatorname{div} \nabla f(\mathbf{u}) \, \mathrm{d}A = \int_{\partial A_i} \nabla f(\mathbf{u}) \cdot \mathbf{n}(\mathbf{u}) \, \mathrm{d}s$$

Discrete Laplace-Beltrami

- Cotangent discretization

$$\Delta_{\mathcal{S}} f(v) := \frac{1}{2A(v)} \sum_{v_i \in \mathcal{N}_1(v)} (\cot \alpha_i + \cot \beta_i)\,(f(v_i) - f(v))$$

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



- Still the most widely used discretization

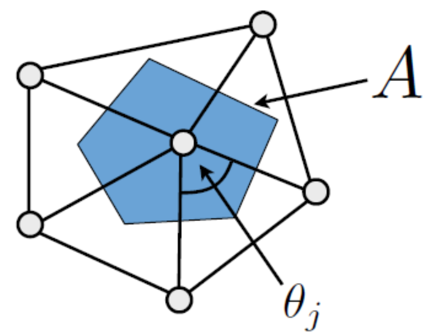
Discrete Curvatures

- Mean curvature (absolute value)

$$H = \frac{1}{2} \|\Delta_{\mathcal{S}} \mathbf{x}\|$$

- Gaussian curvature

$$K = (2\pi - \sum_j \theta_j)/A$$

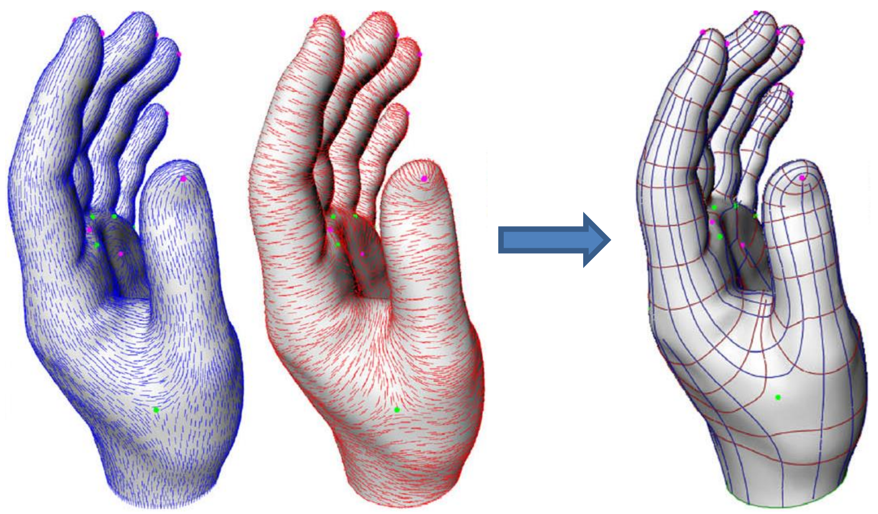


- Principal curvatures

$$\kappa_1 = H + \sqrt{H^2 - K} \qquad \kappa_2 = H - \sqrt{H^2 - K}$$

PRINCIPAL CURVATURES

Principal Curvatures

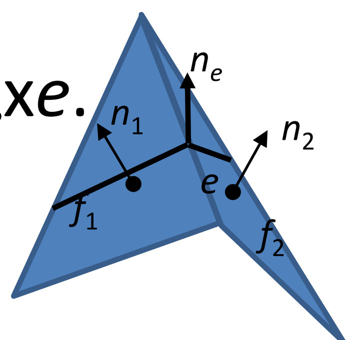


Principal curvature directions Lines of curvatures

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 $n_e \mathrm{x} e$.

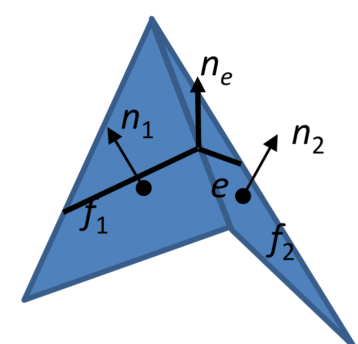


* [Cohen-Steiner *et al*. '03]

