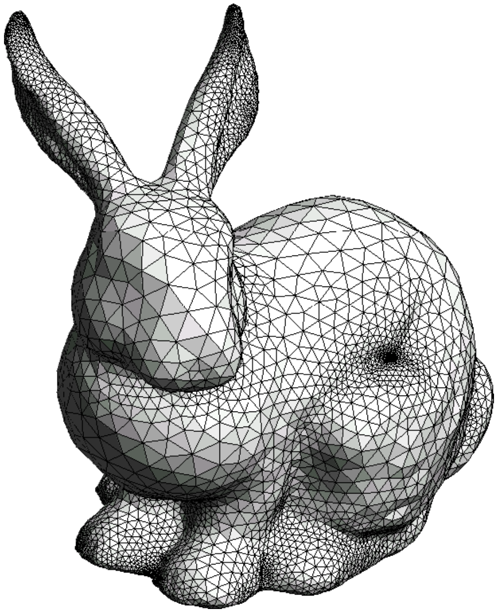


Geometric Modeling in Graphics

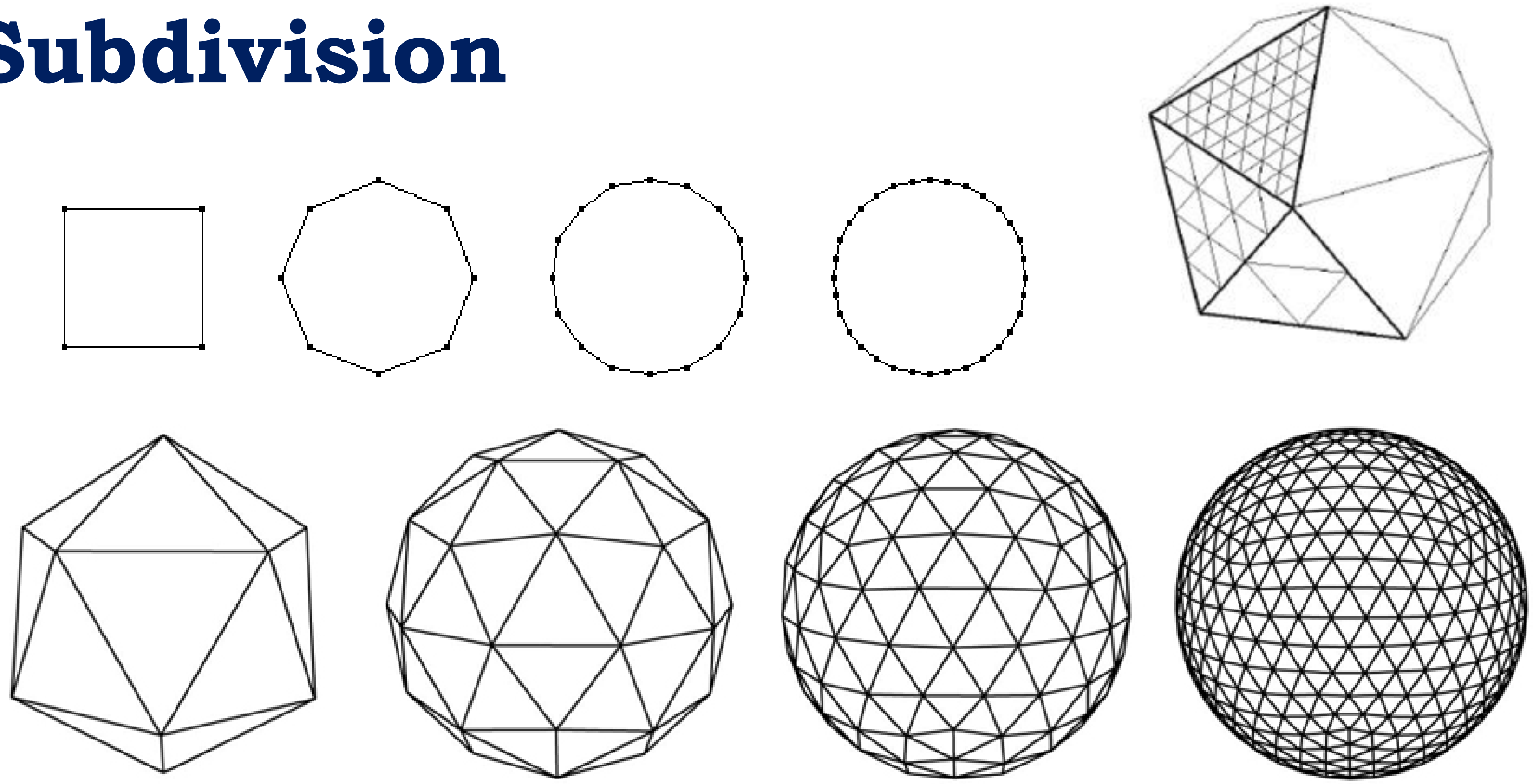
Part 4: Mesh smoothing

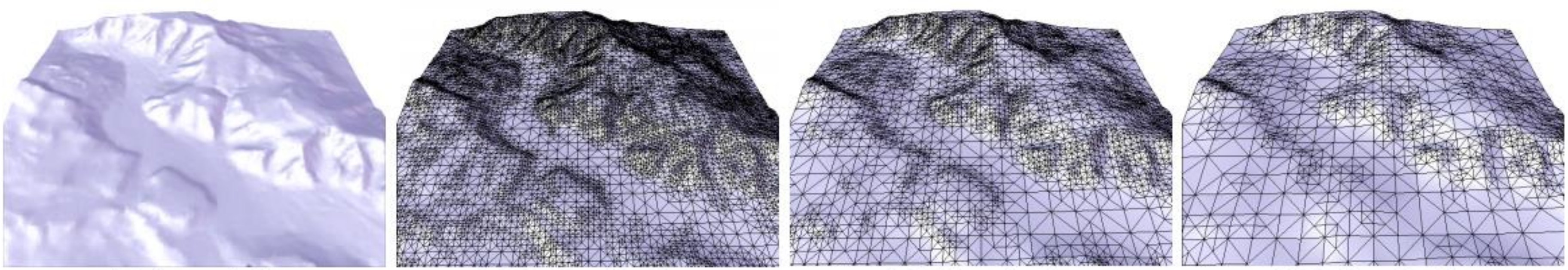
Martin Samuelčík
www.sccg.sk/~samuelcik
samuelcik@sccg.sk

Subdivision

- Generating new object (mesh, polyline) from old one by dividing edges or faces into smaller parts
- Generating new vertices, edges, faces, changing position of old vertices
- One step of subdivision – from object $\mathbf{P}^i$ to object $\mathbf{P}^{i+1}$, i=0,…
  - Subdivision scheme S – $\mathbf{P}^{i+1} = S(\mathbf{P}^i)$
  - Control (starting) object – $\mathbf{P}^0$
  - Limit object – $\mathbf{P}$, $\mathbf{P}^\infty = \lim \mathbf{P}^i, i \to \infty$
- Interpolation schemes – vertices of $\mathbf{P}^i$ are included in $\mathbf{P}^{i+1}$, each $\mathbf{P}^i$ and $\mathbf{P}$ passes through vertices of $\mathbf{P}^0$
- Approximation schemes – Each object $\mathbf{P}^i$ and $\mathbf{P}$ only approximates shape of $\mathbf{P}^0$
- Continuity – continuity of limit object $\mathbf{P}$, usually $C^0, C^1, C^2, \ldots$

