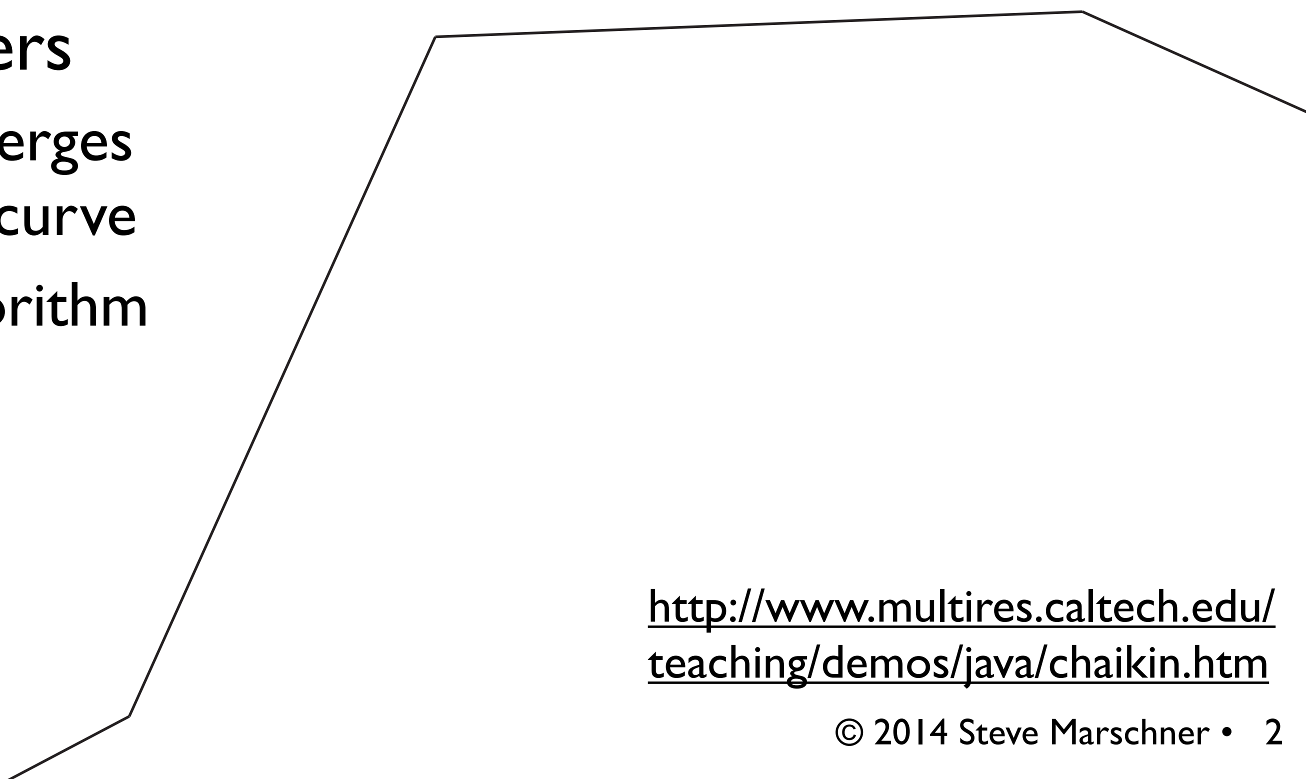
Subdivision overview

CS4620 Lecture 20

Introduction: corner cutting

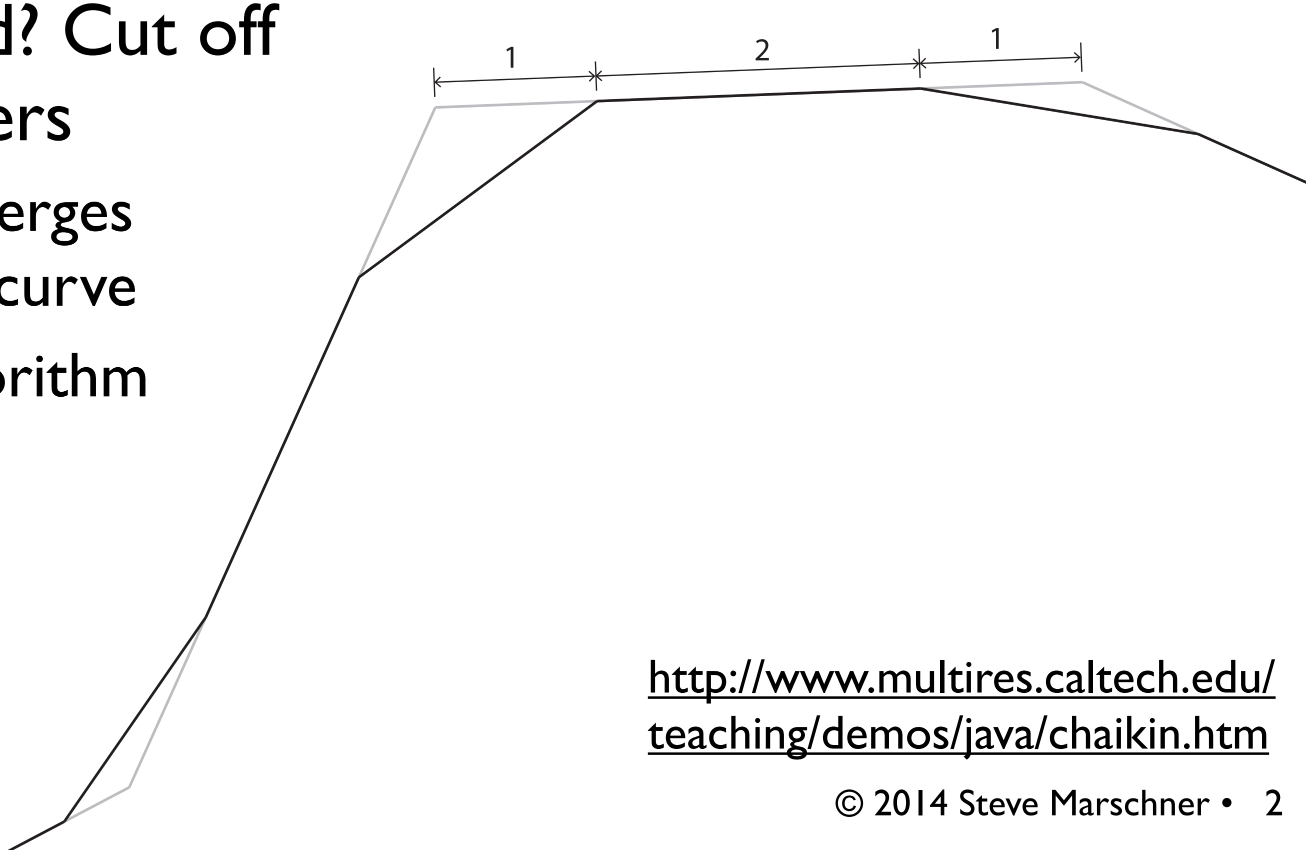
- Piecewise linear curve too jagged for you? Lop off the corners!
  - results in a curve with twice as many corners
- Still too jagged? Cut off the new corners
  - process converges to a smooth curve
  - Chaikin's algorithm



