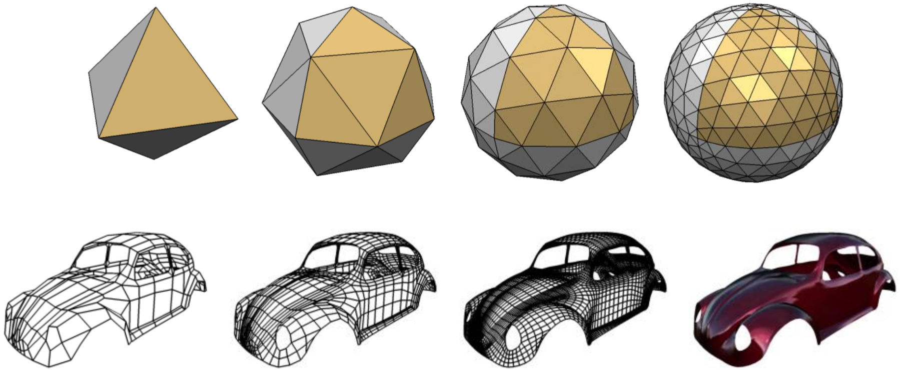
Surface Representations and Geometric Modeling

Prof. Dr. Markus Gross

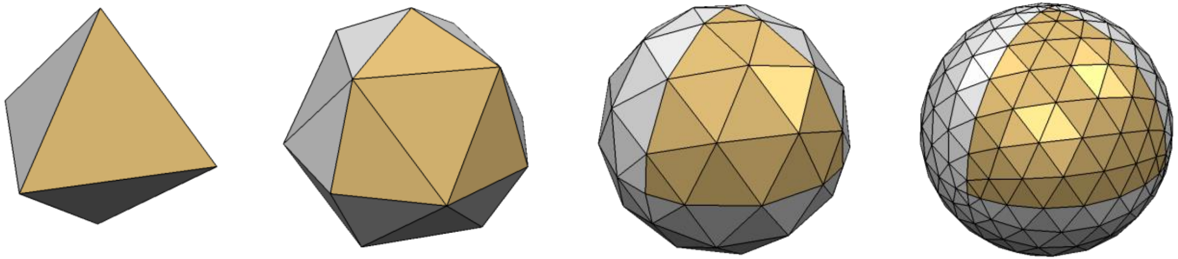


ETH zürich

cgl

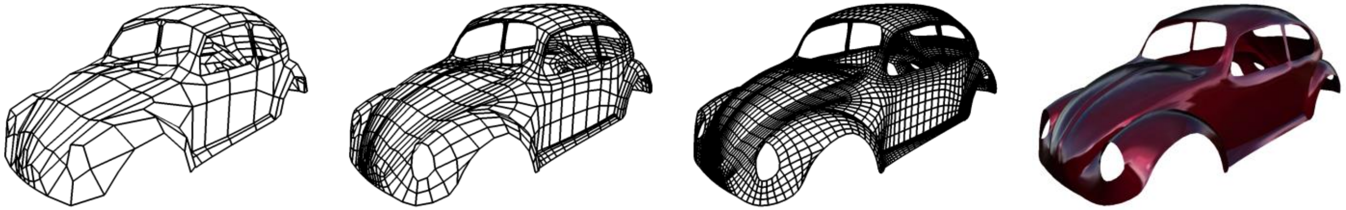
Subdivision Surfaces

- Generalization of spline curves / surfaces
  - Arbitrary control meshes
  - Successive refinement (subdivision)
  - Converges to smooth limit surface
  - Connection between splines and meshes



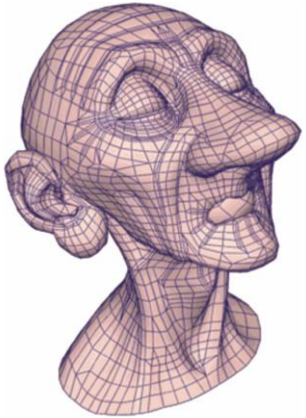
Subdivision Surfaces

- Generalization of spline curves / surfaces
  - Arbitrary control meshes
  - Successive refinement (subdivision)
  - Converges to smooth limit surface
  - Connection between splines and meshes



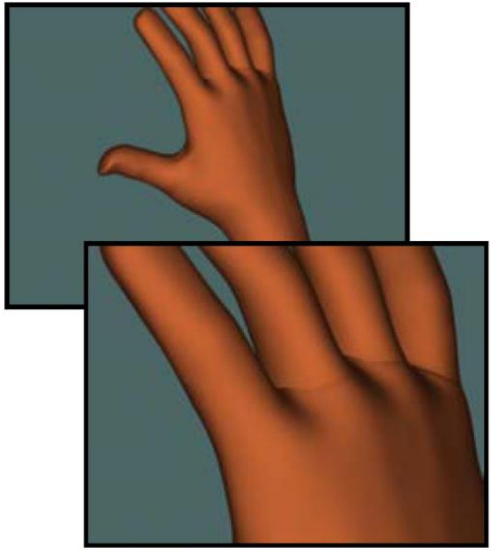
Example: Geri's Game (Pixar)

- Subdivision used for
  - Geri's hands and head
  - Clothing
  - Tie and shoes

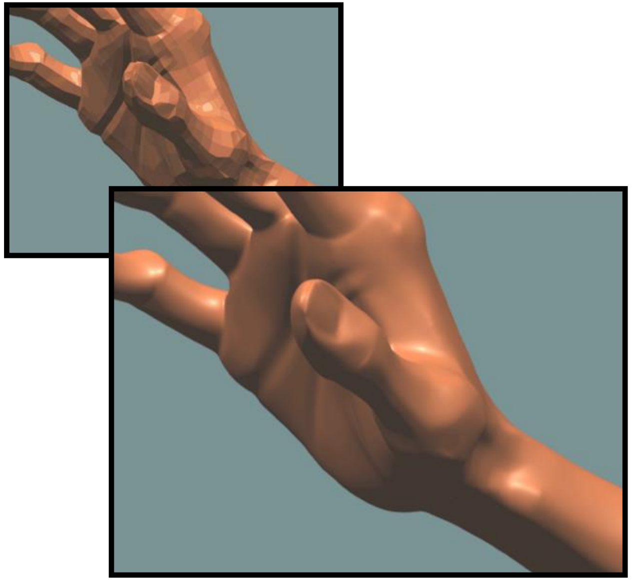


Example: Geri's Game (Pixar)

Woody's hand (NURBS)



Geri's hand (subdivision)

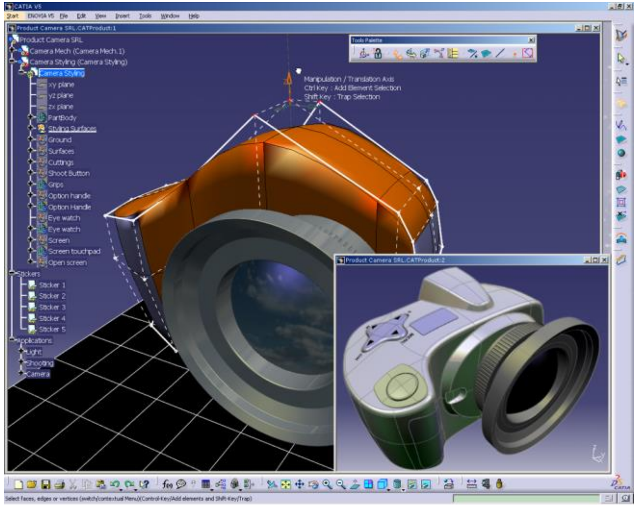


Example: Geri's Game (Pixar)

- Sharp and semi-sharp features

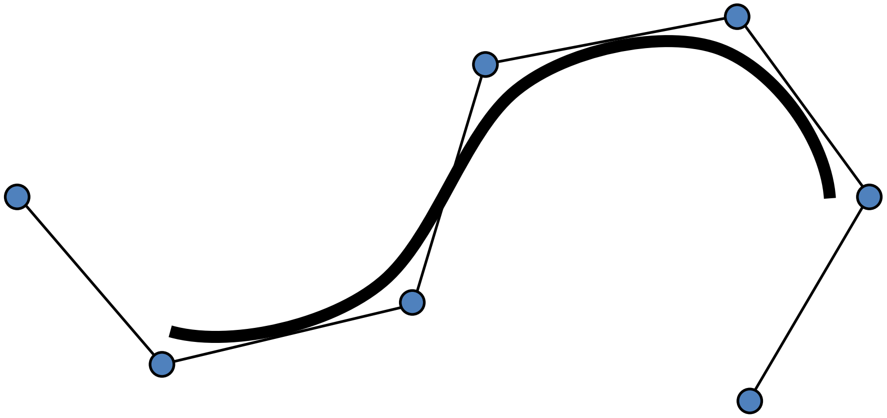
Example: Dassault's CATIA

- Traditional CAD system based on splines
- Latest version also includes subdivision surfaces



Subdivision Curves

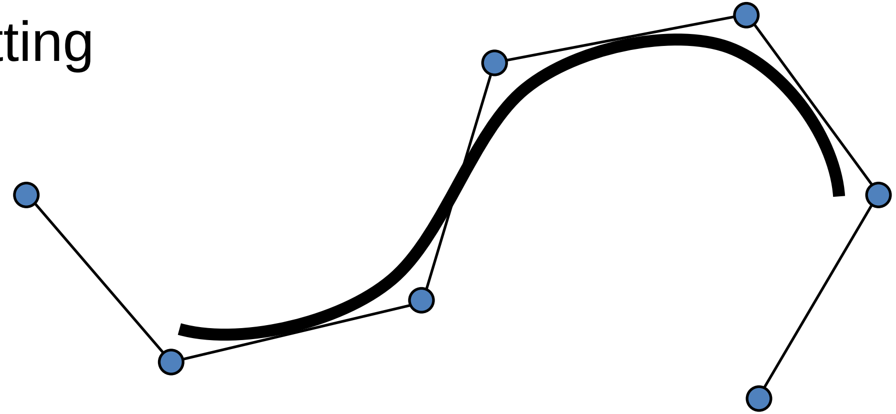
Given a control polygon...



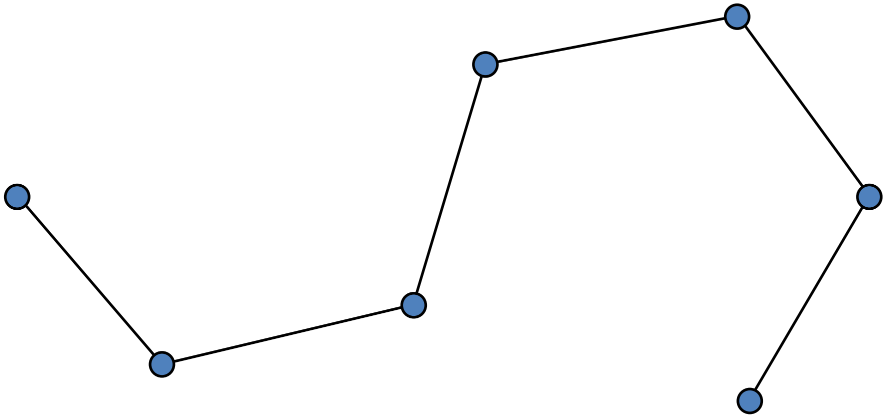
...find a smooth curve related to that polygon.