Subdivision

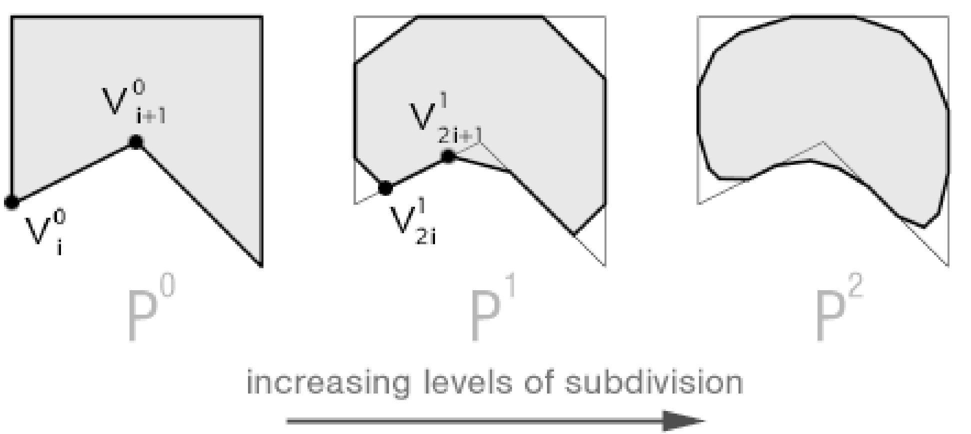




a) full resolution b) 55% reduction c) 82% reduction d) 93% reduction

Polyline subdivision

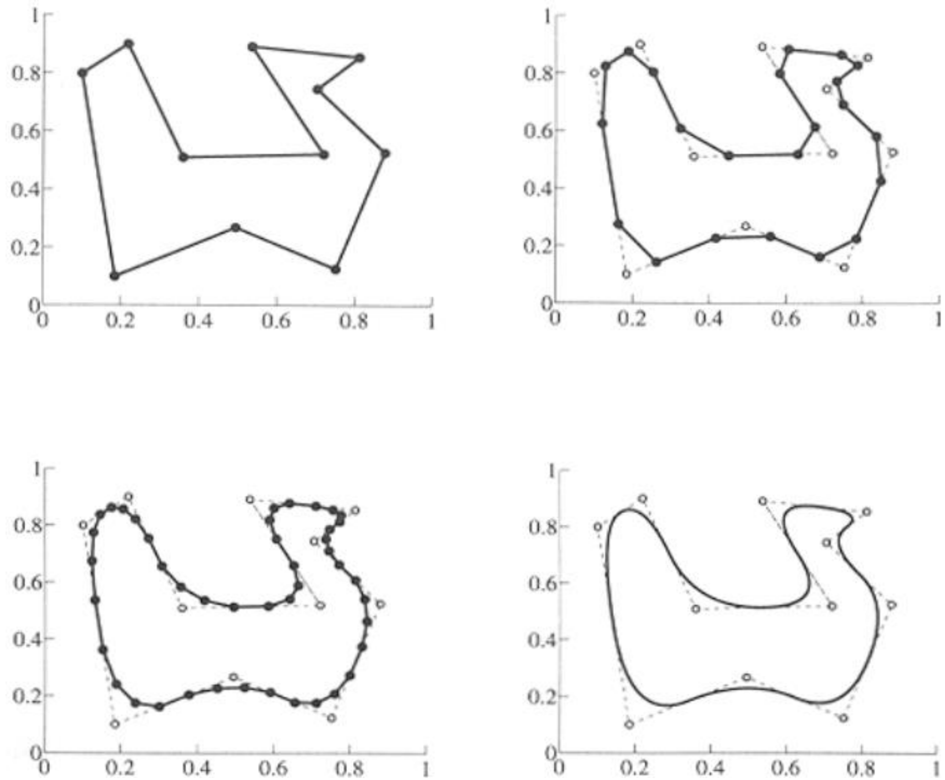
- Chaikin subdivision scheme
- Corner cutting algorithm, approximating scheme
- **P**^{i} has vertices $V_1^i, V_2^i, .., V_n^i$
- **P**$^{i+1}$ has vertices $V_1^{i+1}, V_2^{i+1}, \ldots, V_m^{i+1}$
- $V_{2j}^{i+1} = 0.25 * V_j^i + 0.75 * V_{j+1}^i$, $j=1, 2, \ldots, m/2$
- $V_{2j-1}^{i+1} = 0.75 * V_j^i + 0.25 * V_{j+1}^i$, $j=1, 2, \ldots, m/2$
- Open polyline
  - $m=2n-2$
- Closed polyline
  - $m=2n$
  - $V_{n+1}^i = V_n^i$



Chaikin scheme

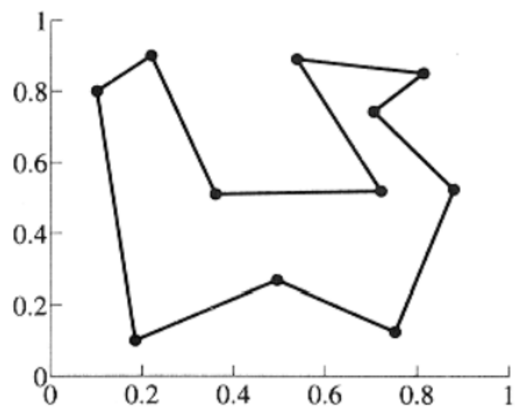
- Limit of Chaikin scheme – C^1 quadratic B-spline curve
- Limit curve in convex hull of control polyline
- Matrix notation, used for determination of mathematical properties

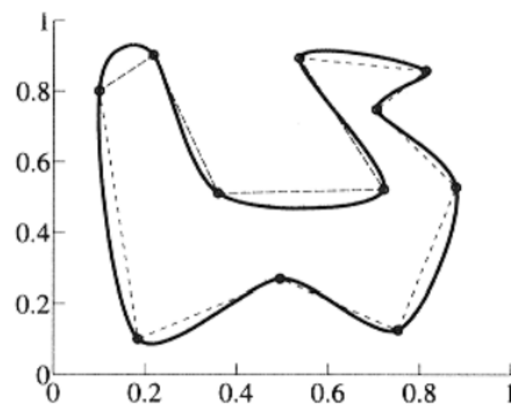
$$\begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \\ f_7 \end{bmatrix} = \begin{bmatrix} \frac{3}{4} & \frac{1}{4} & 0 & 0 \\ \frac{1}{4} & \frac{3}{4} & 0 & 0 \\ 0 & \frac{3}{4} & \frac{1}{4} & 0 \\ 0 & \frac{1}{4} & \frac{3}{4} & 0 \\ 0 & 0 & \frac{3}{4} & \frac{1}{4} \\ 0 & 0 & \frac{1}{4} & \frac{3}{4} \\ \frac{1}{4} & 0 & 0 & \frac{3}{4} \\ \frac{3}{4} & 0 & 0 & \frac{1}{4} \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$



Polyline subdivision

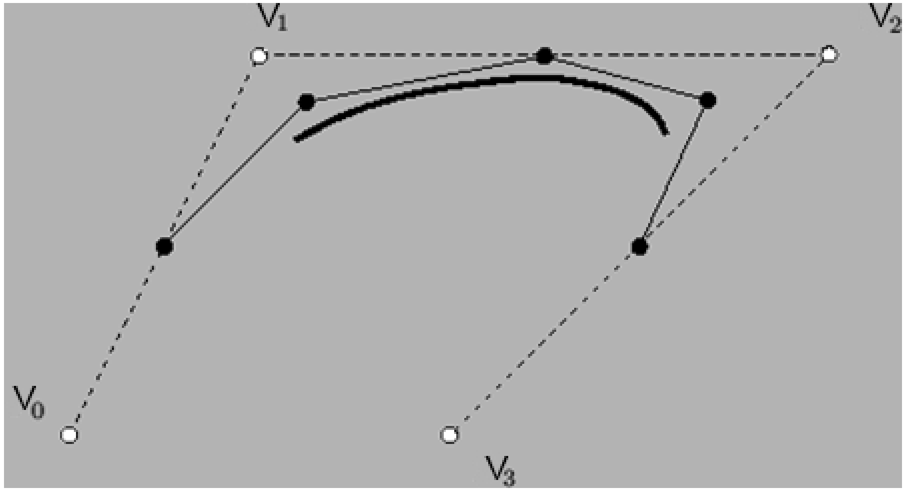
- Interpolation Scheme, limit curve is C^1
- Dyn-Levin-Gregory
- $V_{2j-1}^{i+1} = V_j^i$, $j=1, 2, \ldots, m / 2$
- $V_{2j}^{i+1} = (-1/16)V_{j-1}^i + (9/16)V_j^i + (9/16)V_{j+1}^i + (-1/16)V_{j+2}^i$
  $j=1, 2, \ldots, m / 2$
- Open polyline: $m=2n-1, V_0^i=V_1^i, V_{n+1}^i=V_n^i$
- Closed polyline: $m=2n, V_0^i = V_n^i, V_{n+1}^i=V_2^i$