http://www.multires.caltech.edu/teaching/demos/java/chaikin.htm

Introduction: corner cutting

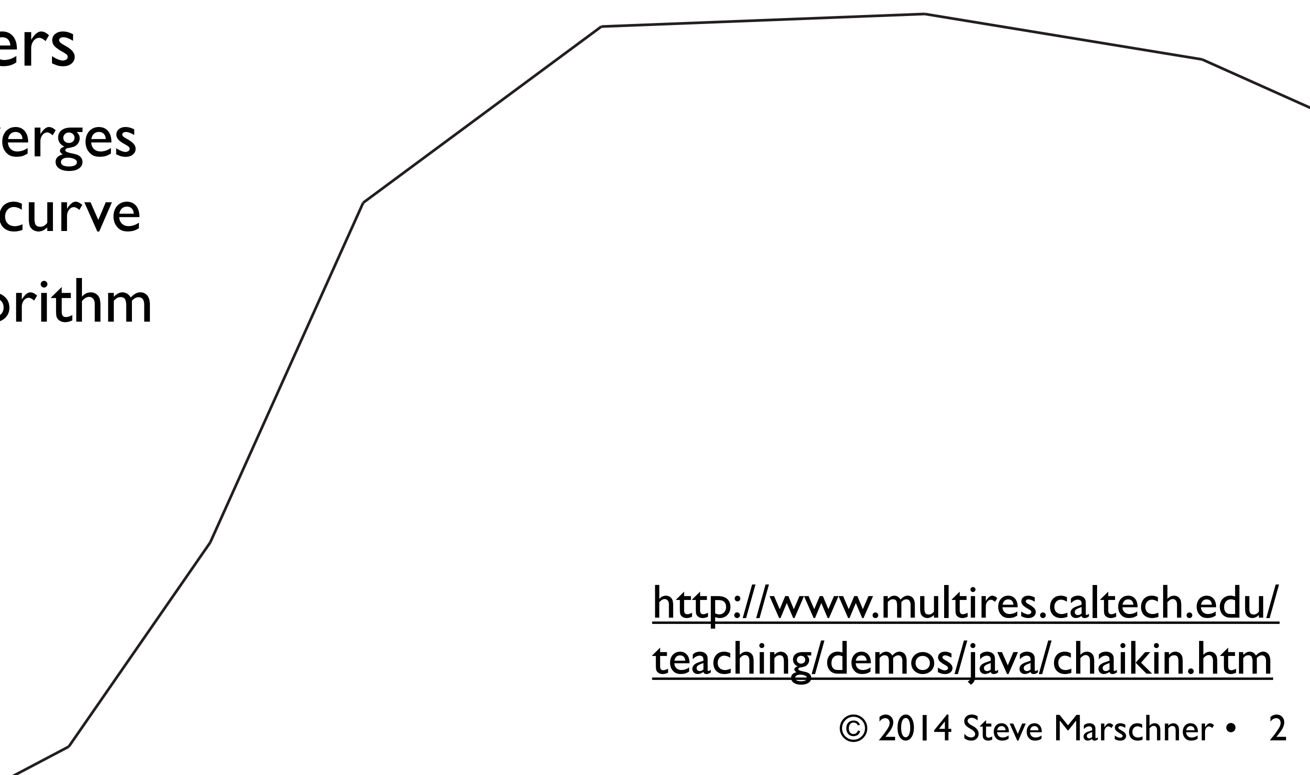
- Piecewise linear curve too jagged for you? Lop off the corners!
  - results in a curve with twice as many corners
- Still too jagged? Cut off the new corners
  - process converges to a smooth curve
  - Chaikin's algorithm



http://www.multires.caltech.edu/teaching/demos/java/chaikin.htm

Introduction: corner cutting

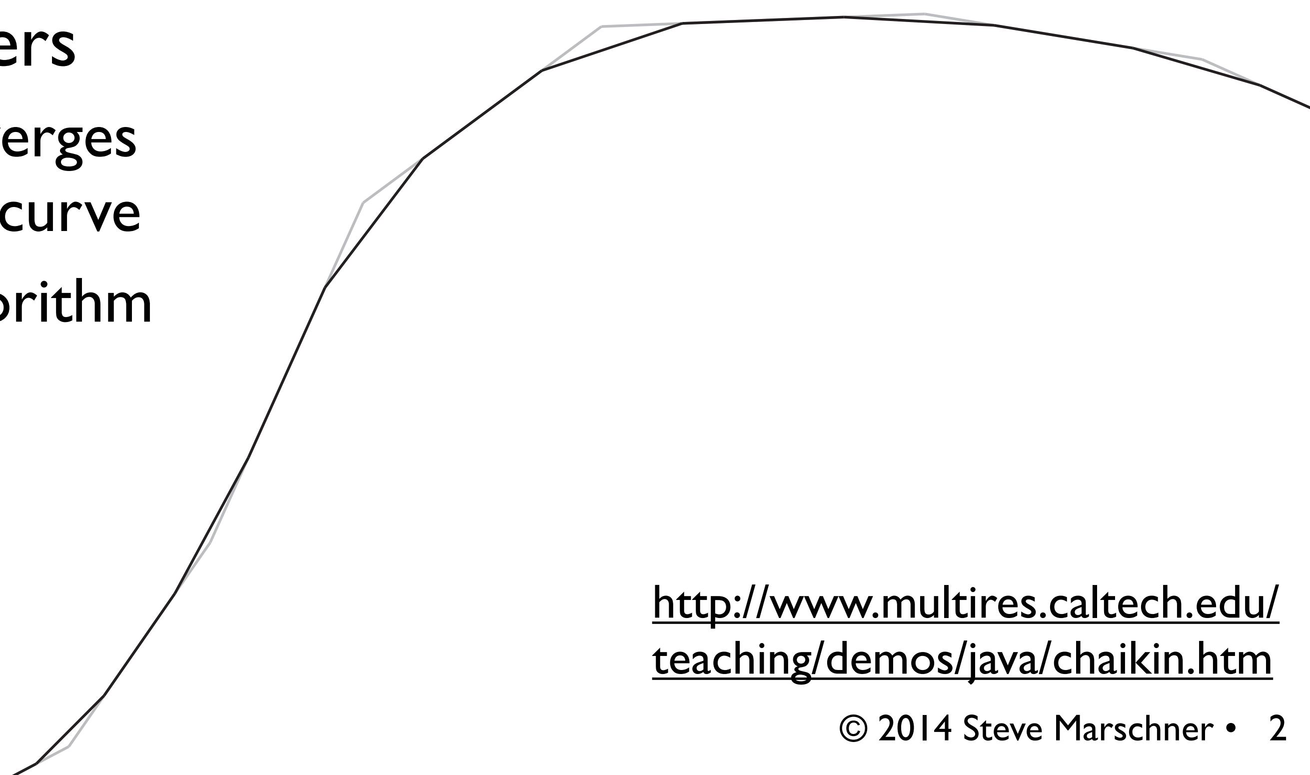
- Piecewise linear curve too jagged for you? Lop off the corners!
  - results in a curve with twice as many corners
- Still too jagged? Cut off the new corners
  - process converges to a smooth curve
  - Chaikin's algorithm