Principal Curvature Directions

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

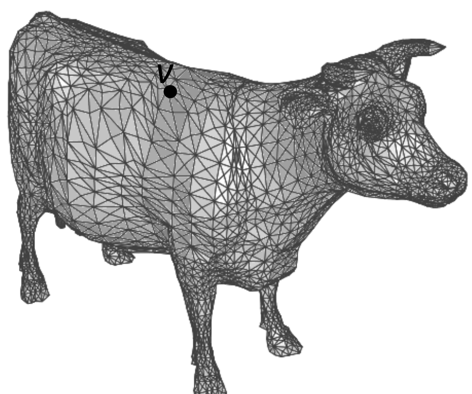
$$\boldsymbol{C}(p \in e) = \beta(e)\frac{1}{\|n_e \times e\|^2}(n_e \times e)(n_e \times e)^t$$

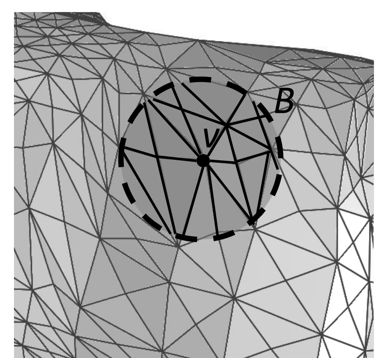


Principal Curvature Directions

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

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





Principal Curvature Directions

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

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

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

Note:

When the two principal directions have the same principal curvature values, the principal directions are not well defined.

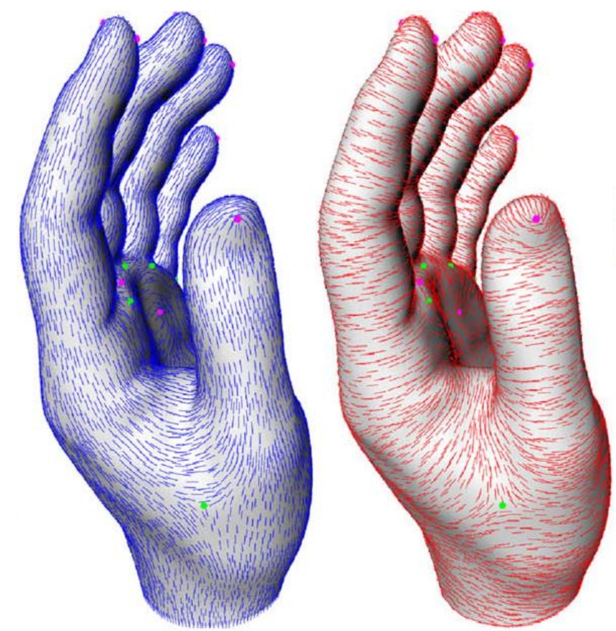
Principal Curvature Directions

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

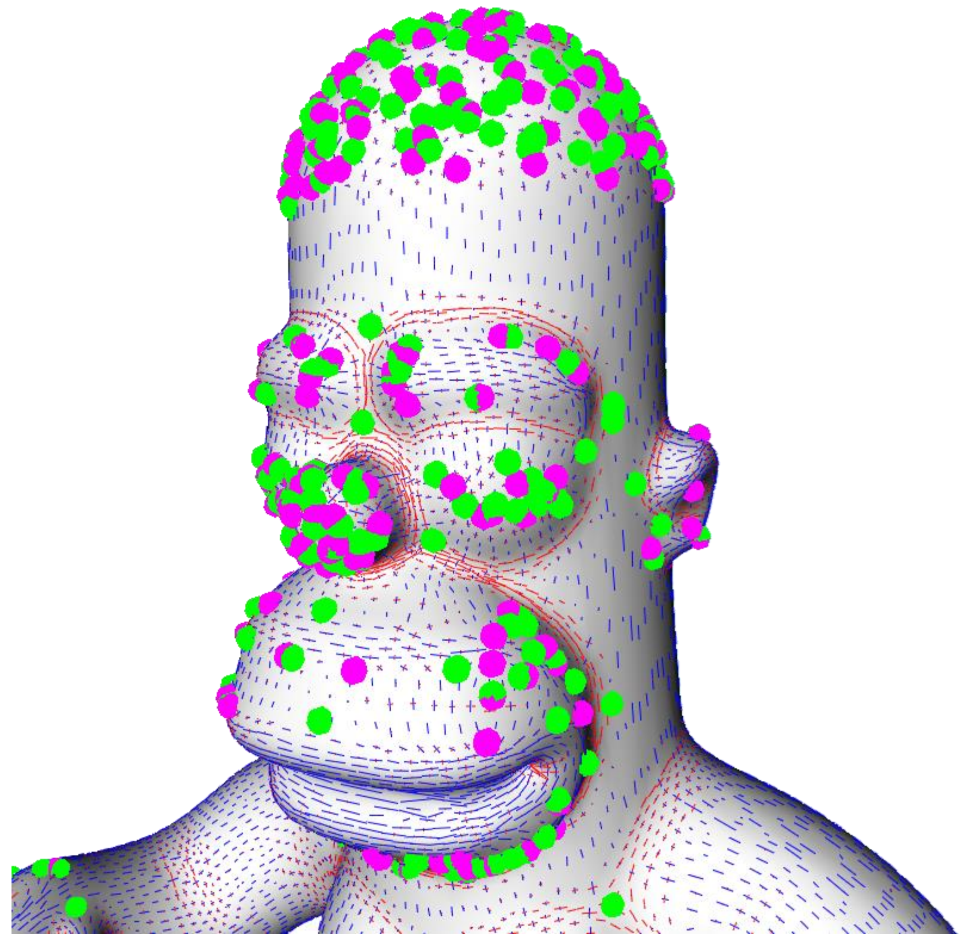
<u>Note</u>:

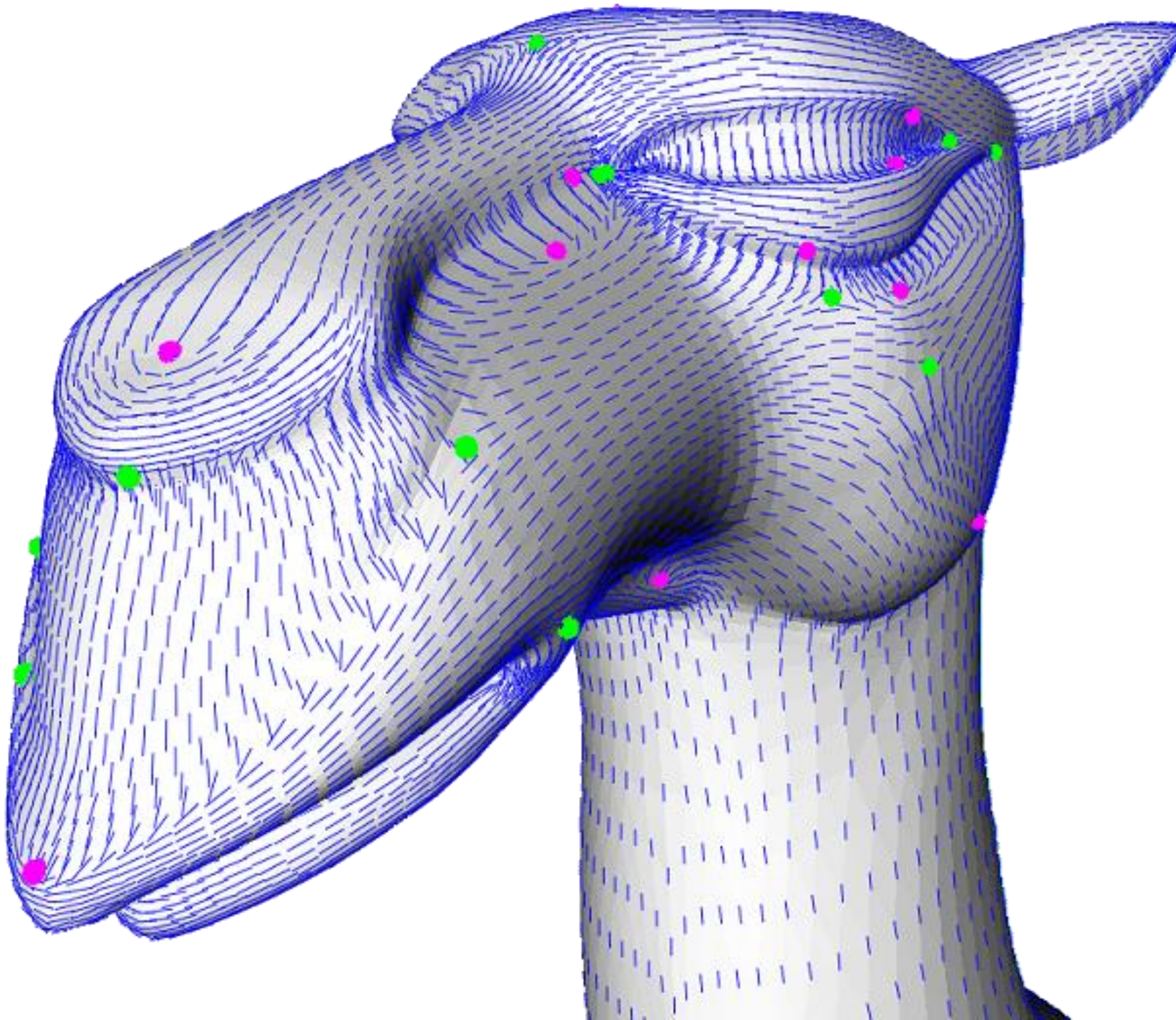
When the two principal directions have the same principal curvature values, the principal directions are not well defined.