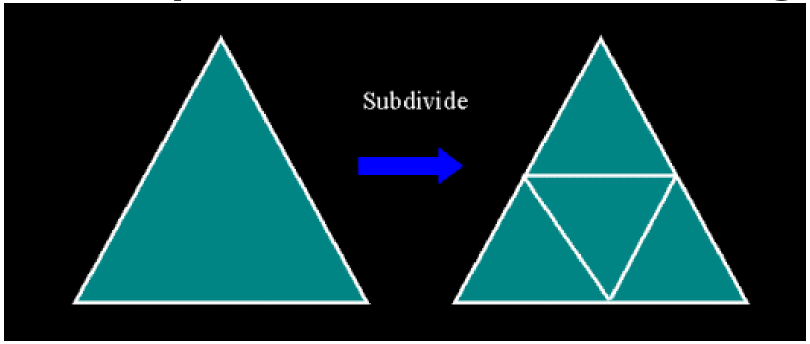
Polyline subdivision

- Catmul-Clark approximating subdivision scheme
- Limit curve is C^2 cubic B-spline curve
- $V_{2j}^{i+1} = (1/2) * V_j^i + (1/2) * V_{j+1}^i$, j=1, 2, …, m / 2
- $V_{2j-1}^{i+1} = (1/8) V_{j-1}^i + (6/8) V_j^i + (1/8) V_{j+1}^i$, j=1, 2, …, m / 2
- Open polyline: m=2n-1, $V_0^i = V_1^i, V_{n+1}^i = V_n^i$
- Closed polyline: m=2n, $V_0^i = V_n^i, V_{n+1}^i = V_2^i$



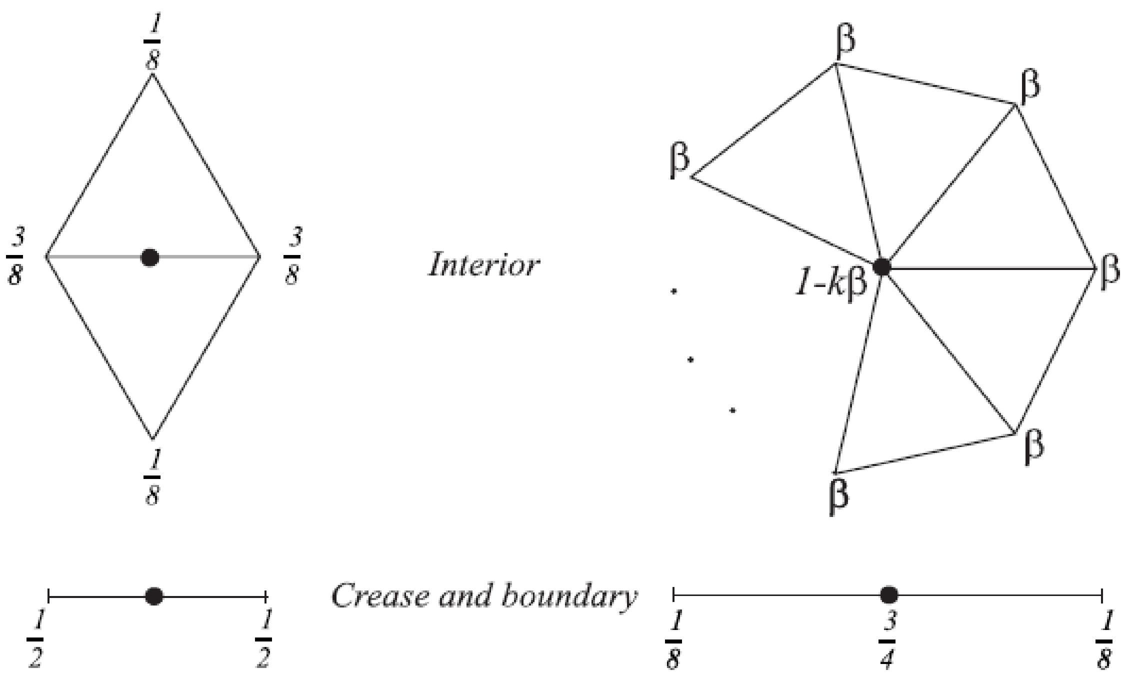
Loop subdivision

- Loop approximating scheme for triangular meshes
- Each subdivision step, triangle is divided into 4 subtriangles
- For each edge of mesh, new vertex is created near center of edge (odd vertex)
- Each old vertex is moved to new position (even vertex)
- Position of odd and even vertex is computed as barycentric combination of old vertices in its neighborhood – barycentric coordinates given as mask



Loop subdivision

- Special rules for vertices, edges lying on boundary or marked as crease
- Using DCEL to get vertex neighborhood info



a. Masks for odd vertices

b. Masks for even vertices

Original Loop

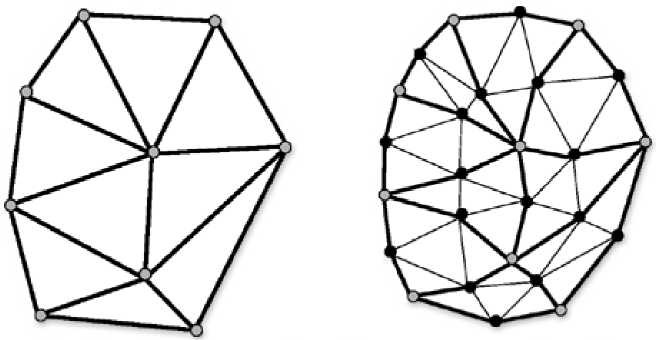
$$\beta = \frac{1}{n}\left(\frac{5}{8} - \left(\frac{3}{8} + \frac{1}{4}\cos\frac{2\pi}{n}\right)^2\right)$$