Subdivision Curve Types

- Approximating
- Interpolating
- Corner Cutting

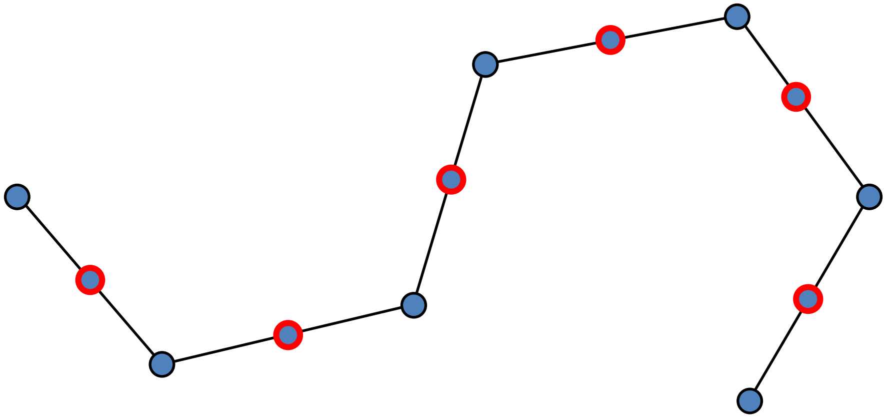


Approximating



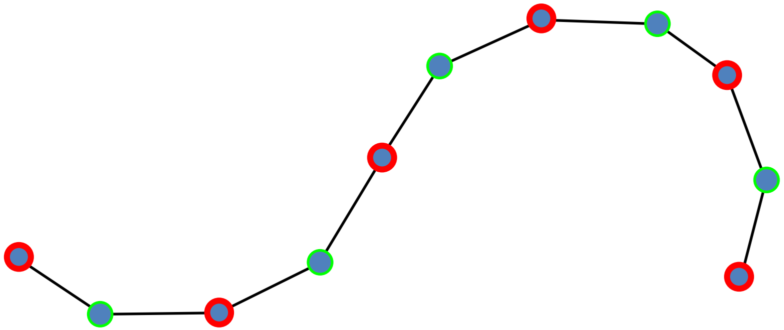
Approximating

Splitting step: split each edge in two



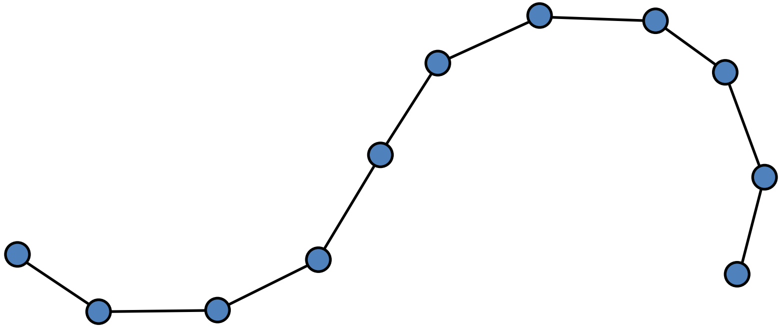
Approximating

Averaging step: relocate each (original) vertex according to some (simple) rule...



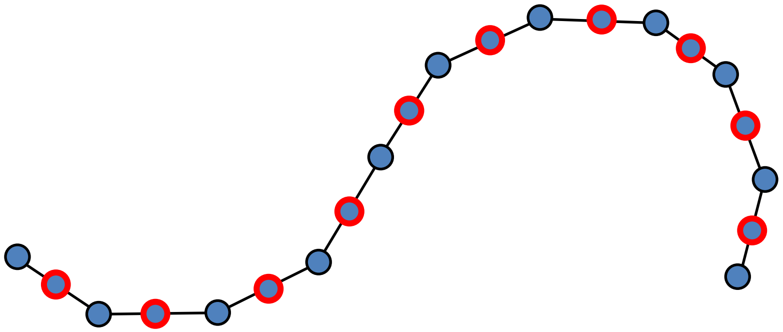
Approximating

Start over ...



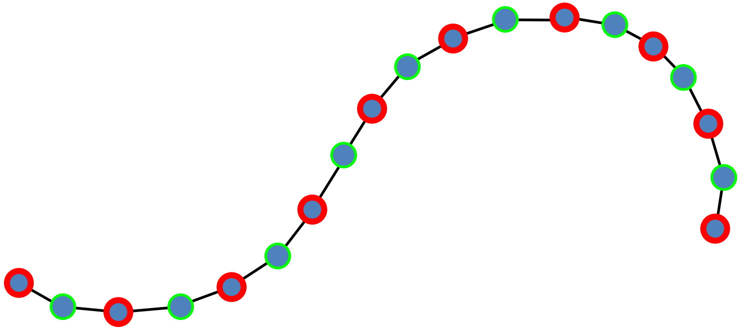
Approximating

...splitting...



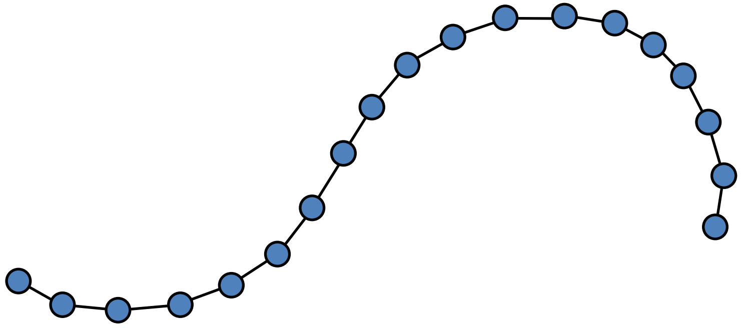
Approximating

...averaging...



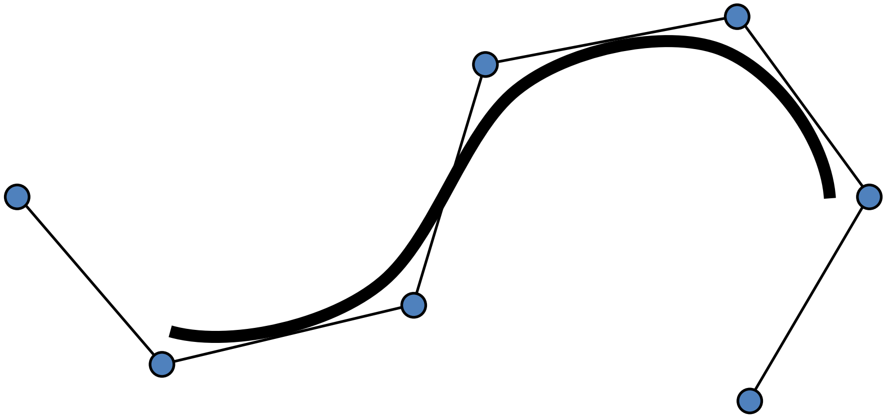
Approximating

...and so on...



Approximating

If the rule is designed carefully...



... the control polygons will converge to a smooth limit curve!

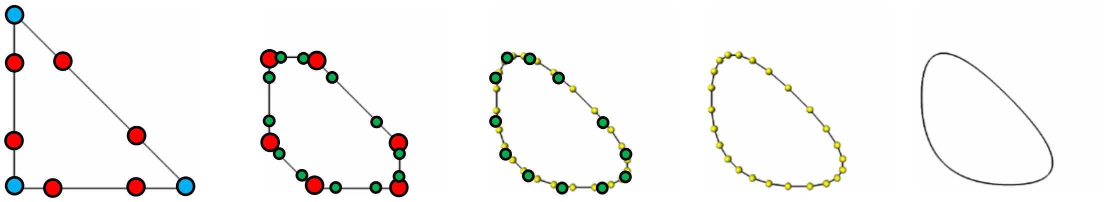
Equivalent to …

- Insert *single* new point at mid-edge
- *Filter* entire set of points.

Catmull-Clark rule: Filter with (1/8, 6/8, 1/8)

Corner Cutting

- Subdivision rule:
  - Insert *two* new vertices at ¼ and ¾ of each edge
  - *Remove* the old vertices
  - Connect the new vertices



B-Spline Curves

- Piecewise polynomial of degree n

B-spline curve

control points

$$\mathbf{s}(u) = \sum_{i=0}^{k} \mathbf{d}_i N_i^n(u)$$

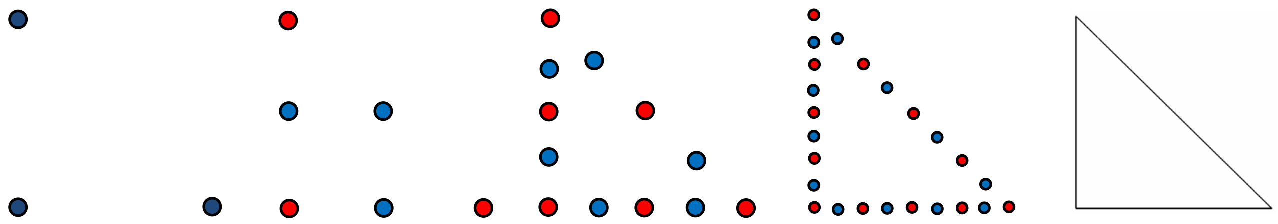
parameter value

basis functions

B-Spline Curves

- Distinguish between odd and even points

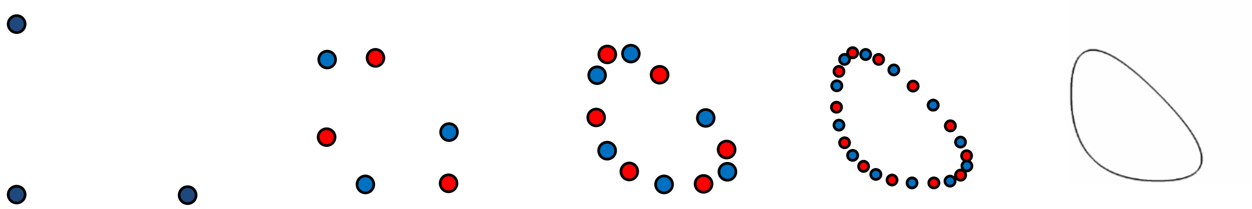
- **Linear B-spline**
  - Odd coefficients (1/2, 1/2)
  - Even coefficient (1)



B-Spline Curves

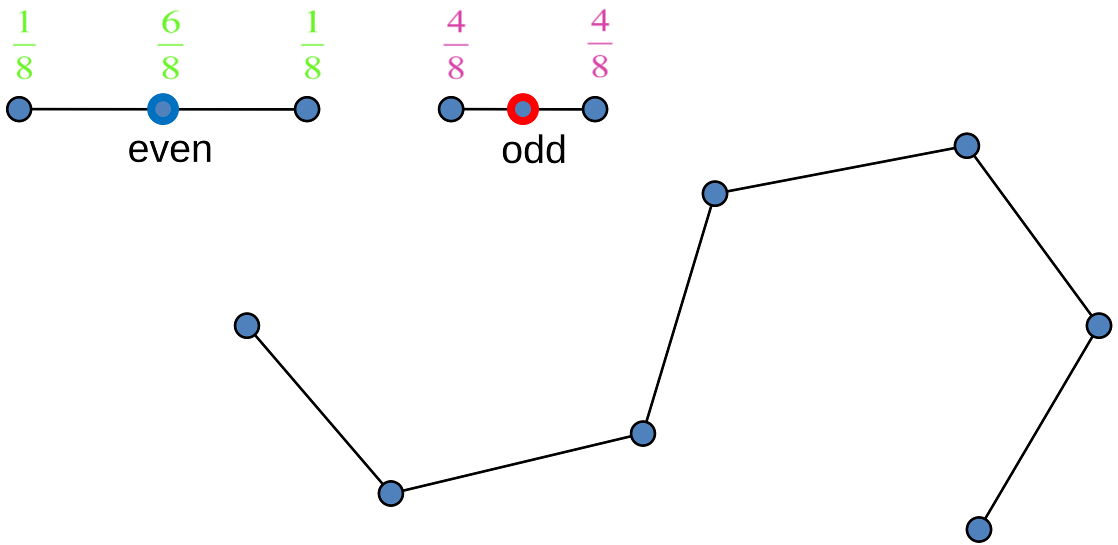
- **Quadratic B-Spline (Chaikin)**
  - Odd coefficients (¼, ¾)
  - Even coefficients (¾ , ¼)

demo

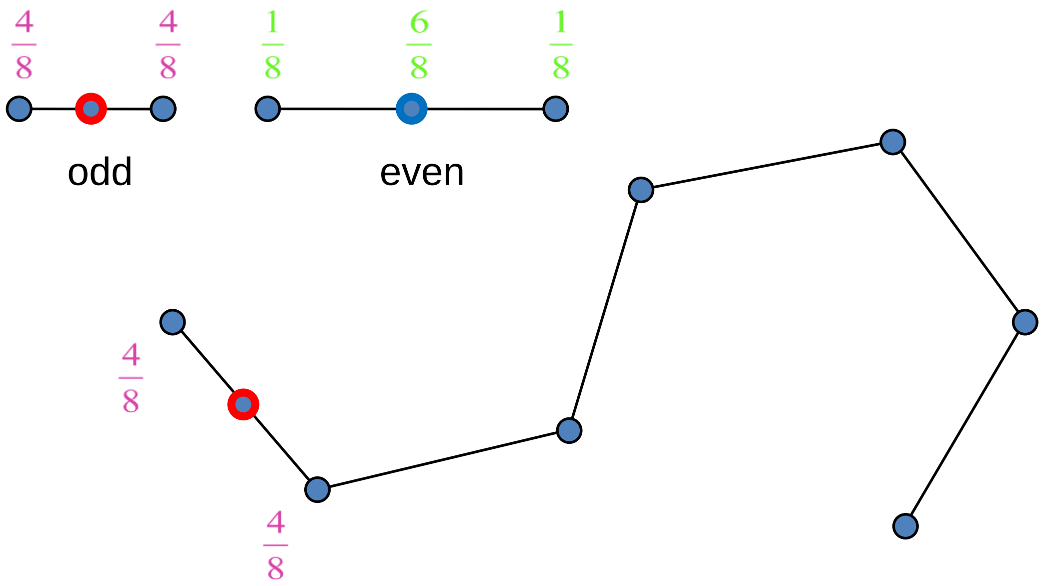


- **Cubic B-Spline (Catmull-Clark)**
  - Odd coefficients (4/8, 4/8)
  - Even coefficients (1/8, 6/8, 1/8)

