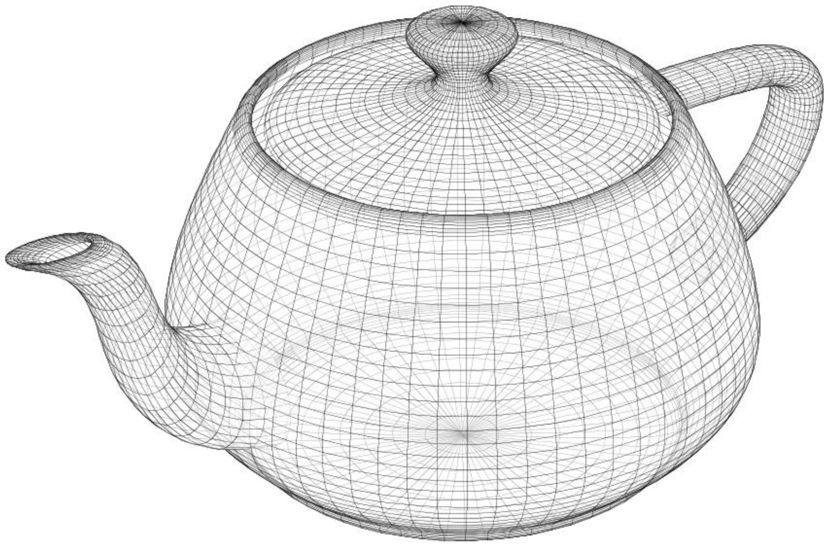
Subdivision Surfaces



Interactive Computer Graphics

Eric Shaffer

Based on Slides from Alex Benton, University of Cambridge

ILLINOIS

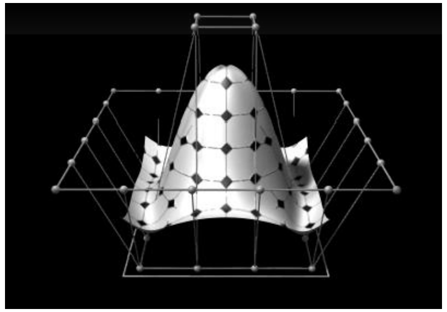
Geometric Modeling

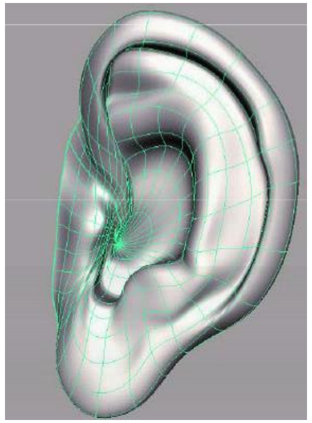
Sometimes need more than polygon meshes

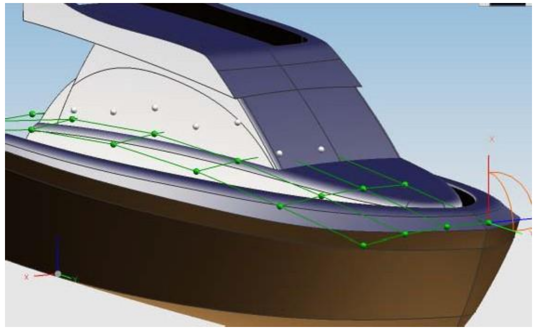
- Smooth surfaces

Traditional geometric modeling used NURBS

- Non uniform rational B-Spline

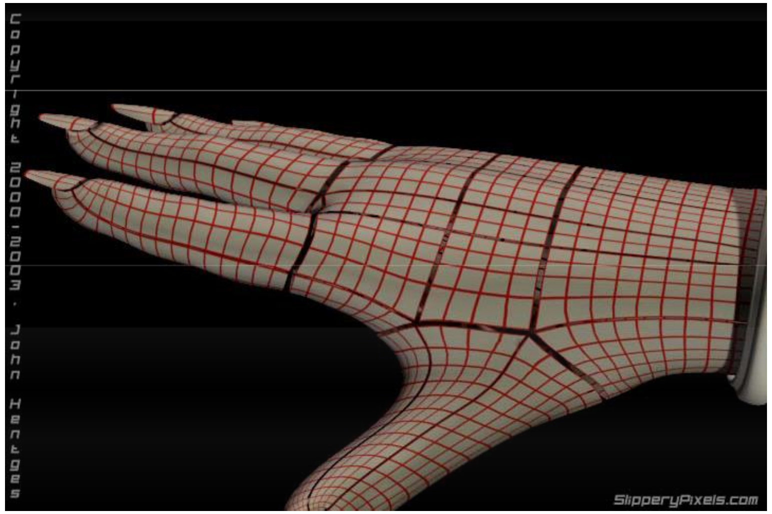






Problems with NURBS

- A single NURBS patch is either a topological disk, a tube or a torus



- Must use many NURBS patches to model complex geometry

- When deforming a surface made of NURBS patches, cracks arise at the seams

ILLINOIS

Subdivision surfaces

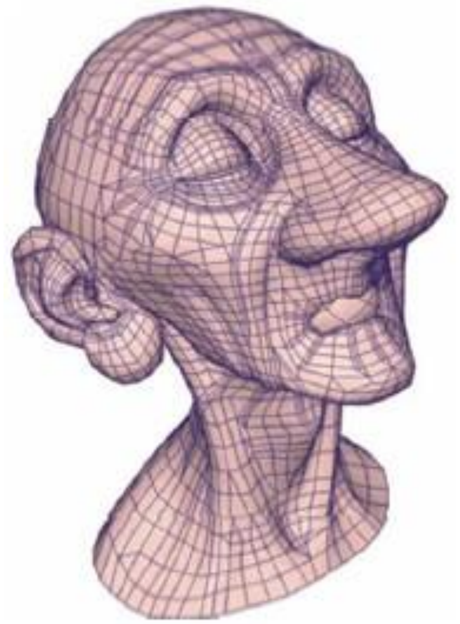
- Beyond shipbuilding: we want guaranteed continuity, without having to build everything out of rectangular patches.
  - Applications include CAD/CAM, 3D printing, museums and scanning, medicine, movies…

- The solution: *subdivision surfaces*.

Geri's Game, by Pixar (1997)

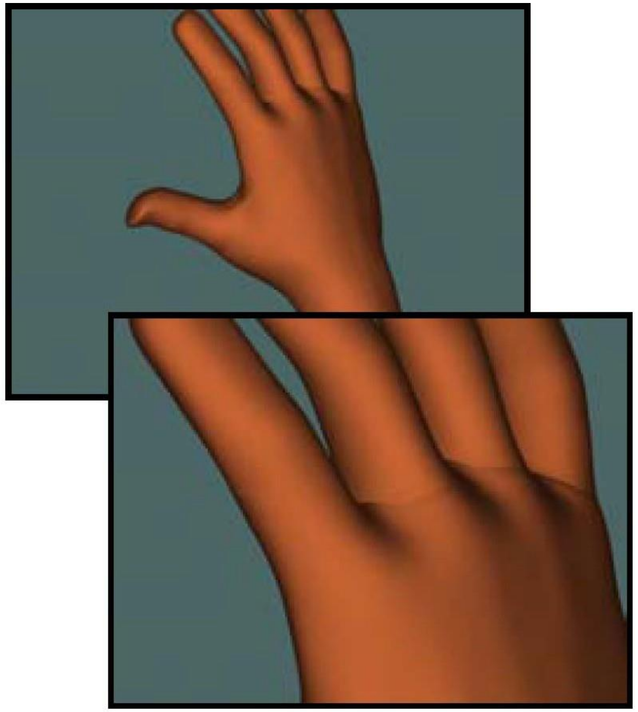
Example: Geri's Game (Pixar 1997)

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

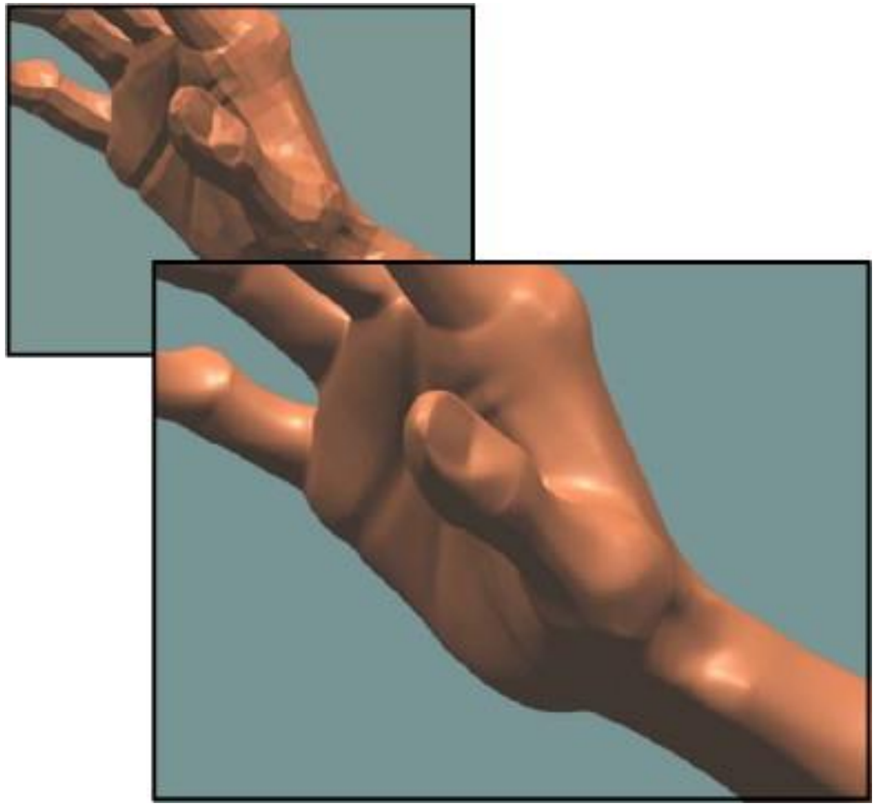


Example: Geri's Game (Pixar)

Woody's hand (NURBS)



Geri's hand (subdivision)

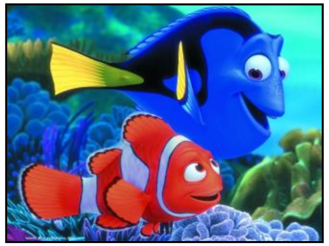


Example: Geri’s Game (Pixar)

- Sharp and semi-sharp features

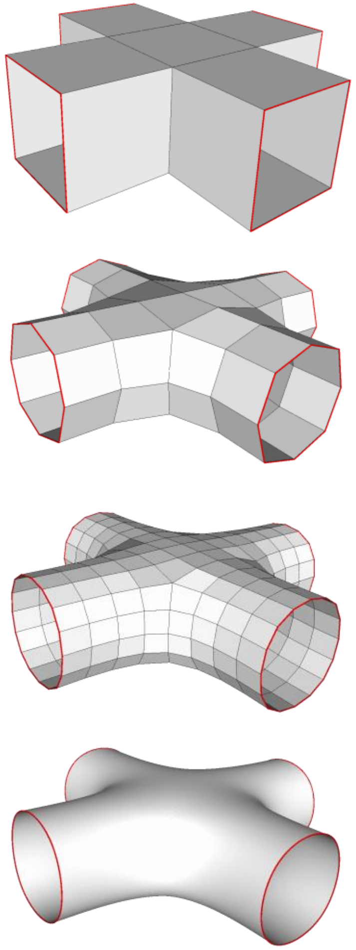
Subdivision surfaces and the movies

- Pixar first demonstrated subdivision surfaces in 1997 with Geri’s Game
  - Up until then they’d done everything in NURBS (Toy Story, A Bug’s Life.)
  - From 1999 onwards everything they did was with subdivision surfaces (Toy Story 2, Monsters Inc, Finding Nemo...)



Subdivision Surfaces

- Instead of ticking a parameter *t* along a parametric curve (or the parameters *u,v* over a parametric grid), subdivision surfaces repeatedly refine from a coarse set of *control points*.
- Each step of refinement adds new faces and vertices.
- The process converges to a smooth *limit surface*.



(Catmull-Clark in action)

Subdivision Surfaces – History

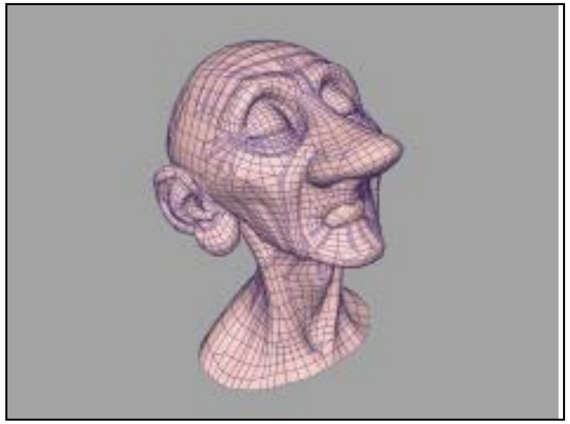
- de Rahm described a 2D (curve) subdivision scheme in 1947; rediscovered in 1974 by Chaikin

- Concept extended to 3D (surface) schemes by two separate groups during 1978:
  - Doo and Sabin found a biquadratic surface
  - Catmull and Clark found a bicubic surface

- Subsequent work in the 1980s (Loop, 1987; Dyn [Butterfly subdivision], 1990) led to tools suitable for CAD/CAM and animation

Useful Terms

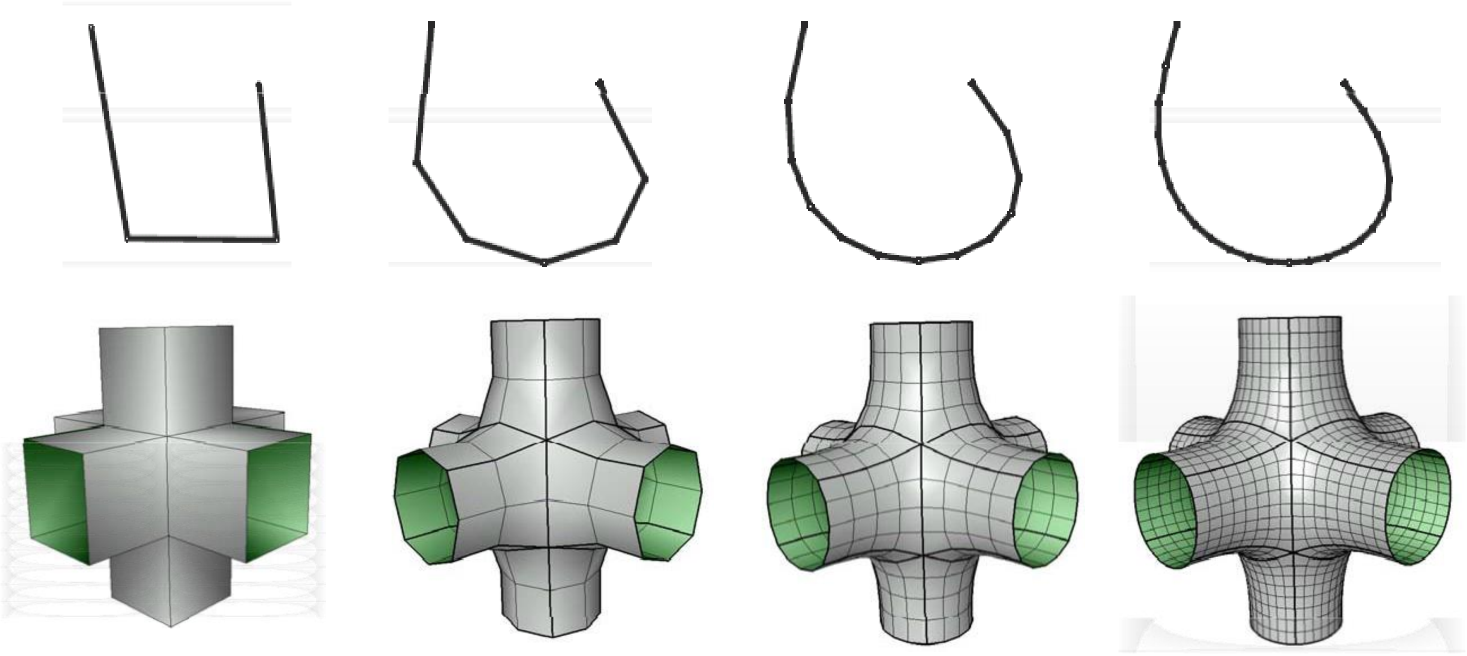
- A scheme which describes a 1D curve (even if that curve is travelling in 3D space, or higher) is called *univariate*, referring to the fact that the limit curve can be approximated by a polynomial in one variable (t).
- A scheme which describes a 2D surface is called *bivariate*, the limit surface can be approximated by a u,v parameterization.
- A scheme which retains and passes through its original control points is called an *interpolating* scheme.
- A scheme which moves away from its original control points, converging to a limit curve or surface nearby, is called an *approximating* scheme.