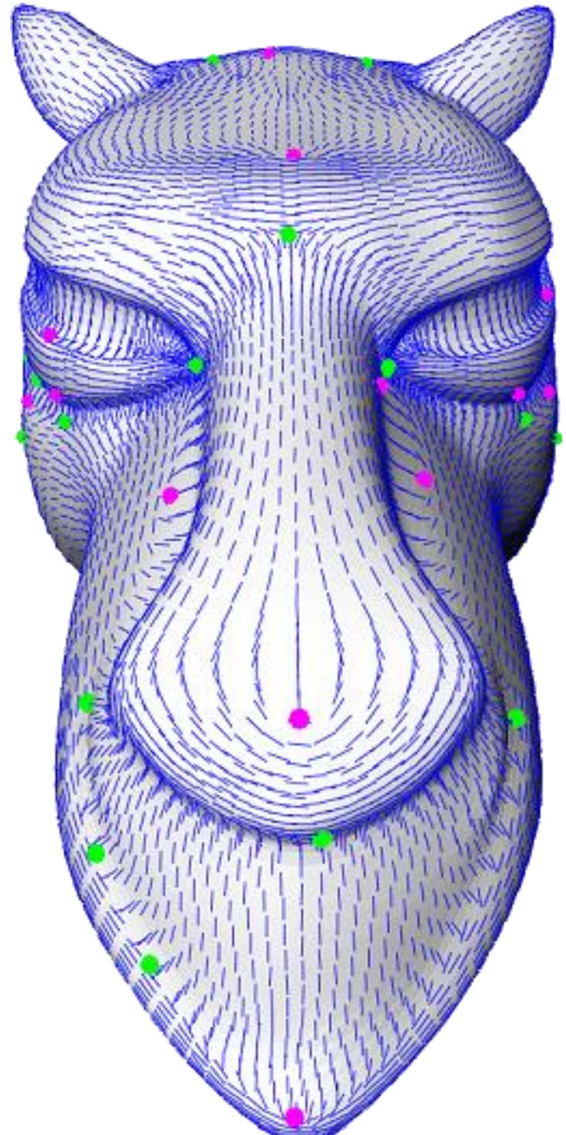
Such points are called <u>umbilical</u> points.