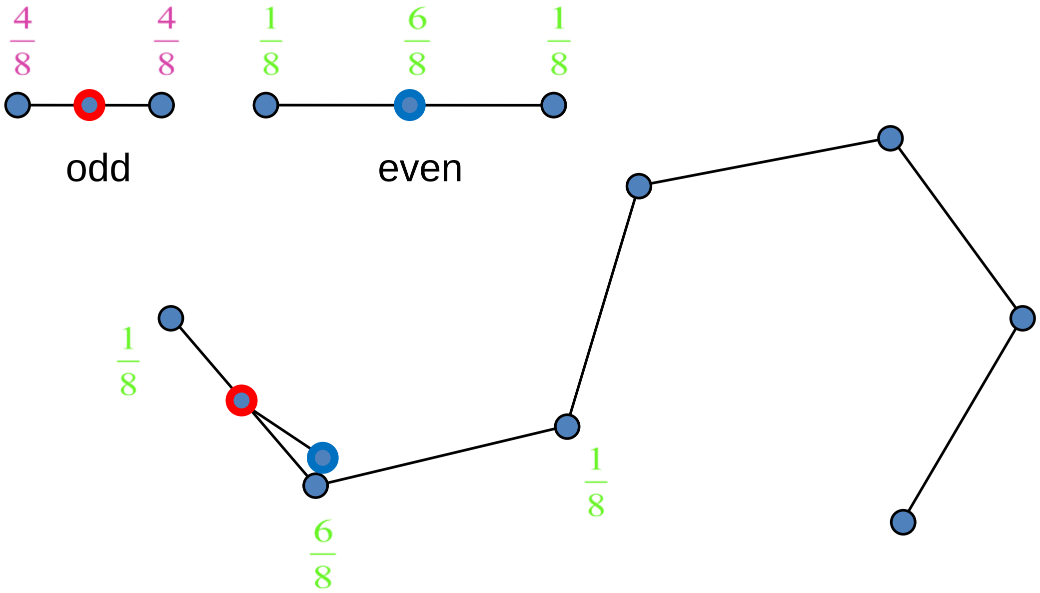
Cubic B-Spline



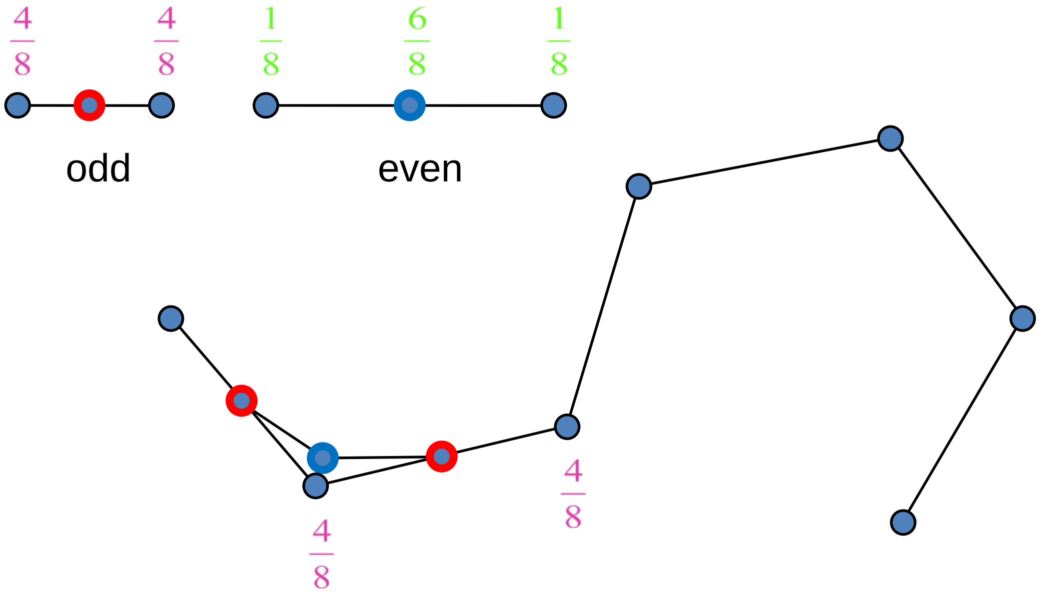
Cubic B-Spline



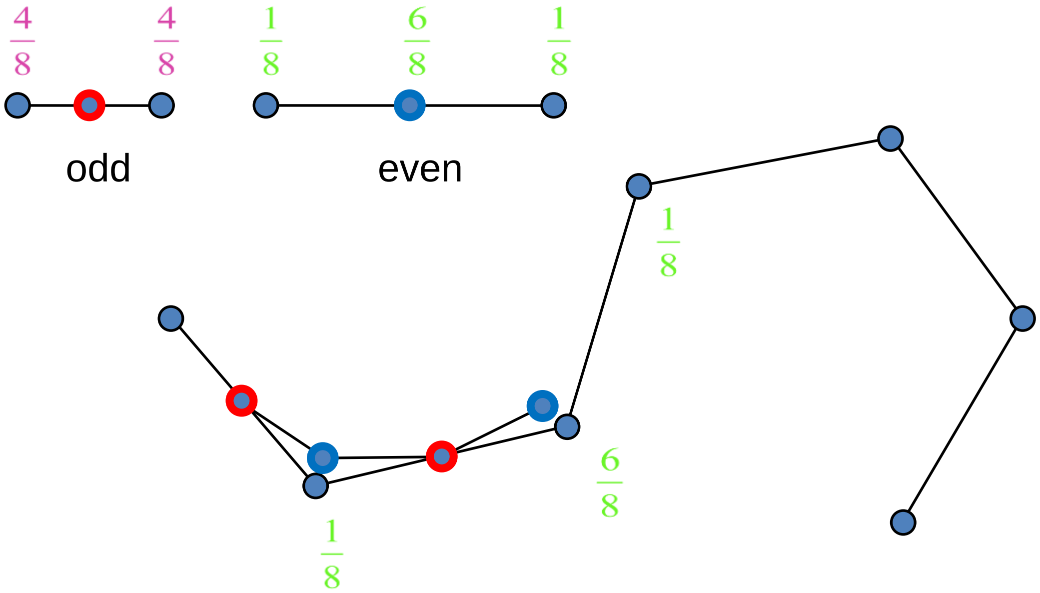
Cubic B-Spline



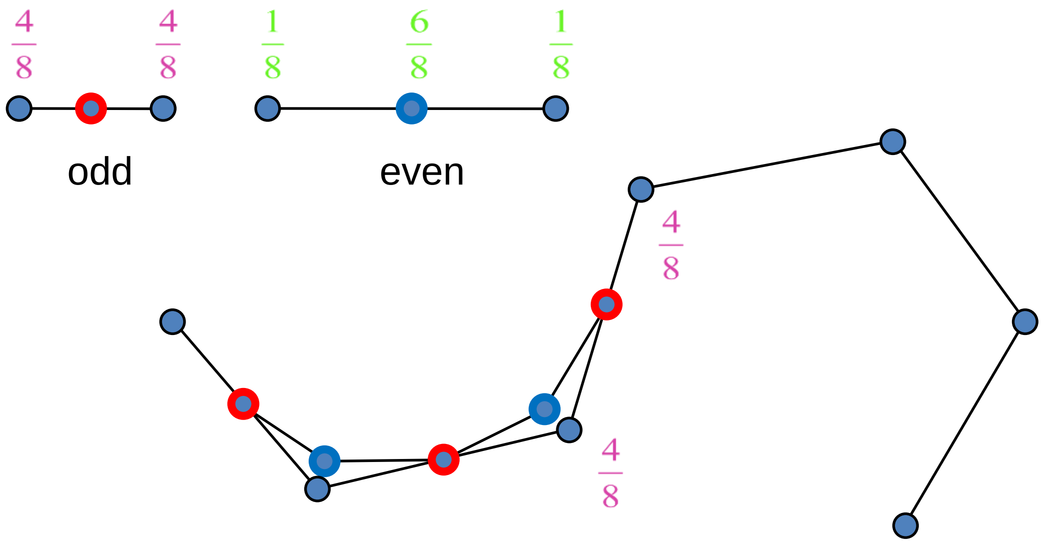
Cubic B-Spline



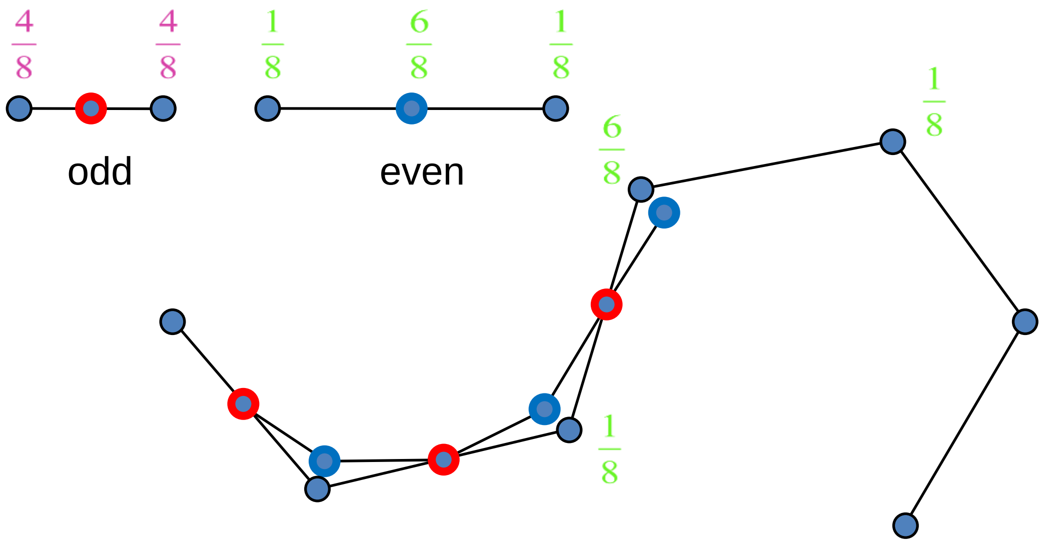
Cubic B-Spline



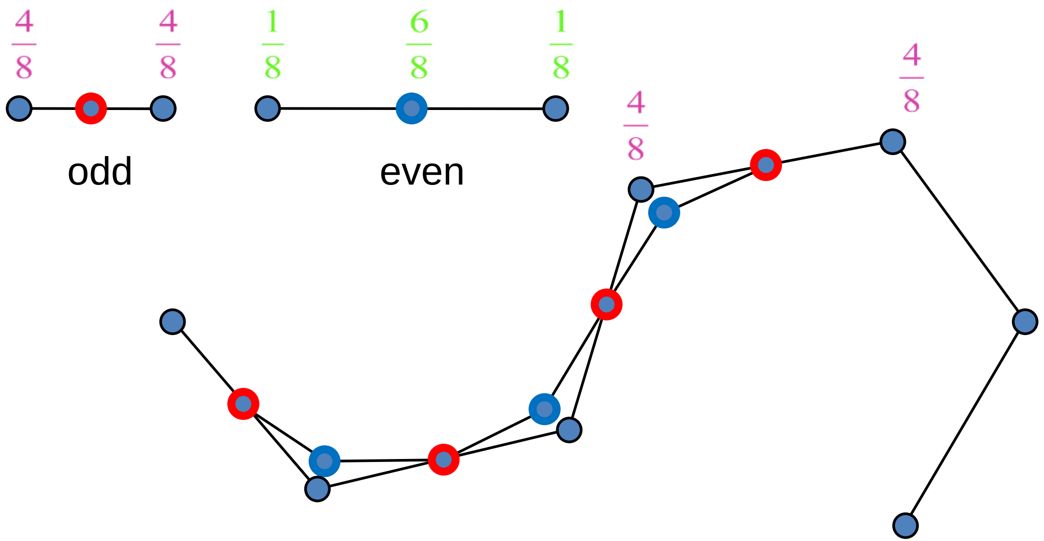
Cubic B-Spline



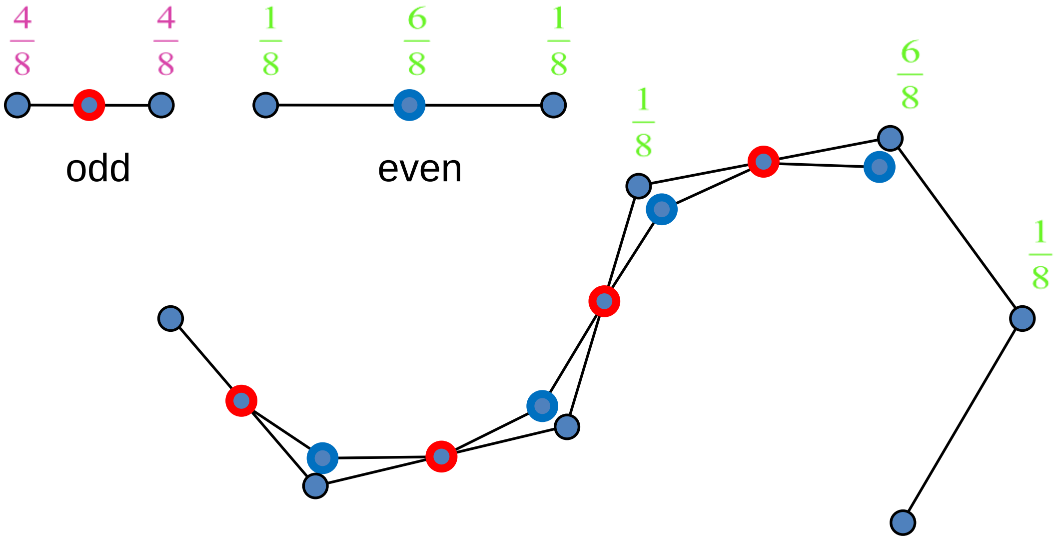
Cubic B-Spline



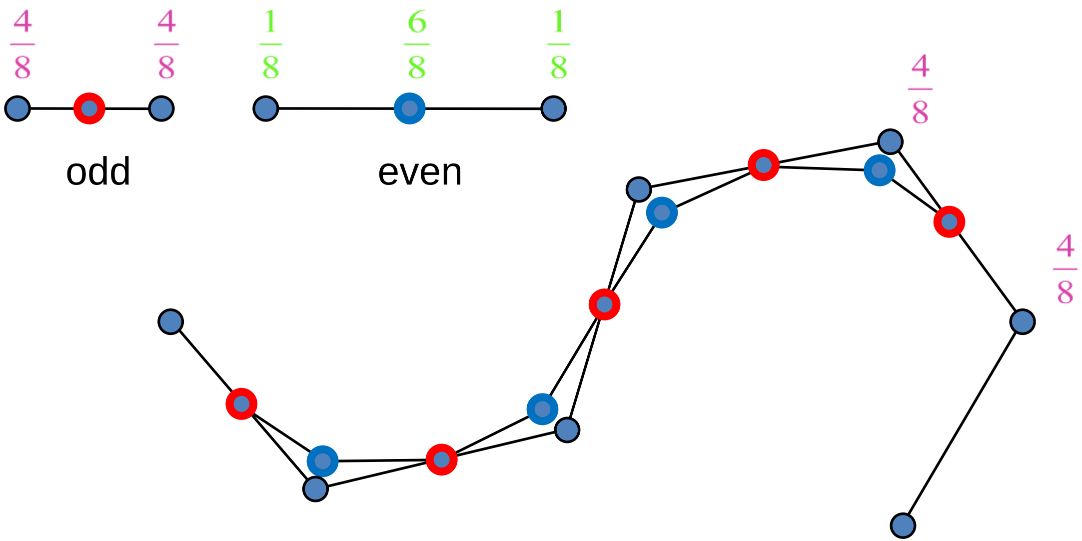
Cubic B-Spline



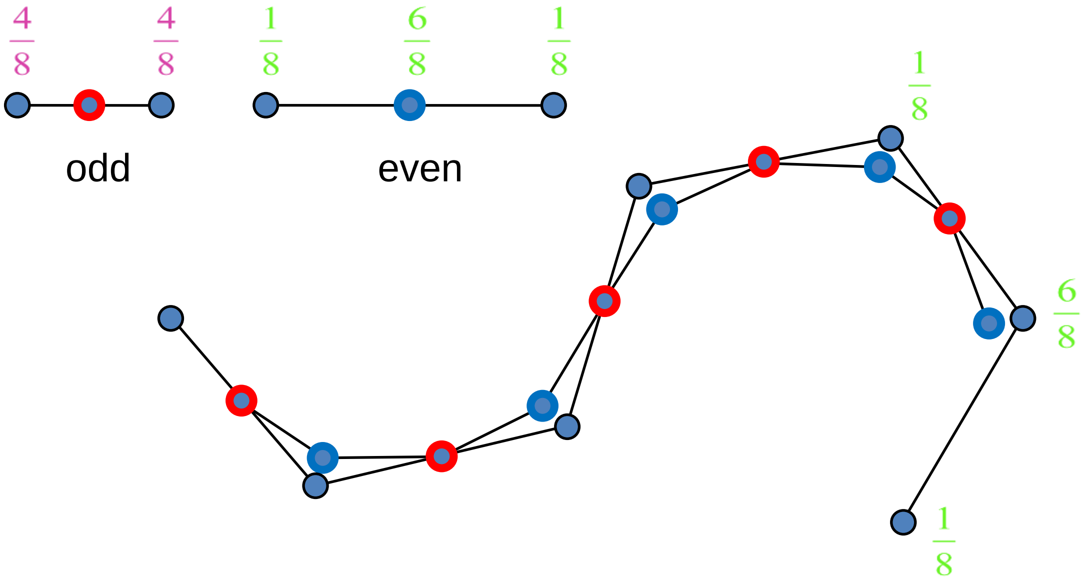
Cubic B-Spline



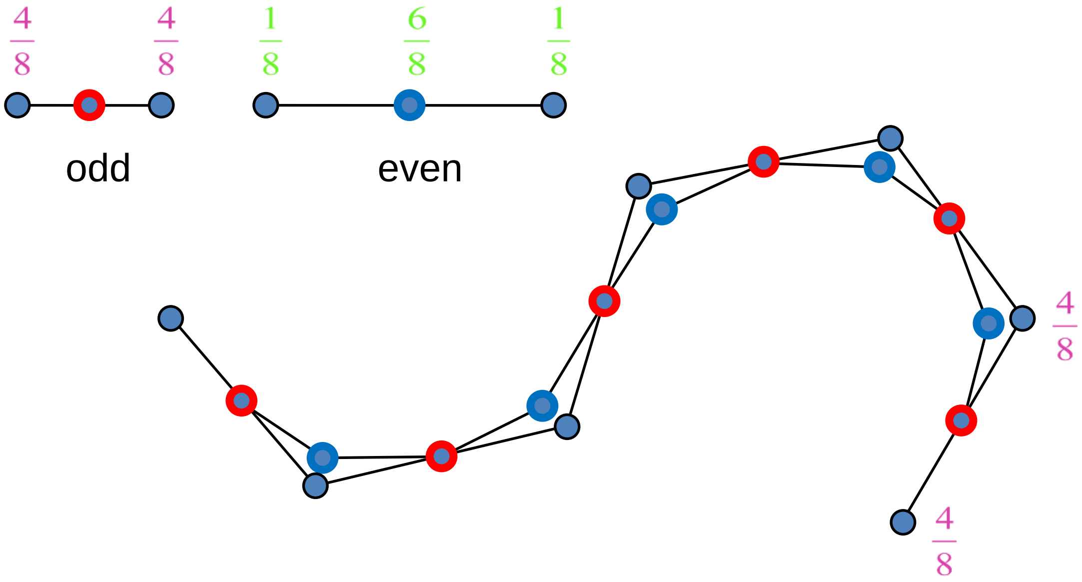
Cubic B-Spline