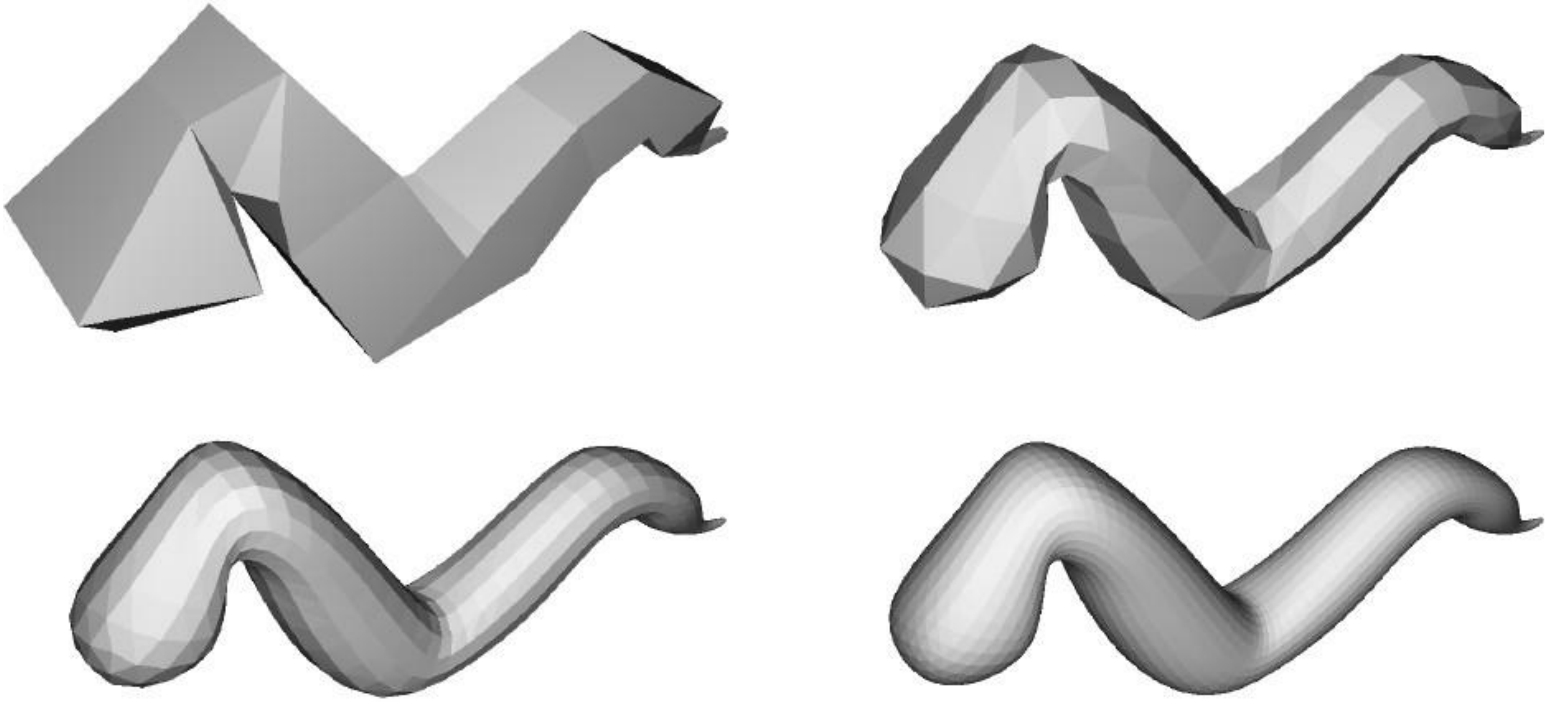
Control surface for Geri's head

Subdivision

"Subdivision defines a smooth curve or surface as the limit of a sequence of successive refinements"



Subdivision

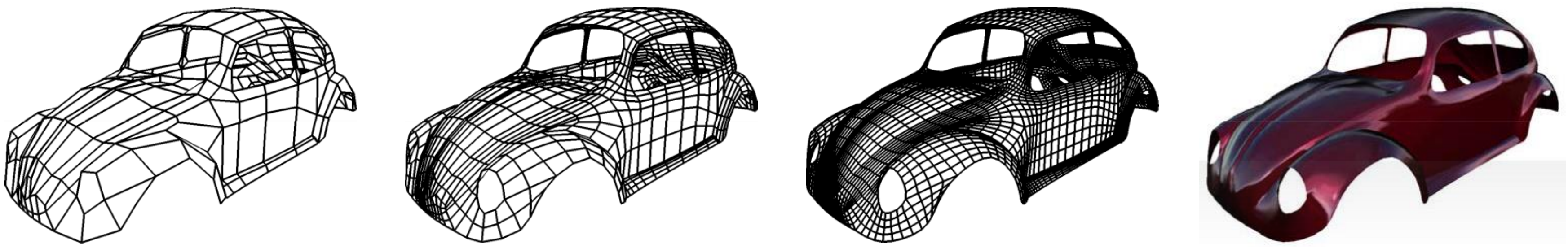


Subdivision Surfaces

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

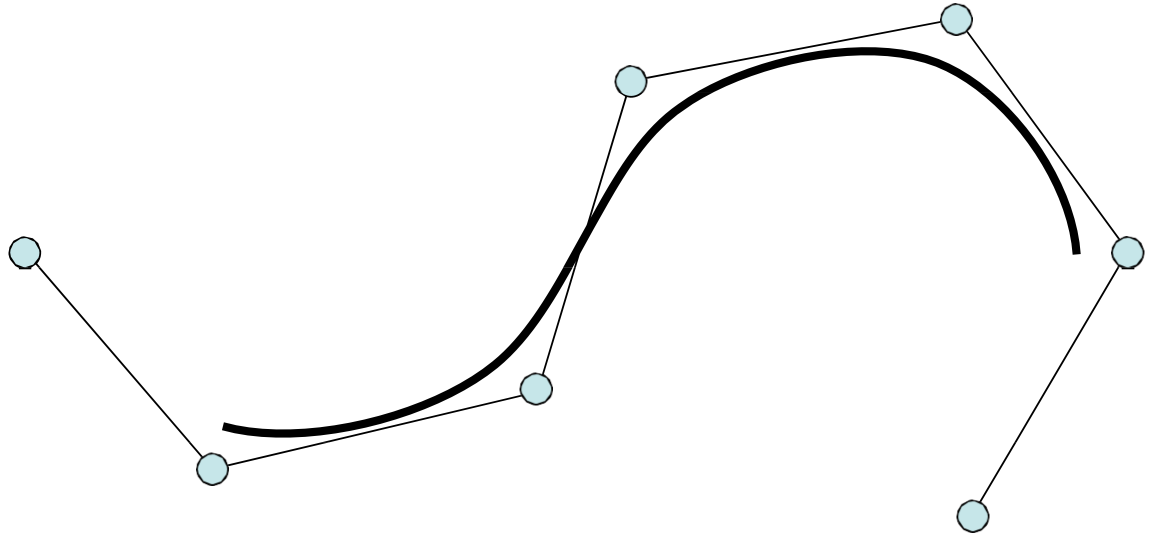
Subdivision Surfaces

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



Subdivision Curves

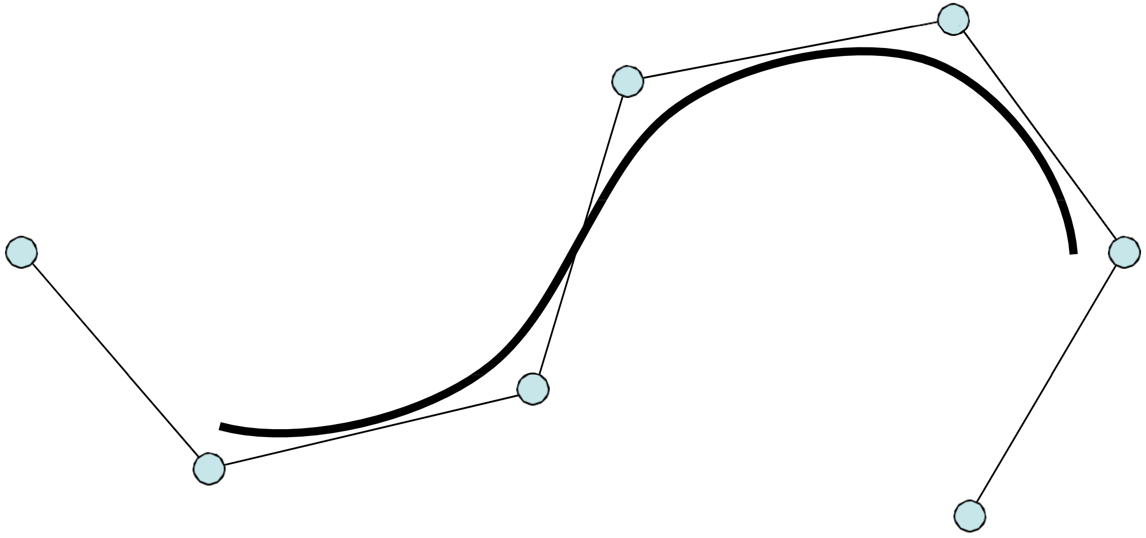
Given a control polygon...



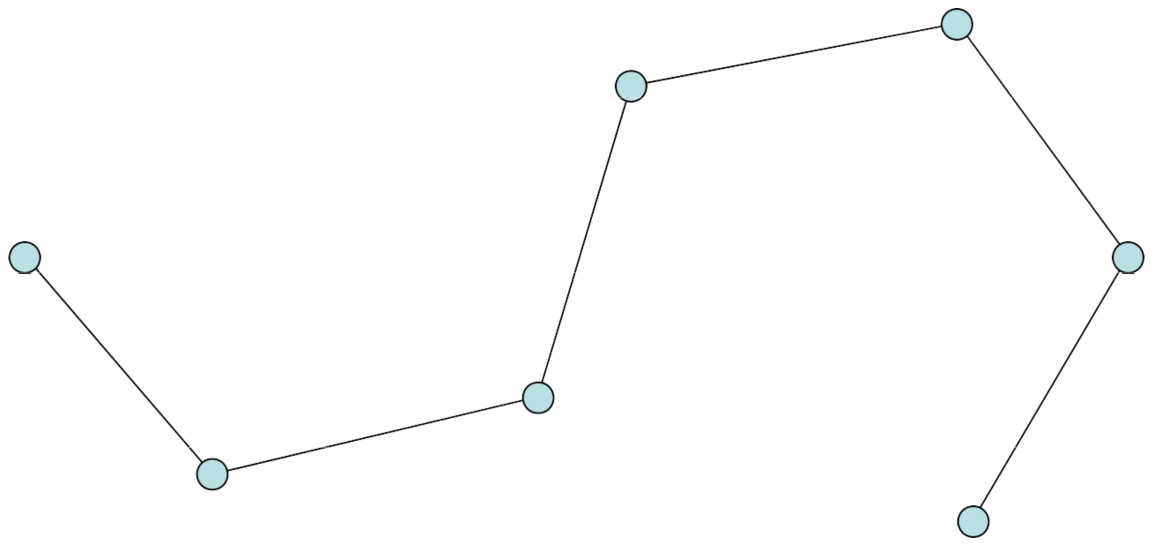
...find a smooth curve related to that polygon.

Subdivision Curve Types

- Approximating
- Interpolating

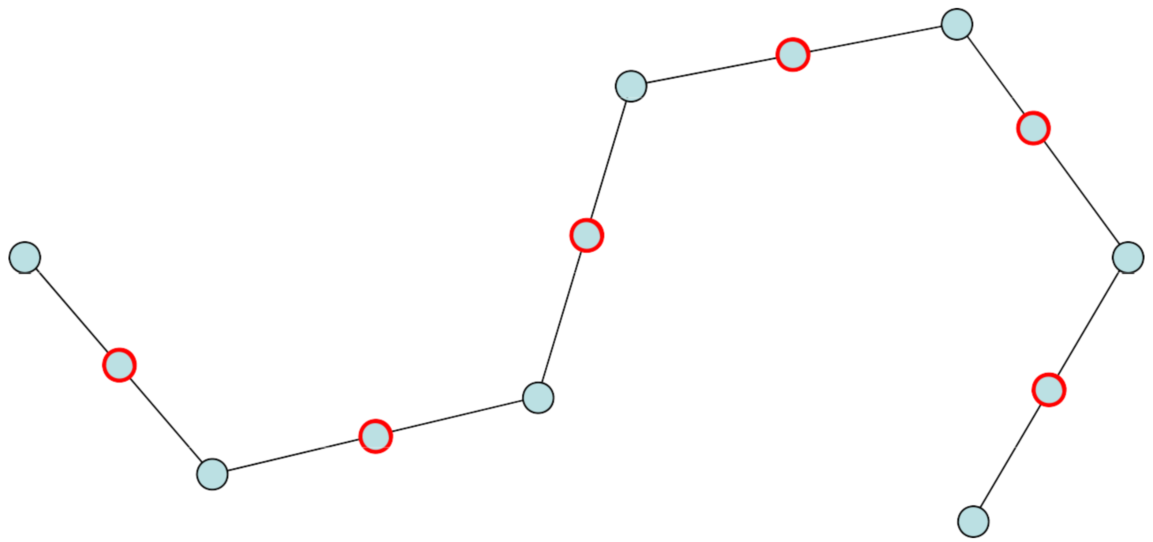


Approximating



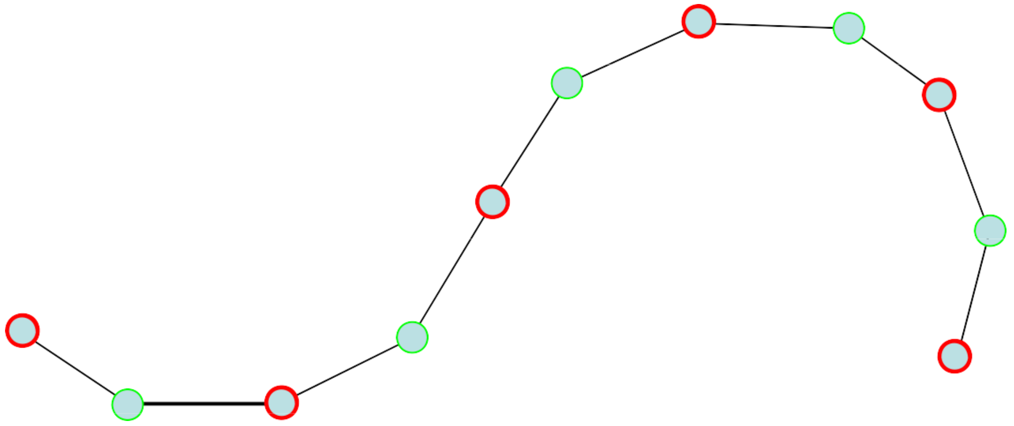
Approximating

Splitting step: split each edge in two



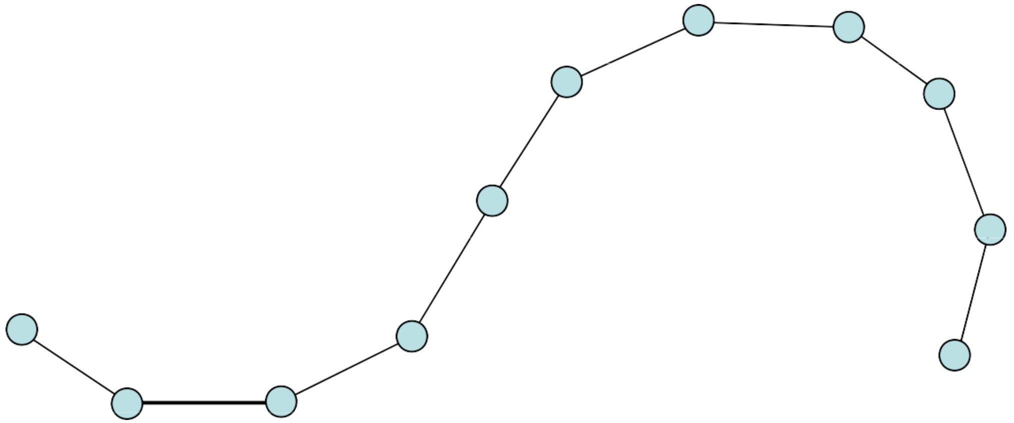
Approximating

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



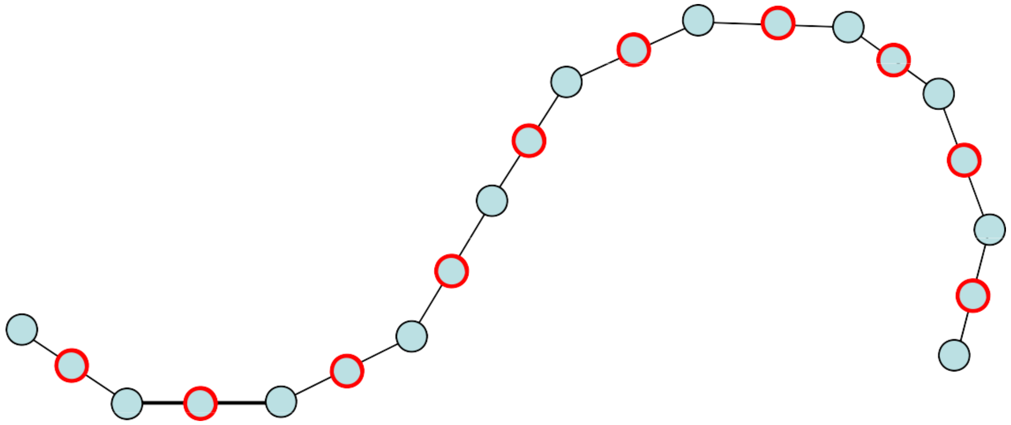
Approximating

Start over ...