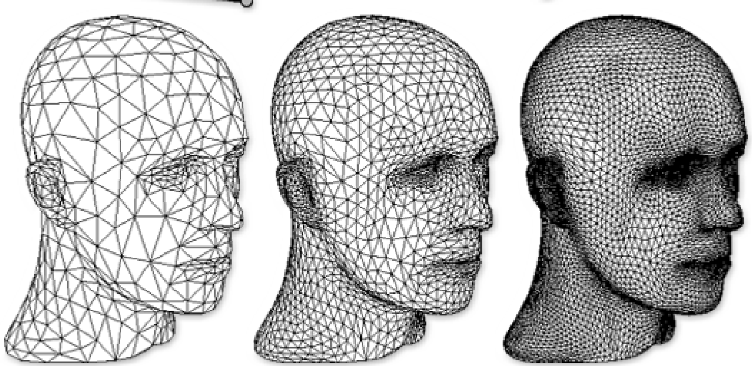
Warren

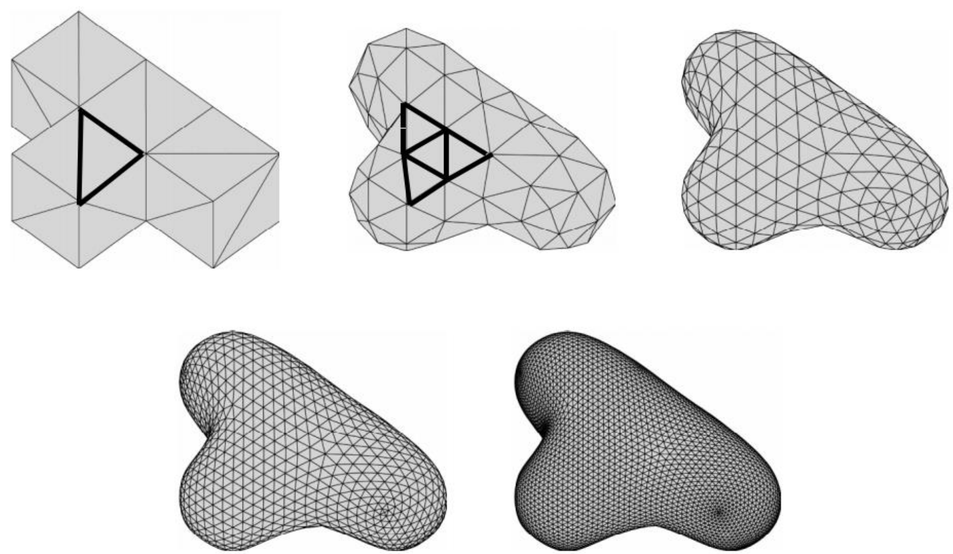
$$\beta = \begin{cases} \frac{3}{8n} & n > 3 \\ \frac{3}{16} & n = 3 \end{cases}$$

Loop subdivision

- Control mesh $\mathbf{M}^0$ – arbitrary triangular mesh
- Limit mesh $\mathbf{M}^\infty$ – C^2 smooth except small neighborhood of extraordinary vertices
- Extraordinary vertex – with valence not equal to 6



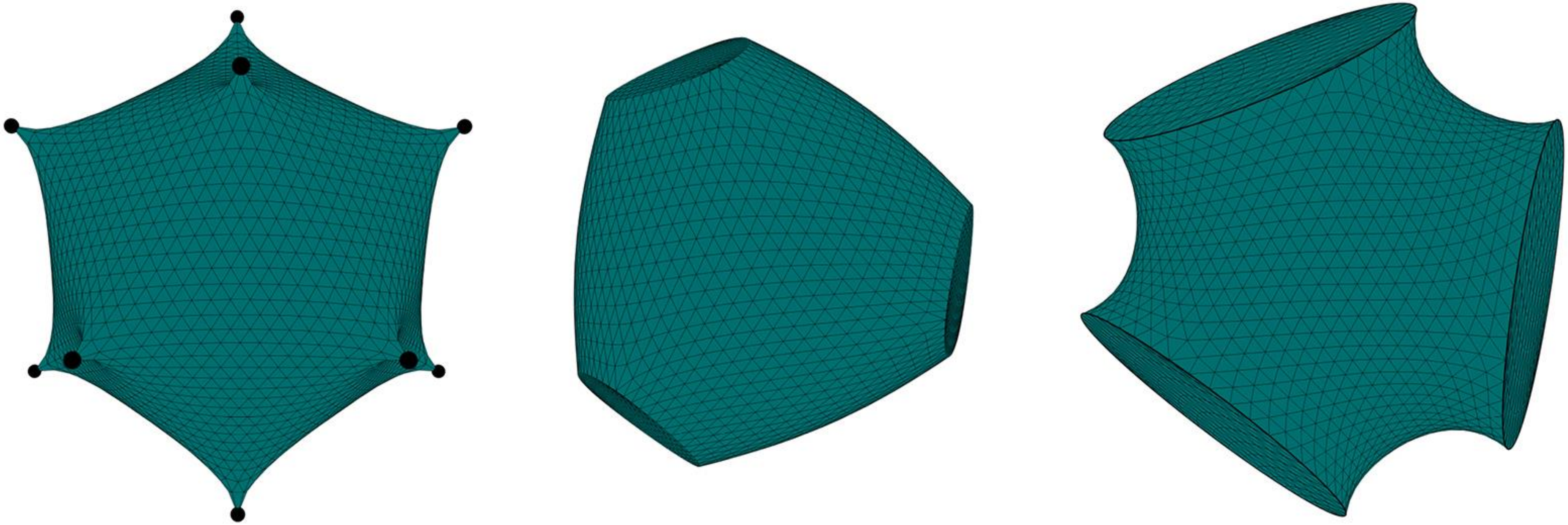




Loop subdivision & DCEL

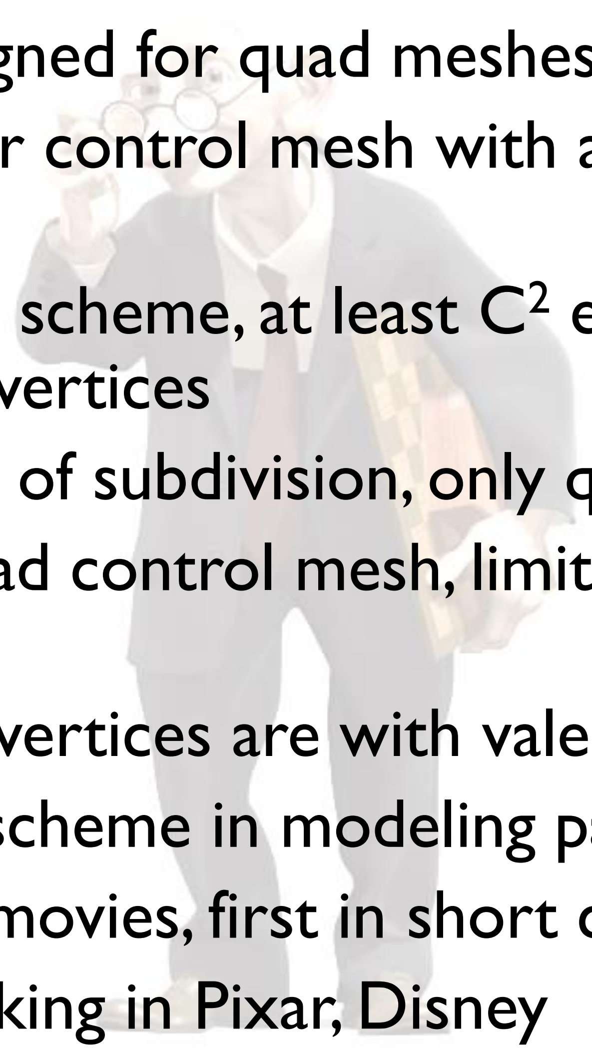
- One Loop subdivision step from mesh $\mathbf{M}^i$ to mesh $\mathbf{M}^{i+1}$
- Creating DCEL structure for mesh $\mathbf{M}^{i+1}$
  - 1. For each vertex of $\mathbf{M}^i$, create new even vertex of $\mathbf{M}^{i+1}$ and compute its position from positions of $\mathbf{M}^i$ vertices, remember connection of new and old vertex
  - 2. For each edge of $\mathbf{M}^i$, create and compute coordinates of new odd vertex, remember connection of new vertex with both half-edges of edge
  - 3. For each face of $\mathbf{M}^i$, create 4 new DCEL faces (triangles) and 12 new half-edges and fill its properties based on connections from previous steps except opposite pointers, remember connection of new faces with old faces
  - 4. Fill opposite half-edges of mesh $\mathbf{M}^{i+1}$ using connections from step 3