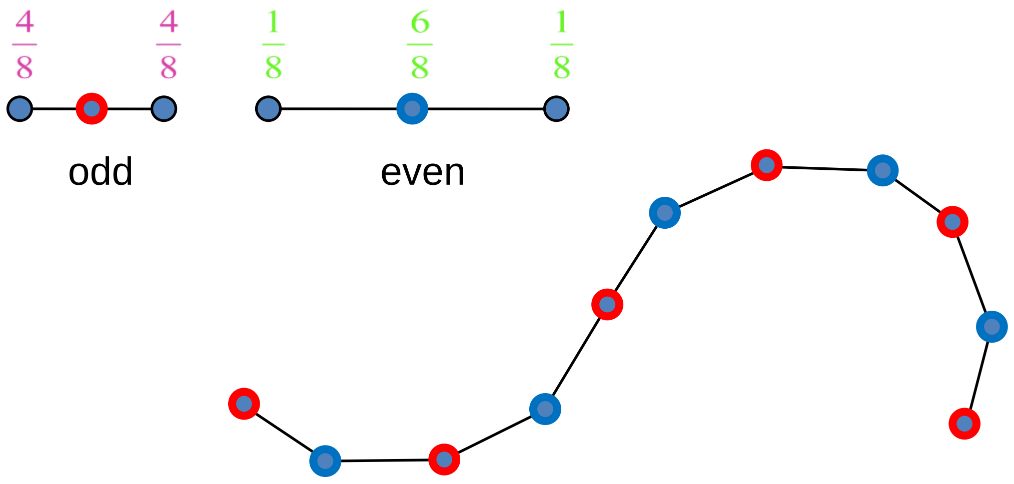
Cubic B-Spline



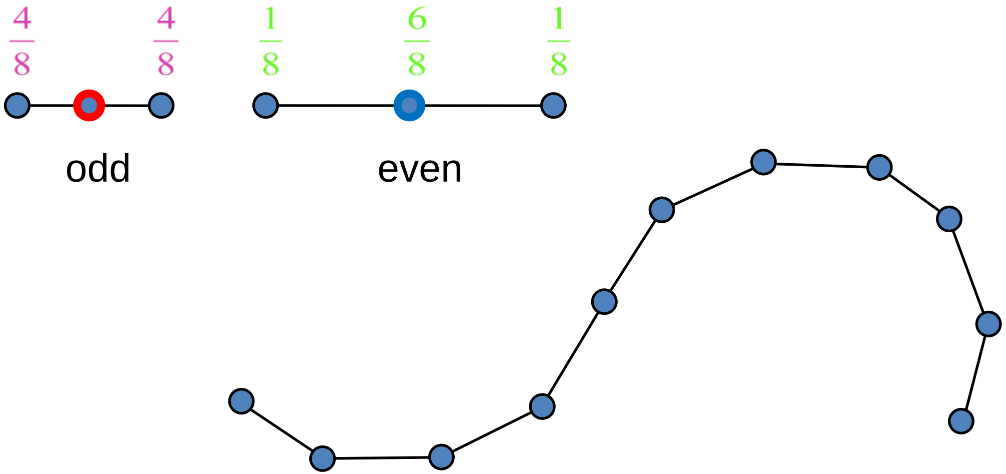
Cubic B-Spline



Cubic B-Spline



Cubic B-Spline



B-Spline Curves

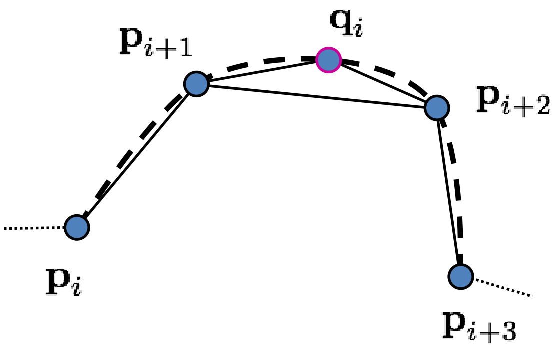
- Subdivision rules for control polygon

$$\mathbf{d}^0 \;\rightarrow\; \mathbf{d}^1 = S\mathbf{d}^0 \;\rightarrow\; \ldots \;\rightarrow\; \mathbf{d}^j = S\mathbf{d}^{j-1} = S^j\mathbf{d}^0$$

- Mask of size *n* yields C^{n-1} curve

Interpolating (4-point Scheme)

- Keep old vertices
- Generate new vertices by fitting cubic curve to old vertices
- C^1 continuous limit curve