http://www.multires.caltech.edu/teaching/demos/java/chaikin.htm

Introduction: corner cutting

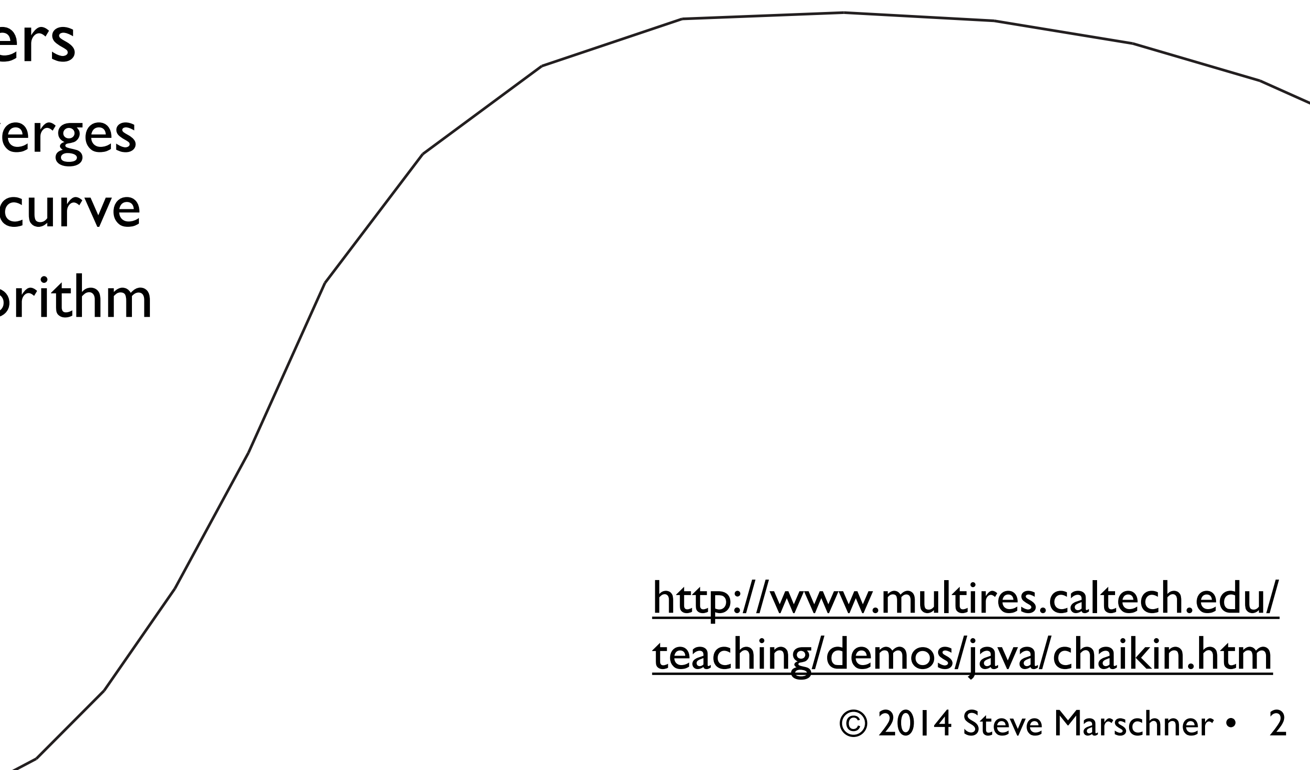
- Piecewise linear curve too jagged for you? Lop off the corners!
  - results in a curve with twice as many corners
- Still too jagged? Cut off the new corners
  - process converges to a smooth curve
  - Chaikin's algorithm



http://www.multires.caltech.edu/teaching/demos/java/chaikin.htm

Introduction: corner cutting

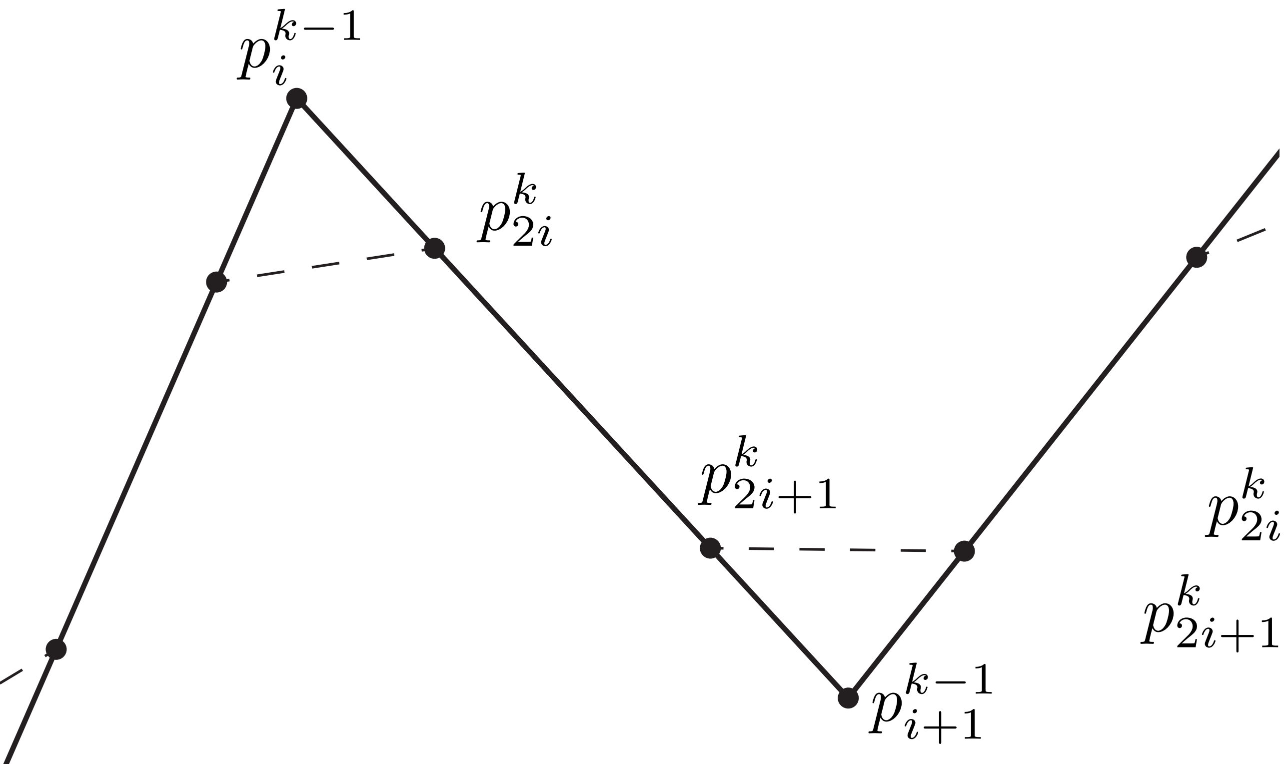
- Piecewise linear curve too jagged for you? Lop off the corners!
  - results in a curve with twice as many corners
- Still too jagged? Cut off the new corners
  - process converges to a smooth curve
  - Chaikin's algorithm



http://www.multires.caltech.edu/teaching/demos/java/chaikin.htm

Corner cutting in equations

- New points are linear combinations of old ones
- Different treatment for odd-numbered and even-numbered points.



$$p_{2i}^k = (3p_i^{k-1} + p_{i+1}^{k-1})/4$$
$$p_{2i+1}^k = (p_i^{k-1} + 3p_{i+1}^{k-1})/4$$

Spline-splitting math for B-splines

- Can use spline-matrix math from previous lecture to split a B-spline segment in two at $s = t = 0.5$.
- Result is especially nice because the rules for adjacent segments agree (not true for all splines).

$$S_L = \begin{bmatrix} s^3 & & & \\ & s^2 & & \\ & & s & \\ & & & 1 \end{bmatrix}$$

$$P_L = M^{-1} S_L M P$$
$$P_R = M^{-1} S_R M P$$

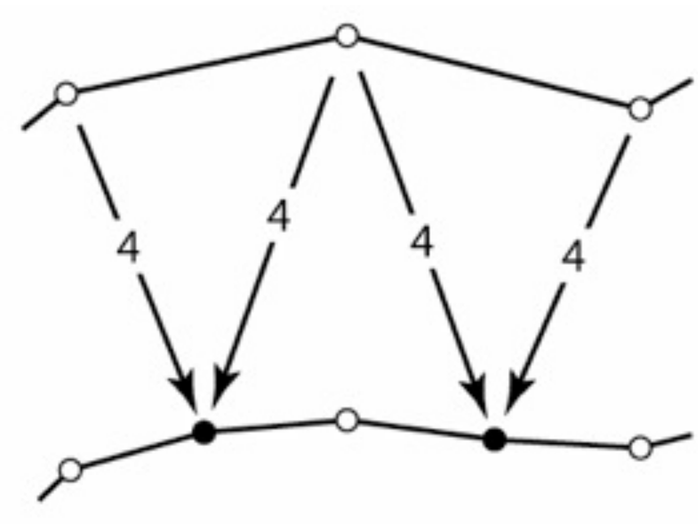
$$P_L = \begin{bmatrix} 4 & 4 & 0 & 0 \\ 1 & 6 & 1 & 0 \\ 0 & 4 & 4 & 0 \\ 0 & 1 & 6 & 1 \end{bmatrix}$$