Loop subdivision - Creases



http://www.bespokegeometry.com/2015/01/29/mesh-subdivision-loop-and-catmull-clark/

Catmull-Clark subdivision



- Originally designed for quad meshes
- Generalized for control mesh with arbitrary simple polygons
- Approximating scheme, at least C^2 except neighborhood of extraordinary vertices
- After first step of subdivision, only quads are present
- For regular quad control mesh, limit surface is bicubic B-spline surface
- Extraordinary vertices are with valence not equal to 4
- Most popular scheme in modeling packages
- Used in many movies, first in short called Geri's game
- Catmull – working in Pixar, Disney

Catmull-Clark subdivision

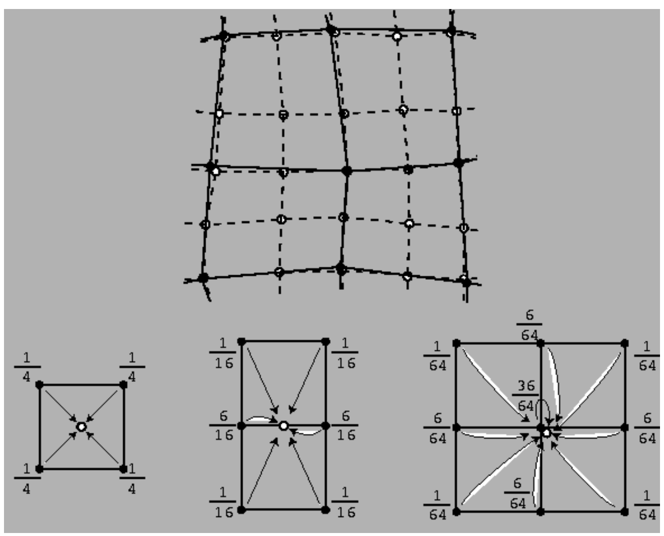
- One step of subdivision process, $\mathbf{M}^i$ to mesh $\mathbf{M}^{i+1}$
- 3 kinds of new vertices of mesh $\mathbf{M}^{i+1}$, created for each element (face, edge, vertex) of $\mathbf{M}^i$
  - Face point – average of all vertices of face
  - Edge point – average of two points from neighboring faces
  - Vertex point
    - F – average of all face points for faces touching vertex
    - R – average of all edge points for edges touching vertex
    - P – position of vertex
    - n – valence of vertex

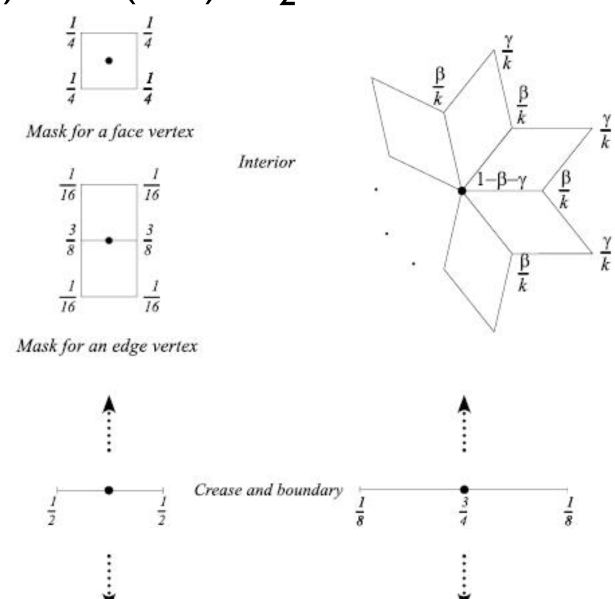
$$\frac{F + 2R + (n-3)P}{n}.$$

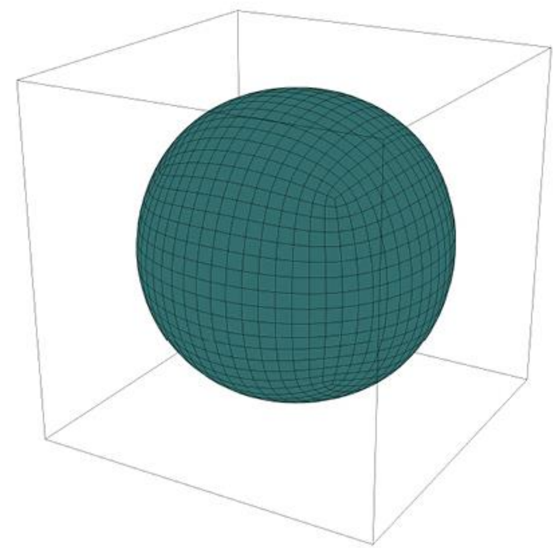
- To create mesh $\mathbf{M}^{i+1}$, connect each face point with corresponding edge points and each vertex point with corresponding edge points

Catmull-Clark subdivision

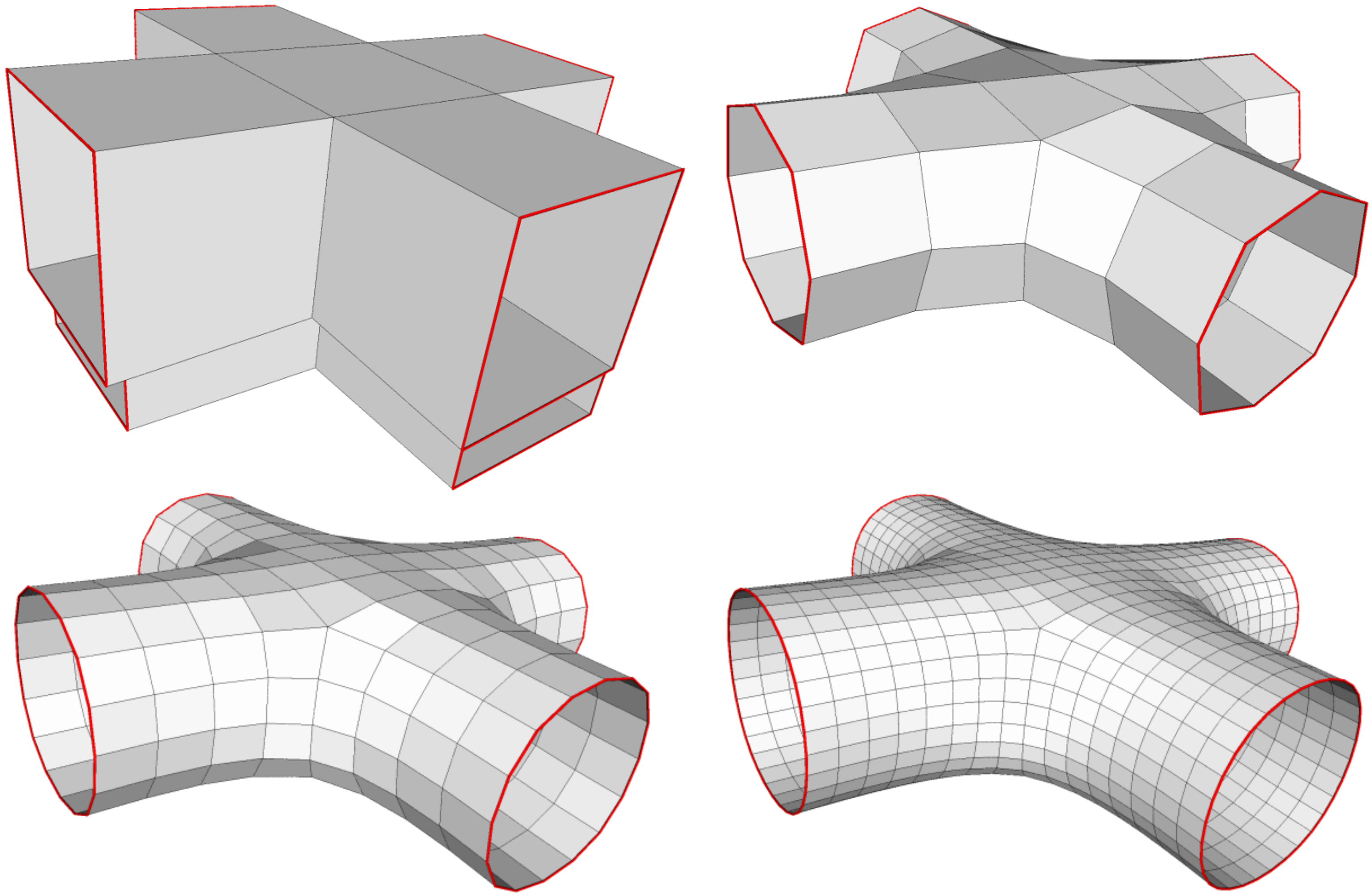
- Rules for boundary and crease are given by curve rules
  - Edge Point is computed as center of edge
    - $EP = (1/2) * V_1 + (1/2) * V_2$
  - Vertex point is computed as combination of vertex and its neighbor vertices on boundary (crease)
    - $VP = (1/8) N_1 + (6/8) V + (1/8) N_2$