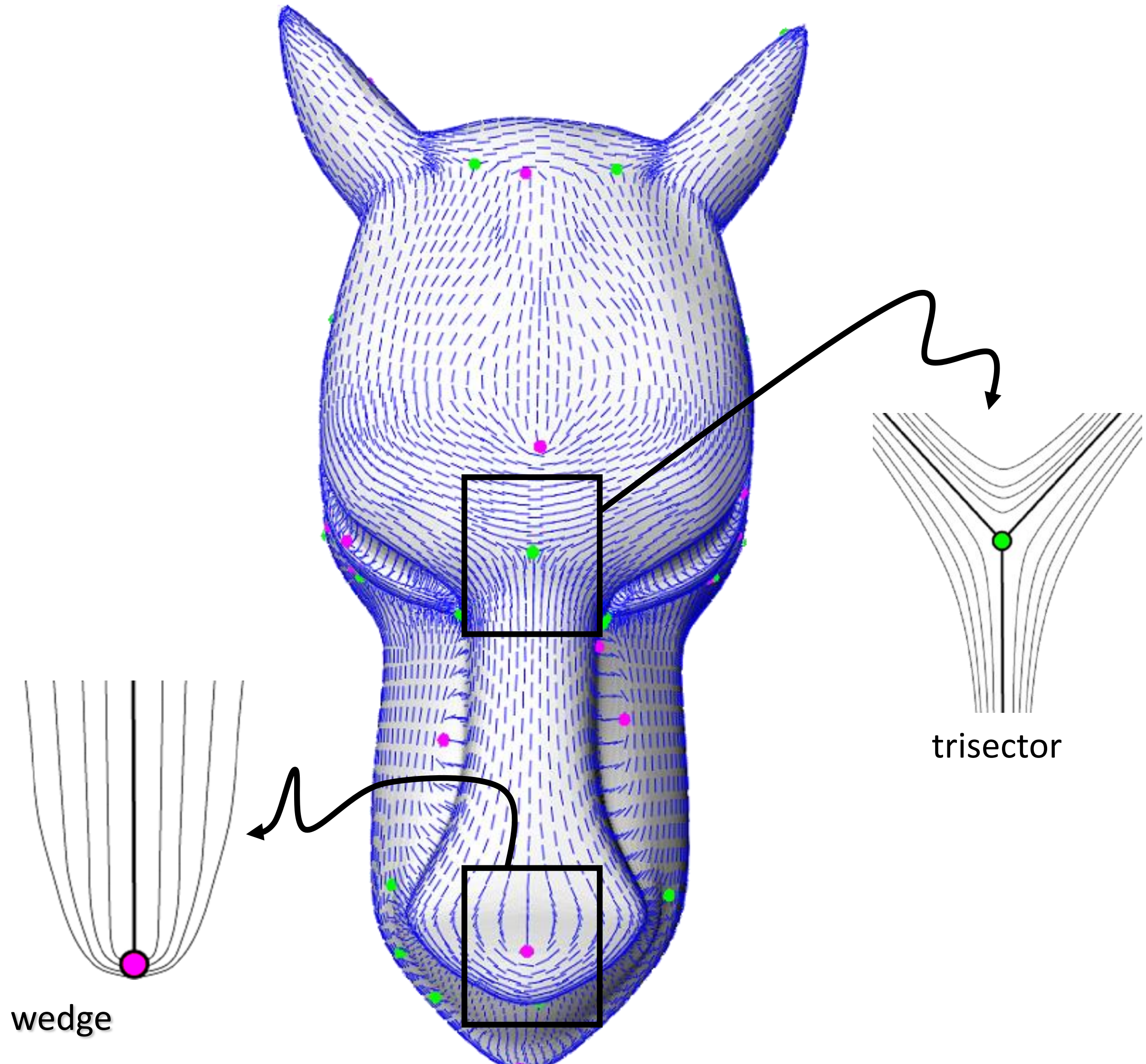


Umbilic Points





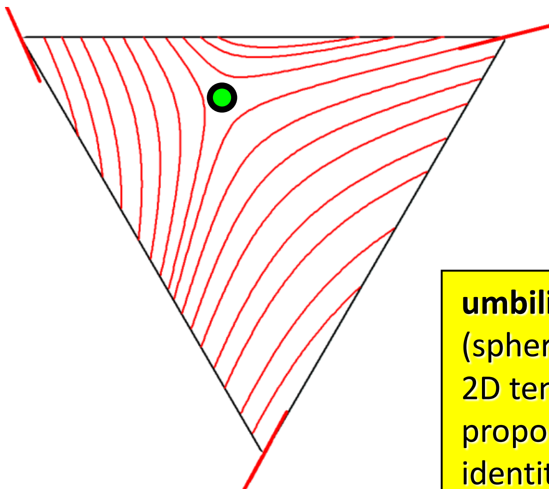



trisector
wedge

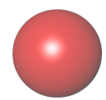
Principal Direction Fields

Linear singularities

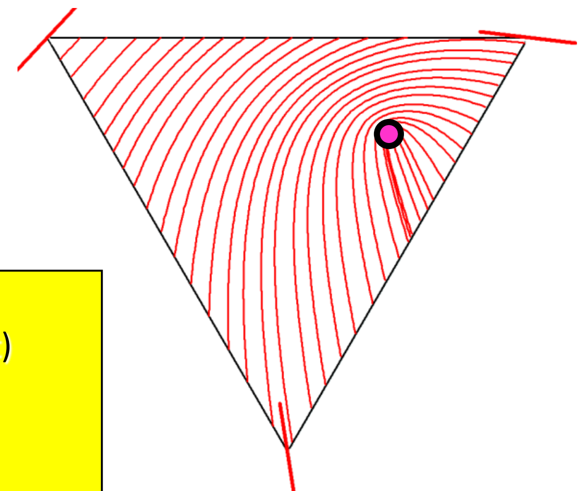
Topology of tensor fields.
[Delmarcelle & Hesselink 94]
[Tricoche 02]



trisector



umbilic
(spherical point)
2D tensor
proportional to
identity



wedge

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_{min}: Aligns with the min. curvature
2. v_{max}: Aligns with the max. curvature

Principal Curvature Lines

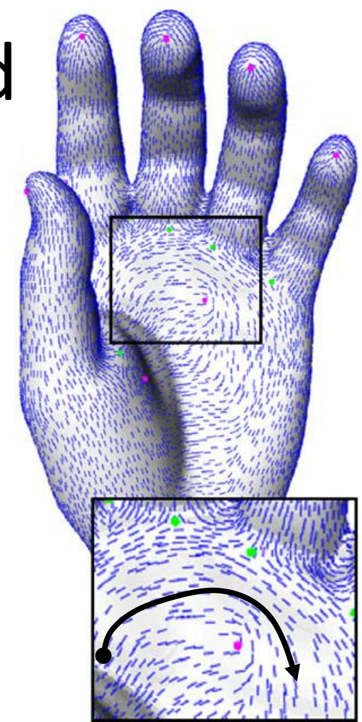
What are the principal curvature lines?

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

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

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

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