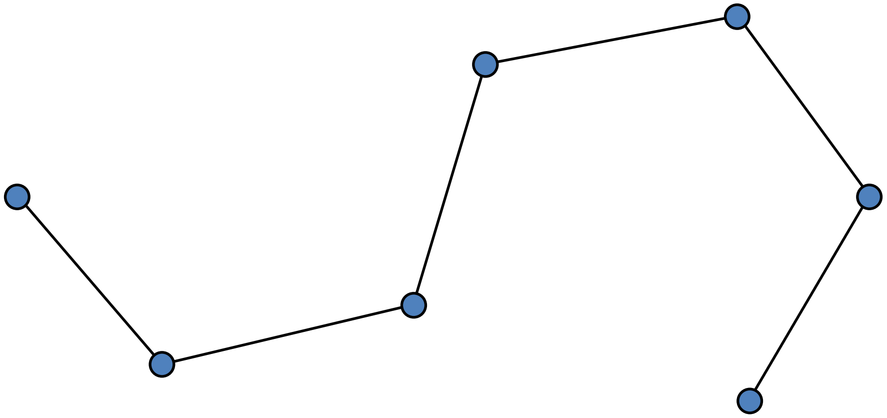


$$f(x) = ax^3 + bx^2 + cx + d$$

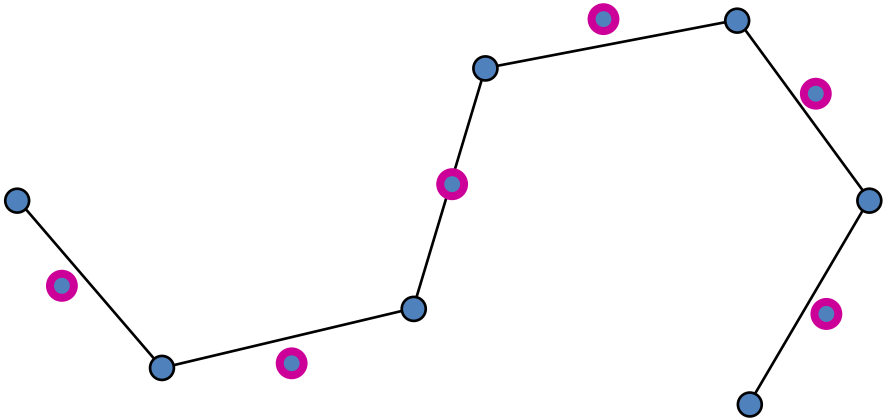
$$f(j) = \mathbf{p}_{i+j}\ , \quad j = 0, \ldots, 3$$

$$\begin{aligned} \mathbf{q}_i &= f(3/2) \\ &= \frac{1}{16}\left(-\mathbf{p}_i + 9\mathbf{p}_{i+1} + 9\mathbf{p}_{i+2} - \mathbf{p}_{i+3}\right) \end{aligned}$$

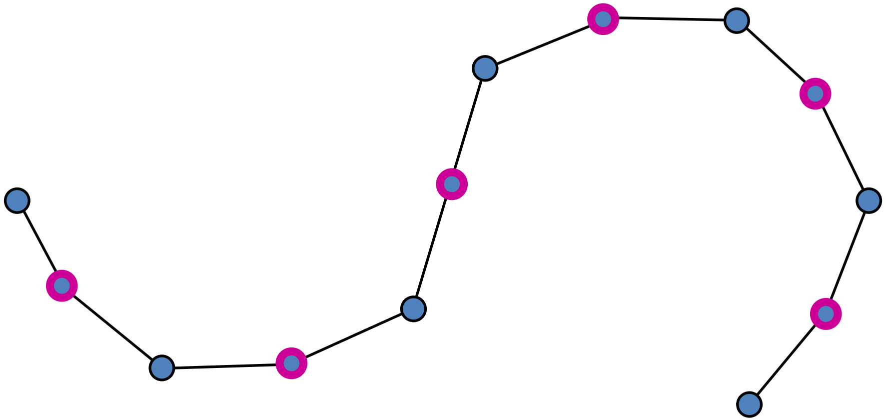
Interpolating



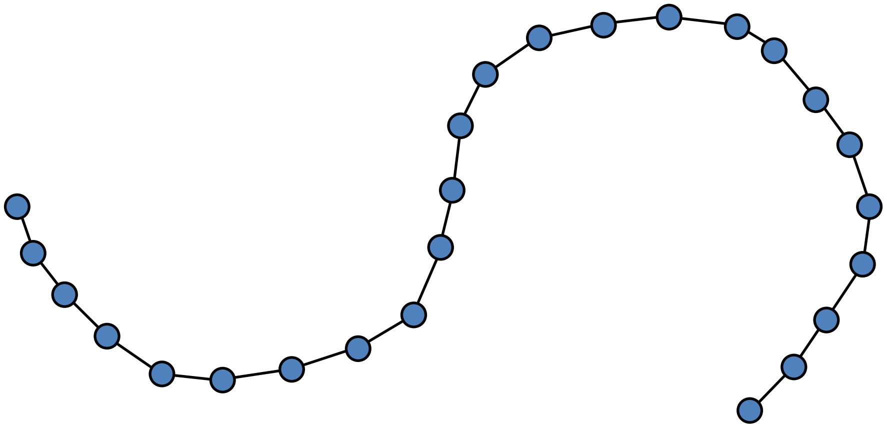
Interpolating



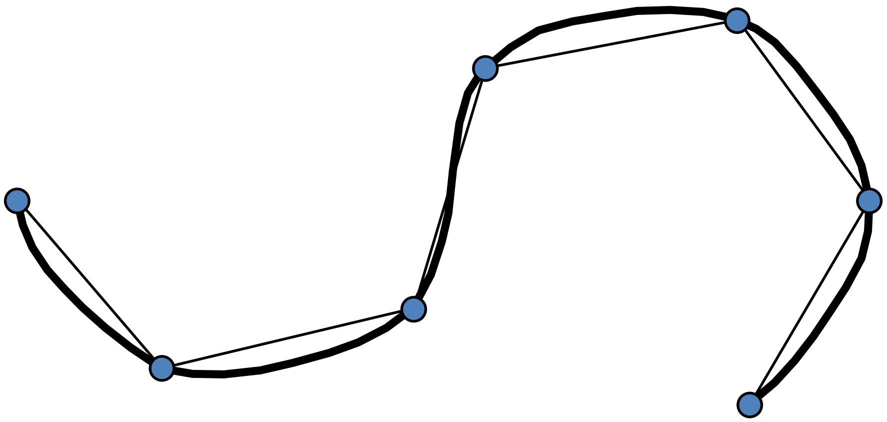
Interpolating



Interpolating



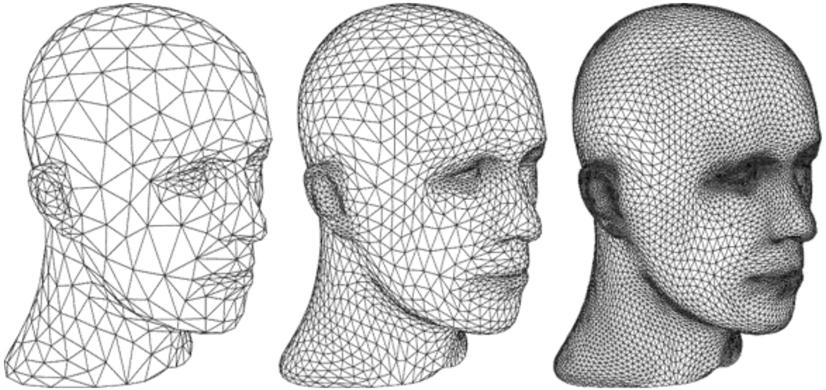
Interpolating



demo

Subdivision Surfaces

- No regular structure as for curves
  - Arbitrary number of edge-neighbors
  - Different subdivision rules for each valence



Classic Subdivision Operators

- Classification of subdivision schemes

Primal	Faces are split into sub-faces
Dual	Vertices are split into multiple vertices

Approximating	Control points are not interpolated
Interpolating	Control points are interpolated

Classic Subdivision Operators

- Classification of subdivision schemes

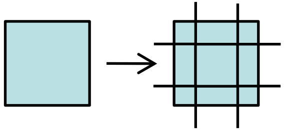
	Primal		Dual
	Triangles	Rectangles	
Approximating	Loop	Catmull-Clark	Doo-Sabin Midedge
Interpolating	Butterfly	Kobbelt	

Classic Subdivision Operators

- Classification of subdivision schemes

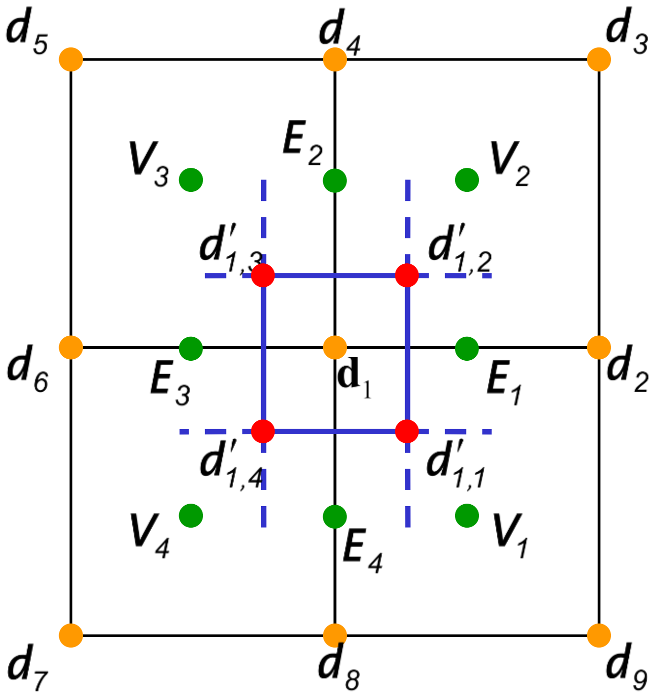
	Primal		Dual
	Triangles	Rectangles	
Approximating	Loop	Catmull-Clark	Doo-Sabin Midedge
Interpolating	Butterfly	Kobbelt	

Doo-Sabin Subdivision



- Generalization of *bi-quadratic B-Splines*
- Dual, approximating subdivision scheme
- Applied to *polygonal* meshes
- Generates G^1 *continuous* limit surfaces:
  - C^0 for the set of finite extraordinary points
    - Center of irregular polygons after 1 subdivision step
  - C^1 continuous everywhere else