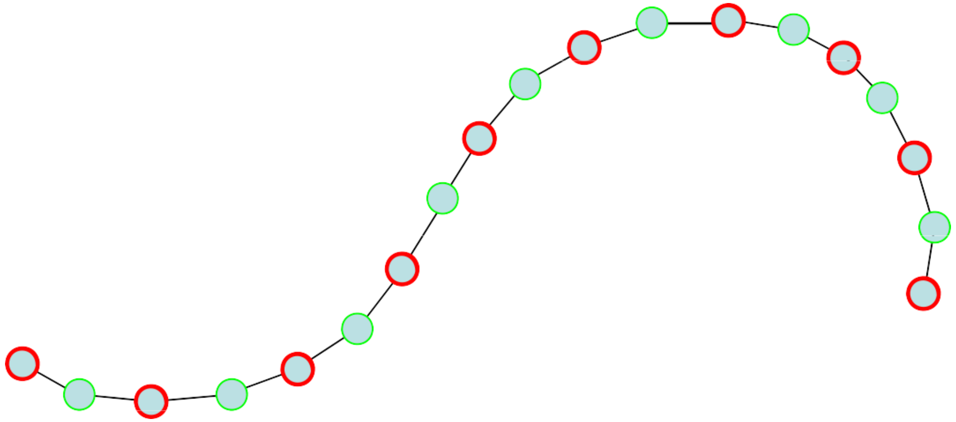
Approximating

...splitting...



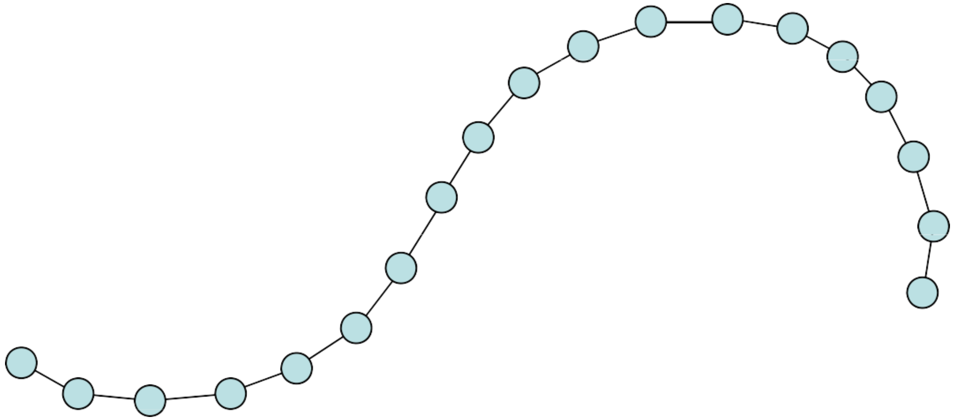
Approximating

...averaging...



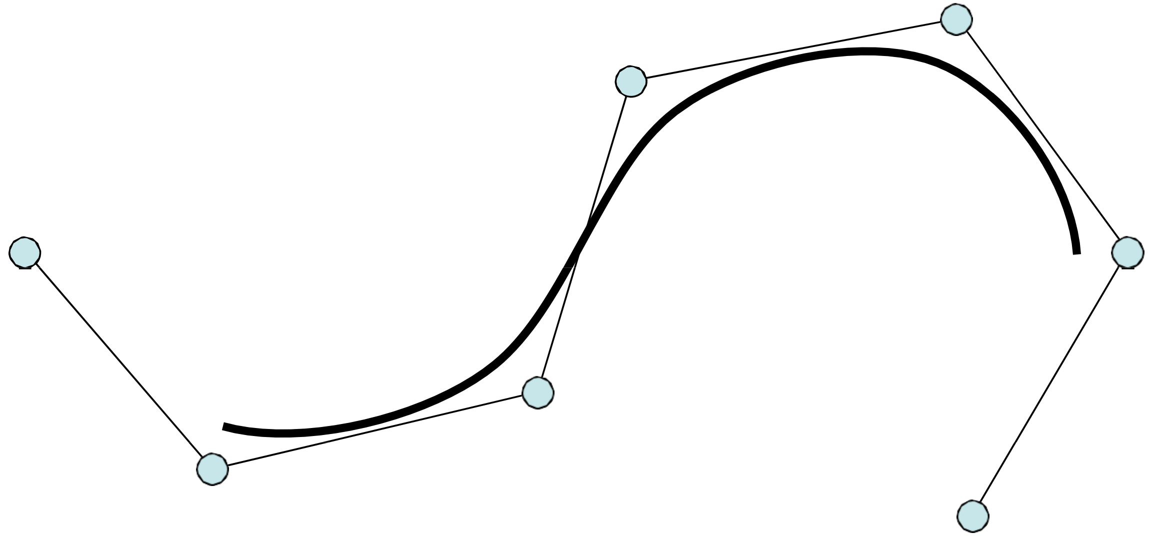
Approximating

...and so on...



Approximating

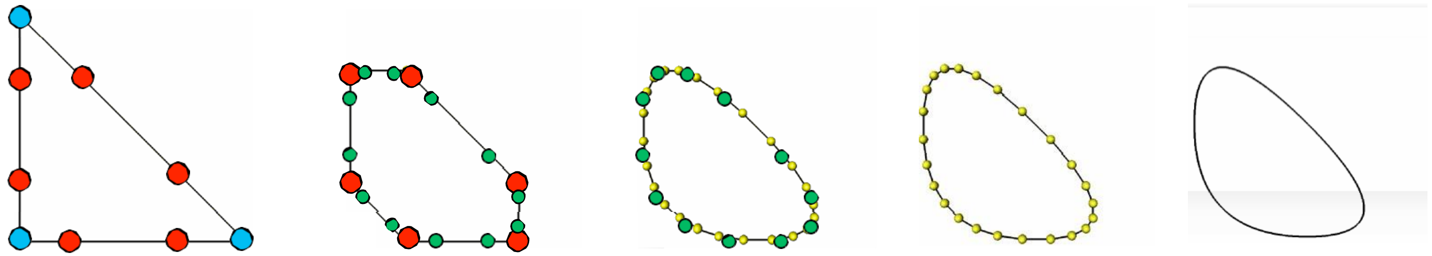
If the rule is designed carefully...



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

Corner Cutting

- Subdivision rule:
  - Insert *two* new vertices at ¼ and ¾ of each edge
  - *Remove* the old vertices
  - Connect the new vertices
  - In the limit, generates a curve called a quadratic B-Spline



B-Spline Curves

- Piecewise polynomial of degree n

B-spline curve

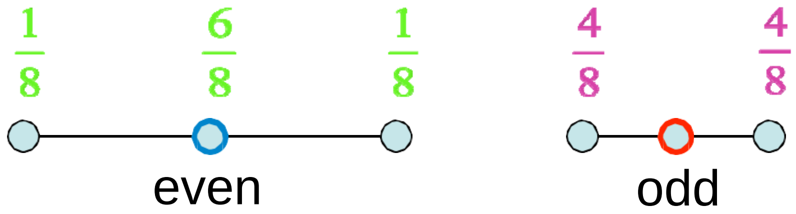
control points

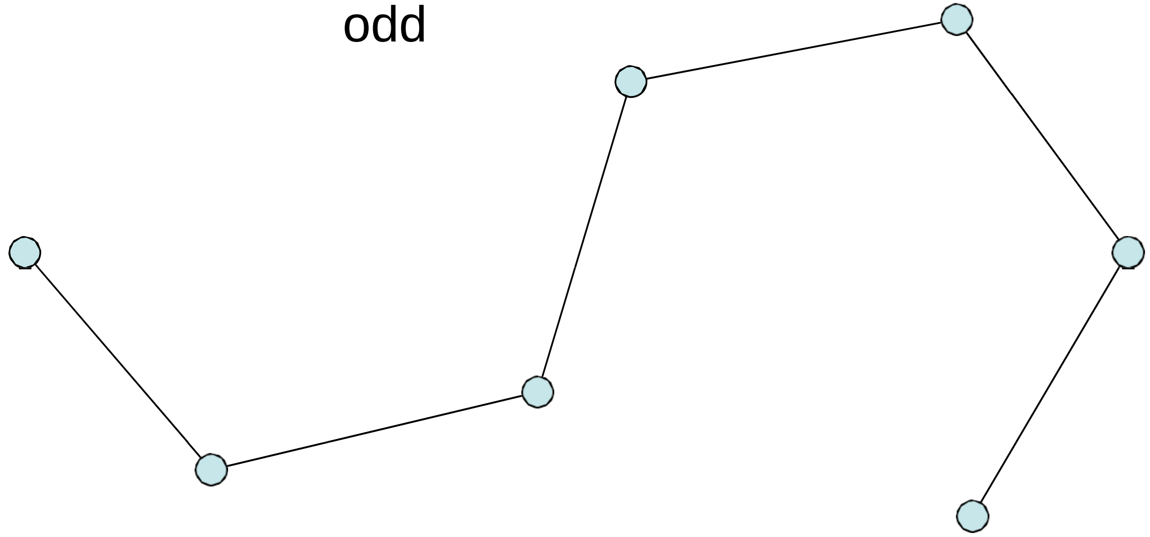
$$\mathbf{s}(u) = \sum_{i=0}^{k} \mathbf{d}_i N_i^n(u)$$