Catmull-Clark subdivision

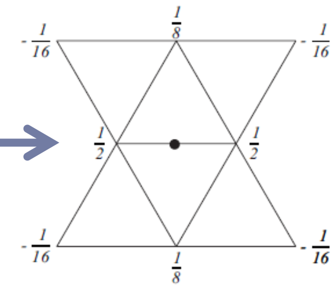


Modified Butterfly subdivision

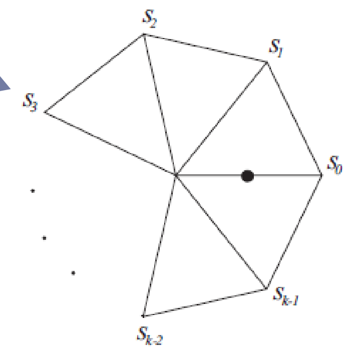
- Interpolation scheme on triangular meshes
- C1smooth everywhere except vertices with valence equal to 3 or greater then 7
- Extraordinary vertices with valence not equal to 6
- Dividing each triangle of mesh $\mathbf{M}^i$ into 4 triangles of $\mathbf{M}^{i+1}$
- All vertices of $\mathbf{M}^i$ are present in $\mathbf{M}^{i+1}$
- For each edge of $\mathbf{M}^i$, new edge point (odd vertex) is created and used when triangle is divided into 4 new triangles
- Rules for edge point are based on valence of end vertices of that edge

Modified Butterfly subdivision

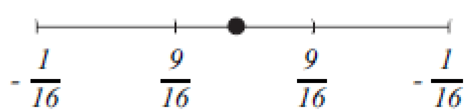
- Computing edge point (odd vertex) for edge (V_1,V_2) leads to 4 possibilities
  - V_1 and V_2 both have valence k=6 (are regular)



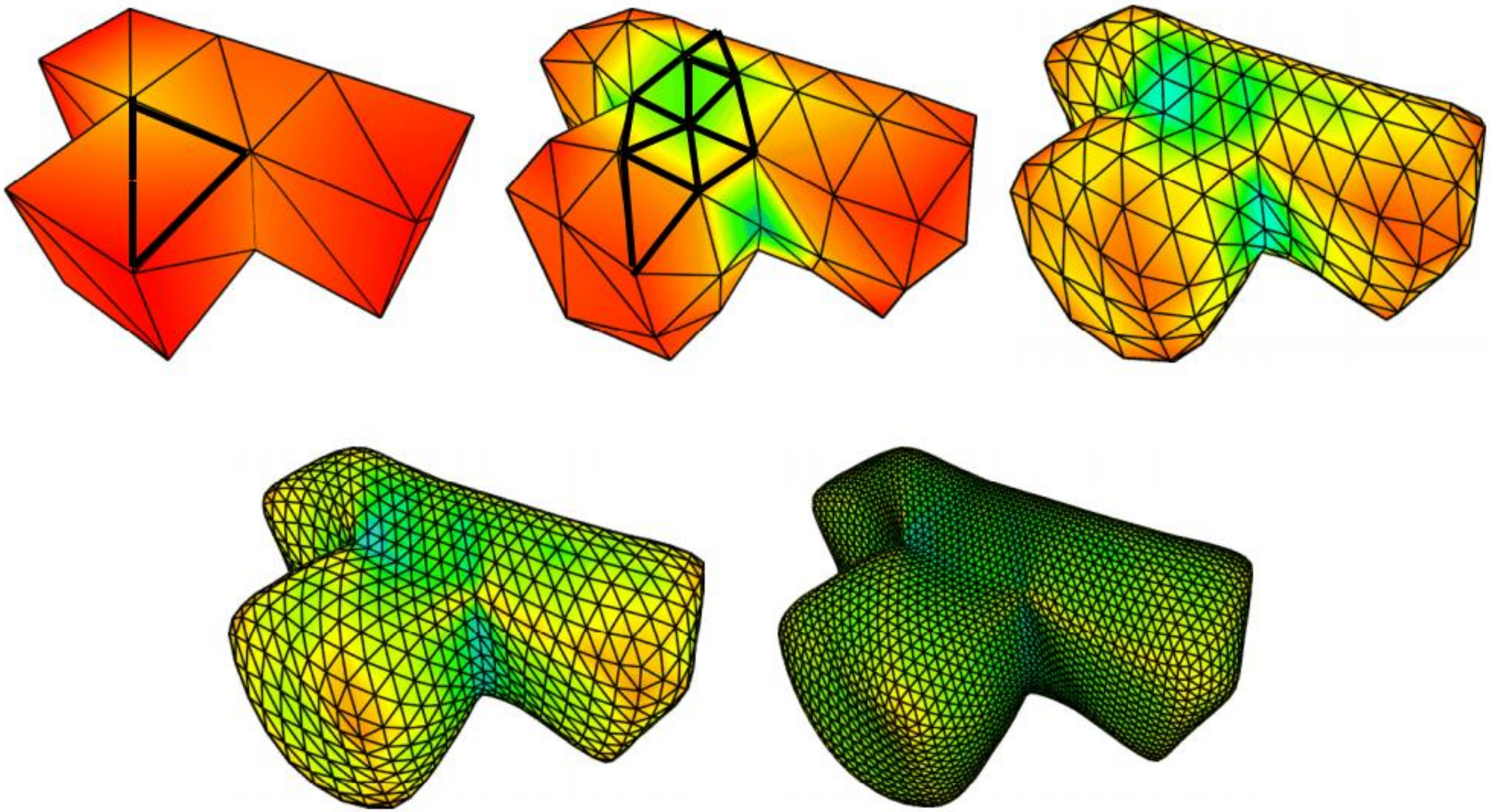
  - V_1 has valence k=6 and V_2 has not
    - for k = 3 $s_0 = \frac{5}{12},\ s_{1,2} = -\frac{1}{12}$
    - for k = 4 $s_0 = \frac{3}{8},\ s_2 = -\frac{1}{8},\ s_{1,3} = 0$
    - for k>= 5 $s_i = \frac{1}{k}\left(\frac{1}{4} + \cos\frac{2i\pi}{k} + \frac{1}{2}\cos\frac{4i\pi}{k}\right)$



  - V_1 and V_2 have valence not equal to 6
    - Average the results of using the extraordinary stencil on each of them
  - Edge is on boundary