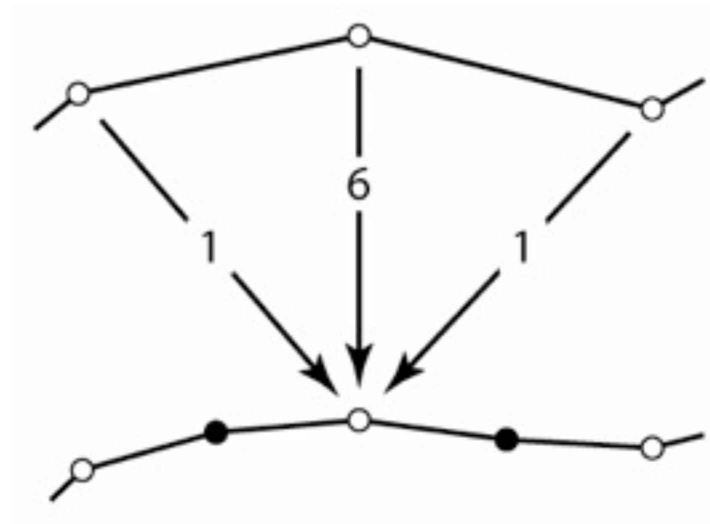
$$S_R = \begin{bmatrix} s^3 & & & \\ 3s^2(1-s) & s^2 & & \\ 3s(1-s)^2 & 2s(1-s) & s & \\ (1-s)^3 & (1-s)^2 & (1-s) & 1 \end{bmatrix}$$

$$P_R = \begin{bmatrix} 1 & 6 & 1 & 0 \\ 0 & 4 & 4 & 0 \\ 0 & 1 & 6 & 1 \\ 0 & 0 & 4 & 4 \end{bmatrix}$$

Subdivision for B-splines

- Control vertices of refined spline are linear combinations of the c.v.s of the coarse spline

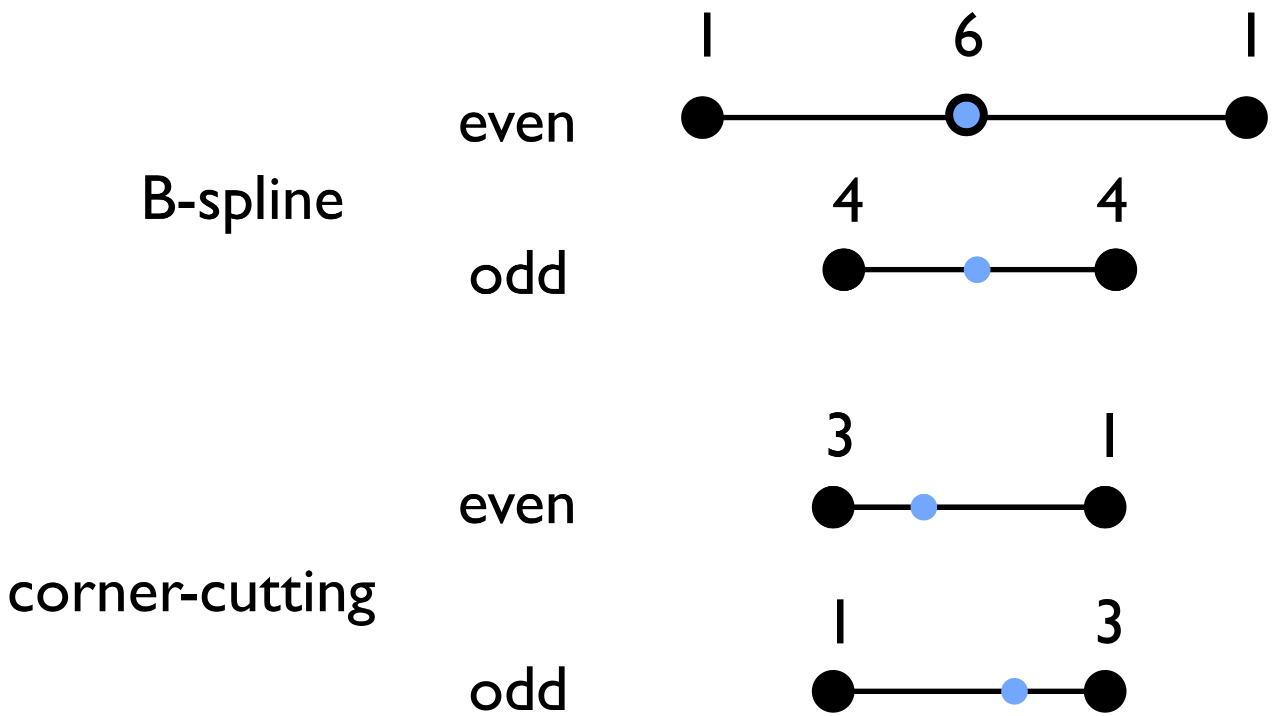




ODD **EVEN**

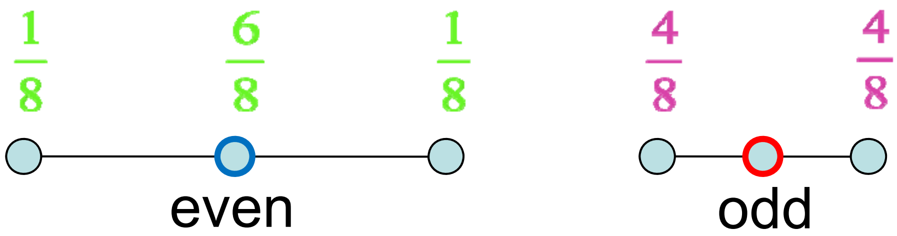
Drawing a picture of the rule

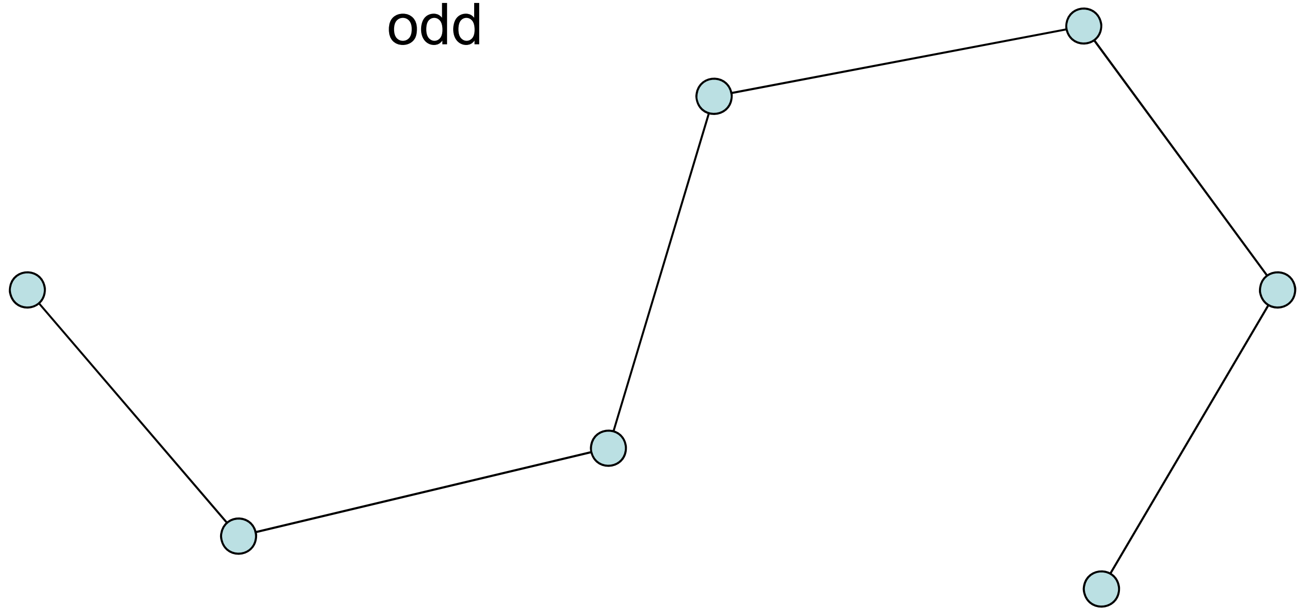
- Conventionally illustrate subdivision rules as a "mask" that you match against the neighborhood
  - often implied denominator = sum of weights