parameter value

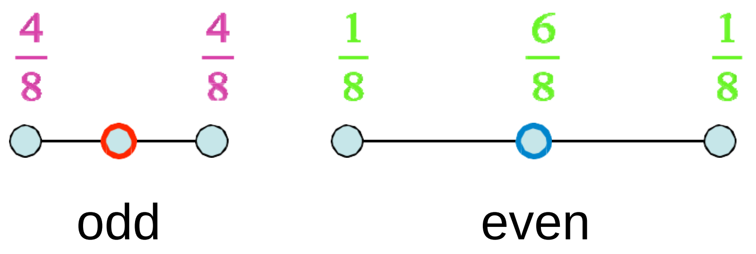
basis functions

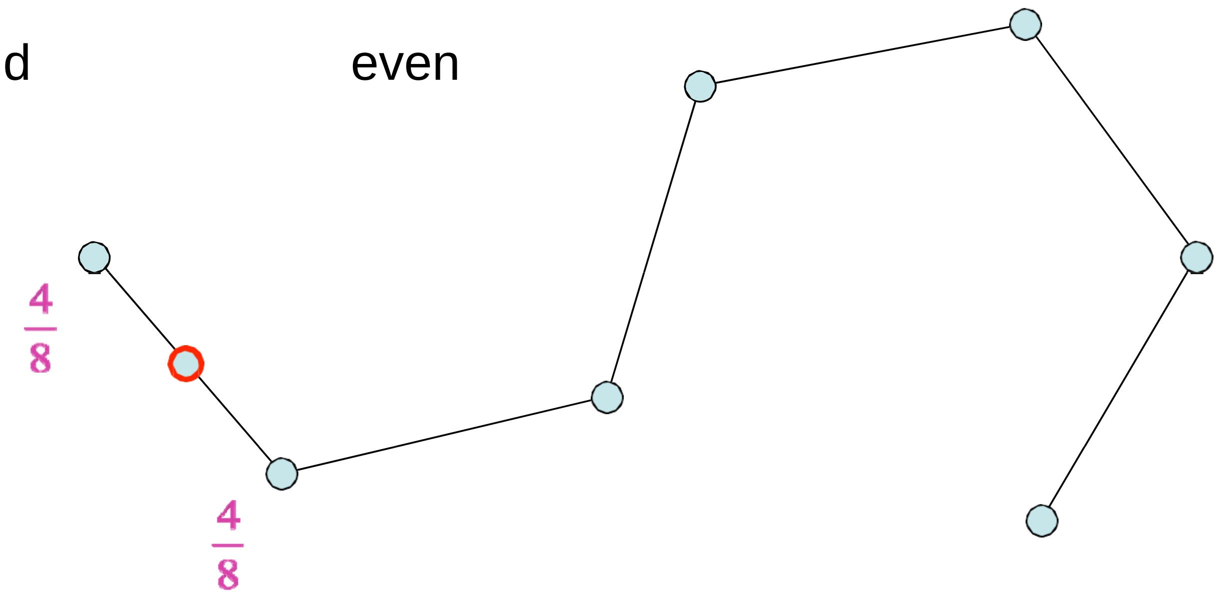
Cubic B-Spline



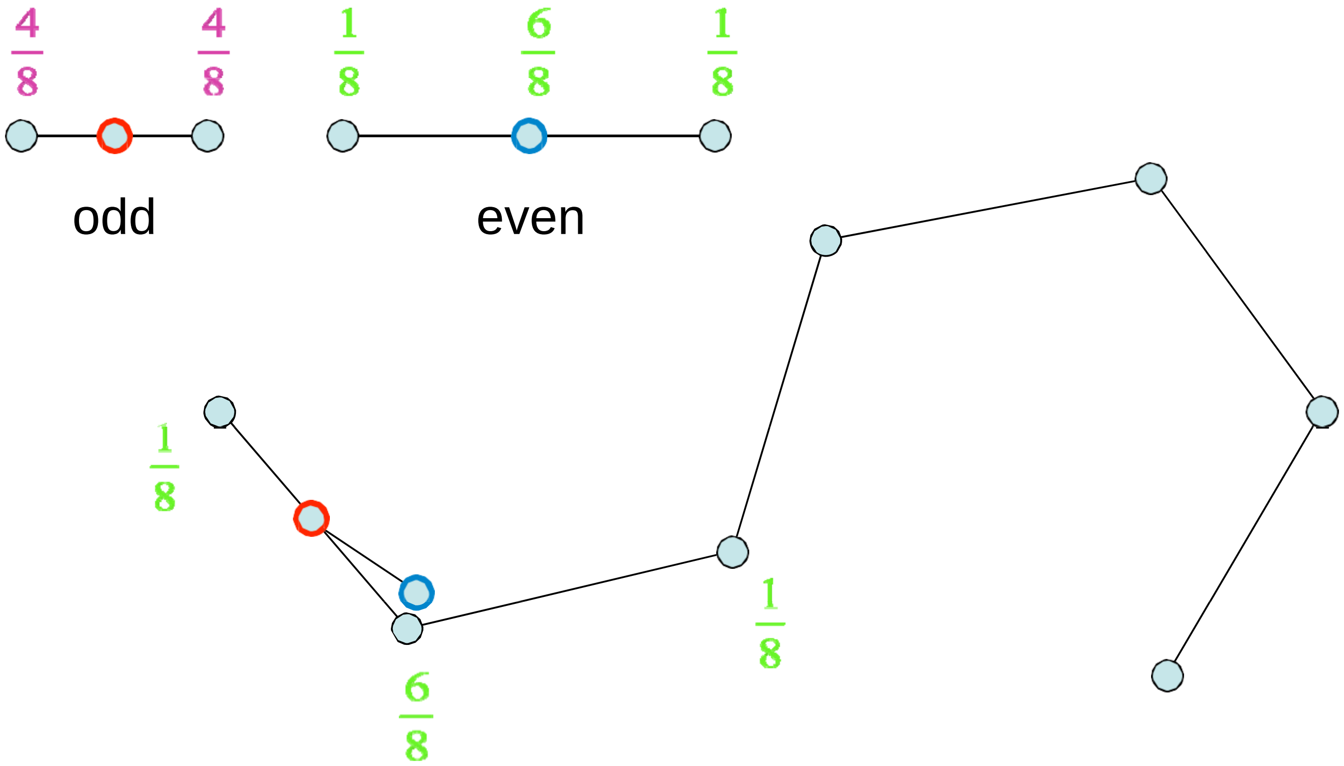


Cubic B-Spline

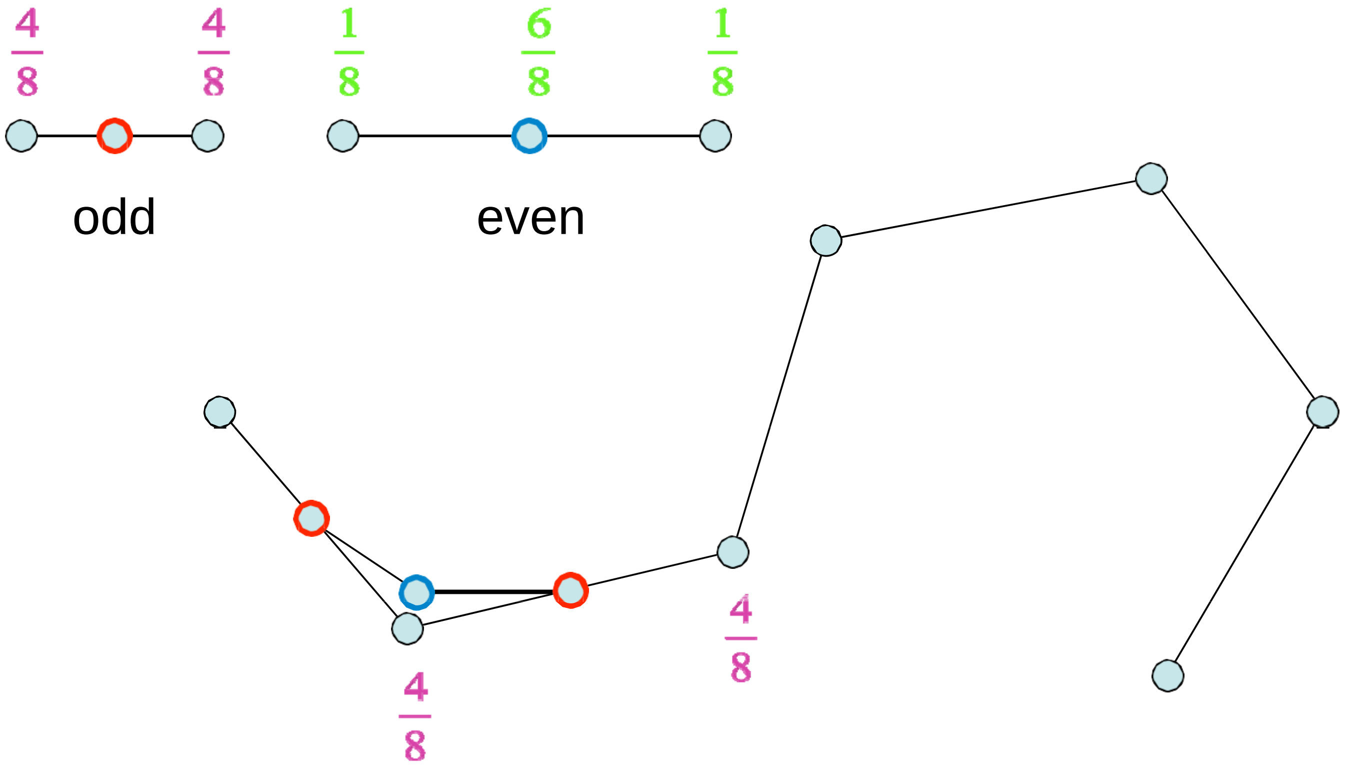




Cubic B-Spline

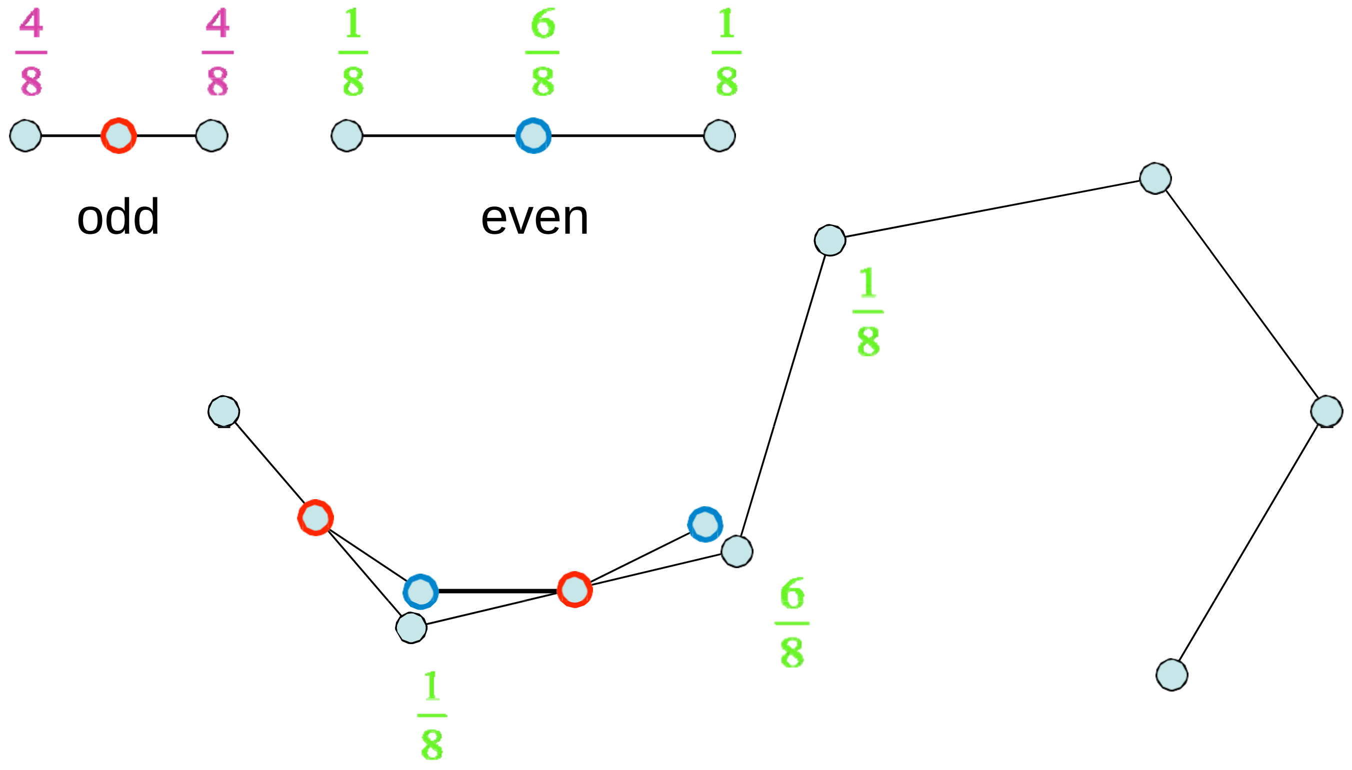


Cubic B-Spline

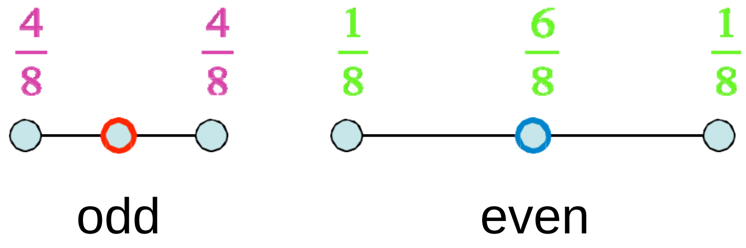


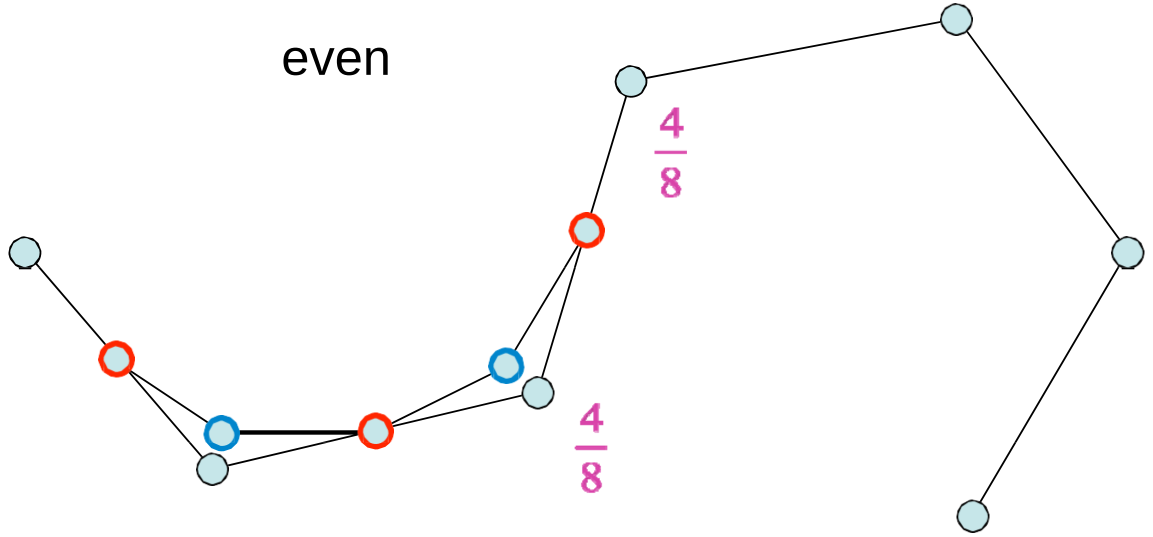
ILLINOIS

Cubic B-Spline



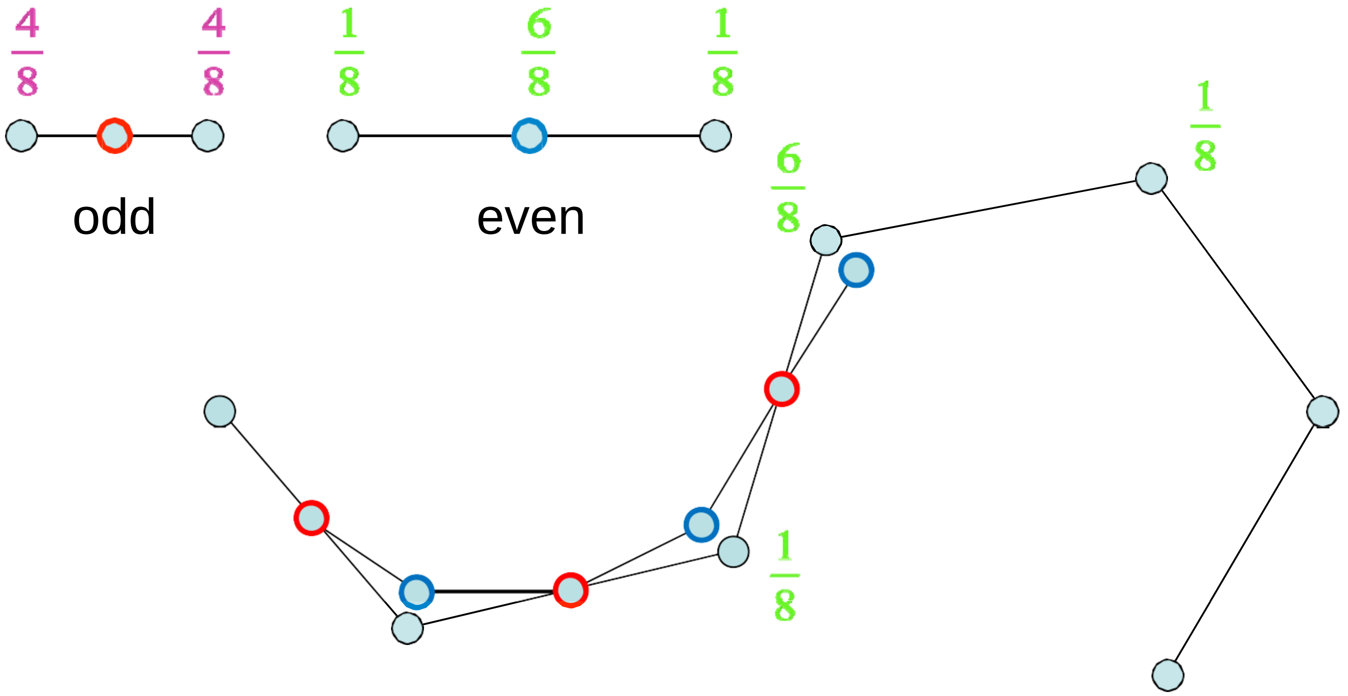
Cubic B-Spline



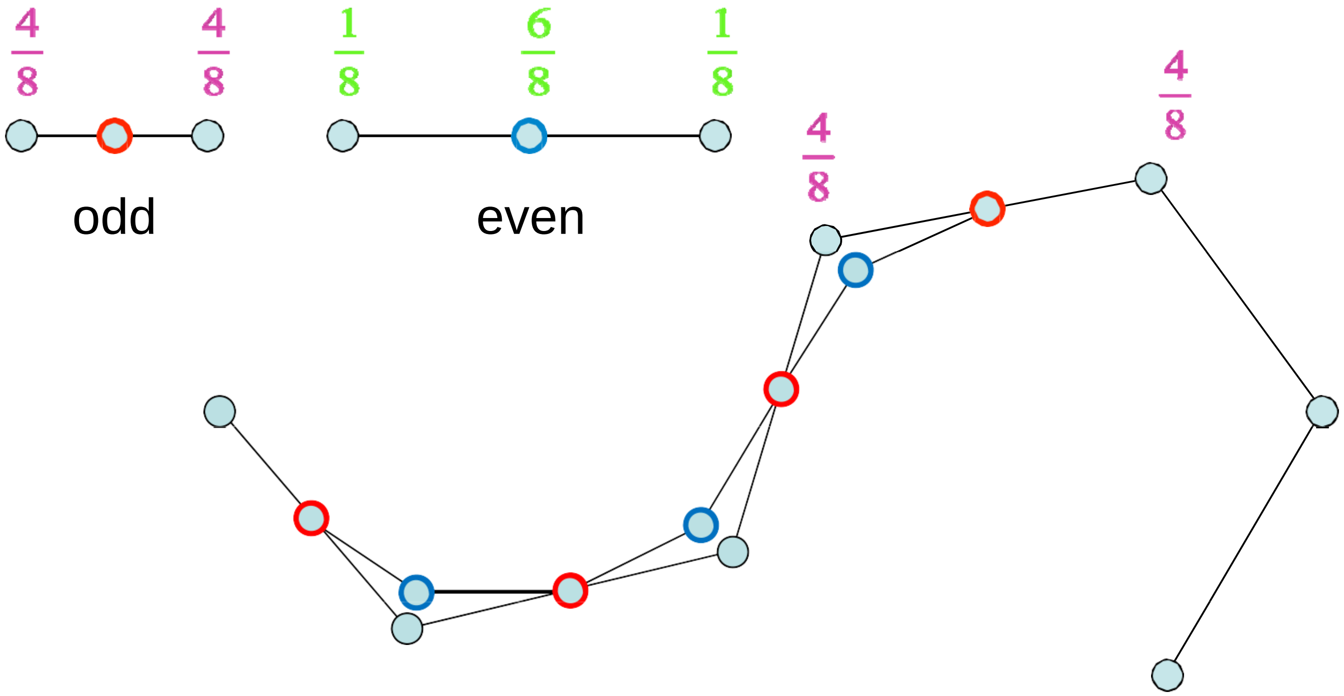


ILLINOIS

Cubic B-Spline

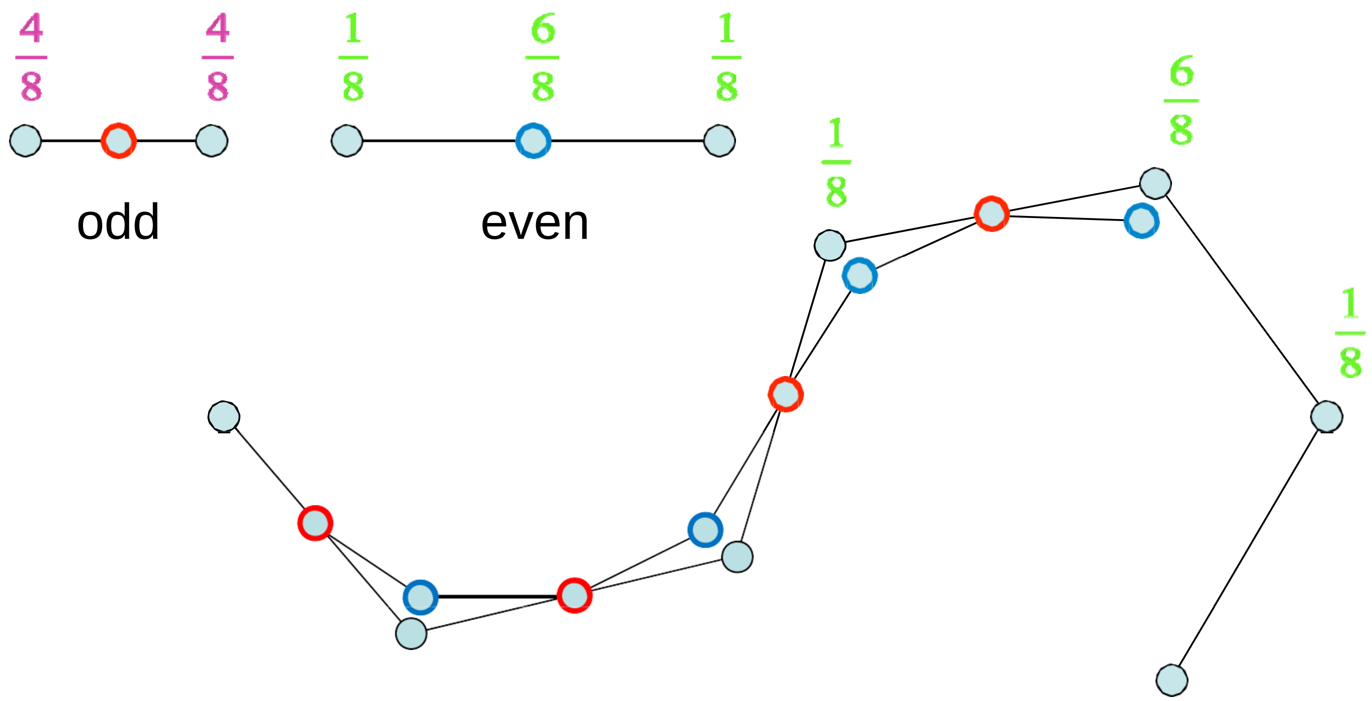


Cubic B-Spline

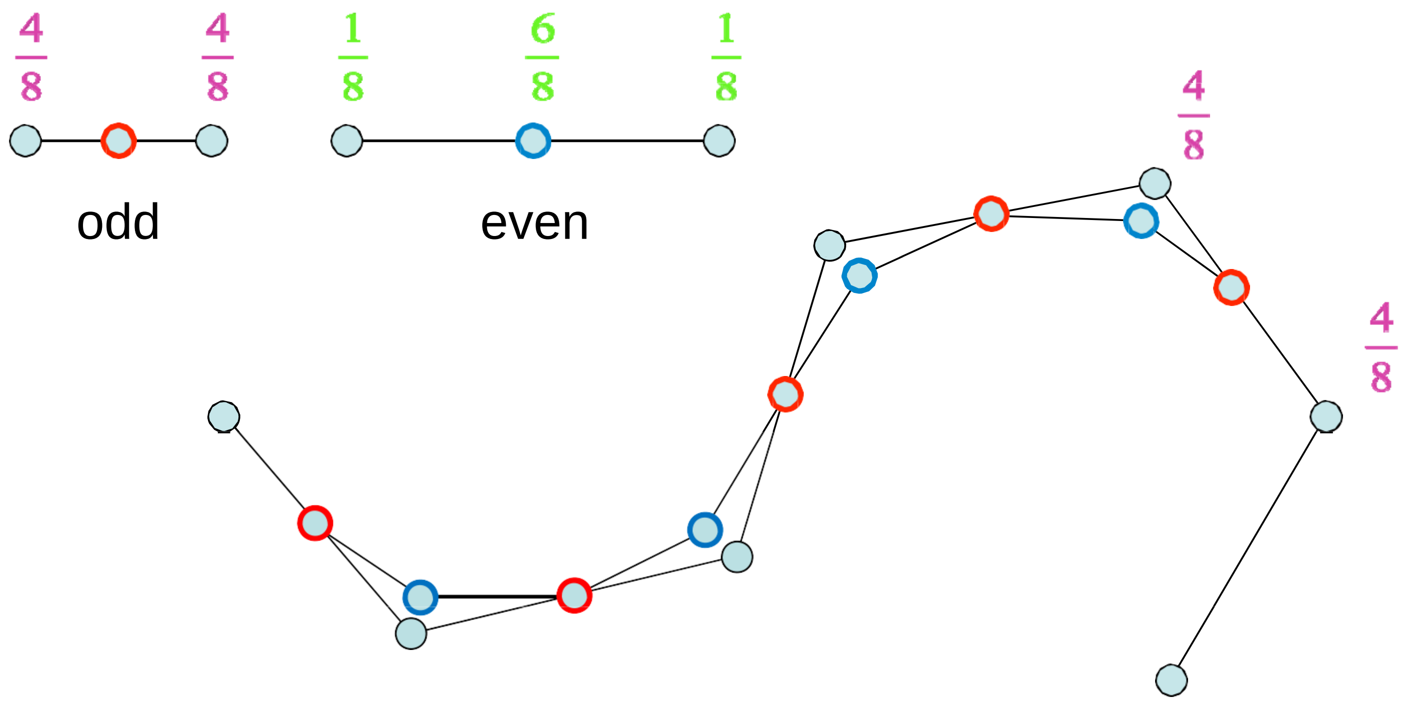


ILLINOIS

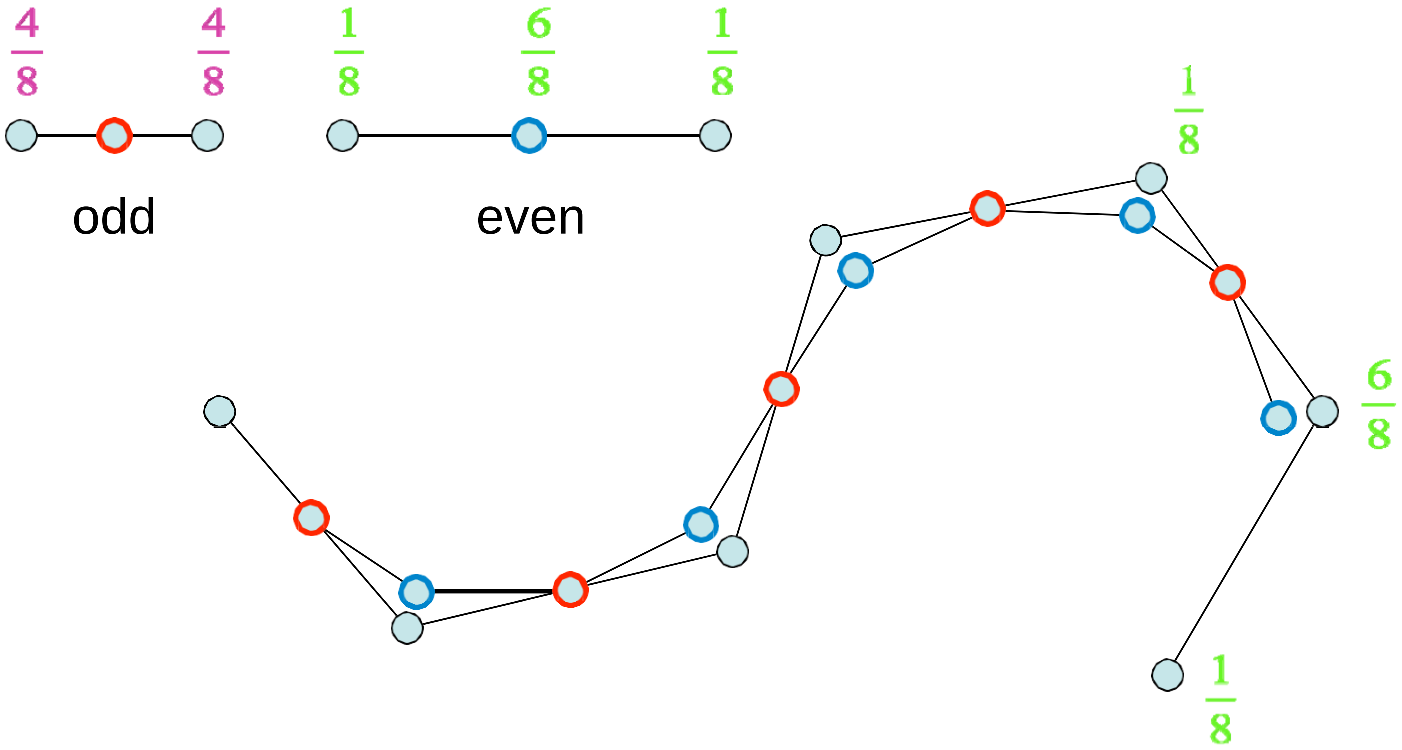
Cubic B-Spline



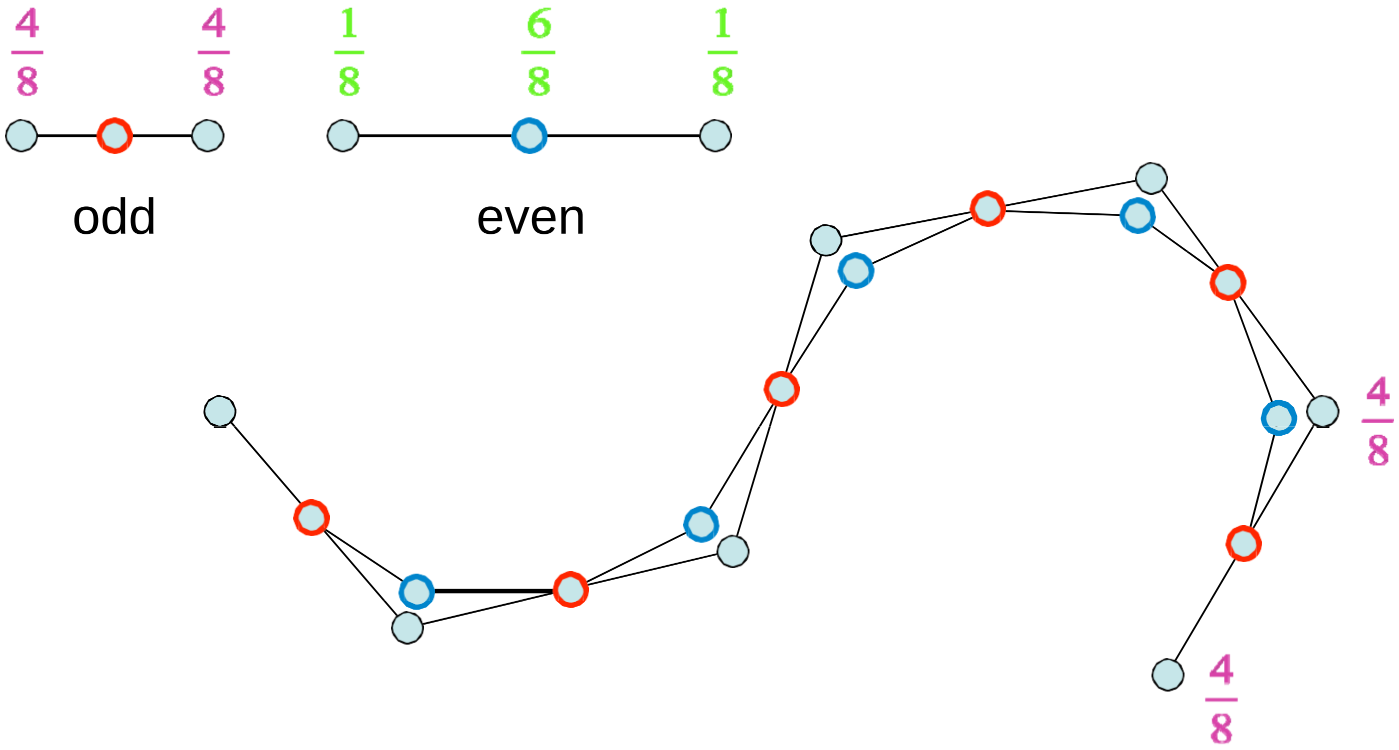
Cubic B-Spline



Cubic B-Spline

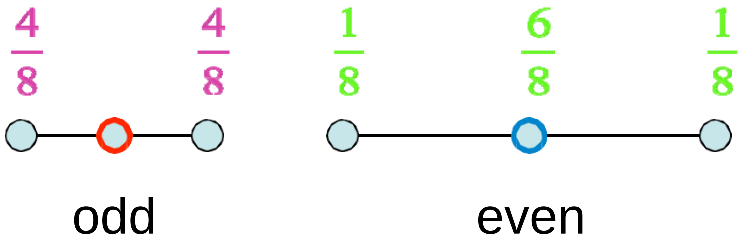


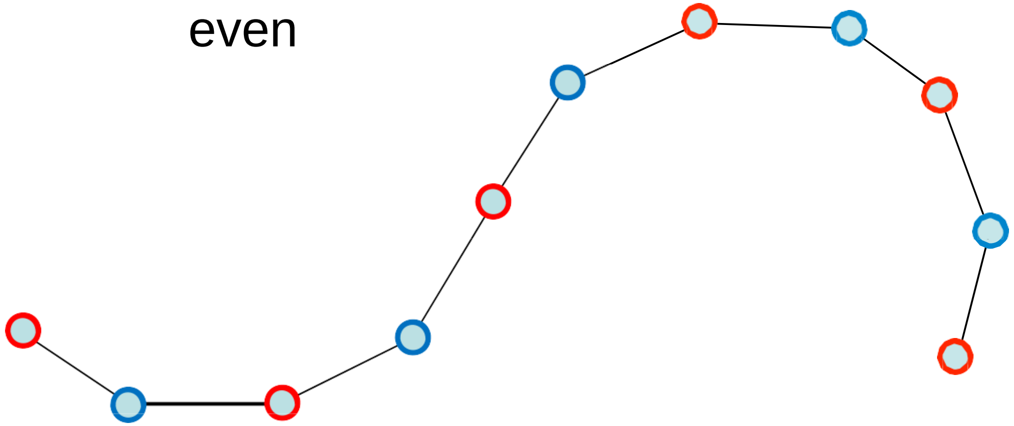
Cubic B-Spline



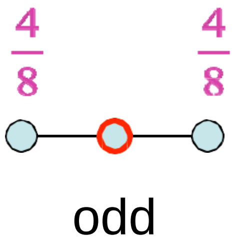
ILLINOIS

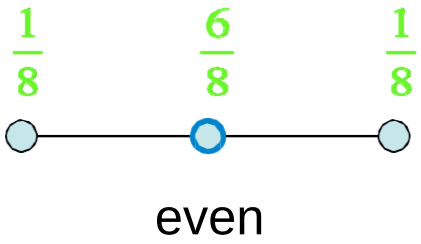