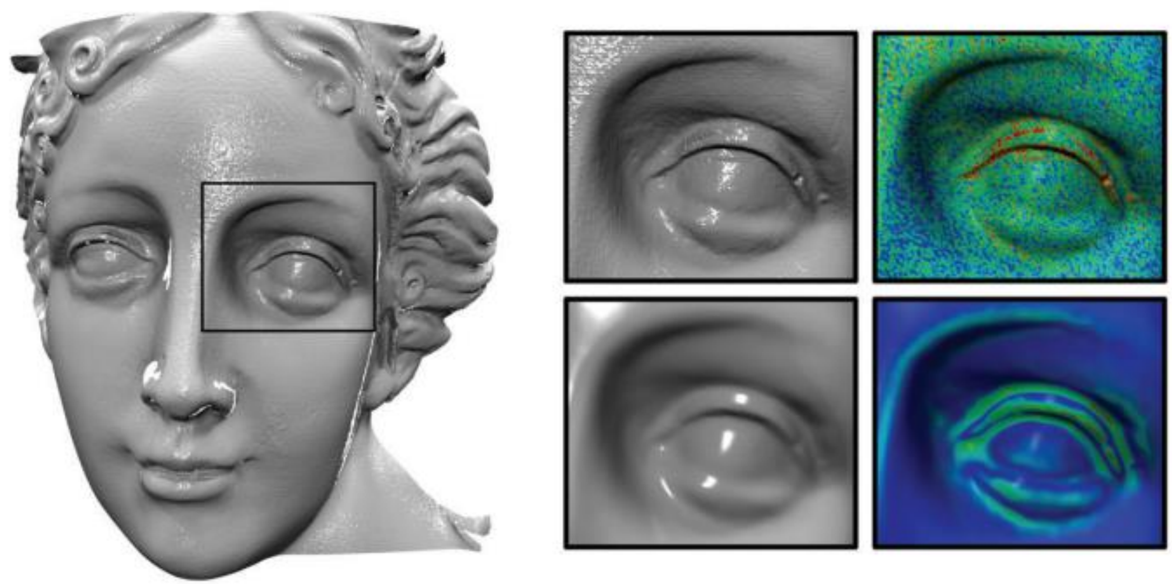
Modified Butterfly subdivision



http://graphics.stanford.edu/courses/cs468-10-fall/LectureSlides/10_Subdivision.pdf

Mesh smoothing

- Changing position of mesh vertices such that updated mesh is more smooth than given mesh
- No topology change inside mesh
- Increasing continuity of function over mesh
- Simulating (heat) diffusion, low pass filter
- Noise removal



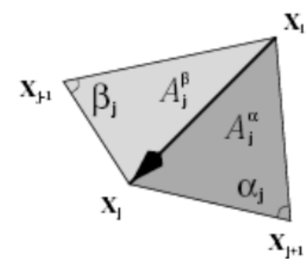
http://staff.ustc.edu.cn/~tongwh/GM_2011/textbooks/Polygon Mesh Processing Mario Botsch et.al 2010.pdf

Laplacian smoothing

- Based on Fourier analysis
- Vertices of mesh are incrementally moved in direction of Vector Laplacian $\Delta f = \nabla^2 f = \sum_{i=1}^{n} \frac{\partial^2 f}{\partial x_i^2}$ $\nabla^2 \mathbf{A} = (\nabla^2 A_x, \nabla^2 A_y, \nabla^2 A_z),$
- Approximating Laplacian on meshes
  - Computation of mesh Laplacian for each vertex $x_i - L(x_i)$
  - Mesh Laplacian is vector created as linear combination of vertex x_i and vertices from its 1-ring neighborhood $N_1(i)$

$$L(x_i) = \sum_{j \in N_1(i)} w_{ij}(x_j - x_i)$$

  - Simple Laplacian, uniform weights $w_{ij} = \frac{1}{m}$
  - Scale-dependent Laplacian, Fujiwara weights $w_{ij} = \frac{1}{|e_{ij}|}$
  - Cotangent Laplacian $w_{ij} = \cot \alpha_j + \cot \beta_j$
    - $L(x_i) = -2H\boldsymbol{n}_i$ (Laplacian is mean curvature normal)

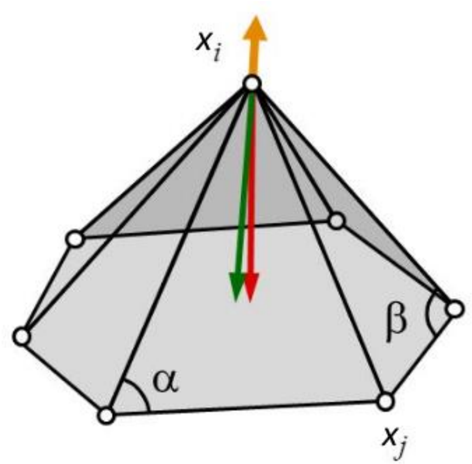


Laplacian smoothing

- Simulating diffusion on mesh using forward difference

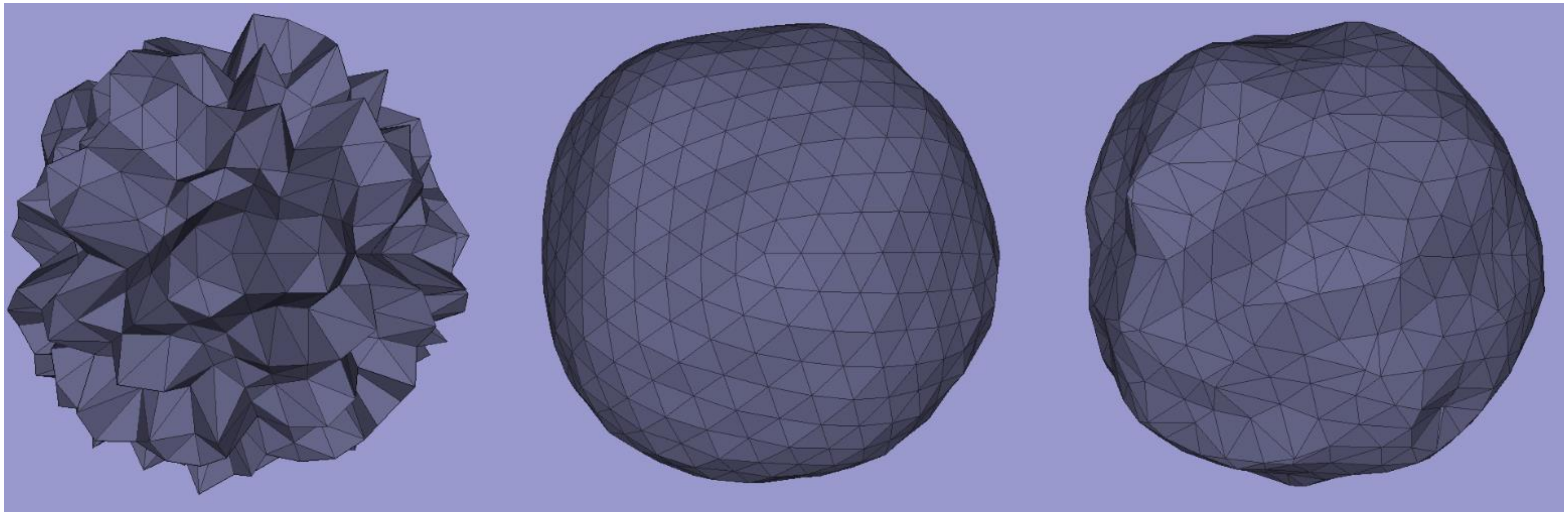
$$\frac{\partial X}{\partial t} = \lambda L(X)$$

