Notes - Theory session 5

Geometry Processing (GPR)

1 Reconstruction

1.1 Radial Basis Functions (RBF)

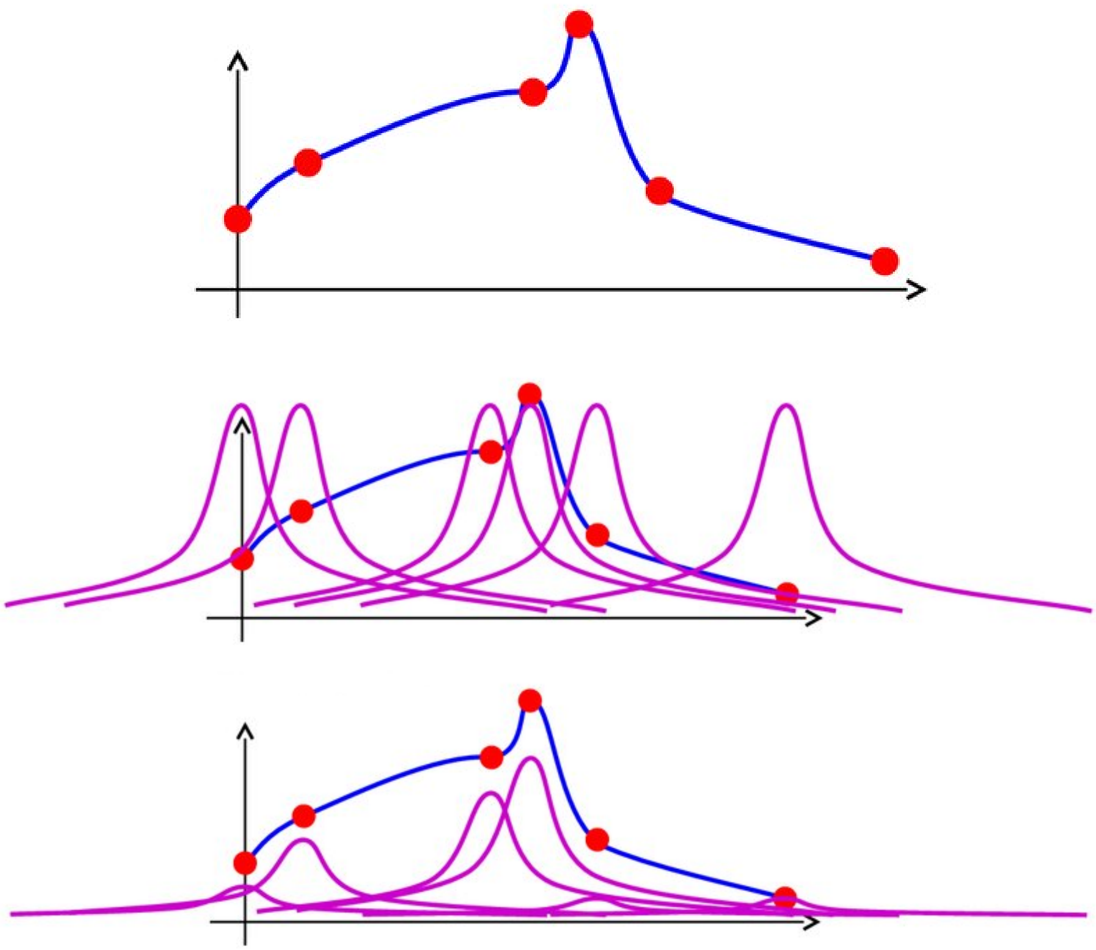


Figure 1: Interpolating with RBFs.