Cubic B-Spline

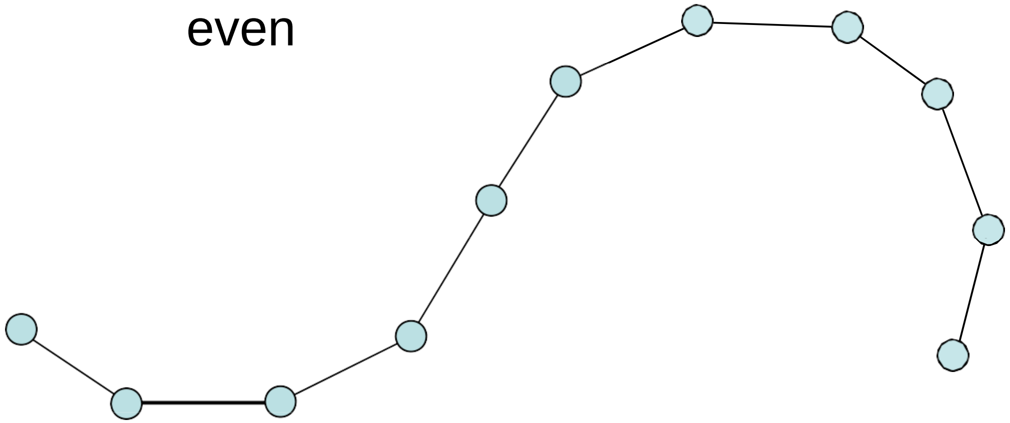




Cubic B-Spline

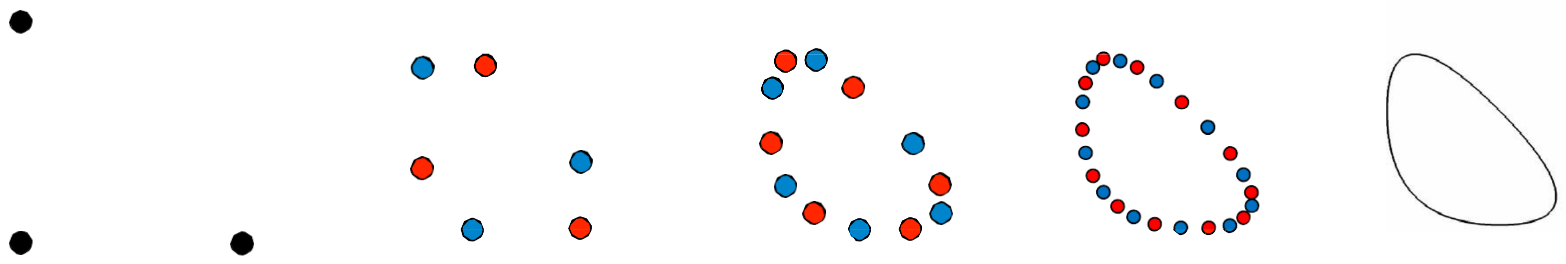






B-Spline Curves

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



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

B-Spline Curves

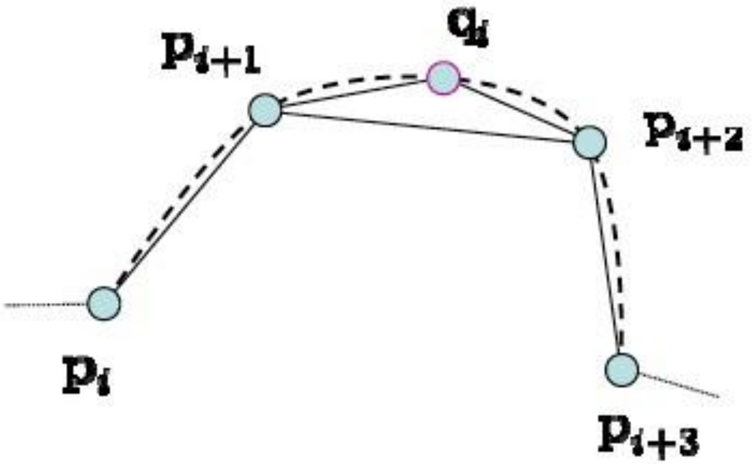
- Subdivision rules for control polygon

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

- 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

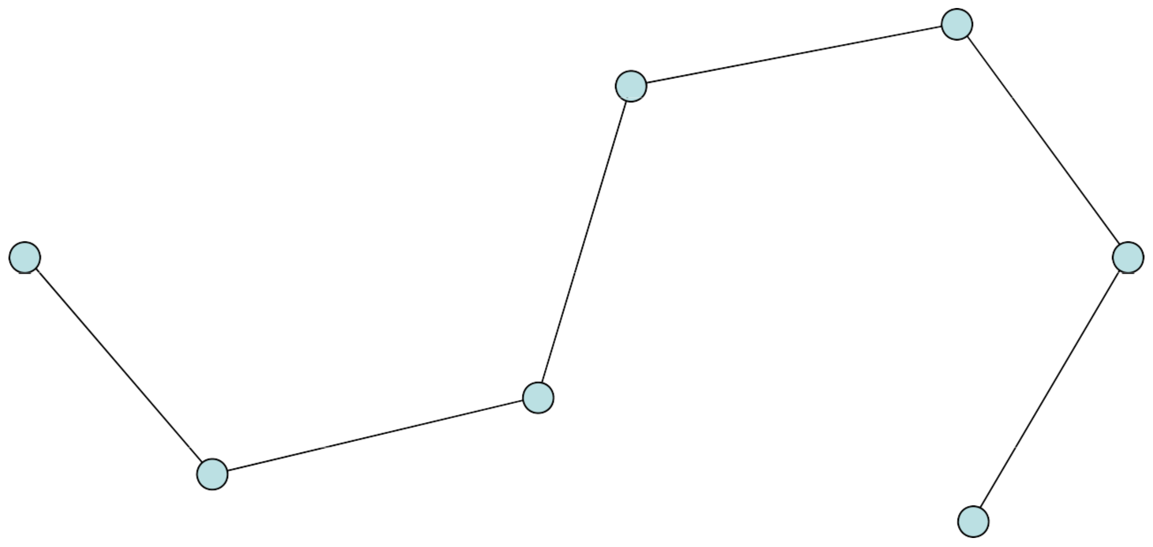


$$f(x) = ax^3 + bx^2 + cx + d$$

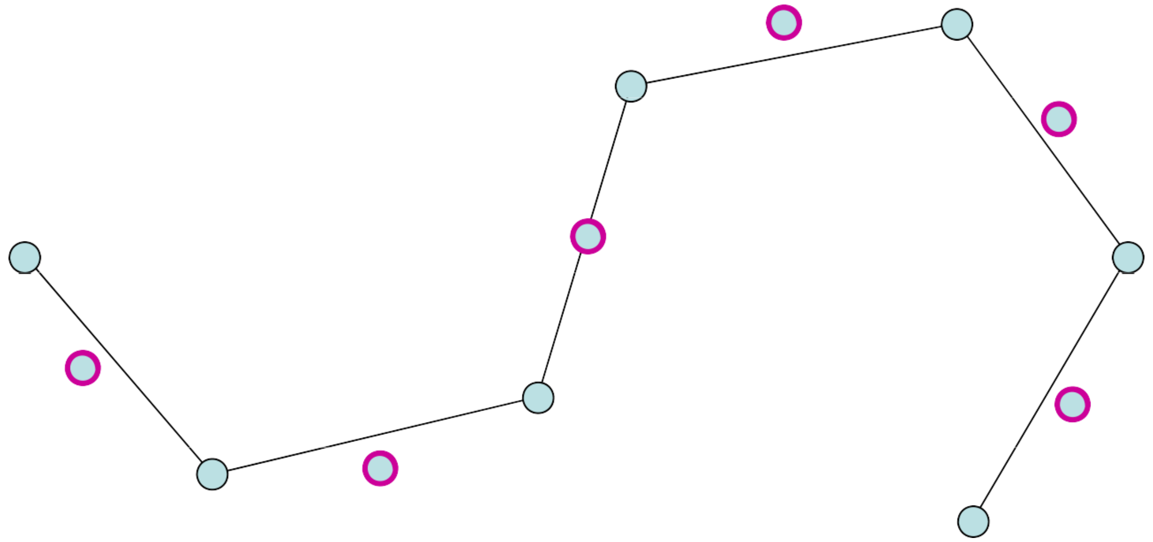
$$f(j) = \mathbf{p}_{i+j}, \quad j = 0, \ldots, 3$$

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

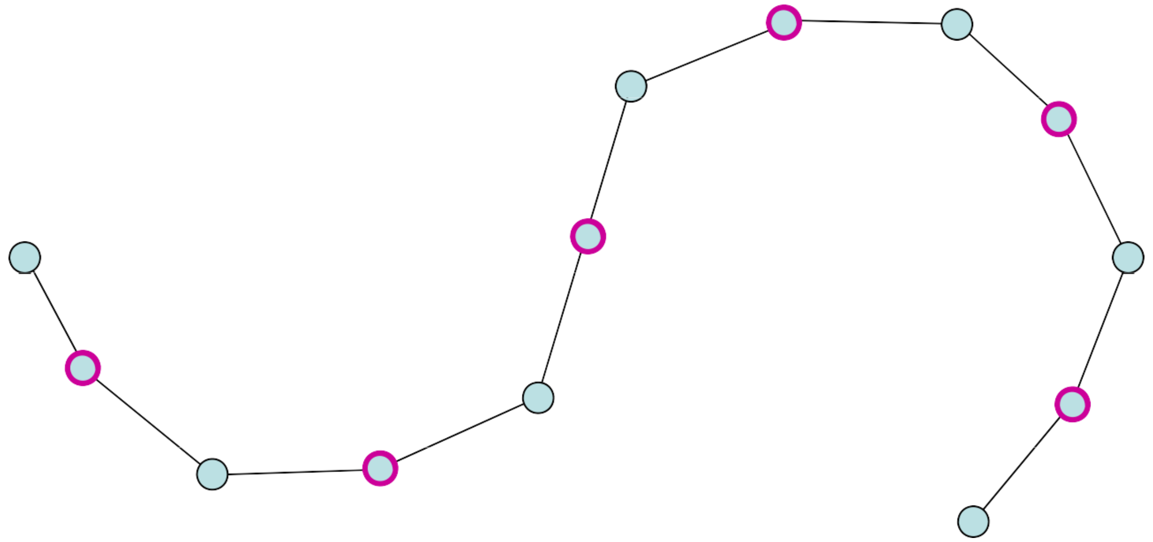
Interpolating



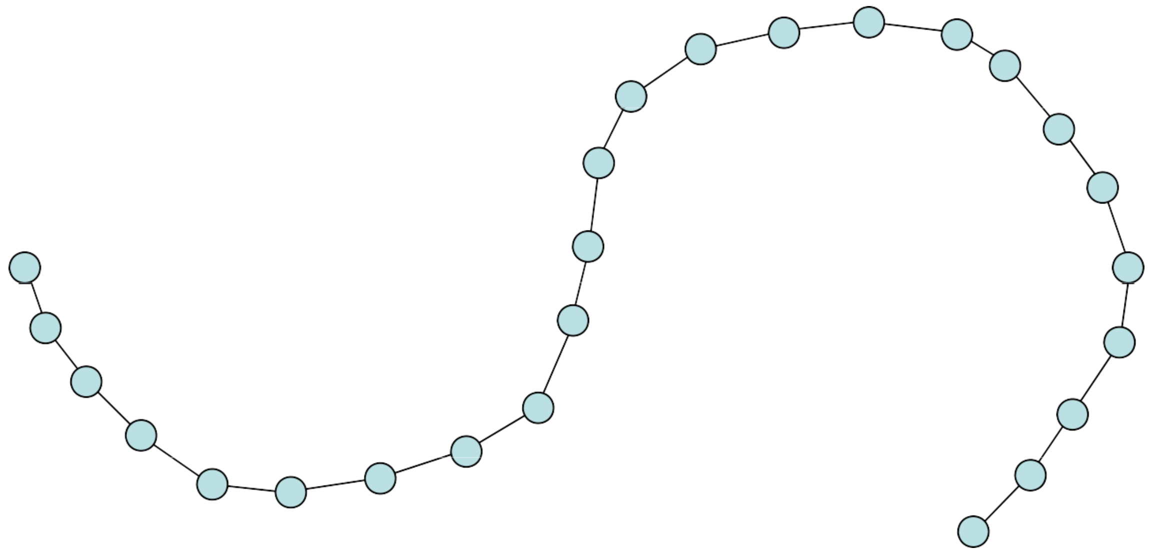
Interpolating



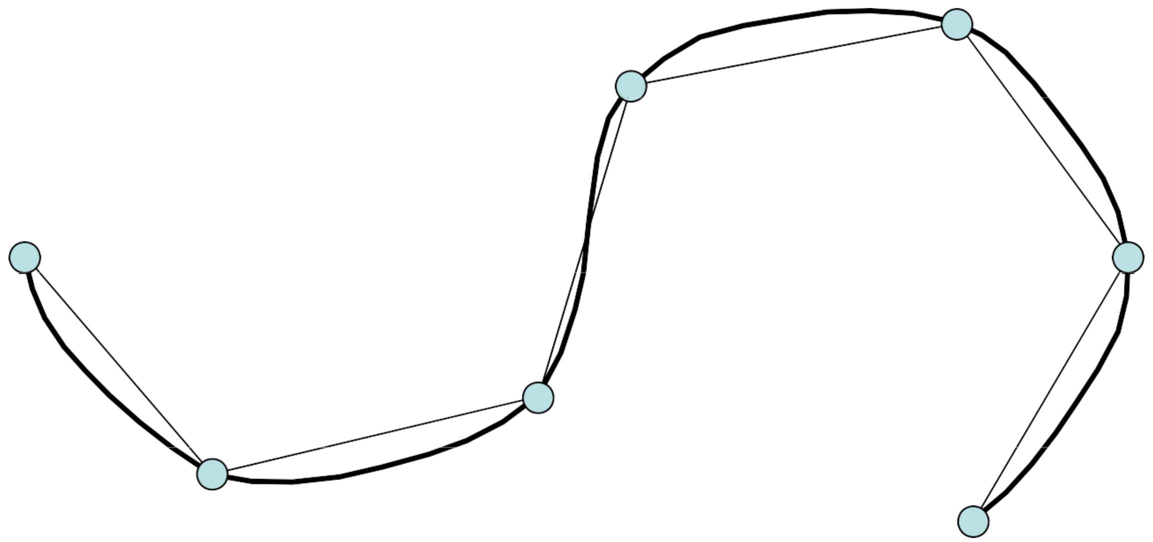
Interpolating



Interpolating

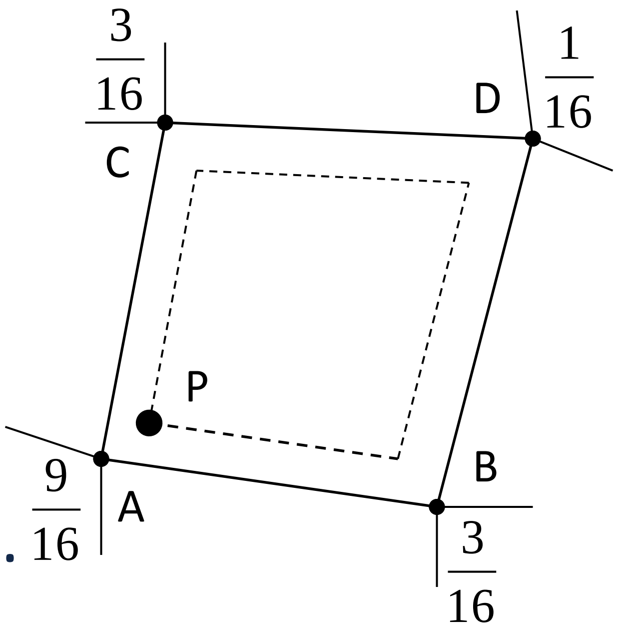


Interpolating

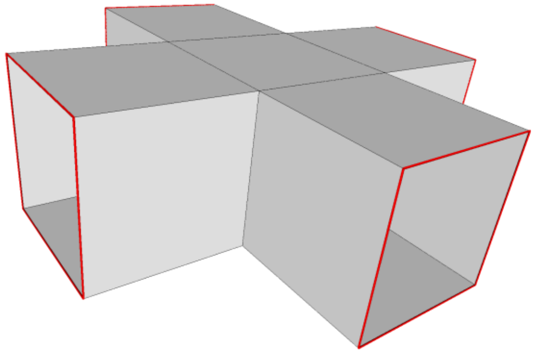