- Iterative process over each vertex, starting with base mesh
  - $(x_1, x_2, \ldots, x_n) = (x_1^0, x_2^0, \ldots, x_n^0)$
- One step of process computes new positions of vertices
  - $(x_1^j, x_2^j, \ldots, x_n^j) \longrightarrow (x_1^{j+1}, x_2^{j+1}, \ldots, x_n^{j+1})$
  - Compute $L(x_i^j)$
  - $x_i^{j+1} = x_i^j + \lambda dt L(x_i^j)$, I = 1,2,…,n
  - λ - scalar that controls the diffusion speed
  - dt - sufficiently small time step
- Finish after user defined number of steps

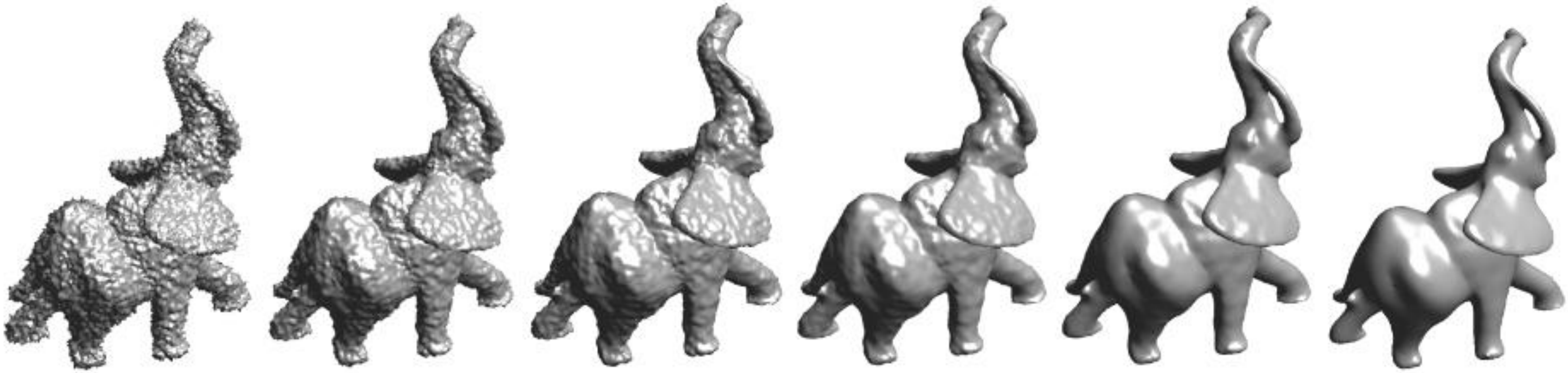


Laplacian smoothing

http://w3.impa.br/~zang/pg2012/exe3.html

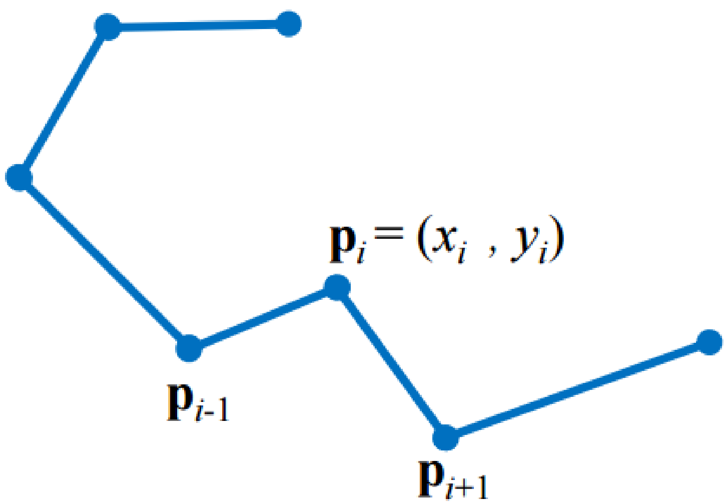


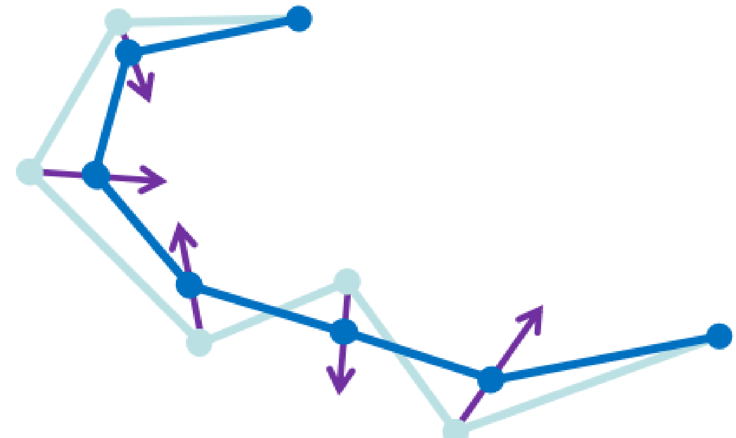
Base mesh with noise　　Uniform Laplacian smooth, 3 iterations　　Cotangent Laplacian smooth, 3 iterations



https://www.ceremade.dauphine.fr/~peyre/teaching/manifold/tp4.html

Laplacian smoothing - Curves





$$\mathbf{p}_i \leftarrow \mathbf{p}_i + \frac{1}{2} L(\mathbf{p}_i)$$

Finite difference discretization of second derivative
= Laplace operator in one dimension

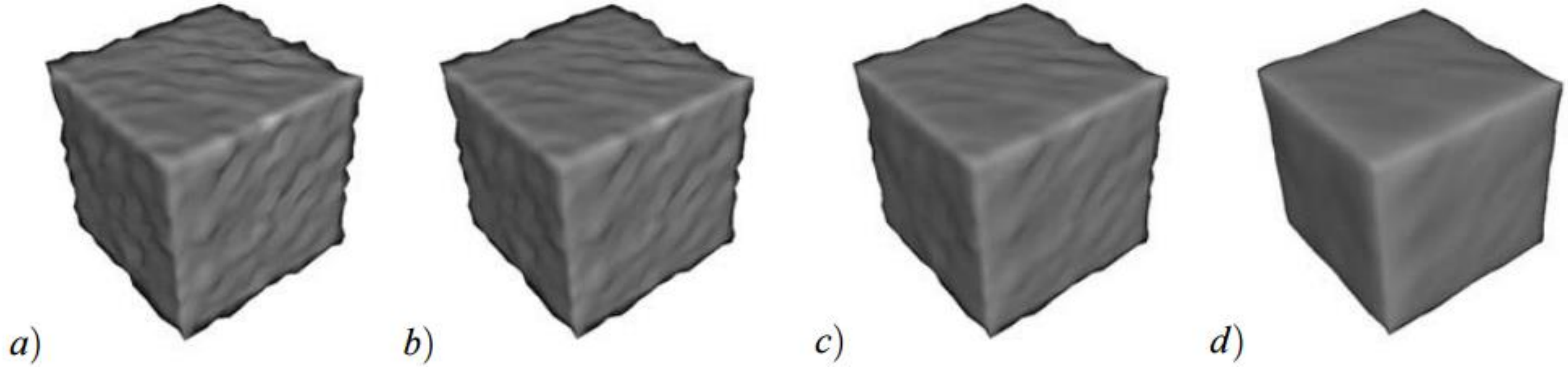
$$L(\mathbf{p}_i) = \frac{1}{2}(\mathbf{p}_{i+1} - \mathbf{p}_i) + \frac{1}{2}(\mathbf{p}_{i-1} - \mathbf{p}_i)$$

http://graphics.stanford.edu/courses/cs468-12-spring/LectureSlides/06_smoothing.pdf

Other smoothing algorithms

- http://www.geometry.caltech.edu/pubs/JDD03.pdf

- https://otik.uk.zcu.cz/bitstream/handle/11025/10872/Svub.pdf?sequence=1



The End
for today