Doo-Sabin Subdivision

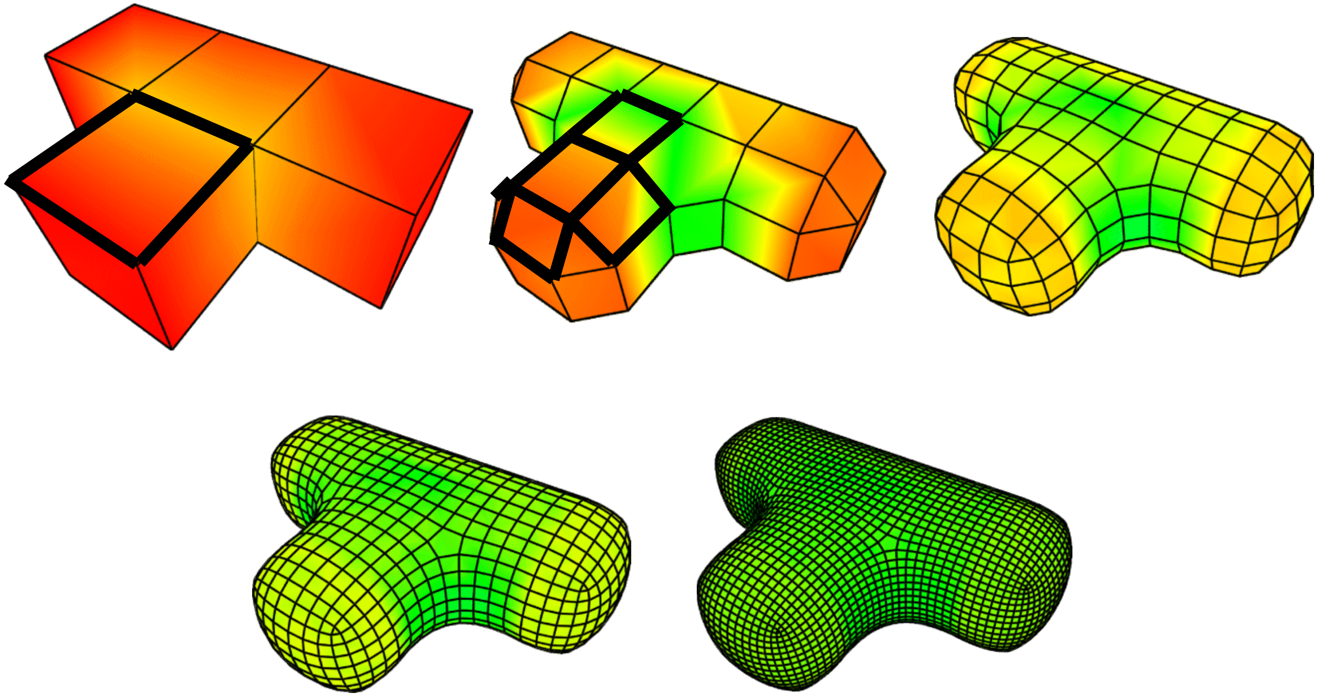


$$V_2 = \frac{1}{n} \times \sum_{j=1}^{n} d_j$$

$$E_i = \frac{1}{2}\left(d_1 + d_{2i}\right)$$

$$d'_{1,j} = \frac{1}{4}\left(d_1 + E_j + E_{j-1} + V_j\right)$$

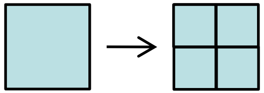
Doo-Sabin Subdivision



Classic Subdivision Operators

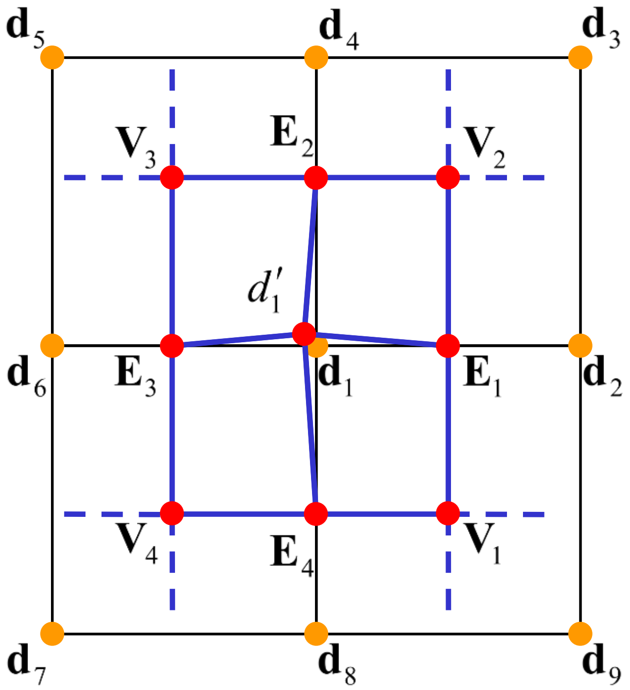
- Classification of subdivision schemes

	Primal		Dual
	Triangles	Rectangles	
Approximating	Loop	Catmull-Clark	Doo-Sabin Midedge
Interpolating	Butterfly	Kobbelt	

Catmull-Clark Subdivision 

- Generalization of *bi-cubic B-Splines*
- Primal, approximation subdivision scheme
- Applied to *polygonal* meshes
- Generates G^2 *continuous* limit surfaces:
  - C^1 for the set of finite extraordinary points
    - Vertices with valence $\neq 4$
  - C^2 continuous everywhere else

Catmull-Clark Subdivision



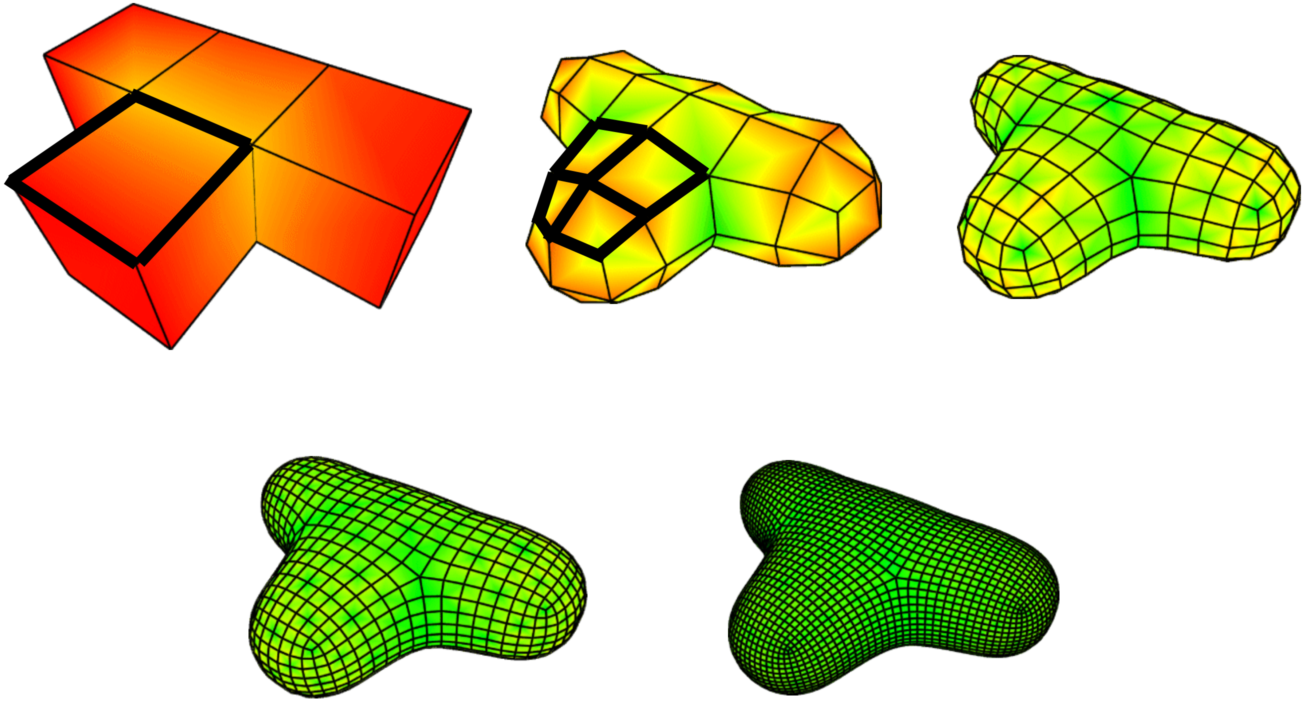
$$\mathbf{V}_2 = \frac{1}{n} \times \sum_{j=1}^{n} \mathbf{d}_j$$

$$\mathbf{E}_i = \frac{1}{4}\left(\mathbf{d}_1 + \mathbf{d}_{2i} + \mathbf{V}_i + \mathbf{V}_{i+1}\right)$$

$$\mathbf{d}'_1 = \frac{(n-3)}{n}\mathbf{d}_1 + \frac{2}{n}\mathbf{R} + \frac{1}{n}\mathbf{S}$$

$$\mathbf{R} = \frac{1}{m}\sum_{i=1}^{m} \mathbf{E}_i \quad \mathbf{S} = \frac{1}{m}\sum_{i=1}^{m} \mathbf{V}_i$$

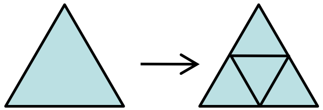
Catmull-Clark Subdivision



Classic Subdivision Operators

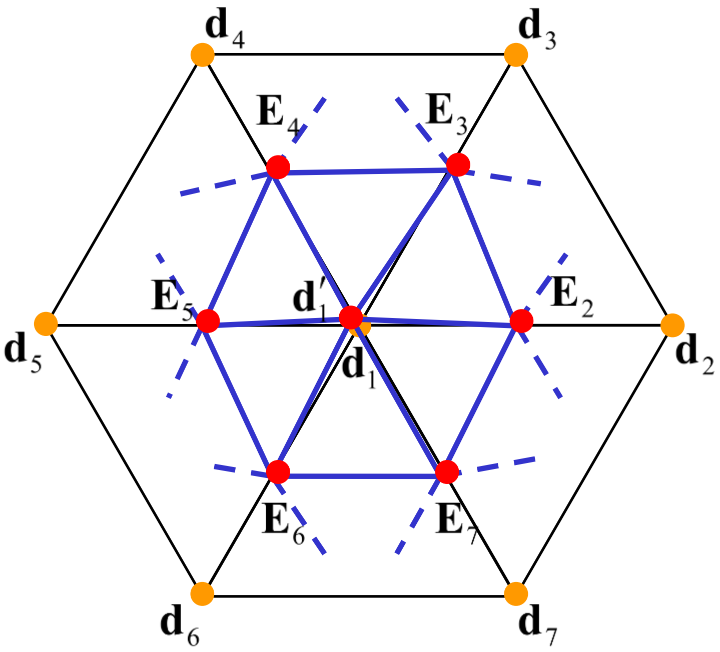
- Classification of subdivision schemes

	Primal		Dual
	Triangles	Rectangles	
Approximating	Loop	Catmull-Clark	Doo-Sabin Midedge
Interpolating	Butterfly	Kobbelt	

Loop Subdivision 

- Generalization of *box splines*
- Primal, approximating subdivision scheme
- Applied to *triangle* meshes
- Generates *G^2 continuous* limit surfaces:
  - C^1 for the set of finite extraordinary points
    - Vertices with valence $\neq 6$
  - C^2 continuous everywhere else

Loop Subdivision

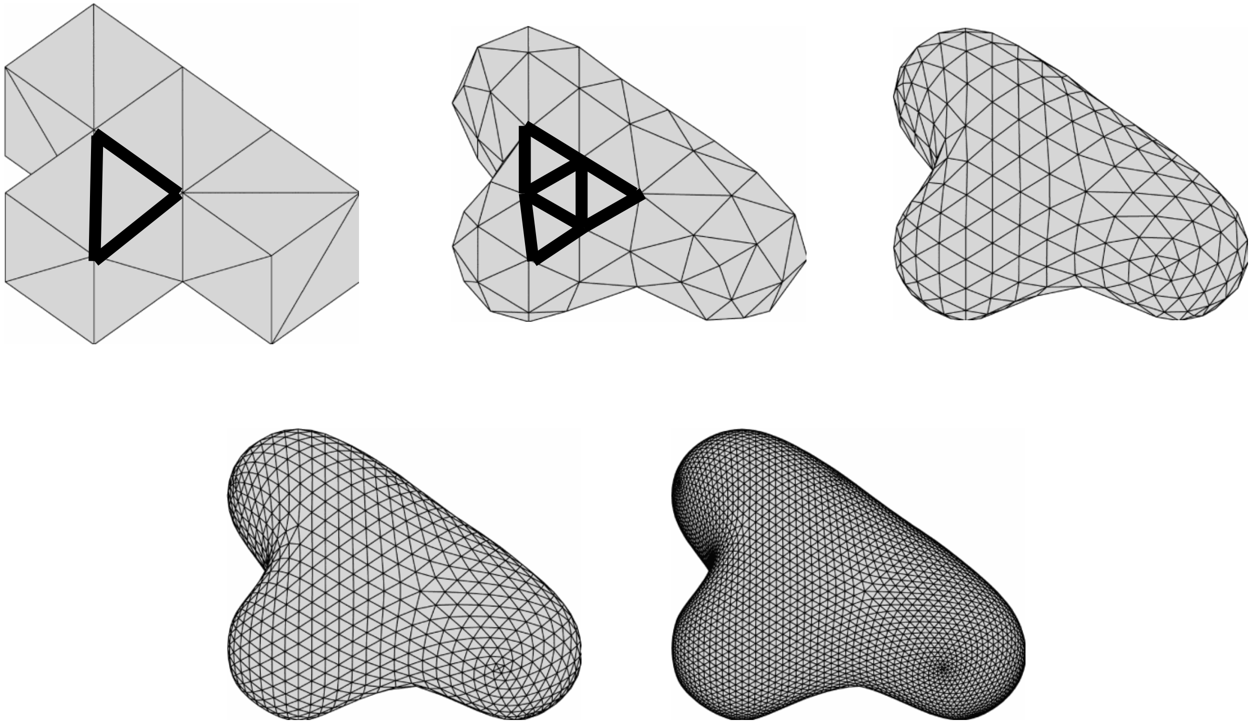


$$E_i = \frac{3}{8}(d_1 + d_i) + \frac{1}{8}(d_{i-1} + d_{i+1})$$

$$\mathbf{d}_1' = \alpha_n \mathbf{d}_1 + \frac{(1-\alpha_n)}{n} \sum_{j=2}^{n+1} \mathbf{d}_j$$