Input: Set of pairs $(p_i, v_i), p_i \in \mathbb{R}^3, v_i \in \mathbb{R}$

Output: Smooth interpolating function f.

$$f(p) = \sum_{i=1}^{m} f_i(p) \qquad f_i(p) = \phi(\|p - p_i\|) \cdot c_i$$

Gaussian RBF:

$$\phi(r) = exp(-r^2/2c^2)$$

Interpolating conditions:

$$f(p_i) = v_i \implies f(p_i) = \sum_{j=1}^{m} f_j(p_i) = v_i \implies \sum_{j=1}^{m} \phi(\|p_i - p_j\|) \cdot c_j = v_i \implies \mathbf{A} \cdot \mathbf{c} = \mathbf{v}$$

$$\mathbf{A} = \begin{pmatrix} \phi(\|p_1 - p_1\|) & \phi(\|p_1 - p_2\|) & \cdots & \phi(\|p_1 - p_m\|) \\ \phi(\|p_2 - p_1\|) & \phi(\|p_2 - p_2\|) & \cdots & \phi(\|p_2 - p_m\|) \\ \vdots & \vdots & \ddots & \vdots \\ \phi(\|p_m - p_1\|) & \phi(\|p_m - p_2\|) & \cdots & \phi(\|p_m - p_m\|) \end{pmatrix}$$

$$\mathbf{c} = \begin{pmatrix} c_1 & c_2 & \cdots & c_m \end{pmatrix}^T \qquad \mathbf{v} = \begin{pmatrix} v_1 & v_2 & \cdots & v_m \end{pmatrix}^T$$

Problems:

- We want to interpolate, so $f(p_i) = 0$ for all input points. Which means $\mathbf{A} \cdot \mathbf{c} = \mathbf{0}$. The trivial solution is $\mathbf{c} = \mathbf{0}$.
  - Create artificial points. For each p_i, create $p_i^+ = p_i + d \cdot n_i$ and $p_i^- = p_i - d \cdot n_i$, such that $f(p_i^+ = d)$ and $f(p_i^- = -d)$.
- As m increases matrix $\mathbf{A}$ is going to become ill-conditioned.
  - Apply a regularization by $\mathbf{A'} = \mathbf{A} + \lambda \mathbf{I}$.
- Matrix $\mathbf{A}$ is dense and can be very large ($m \times m$).
  - Use gaussian RBFs with compact support for a sparse matrix.
  - Loses its generalization power (holes are undefined).
  - Solution: Give increasing support to a decreasing number of points.

$$\phi(r) = \begin{cases} exp(-r^2/2c^2) & \text{if } r < 3c \\ 0 & \text{otherwise} \end{cases}$$

2 Curvatures

Characteristics:

- Coordinate system independent.
- Local surface characteristic → Geometric signature
- For curves:
  - Connected to the second derivative.
  - Inverse of the radius of the *osculating circle*.
- For surfaces:
  - Plane intersection → Curve → Curvature
  - Only two directions needed: *principal curvature* directions (min and max curvature - K_{min} and K_{max}).
  - Positive curvature → Convex, Negative → Concave, Zero → No curvature

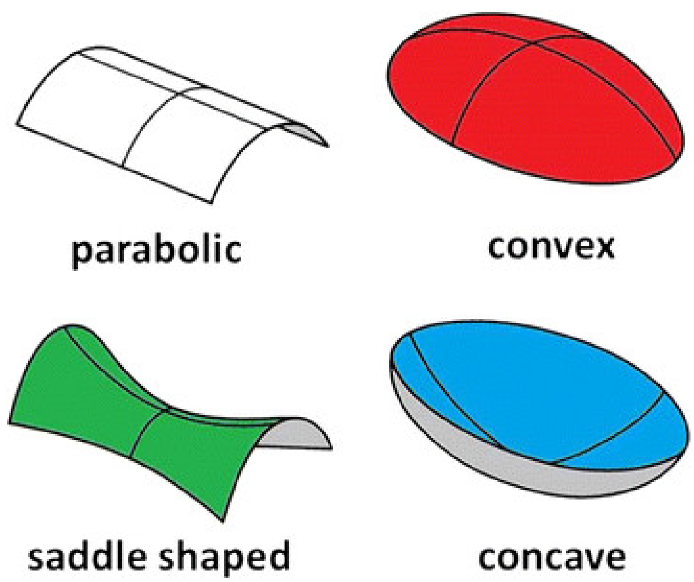


Figure 2: Local shapes depending on principal curvatures.