[Stanford CS468 Fall 2010 slides]

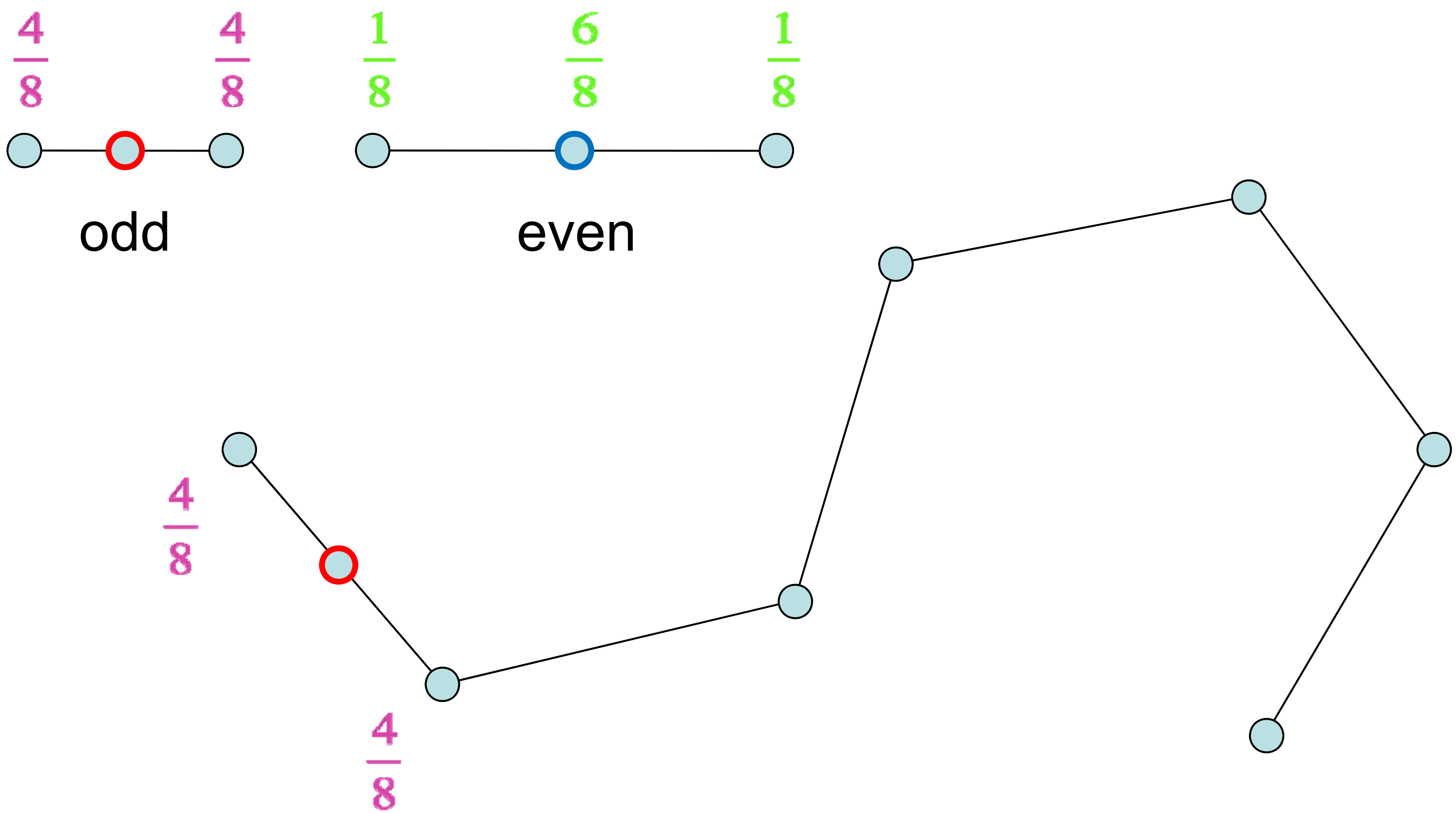
Cubic B-Spline





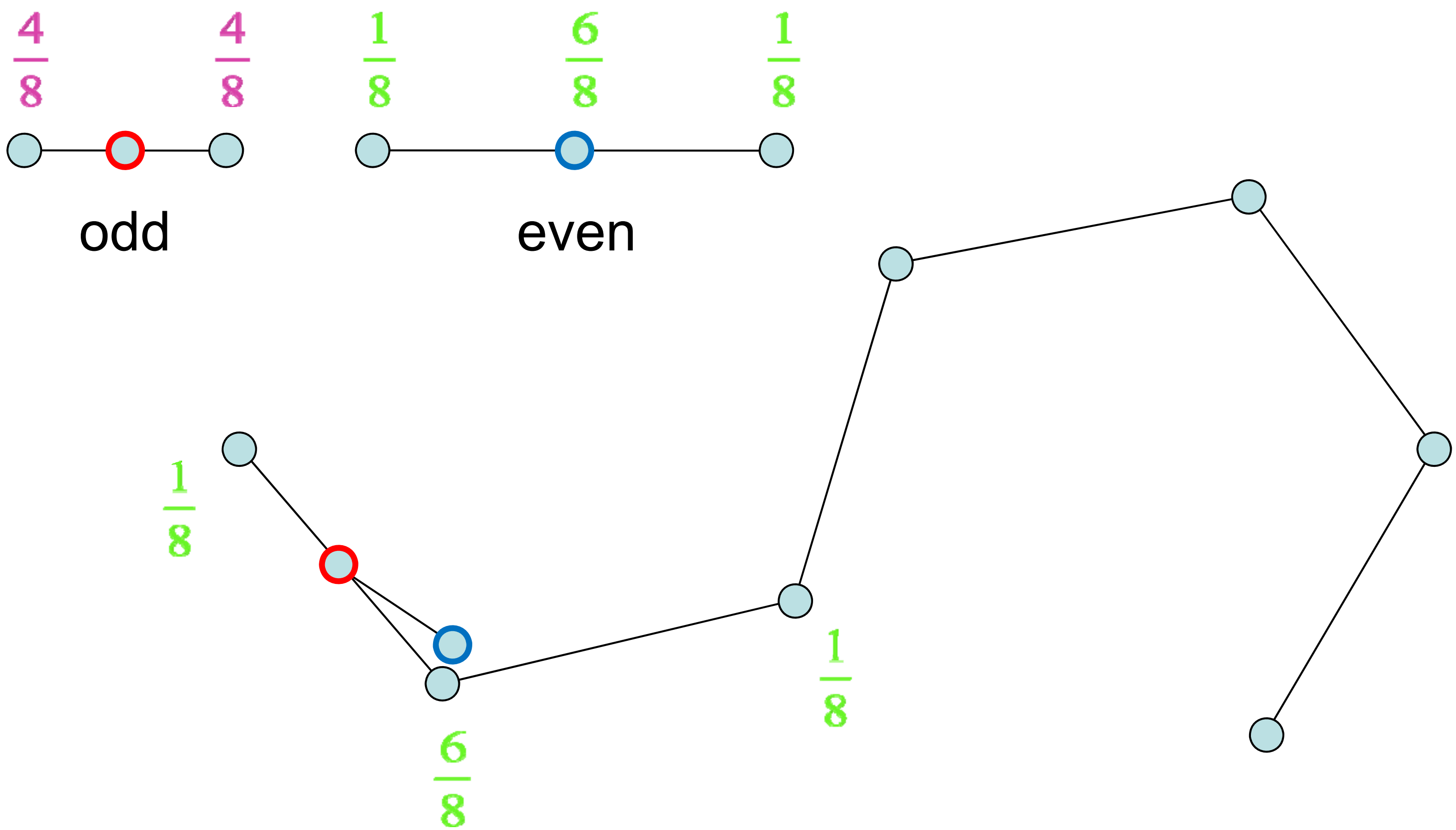
[Stanford CS468 Fall 2010 slides]