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

Loop Subdivision



Classic Subdivision Operators

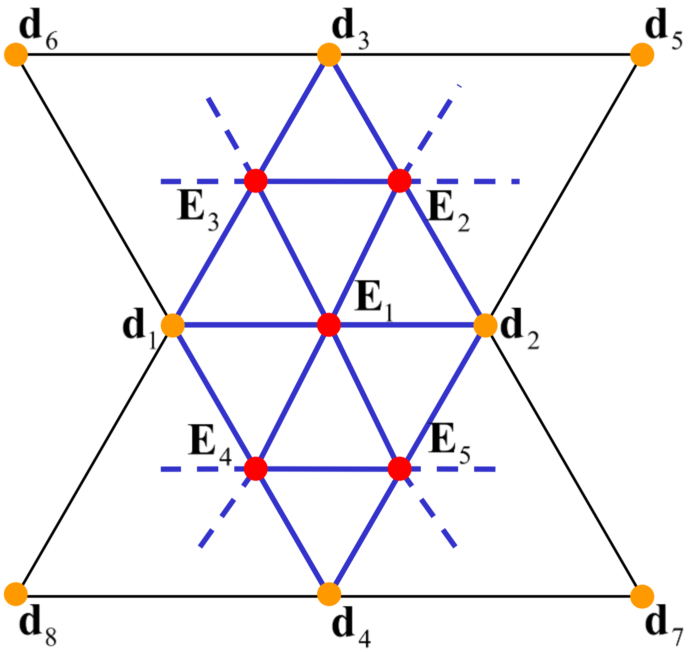
- Classification of subdivision schemes

	Primal		Dual
	Triangles	Rectangles	
Approximating	Loop	Catmull-Clark	Doo-Sabin Midedge
Interpolating	Butterfly	Kobbelt	

Butterfly Subdivision

- Primal, interpolating scheme
- Applied to *triangle* meshes
- Generates G^1 *continuous* limit surfaces:
  - C^0 for the set of finite extraordinary points
    - Vertices of valence = 3 or > 7
  - C^1 continuous everywhere else

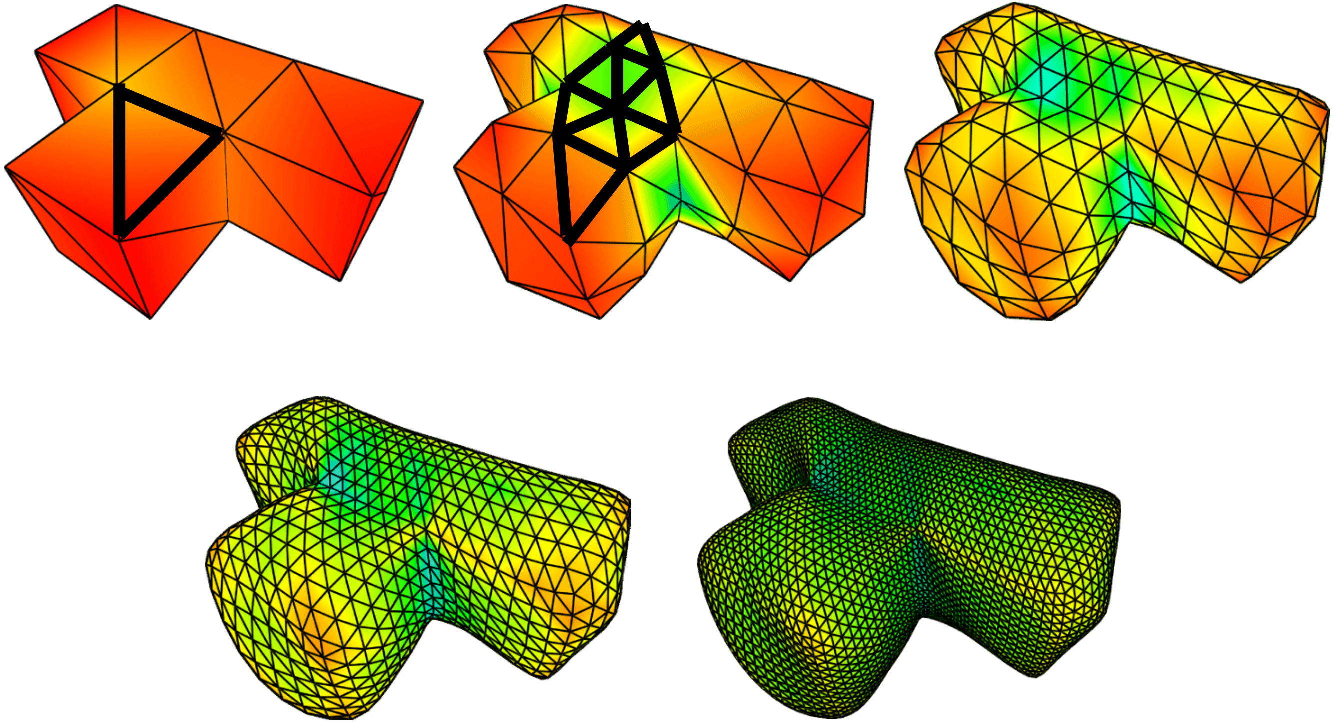
Butterfly Subdivision



$$\mathbf{E}_1 = \frac{1}{2}(\mathbf{d}_1 + \mathbf{d}_2) + \omega(\mathbf{d}_3 + \mathbf{d}_4) - \frac{\omega}{2}(\mathbf{d}_5 + \mathbf{d}_6 + \mathbf{d}_7 + \mathbf{d}_8)$$

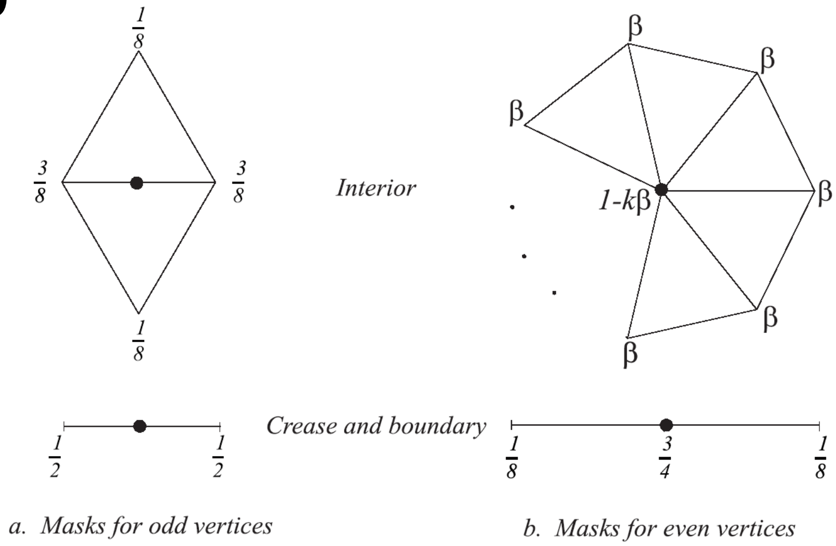
$$\mathbf{d}'_i = \mathbf{d}_i$$

Butterfly Subdivision

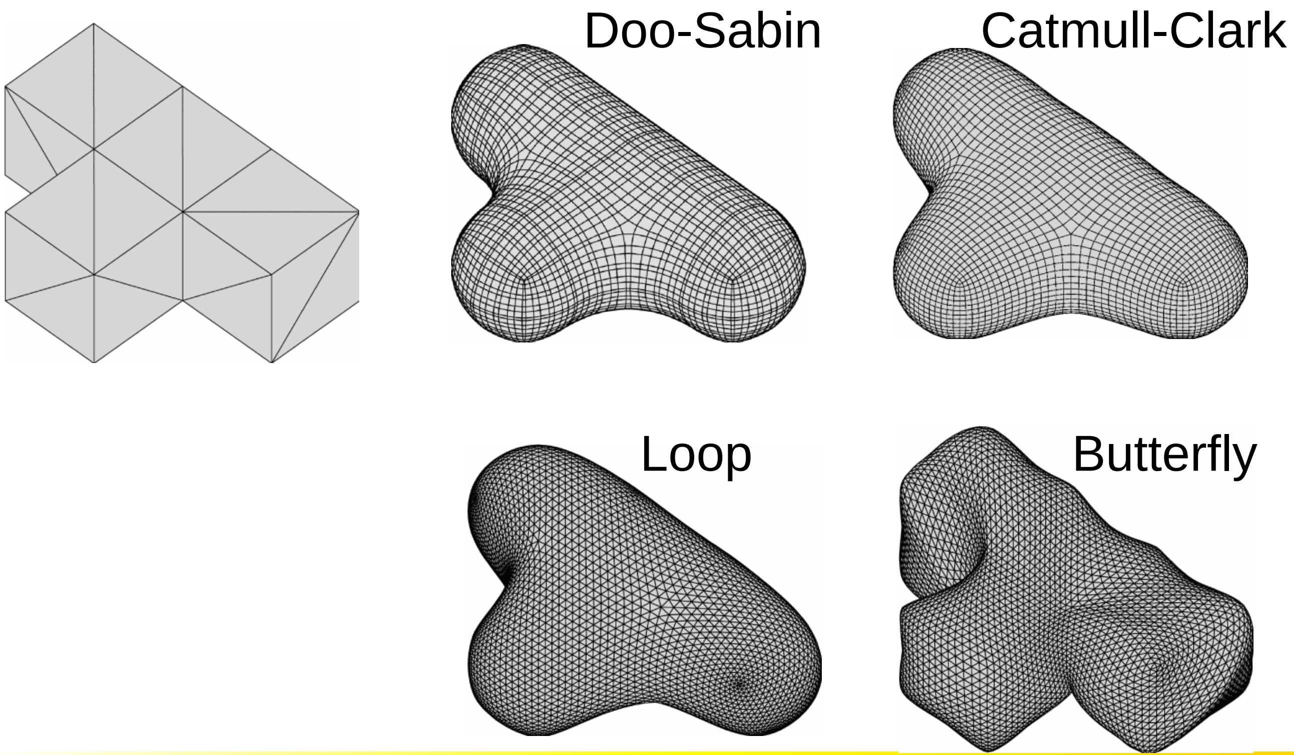


Remark

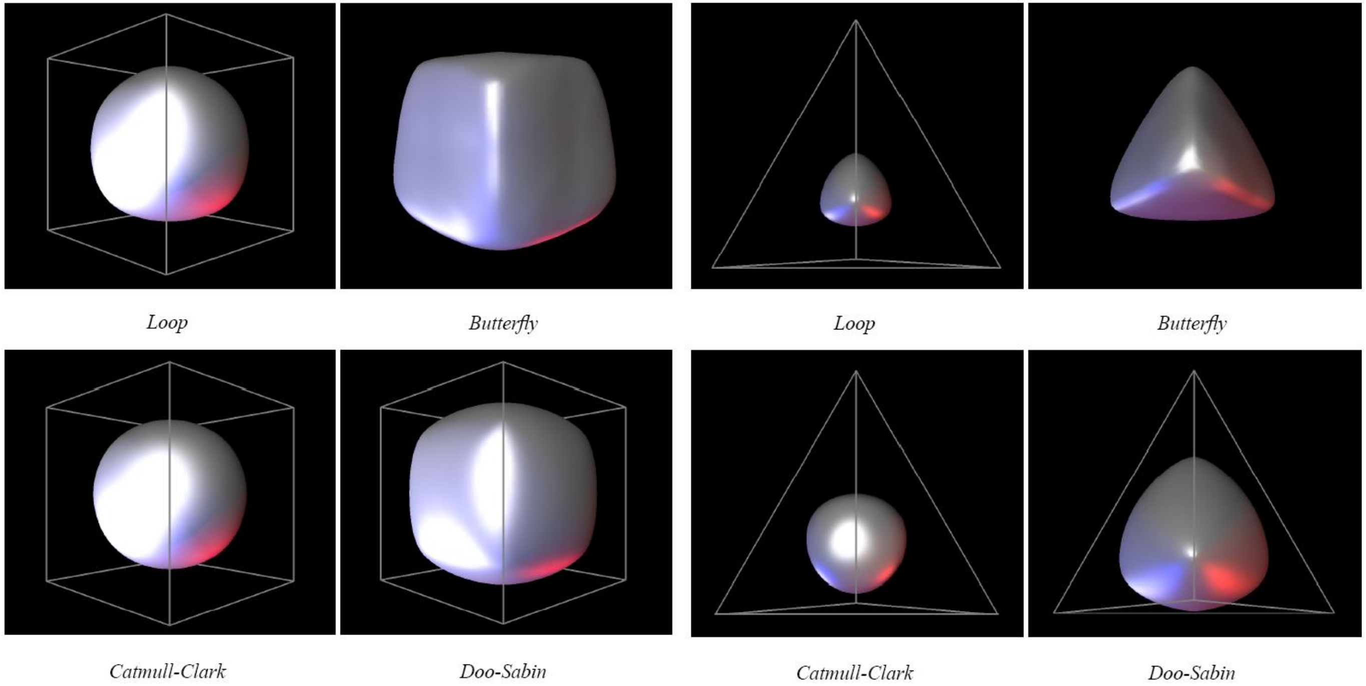
- Different masks apply on the boundary
- Example: Loop