Given a surface expressed as $w(u, v)$, the curvature at $(0, 0)$ can be computed from its Hessian matrix, which collects the second derivatives:

$$\mathbf{H_w} = \begin{pmatrix} w_{uu} & w_{vu} \\ w_{uv} & w_{vv} \end{pmatrix} = \begin{pmatrix} \frac{\partial^2 w}{\partial u^2} & \frac{\partial^2 w}{\partial v \partial u} \\ \frac{\partial^2 w}{\partial u \partial v} & \frac{\partial^2 w}{\partial v^2} \end{pmatrix}$$

The principal curvatures are the eigenvalues of $\mathbf{H_w}$ and its directions are the corresponding eigenvectors.

We can combine both values into a single one. The *Gaussian curvature*:

$$K = K_{min} \cdot K_{max}$$

and the *Mean curvature*:

$$H = \frac{K_{min} + K_{max}}{2}$$

and use both to characterize the shape:

- $K = 0 \rightarrow$ Planar or cylindrical
- $K < 0 \rightarrow$ Saddle point
- $K > 0 \rightarrow$ Convex or concave
  - $H > 0 \rightarrow$ Convex
  - $H < 0 \rightarrow$ Concave

The principal curvatures can also be used to compute K_{min} and K_{max}:

$$K_{max}, K_{min} = H \pm \sqrt{H^2 - K}$$

When we have a point cloud or triangle mesh we compute the Hessian at a point by locally adjusting a function. This is known as a *Monge patch*.

For a given point $\mathbf{p}$, its normal $\mathbf{n}$, and its neighbors $\mathcal{P} = \{p_i\}_{1 \leq i \leq m}$, we need to fit a quadratic function:

$$w(u, v) = au^2 + buv + cv^2 + du + ev + f = \mathbf{q}^T \mathbf{s}$$

$$\mathbf{q} = \begin{pmatrix} u^2 & uv & v^2 & u & v & 1 \end{pmatrix}^T, \qquad \mathbf{s} = \begin{pmatrix} a & b & c & d & e & f \end{pmatrix}^T$$

First we need to transform the neighbors to a coordinate system $(\mathbf{u}, \mathbf{v}, \mathbf{w})$ aligned with the normal $\mathbf{n}$. The origin will be point $\mathbf{p}$ itself.

$$\begin{aligned}\mathbf{w} &= -\mathbf{n}\\ \mathbf{u} &= \mathbf{OX} \times \mathbf{w}\\ \mathbf{v} &= \mathbf{w} \times \mathbf{v}\end{aligned}$$

We transform all the neighbors to this system from $\mathbf{p_i}$ to (u_i, v_i, w_i):

$$\begin{aligned}u_i &=< \mathbf{u}, \mathbf{p_i} - \mathbf{p} >\\ v_i &=< \mathbf{v}, \mathbf{p_i} - \mathbf{p} >\\ w_i &=< \mathbf{w}, \mathbf{p_i} - \mathbf{p} >\end{aligned}$$

Then we fit the function using least squares:

$$\begin{aligned}E(\mathcal{P}, w(u,v)) &= \sum_{i=1}^{m}(w(u,v) - w_i)^2 = \sum_{i=1}^{m}(\mathbf{q_i}^T\mathbf{s} - w_i)^2\\ &= \sum_{i=1}^{m}[(\mathbf{q_i}^T\mathbf{s})^2 + w_i^2 - 2w_i(\mathbf{q_i}^T\mathbf{s})]\\ &= \mathbf{s}^T(\sum_{i=1}^{m}\mathbf{q_i}\mathbf{q_i}^T)\mathbf{s} + \sum_{i=1}^{m} w_i^2 - 2\mathbf{s}^T\sum_{i=1}^{m} w_i\mathbf{q_i}\end{aligned}$$

$$\min_w E(\mathcal{P}, w) \quad \rightarrow \quad \nabla E(\mathcal{P}, w) = 0 \iff 2(\sum_{i=1}^{m}\mathbf{q_i}\mathbf{q_i}^T)\mathbf{s} - 2\sum_{i=1}^{m} w_i\mathbf{q_i} = 0 \iff$$
$$(\sum_{i=1}^{m}\mathbf{q_i}\mathbf{q_i}^T)\mathbf{s} = \sum_{i=1}^{m} w_i\mathbf{q_i} \iff \mathbf{As} = \mathbf{b}$$