Cubic B-Spline



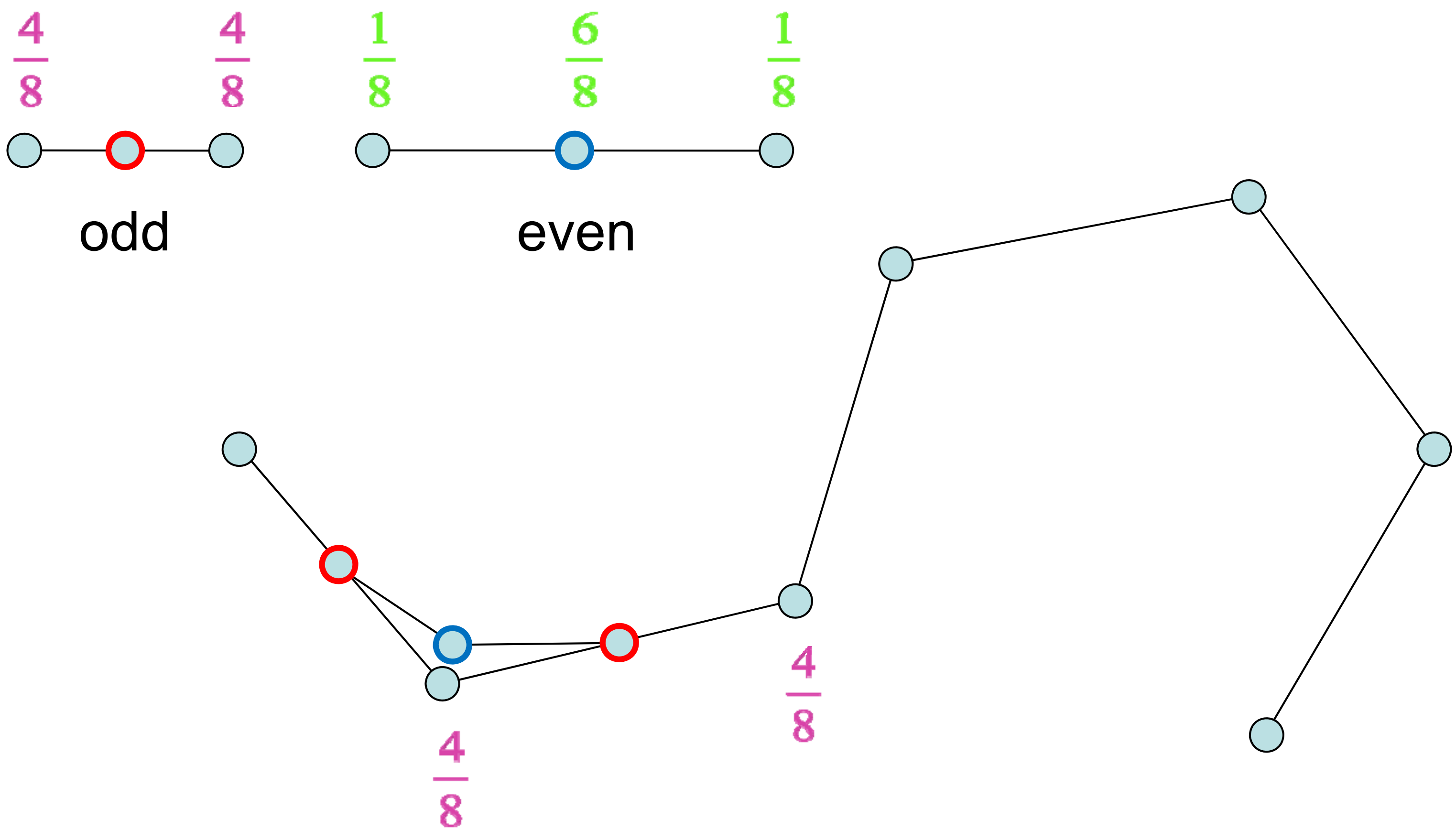
[Stanford CS468 Fall 2010 slides]

Cubic B-Spline



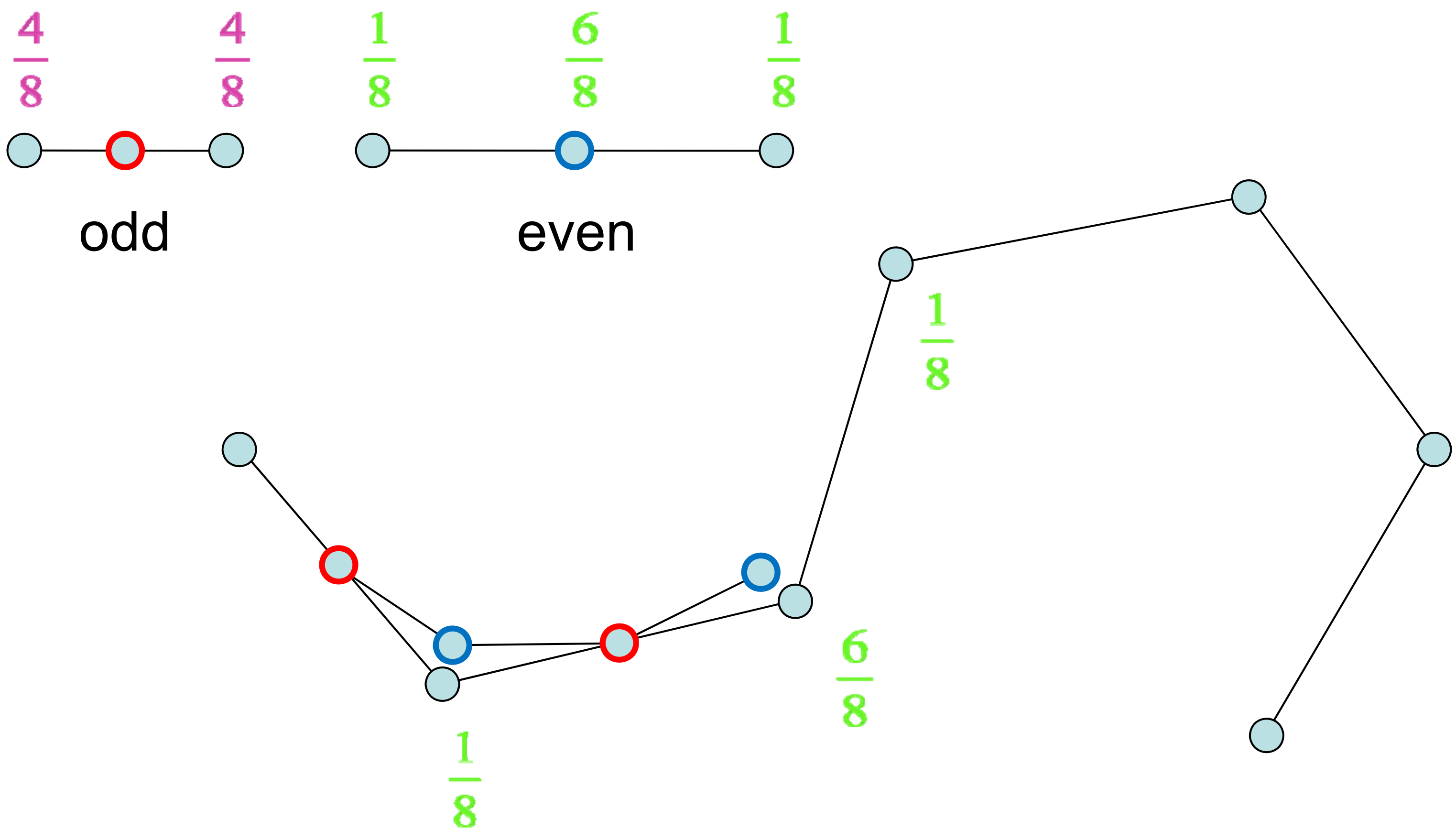
[Stanford CS468 Fall 2010 slides]