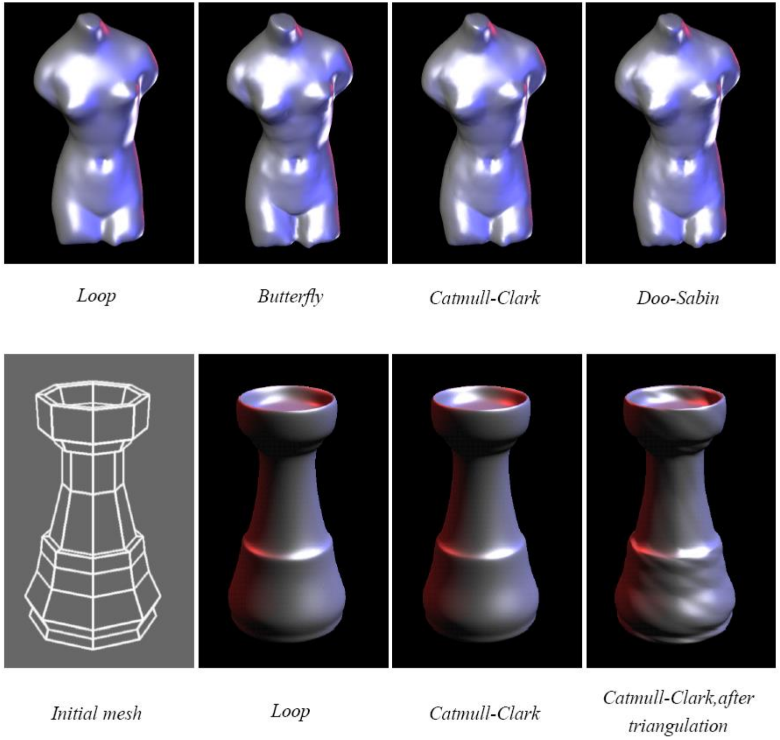
Comparison



Comparison


Loop
Butterfly
Loop
Butterfly
Catmull-Clark
Doo-Sabin
Catmull-Clark
Doo-Sabin

Comparison


Loop
Butterfly
Catmull-Clark
Doo-Sabin
Initial mesh
Loop
Catmull-Clark
Catmull-Clark,after triangulation

Comparison

- Continuity of a scheme determines the quality
- Approximating schemes give best results
- Approximating schemes shrink meshes

- Best schemes: Catmull-Clark & Loop
  - Loop is applied on triangle meshes
  - Catmull-Clark preserves symmetry better

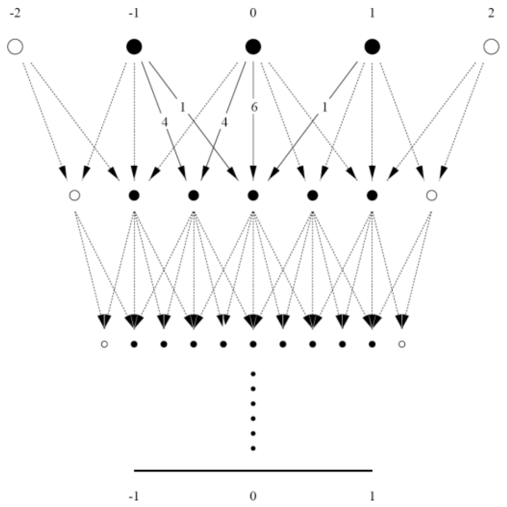
Analysis of Subdivision

- Invariant neighborhoods
  - How many control-points affect a small neighborhood around a point ?
- Subdivision scheme can be analyzed by looking at a *local* subdivision matrix

Local Subdivision Matrix

- Example: Cubic B-Splines

$$\begin{pmatrix} \mathbf{p}_{-2}^{j+1} \\ \mathbf{p}_{-1}^{j+1} \\ \mathbf{p}_{0}^{j+1} \\ \mathbf{p}_{1}^{j+1} \\ \mathbf{p}_{2}^{j+1} \end{pmatrix} = \frac{1}{8} \begin{pmatrix} 1 & 6 & 1 & 0 & 0 \\ 0 & 4 & 4 & 0 & 0 \\ 0 & 1 & 6 & 1 & 0 \\ 0 & 0 & 4 & 4 & 0 \\ 0 & 0 & 1 & 6 & 1 \end{pmatrix} \begin{pmatrix} \mathbf{p}_{-2}^{j} \\ \mathbf{p}_{-1}^{j} \\ \mathbf{p}_{0}^{j} \\ \mathbf{p}_{1}^{j} \\ \mathbf{p}_{2}^{j} \end{pmatrix}$$



- Invariant neighborhood size: 5

Analysis of Subdivision

- Analysis via eigen-decomposition of matrix S
  - Compute the eigenvalues

$$\{\lambda_0, \lambda_1, \ldots, \lambda_{n-1}\}$$

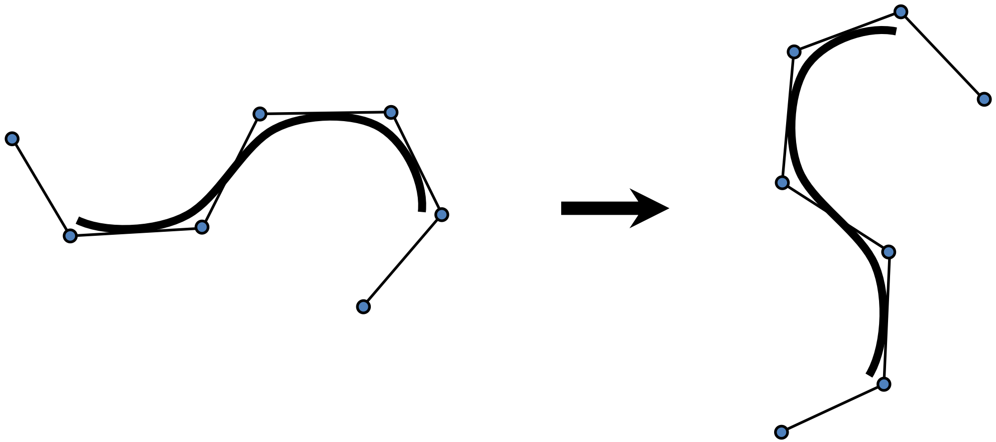
  - and eigenvectors

$$X = \{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{n-1}\}$$

  - Let $\lambda_0 \geq \lambda_1 \geq \cdots \geq \lambda_{n-1}$ be real and X a complete set of eigenvectors

Analysis of Subdivision

- Invariance under affine transformations
  - transform(subdivide(P)) = subdivide(transform(P))



Analysis of Subdivision

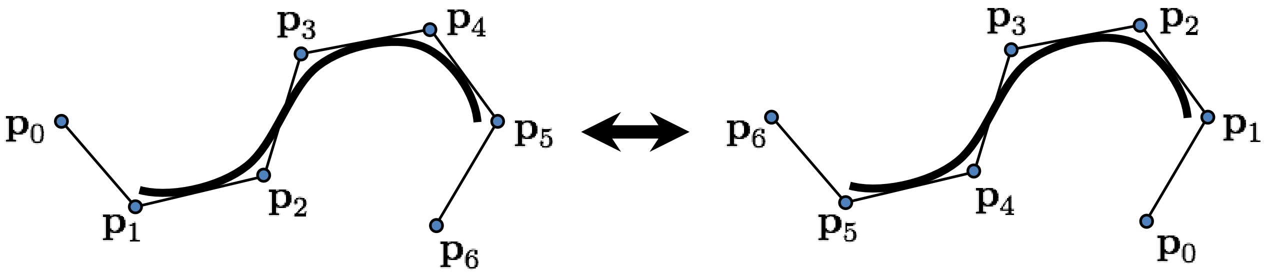
- Invariance under affine transformations
  - transform(subdivide(P)) = subdivide(transform(P))

- Rules have to be affine combinations
  - Even and odd weights each sum to 1

$$\sum_j S_{2i,j} = \sum_j S_{2i+1,j} = 1$$

Analysis of Subdivision

- Invariance under reversion of point ordering
- Subdivision rules (matrix rows) have to be symmetric



Analysis of Subdivision

Conclusion: 1 has to be eigenvector of S with eigenvalue $\lambda_0=1$

Limit Behavior - Position

- Any vector is linear combination of eigenvectors:

$$\mathbf{p} = \sum_{i=0}^{n-1} a_i \mathbf{x}_i \qquad a_i = \tilde{\mathbf{x}}_i^T \mathbf{p}$$

rows of X^{-1}

- Apply subdivision matrix:

$$S\mathbf{p}^0 = S\sum_{i=0}^{n-1} a_i \mathbf{x}_i = \sum_{i=0}^{n-1} a_i S \mathbf{x}_i = \sum_{i=0}^{n-1} a_i \lambda_i \mathbf{x}_i$$

Limit Behavior - Position

- For convergence we need $1 = \lambda_0 > \lambda_1 \geq \cdots \geq \lambda_{n-1}$

- Limit vector:

$$\mathbf{p}^\infty = \lim_{j\to\infty} S^j \mathbf{p}^0 = \lim_{j\to\infty} \sum_{i=0}^{n-1} a_i \lambda_i^j \mathbf{x}_i = a_0 \cdot \mathbf{1}$$

$$\mathbf{p}_i^\infty = a_0 = \tilde{\mathbf{x}}_0^T \mathbf{p}^j$$ independent of *j* !

Limit Behavior - Tangent

- Set origin at a_0:

$$\mathbf{p}^j = \sum_{i=1}^{n-1} a_i \lambda_i^j \mathbf{x}_i$$

- Divide by λ_1^j

$$\frac{1}{\lambda_1^j}\mathbf{p}^j = a_1\mathbf{x}_1 + \sum_{i=2}^{n-1} a_i \left(\frac{\lambda_i}{\lambda_1}\right)^j \mathbf{x}_i$$

- Limit tangent given by:

$$\mathbf{t}_i^\infty = a_1 = \tilde{\mathbf{x}}_1^T \mathbf{p}^j$$

Limit Behavior - Tangent

- Curves:
  - All eigenvalues of S except $\lambda_0=1$ should be less than λ_1 to ensure existence of a tangent, i.e.

$$1 = \lambda_0 > \lambda_1 > \lambda_2 \geq \cdots \geq \lambda_{n-1}$$

- Surfaces:
  - Tangents determined by λ_1 and λ_2

$$1 = \lambda_0 > \lambda_1 = \lambda_2 > \lambda_3 \geq \cdots \geq \lambda_{n-1}$$

Example: Cubic Splines

- Subdivision matrix & rules

$$S = \frac{1}{8}\begin{pmatrix} 1 & 6 & 1 & 0 & 0 \\ 0 & 4 & 4 & 0 & 0 \\ 0 & 1 & 6 & 1 & 0 \\ 0 & 0 & 4 & 4 & 0 \\ 0 & 0 & 1 & 6 & 1 \end{pmatrix}$$

$$\begin{aligned} \mathbf{p}_{2i}^{j+1} &= \frac{1}{8}\mathbf{p}_{i-1}^{j} + \frac{6}{8}\mathbf{p}_{i}^{j} + \frac{1}{8}\mathbf{p}_{i+1}^{j} \\ \mathbf{p}_{2i+1}^{j+1} &= \frac{1}{2}\mathbf{p}_{i}^{j} + \frac{1}{2}\mathbf{p}_{i+1}^{j} \end{aligned}$$

- Eigenvalues

$$(\lambda_0, \lambda_1, \lambda_2, \lambda_3, \lambda_4,) = \left(1, \frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{8}\right)$$

Example: Cubic Splines

- Eigenvectors

$$X = \begin{pmatrix} 1 & -1 & 1 & 1 & 0 \\ 1 & -\frac{1}{2} & \frac{2}{11} & 0 & 0 \\ 1 & 0 & -\frac{1}{11} & 0 & 0 \\ 1 & \frac{1}{2} & \frac{2}{11} & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 \end{pmatrix} \qquad X^{-1} = \begin{pmatrix} 0 & \frac{1}{6} & \frac{4}{6} & \frac{1}{6} & 0 \\ 0 & -1 & 0 & 1 & 0 \\ & & \ldots & & \\ & & \ldots & & \\ & & \ldots & & \end{pmatrix}$$

- Limit position and tangent

$$\mathbf{p}_i^\infty = \tilde{\mathbf{x}}_0^T \mathbf{p}^j = \frac{1}{6}\left(\mathbf{p}_{i-1}^j + 4\mathbf{p}_i^j + \mathbf{p}_{i+1}^j\right)$$

$$\mathbf{t}_i^\infty = \tilde{\mathbf{x}}_1^T \mathbf{p}^j = \mathbf{p}_{i+1}^j - \mathbf{p}_i^j$$

Properties of Subdivision

- Flexible modeling
  - Handle surfaces of arbitrary topology
  - Provably smooth limit surfaces
  - Intuitive control point interaction
- Scalability
  - Level-of-detail rendering
  - Adaptive approximation
- Usability
  - Compact representation
  - Simple and efficient code