Then the Hessian $\mathbf{H_w}$ is simple to extract from the function $w(u,v)$:

$$\mathbf{H_w} = \begin{pmatrix} \frac{\partial^2 w}{\partial u^2} & \frac{\partial^2 w}{\partial v \partial u} \\ \frac{\partial^2 w}{\partial u \partial v} & \frac{\partial^2 w}{\partial v^2} \end{pmatrix} = \begin{pmatrix} 2a & b \\ b & 2c \end{pmatrix}$$

If we have a triangle mesh we can use its topology to compute the curvatures. For the Gaussian curvature:

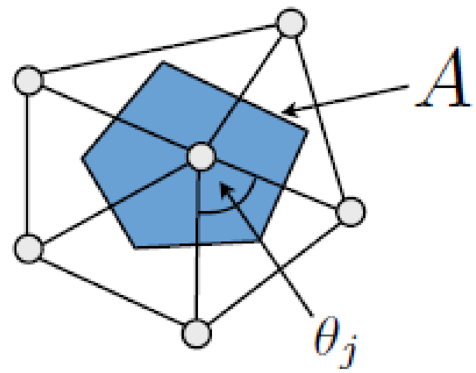


Figure 3: Gaussian curvature using angle deficit.

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

where A is one third of the area of the 1-ring of triangles around a vertex of the mesh.

For the Mean curvature, we can use the norm of the Laplace-Beltrami operator:

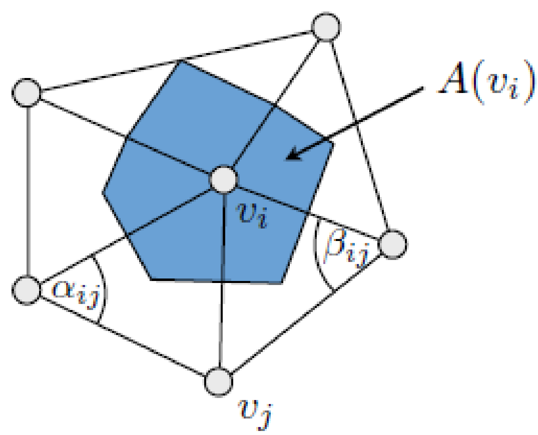


Figure 4: Mean curvature as the norm of Laplace-Beltrami.

$$H(v_i) = \|\Delta_s v_i\|$$

$$\Delta_s v_i = \frac{1}{2 \cdot A(p)} \sum_j (cot\alpha_{ij} + cot\beta_{ij}) \cdot (v_j - v_i)$$

$A(p)$ is the area of influence of vertex p on its adjacent triangles. It can use:

- Barycentric cells
- Voronoi cells
- Mixed cells

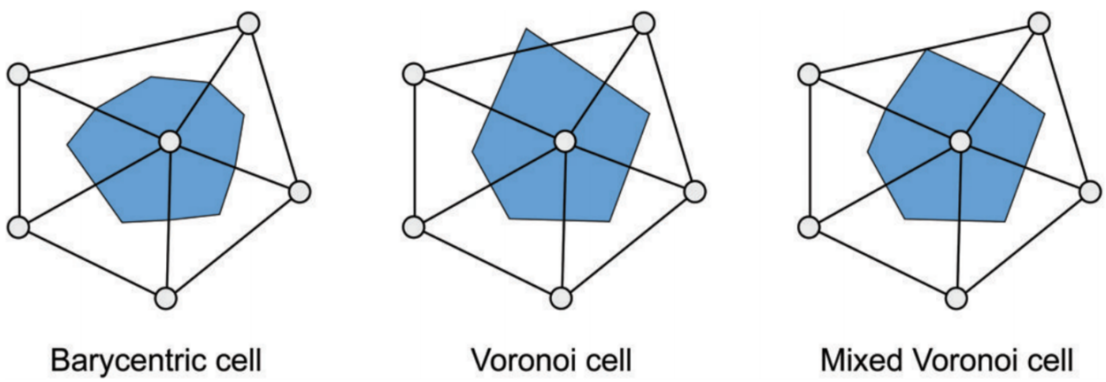


Figure 5: Area of influence of a vertex in a mesh.