Cubic B-Spline



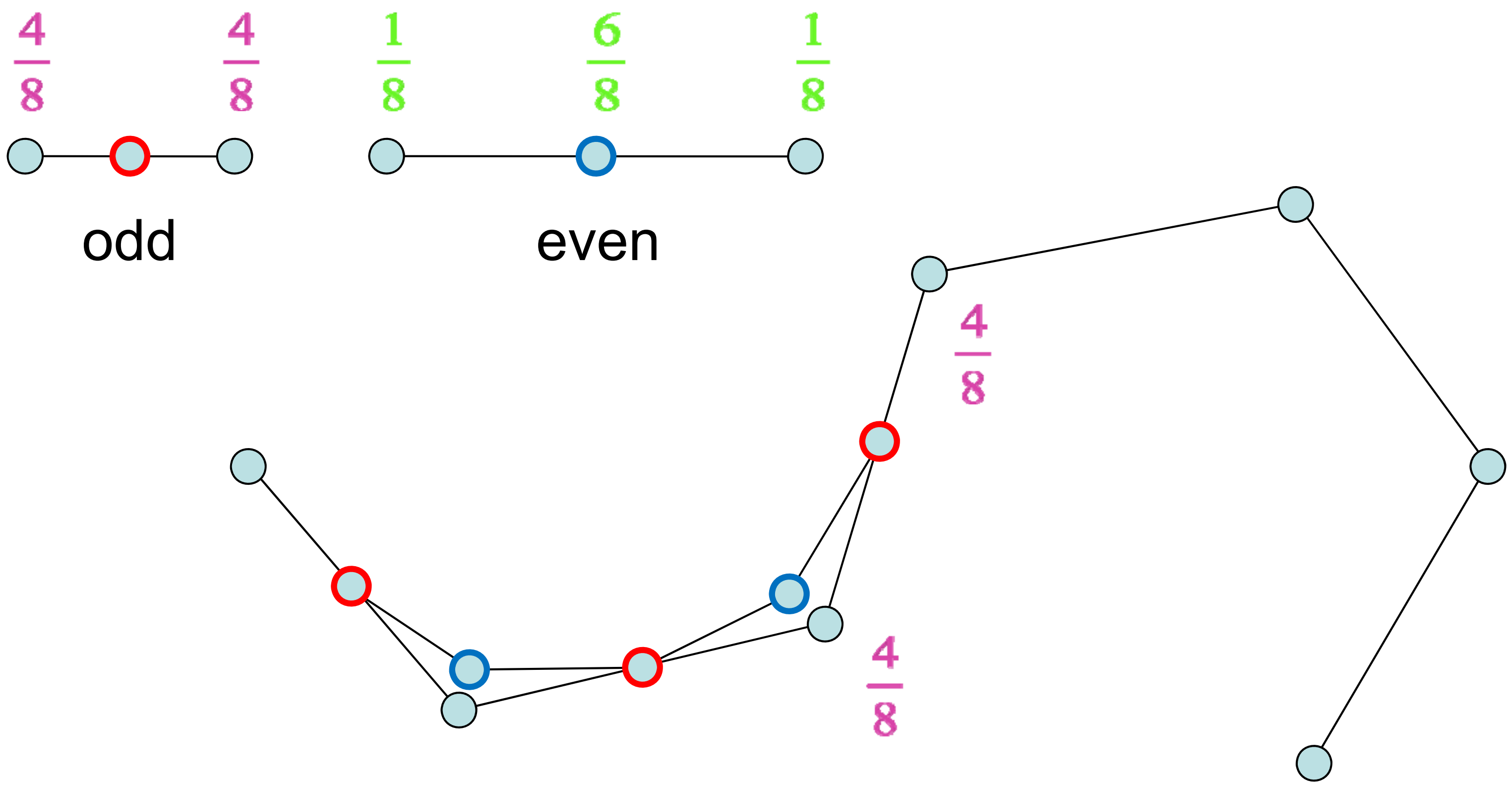
[Stanford CS468 Fall 2010 slides]

Cubic B-Spline



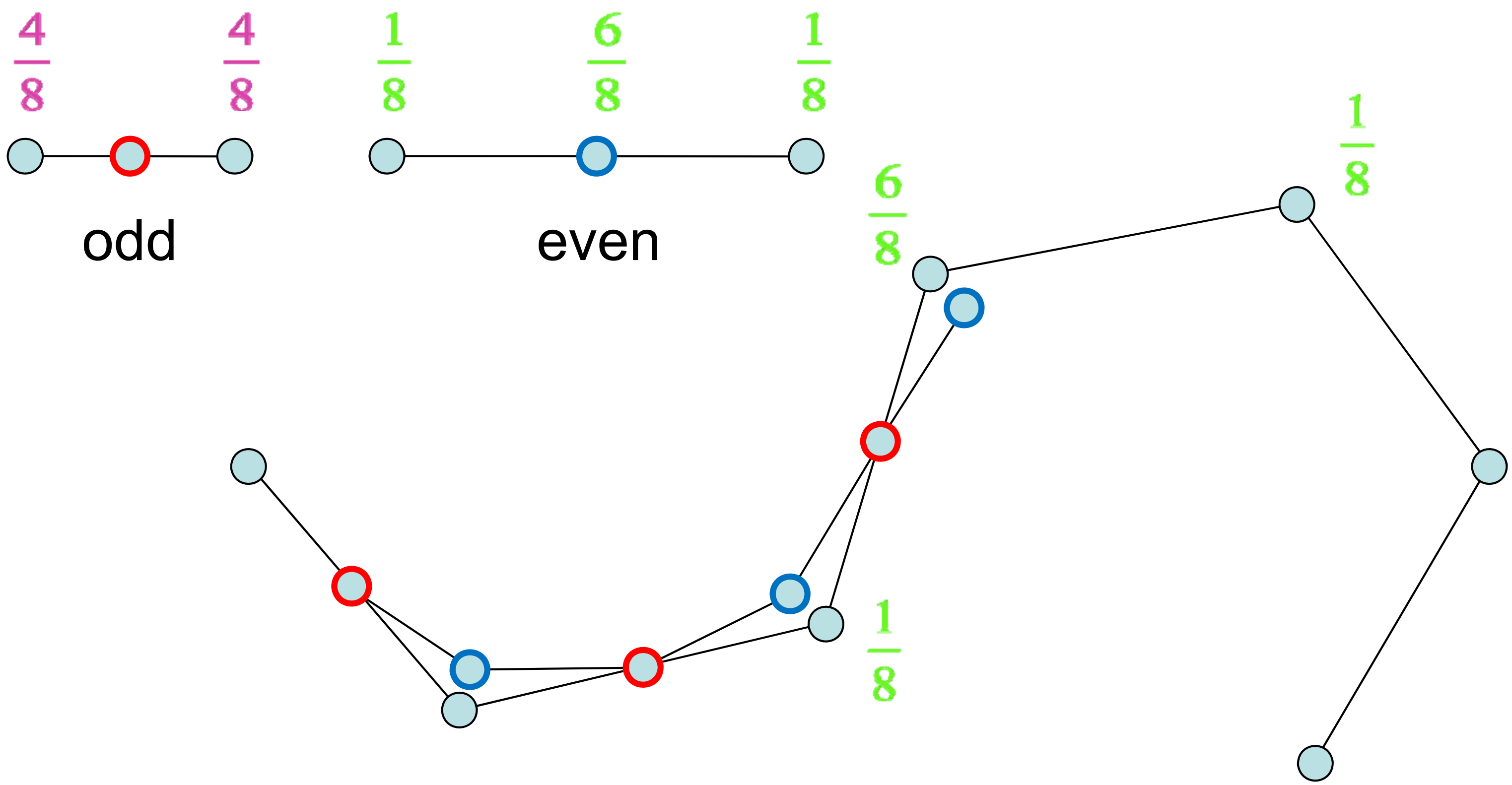
[Stanford CS468 Fall 2010 slides]