Making the jump to 3D: Doo-Sabin

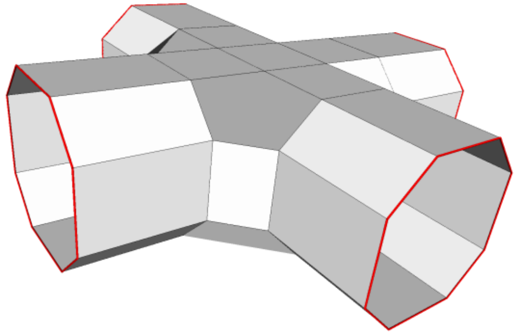
- *Doo-Sabin* takes Chaikin to 3D:
  - $P = (9/16)A +$
    $(3/16)B +$
    $(3/16)C +$
    $(1/16)D$
- This replaces every old vertex with four new vertices.
- The limit surface is biquadratic, C1 continuous everywhere.



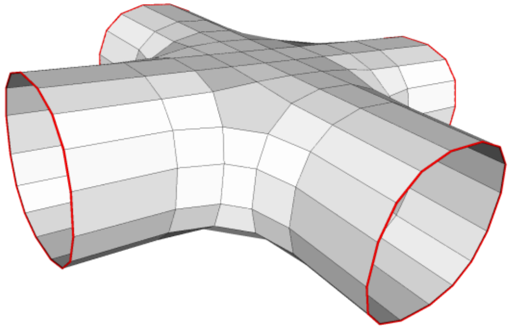
Doo-Sabin in action



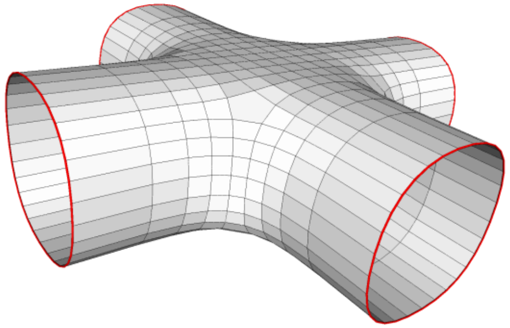
(0) 18 faces



(1) 54 faces



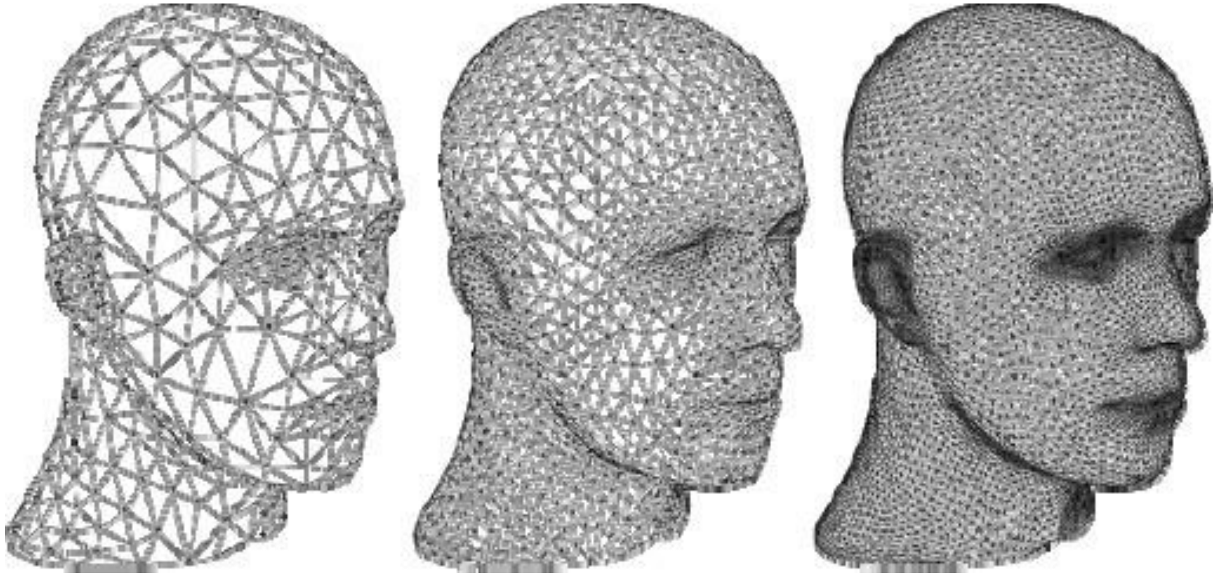
(2) 190 faces



(3) 702 faces

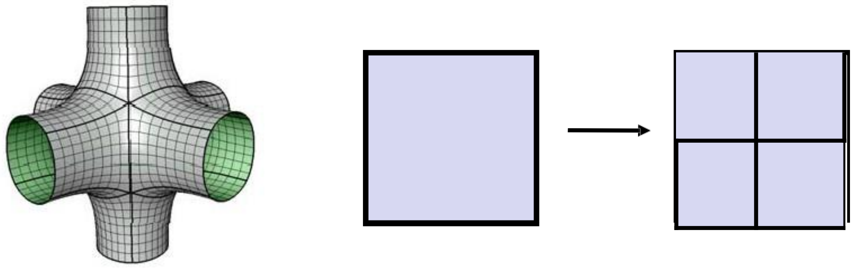
Subdivision Surfaces

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



Subdivision Rules

- How the connectivity changes



- How the geometry changes
  - Old points
  - New points

Subdivision Zoo

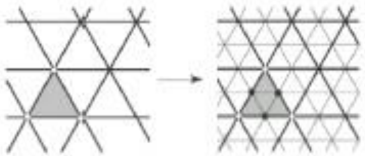
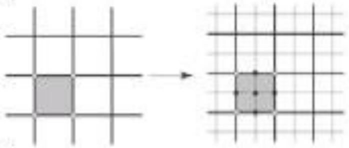
- Classification of subdivision schemes

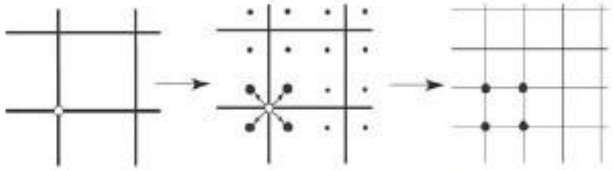
Primal	Faces are split into sub-faces
Dual	Vertices are split into multiple vertices