3 Smoothing

To reduce noise in a mesh or a point cloud we need to apply smoothing algorithms.

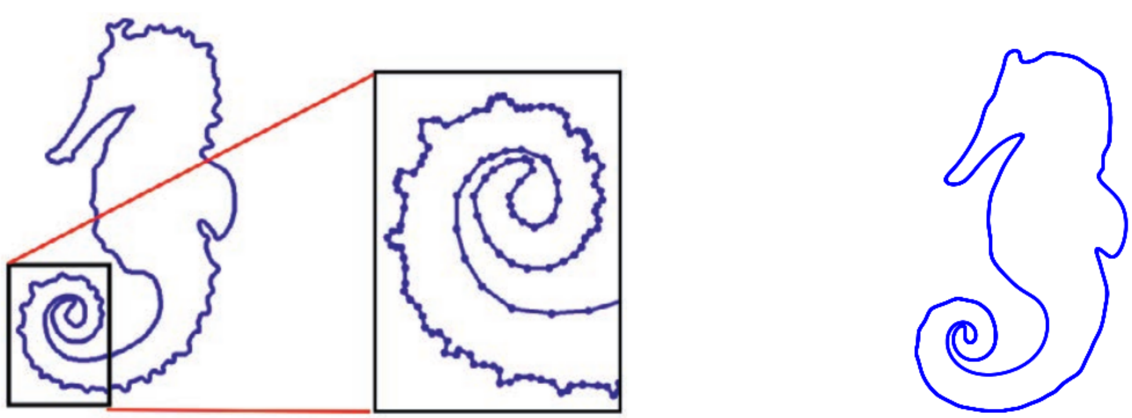


Figure 6: Smoothing a curve.

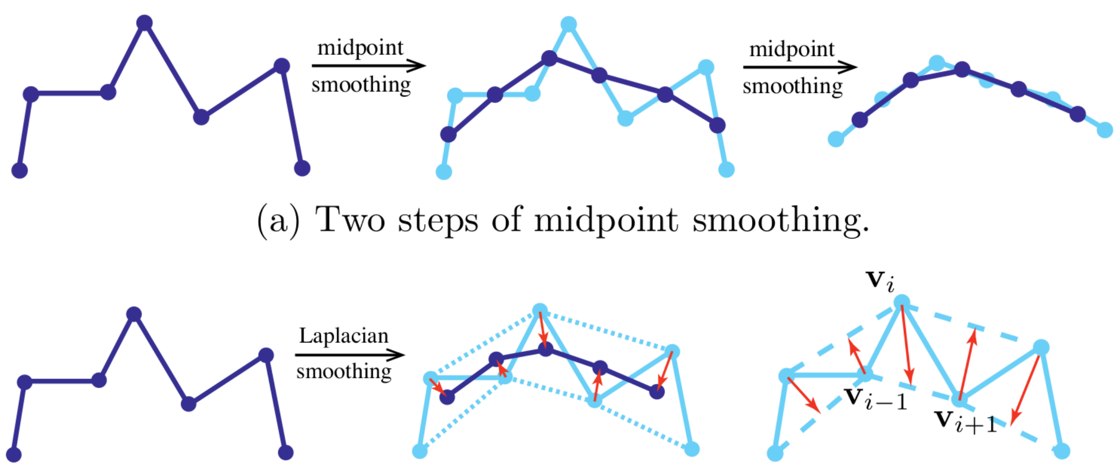
One of the simplest is known as the *midpoint smoothing* algorithm, which can be applied to polygonals. It substitutes the polygonal for a new one where every edge generates a vertex at its midpoint location, and every vertex transforms into an edge. For an edge between vertices v_{i-1} and v_i it generates a vertex at coordinates $(v_{i-1} + v_i)/2$.