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 ε 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:

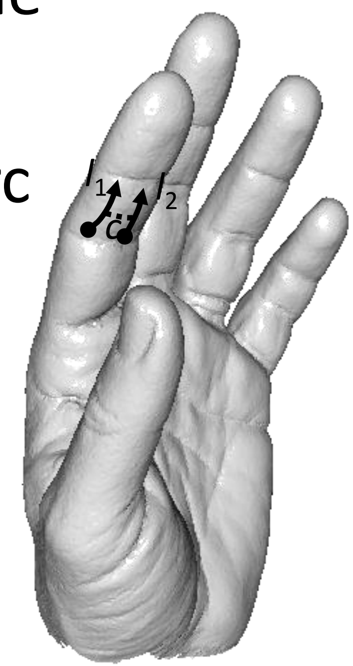
The curve between them will follow the maximal/minimal curvature direction.

Principal Curvature Lines

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

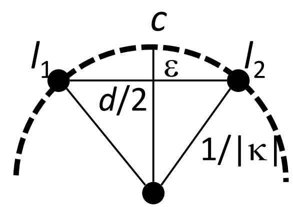
The curve between them will follow the maximal/minimal curvature direction.

The curve will be, roughly, a circular arc with radius equal to one over the maximal/minimal curvature.



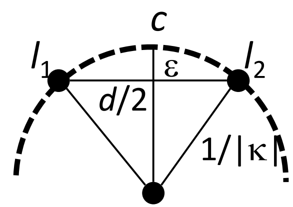
Principal Curvature Lines

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 ε.



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 ε.

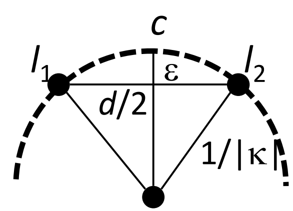


Denoting the distance by ε we get:

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

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 ε.