Approximating	Control points are not interpolated
Interpolating	Control points are interpolated

Subdivision Zoo

- Classification of subdivision schemes

Primal (face split)		
	 Triangular meshes	 Quad Meshes
Approximating	Loop(C^2)	Catmull-Clark(C^2)
Interpolating	Mod. Butterfly (C^1)	Kobbelt (C^1)



Dual (vertex split)
Doo-Sabin, Midedge(C^1)
Biquartic (C^2)

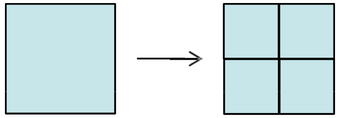
Subdivision Zoo

- Classification of subdivision schemes

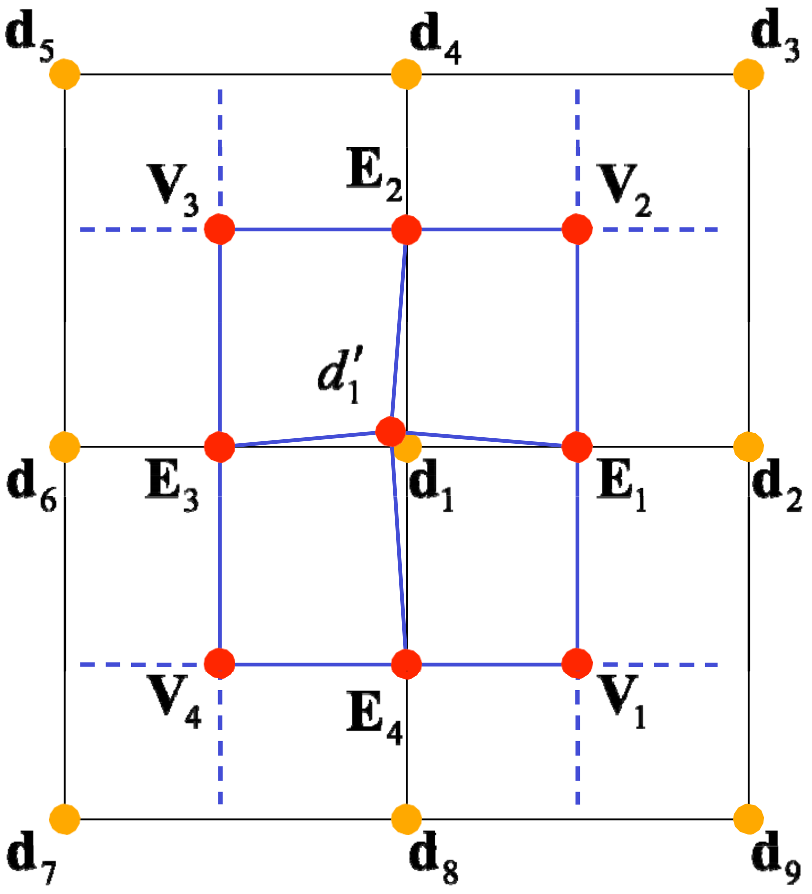
	Primal		Dual
	Triangles	**Rectangles**	
Approximating	Loop	**Catmull-Clark**	Doo-Sabin Midedge
Interpolating	Butterfly	Kobbelt	

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

NOTE: valence = number of neighboring vertices

First subdivision generates quad mesh

Some vertices extraordinary (valence ≠ 4)

Rules

Face vertex = average of face's vertices

Edge vertex = average of edge's two vertices & adjacent face's two vertices

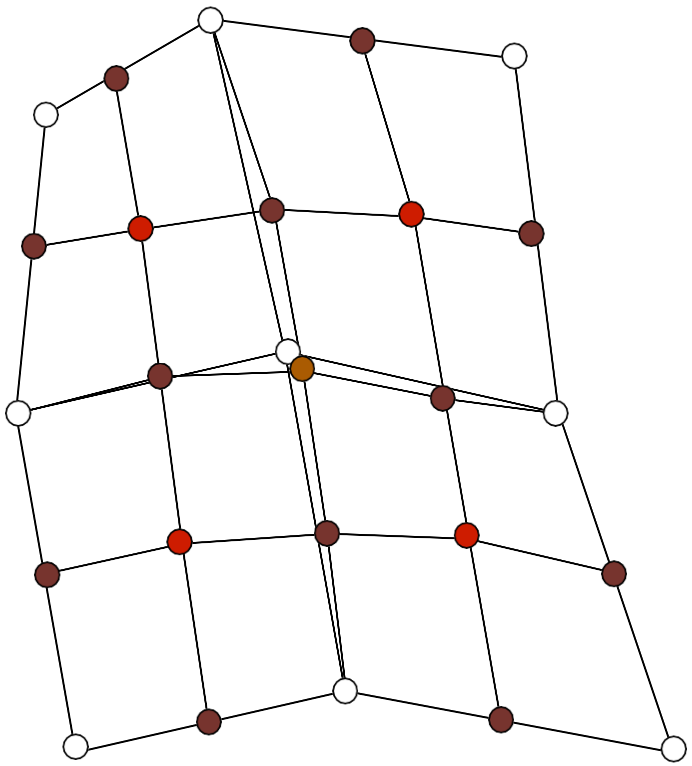
New vertex position = (1/valence) x **sum** of…

- Average of neighboring face points
- 2 x average of neighboring edge points

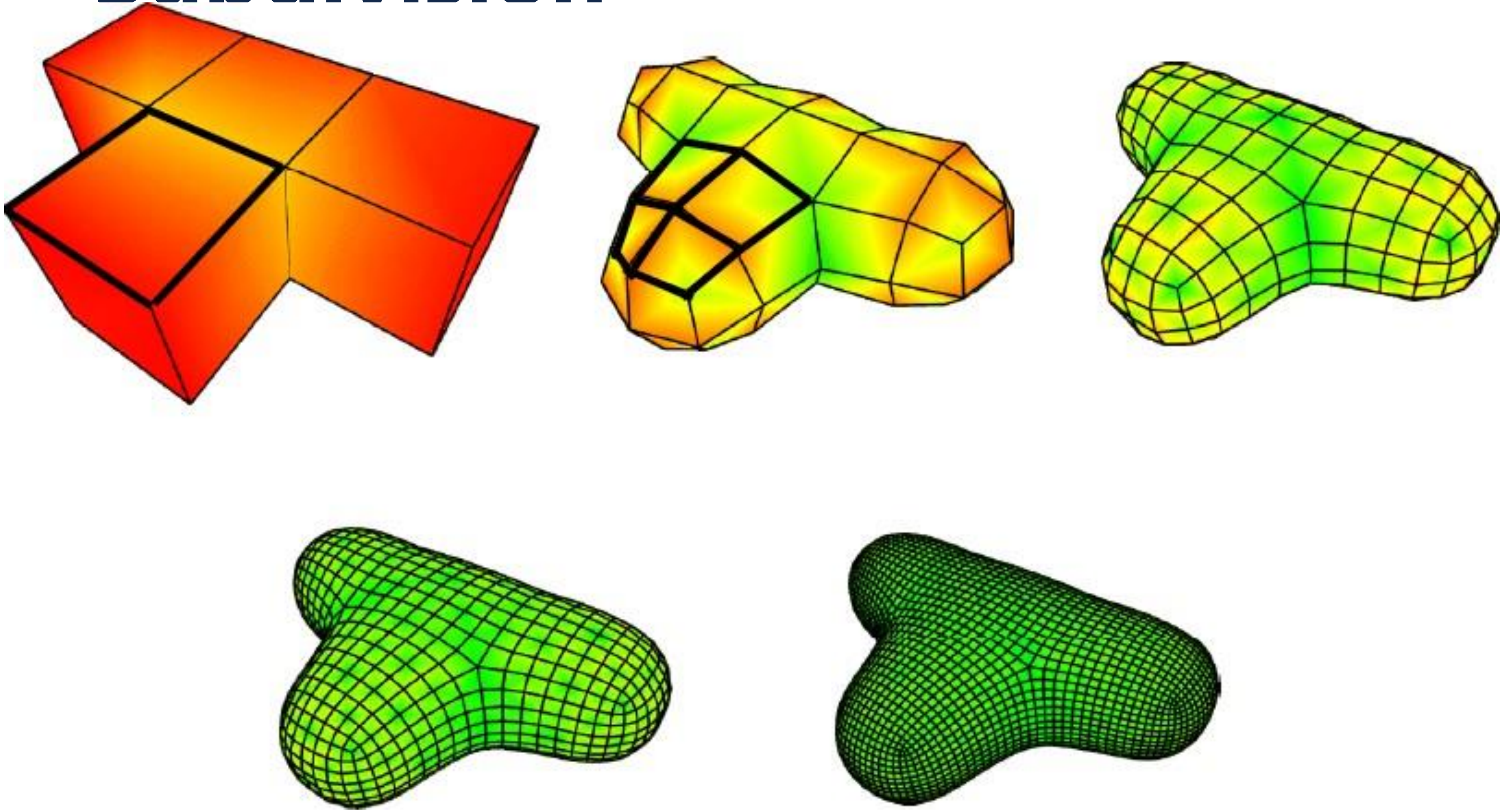
- (valence – 3) x original vertex position

Boundary edge points set to edge midpoints

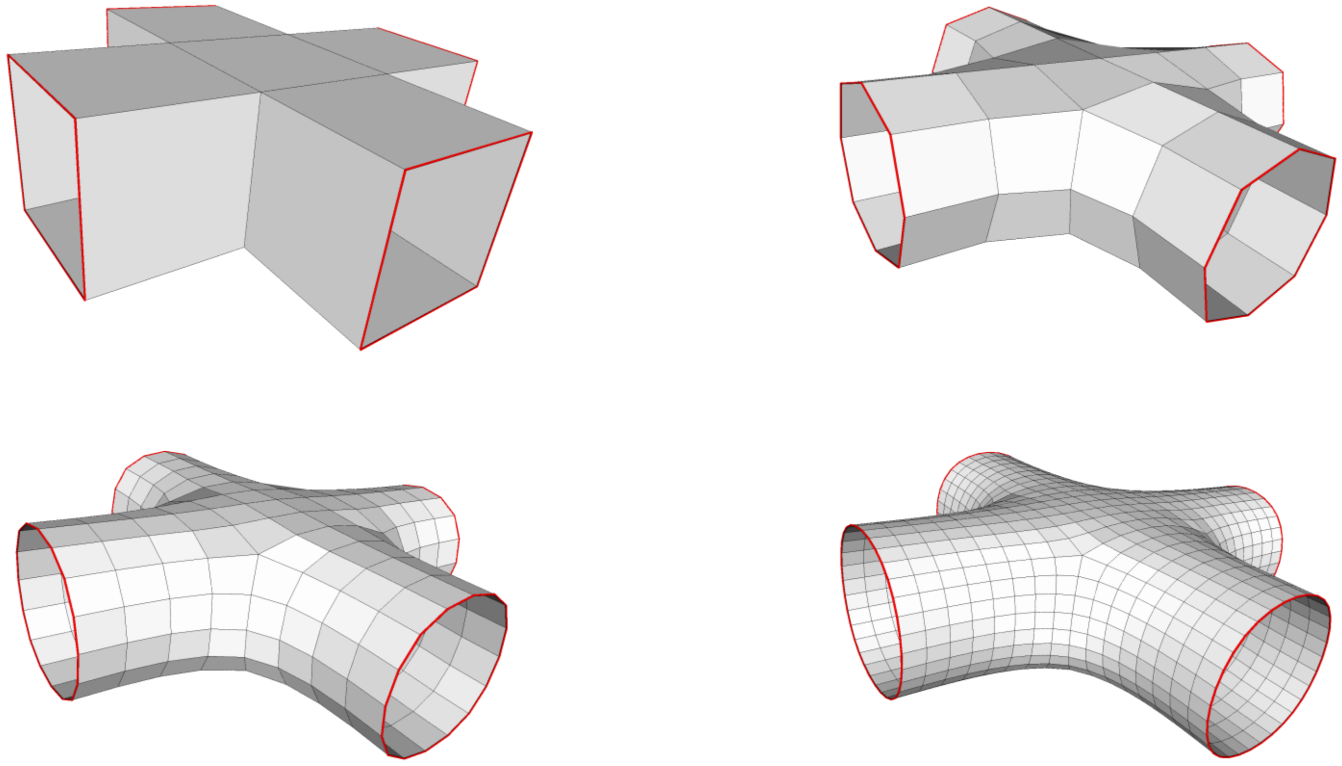
Boundary vertices stay put



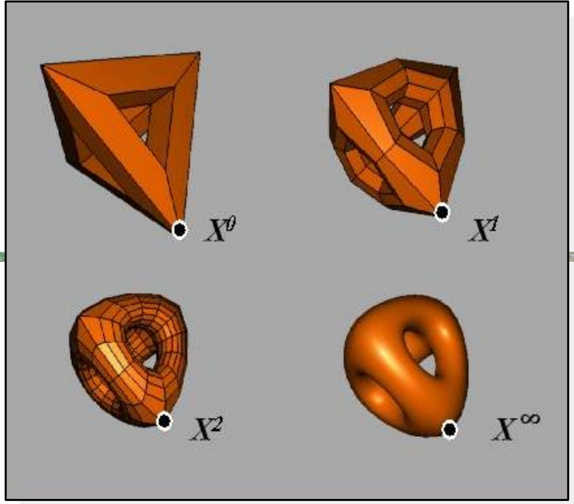
Catmull-Clark Subdivision



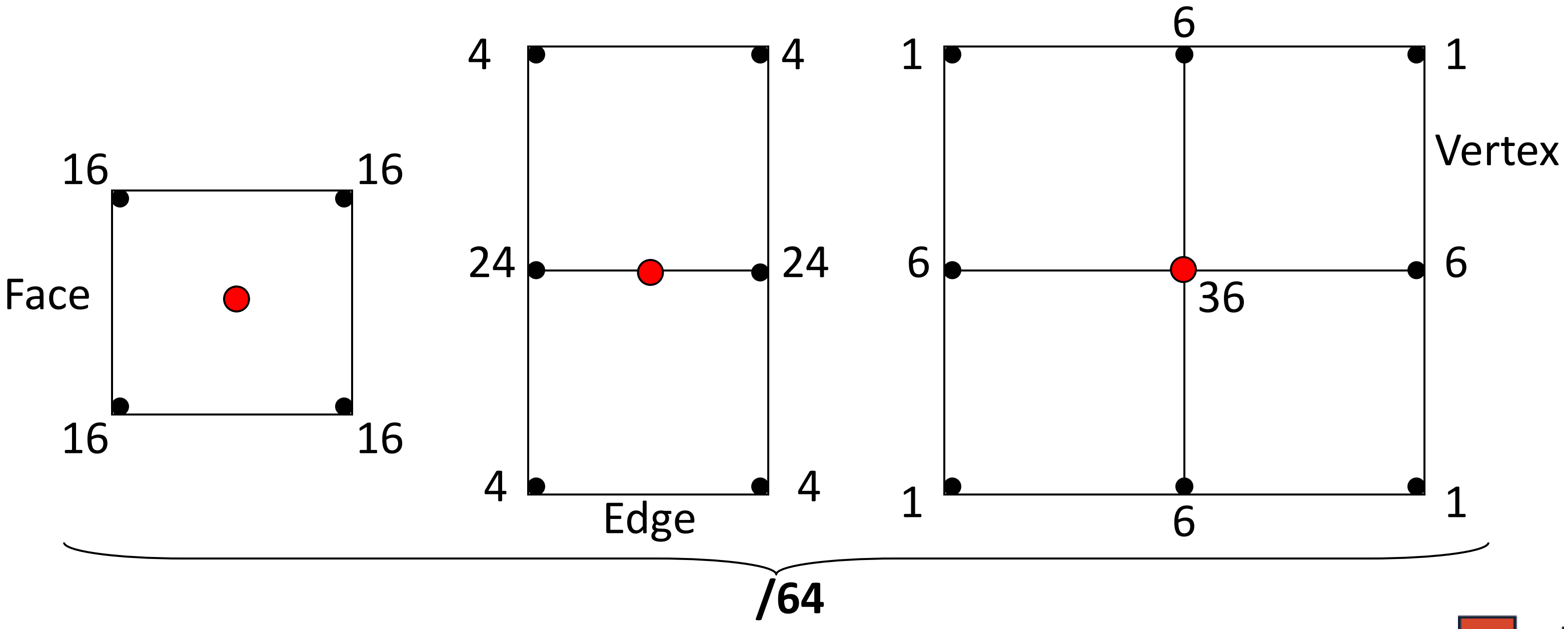
Catmull-Clark in action



Catmull-Clark



- *Catmull-Clark* is a bivariate approximating scheme
  - Limit surface is bicubic, C2-continuous.