Two applications of midpoint smoothing result into *laplacian smoothing*. Here a vertex remains a vertex of the result and its coordinates are:

$$
\begin{aligned}
v_i' &= \frac{1}{2} \cdot \left(\frac{v_{i-1} + v_i}{2} \right) + \frac{1}{2} \cdot \left(\frac{v_i + v_{i+1}}{2} \right) = \\
&= \frac{1}{4} v_{i-1} + \frac{1}{2} v_i + \frac{1}{4} v_{i+1}
\end{aligned}
$$

and the displacement vector from its previous position to its "smoothed" position is called the laplacian of vertex v_i:

$$
\begin{aligned}
\delta(v_i) = v_i' - v_i &= \frac{1}{4} v_{i-1} + \frac{1}{2} v_i + \frac{1}{4} v_{i+1} - v_i = \\
&= \frac{1}{4}(v_{i-1} + v_{i+1}) - \frac{1}{2} v_i
\end{aligned}
$$



(a) Two steps of midpoint smoothing.

(b) Laplacian smoothing.

(c) 1D Laplacians (red).

Figure 7: Midpoint and laplacian smoothing.

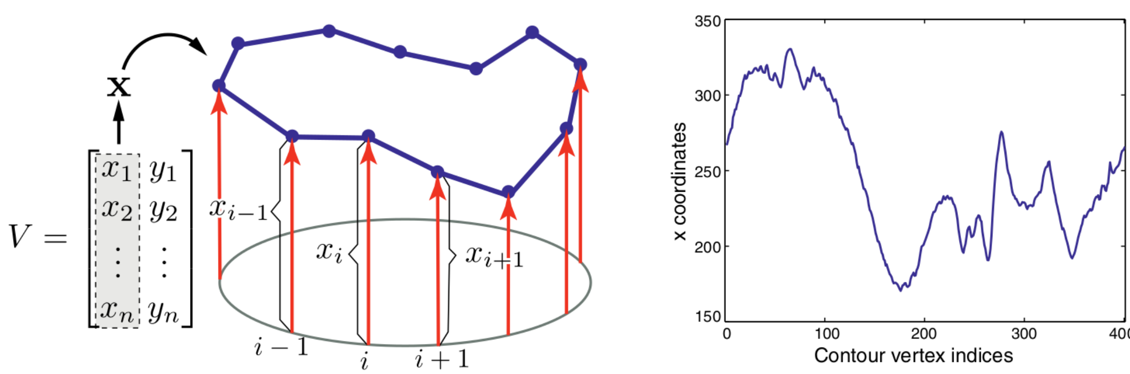
Laplacians can be applied to a full dataset simultaneously in matrix form. As an example, for a 1D periodic signal:

$$\mathbf{x}' = \mathbf{S} \cdot \mathbf{x} \qquad \mathbf{S} = \begin{pmatrix} 1/2 & 1/4 & 0 & \cdots & 0 & 1/4 \\ 1/4 & 1/2 & 1/4 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1/2 & 1/4 \\ 1/4 & 0 & 0 & \cdots & 1/4 & 1/2 \end{pmatrix}$$

which can also be expressed using a laplacian update matrix L:

$$\mathbf{S} = \mathbf{I} - \frac{1}{2} \cdot \mathbf{L} \qquad \mathbf{L} = \begin{pmatrix} 1 & -1/2 & 0 & \cdots & 0 & -1/2 \\ -1/2 & 1 & -1/2 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \\ 0 & 0 & 0 & \cdots & 1 & -1/2 \\ -1/2 & 0 & 0 & \cdots & -1/2 & 1 \end{pmatrix}$$

Matrix $\mathbf{L}$ is called the *1D discrete Laplacian operator*.



(a) (b)

Figure 8: 1D signal.