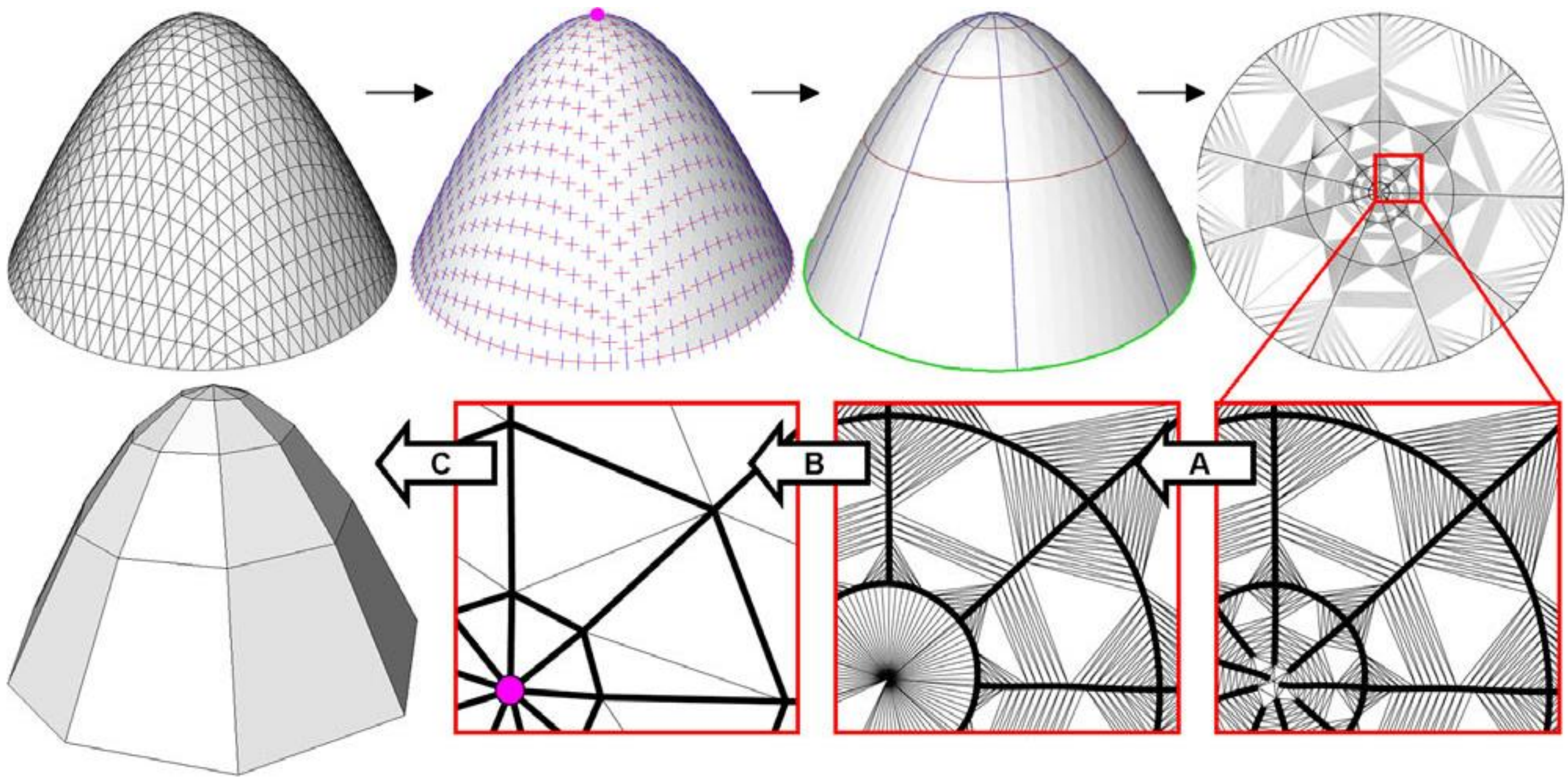


Denoting the distance by ε we get:

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

$$d = 2\sqrt{\varepsilon\left(\frac{2}{|\kappa|} - \varepsilon\right)}$$

Principal Curvature Lines



Literature

Taubin: ***A signal processing approach to fair surface design***, SIGGRAPH 1996.
– Desbrun et al: ***Implicit Fairing of Irregular Meshes using Diffusion and Curvature Flow***, SIGGRAPH 1999.
– Meyer et al: ***Discrete Differential-Geometry Operators for Triangulated 2-Manifolds***, VisMath 2002.
– Wardetzky, Mathur, Kaelberer, Grinspun: ***Discrete Laplace Operators: No free lunch,*** SGP 2007
– Alexa, Wardetzky: ***Discrete Laplacians on General Polygonal Meshes***, SIGGRAPH 2011