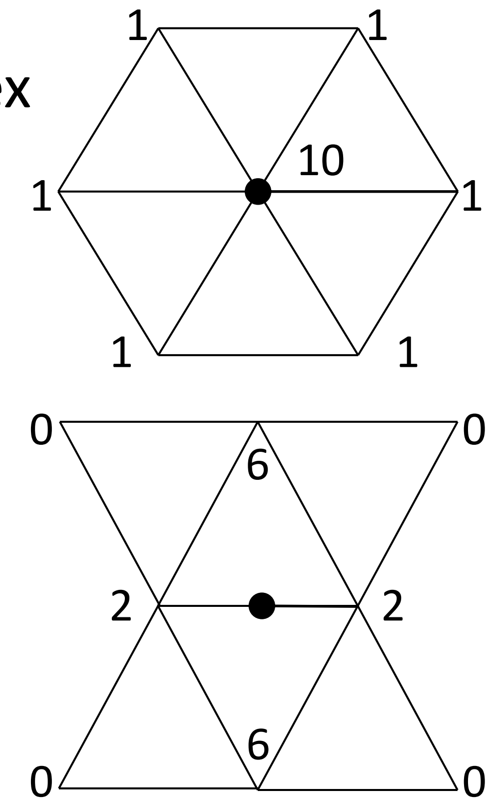
ILLINOIS

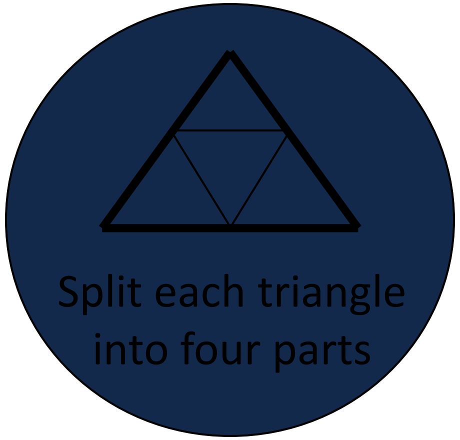
Schemes for simplicial (triangular) meshes

- *Loop* scheme:
- *Butterfly* scheme:

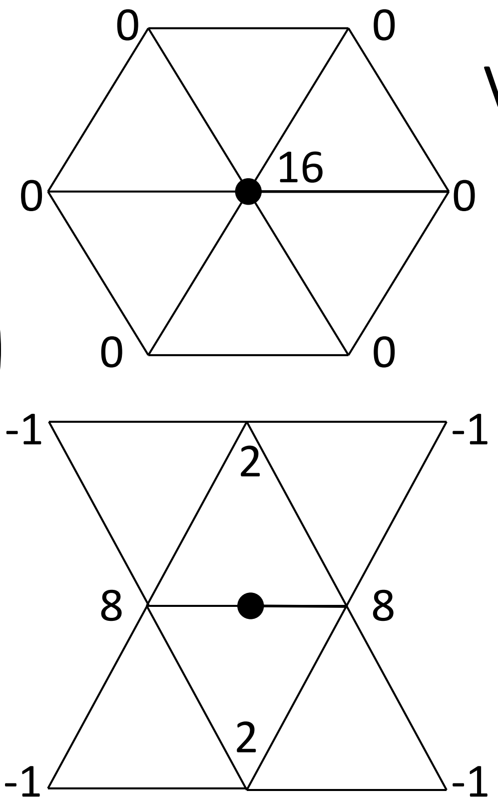
Vertex

Edge





(All weights are /16)

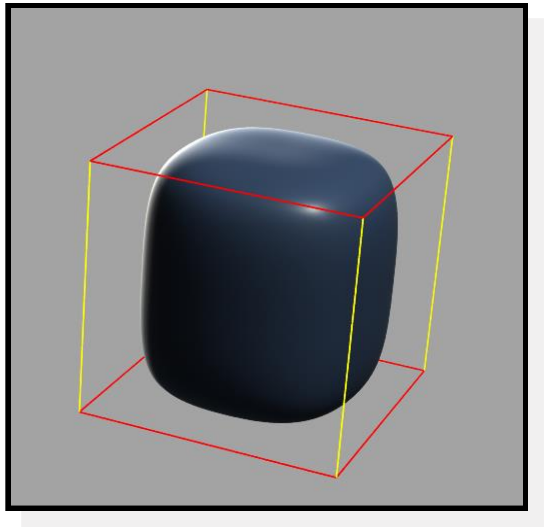


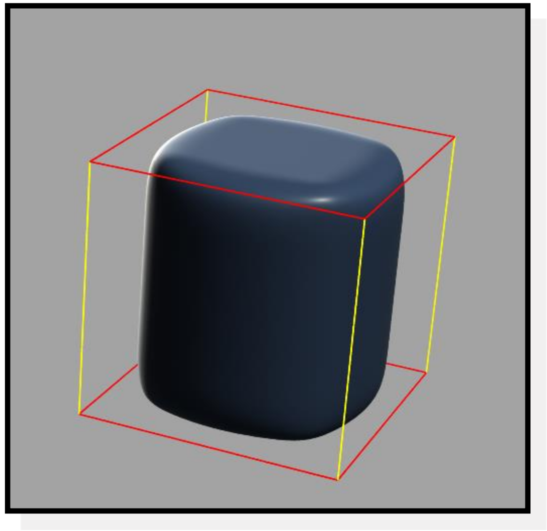
Vertex

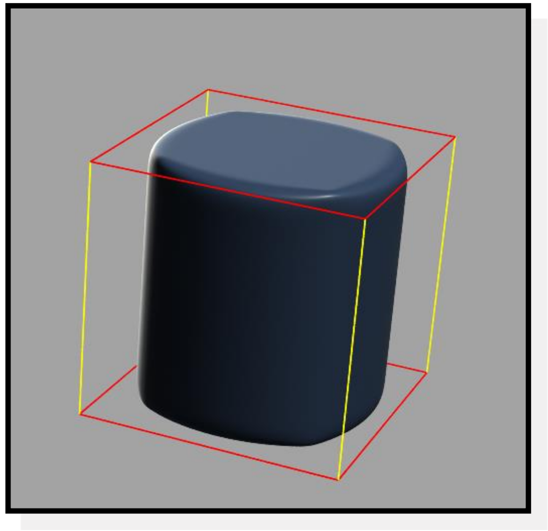
Edge

Creases

- Extensions exist for most schemes to support *creases*, vertices and edges flagged for partial or hybrid subdivision.

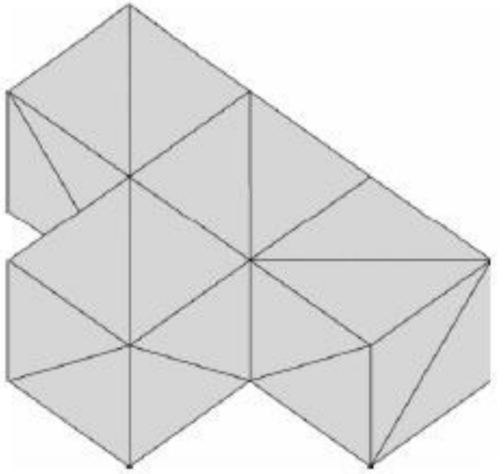


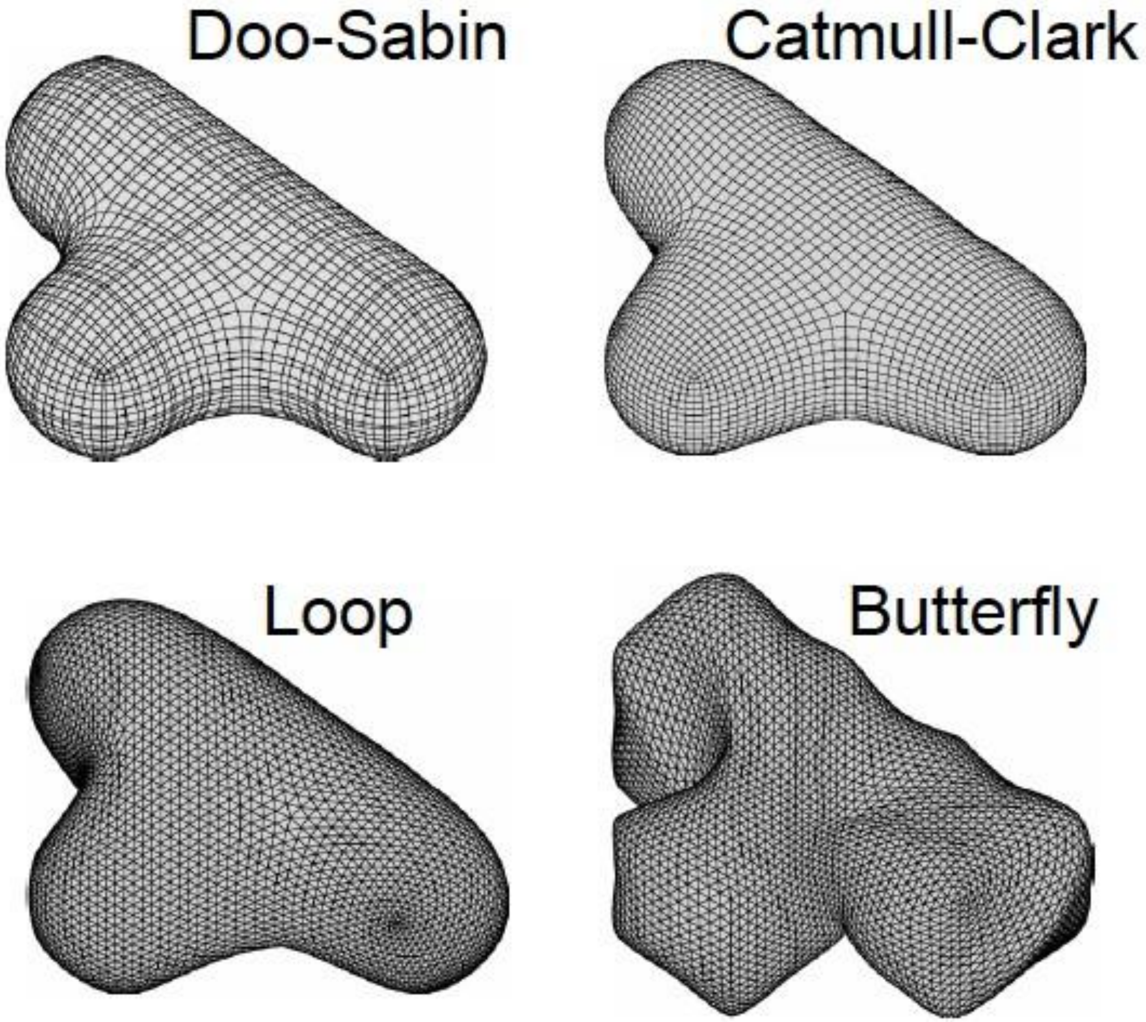




ILLINOIS

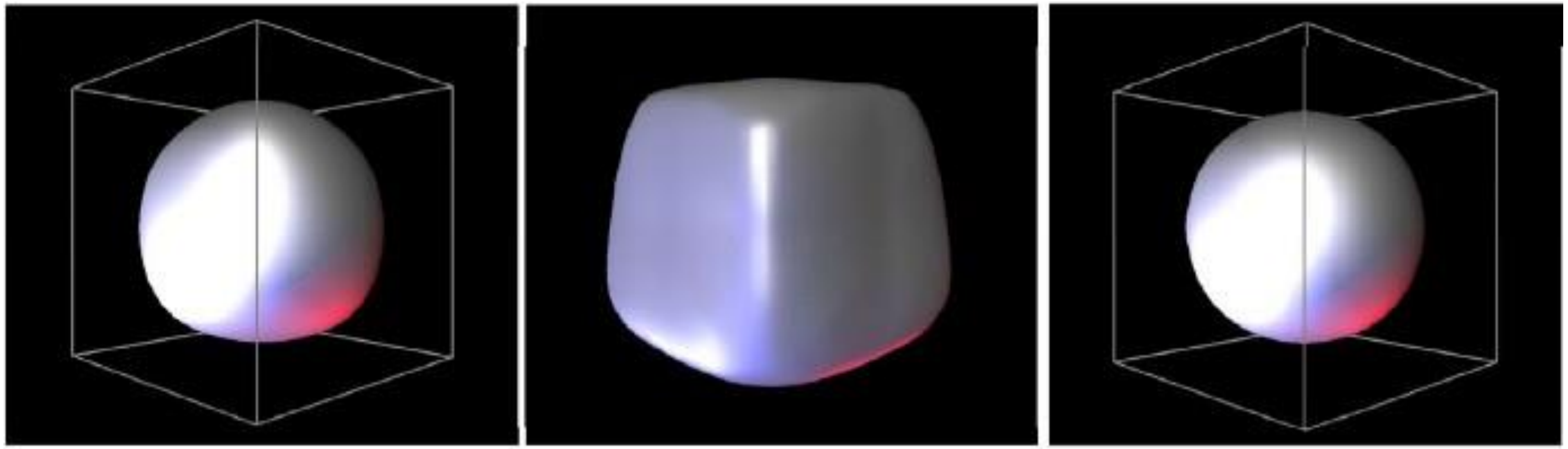
Comparison





Comparison

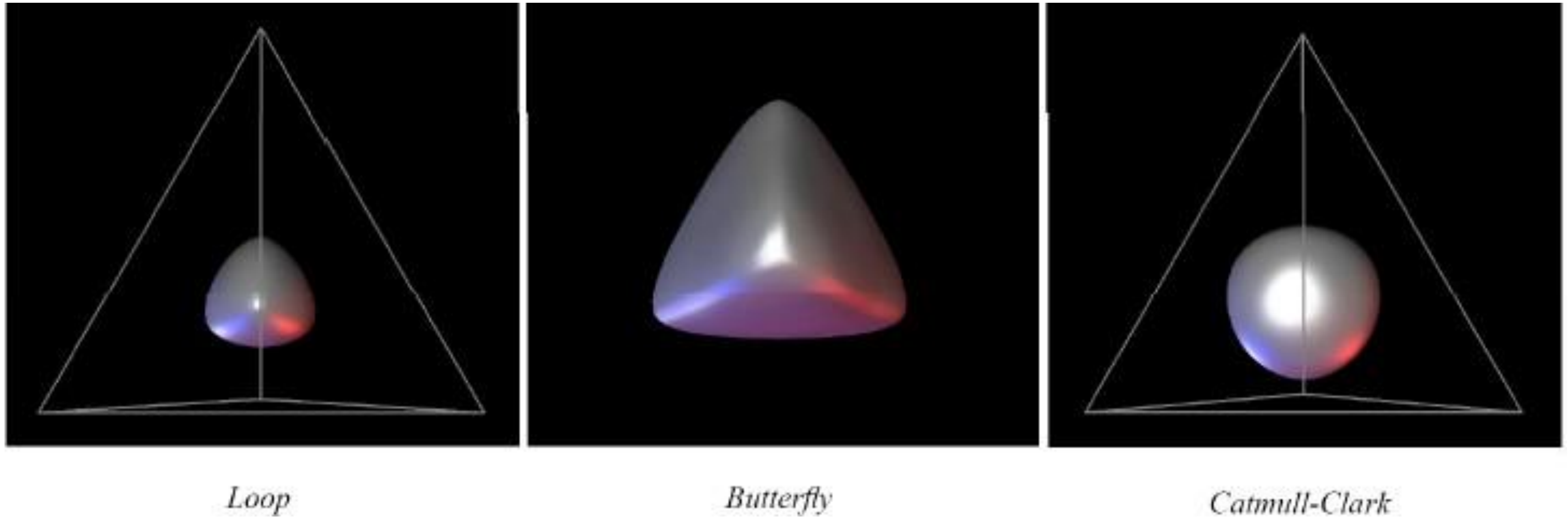
- Subdividing a cube
  - Loop result is assymetric, because cube was triangulated first
  - Both Loop and Catmull-Clark are better then Butterfly (C^2 vs. C^1)
  - Interpolation vs. smoothness



Loop *Butterfly* *Catmull-Clark*

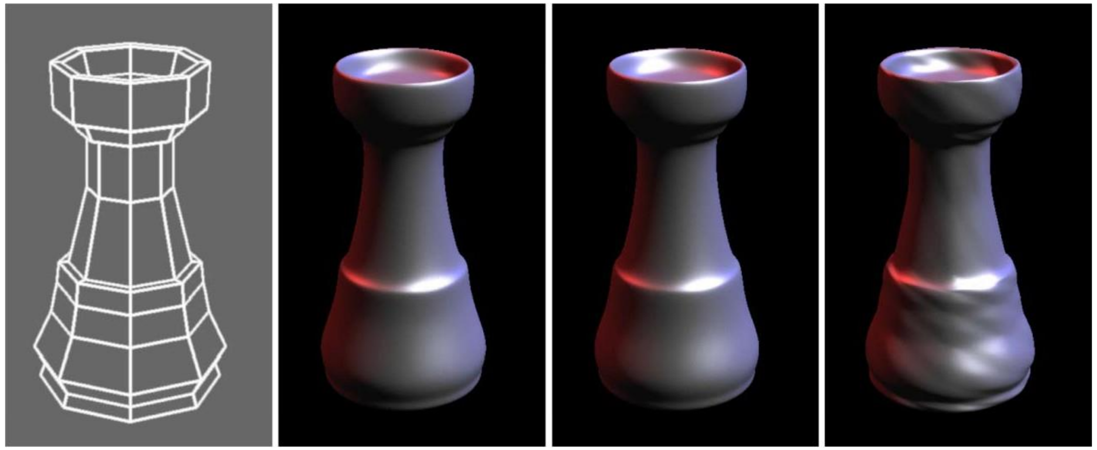
Comparison

- Subdividing a tetrahedron
  - Same insights
  - Severe shrinking for approximating schemes



So Who Wins?

- Loop and Catmull-Clark best when interpolation is not required
- Loop best for triangular meshes
- Catmull-Clark best for quad meshes
  - Don’t triangulate and then use Catmull-Clark



Initial mesh *Loop* *Catmull-Clark* *Catmull-Clark,after triangulation*