Cubic B-Spline



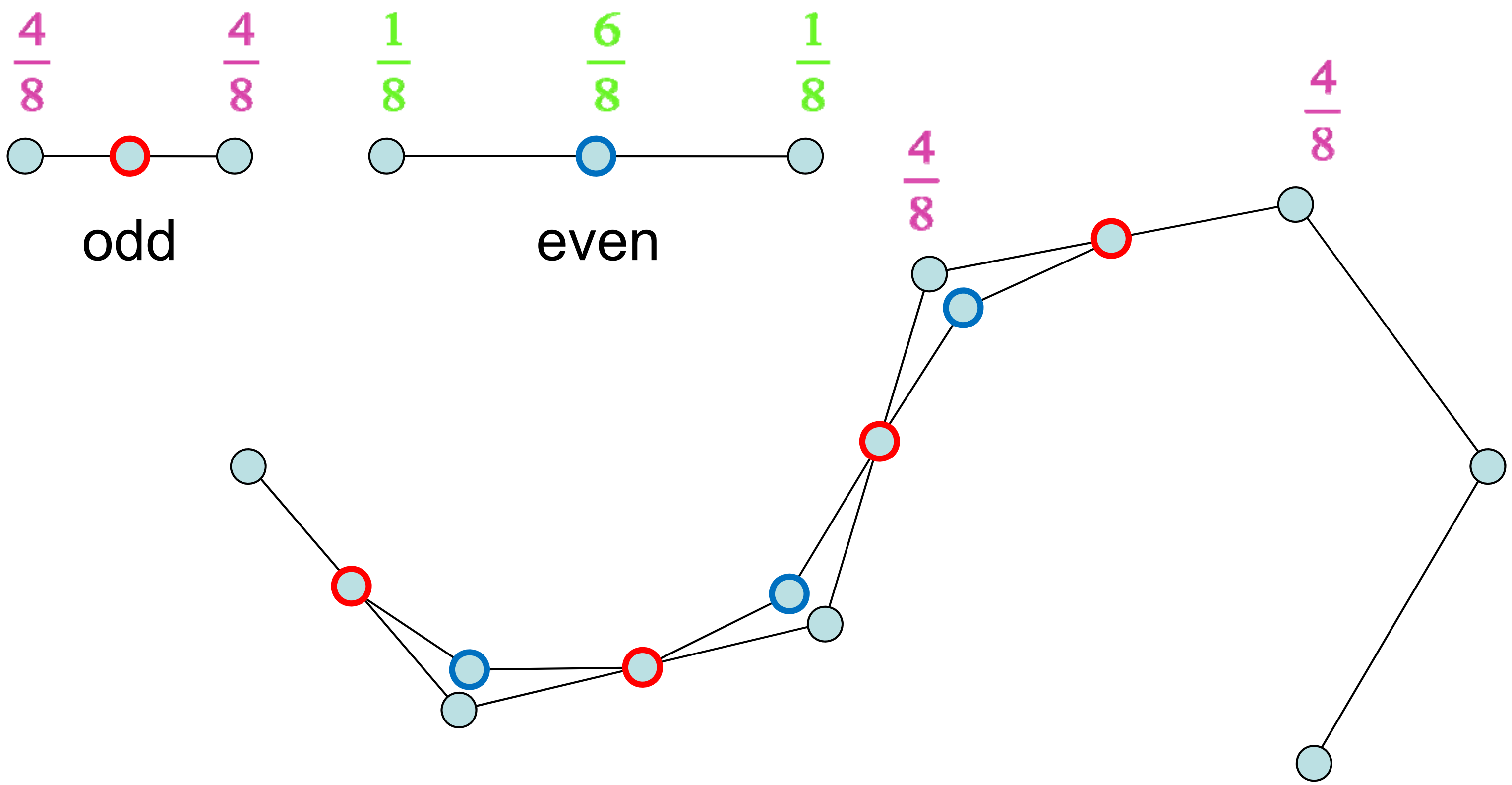
[Stanford CS468 Fall 2010 slides]

Cubic B-Spline



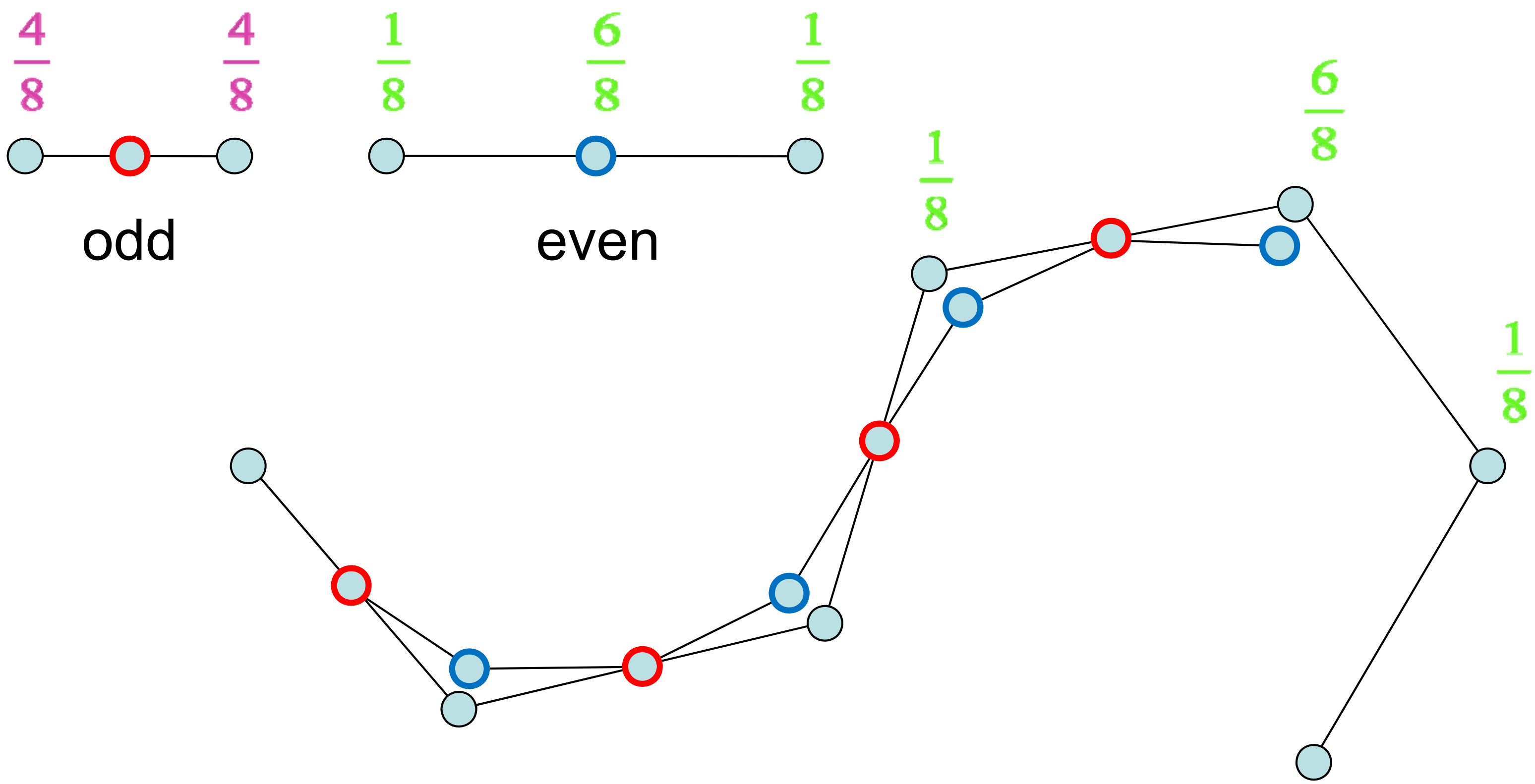
[Stanford CS468 Fall 2010 slides]

Cubic B-Spline



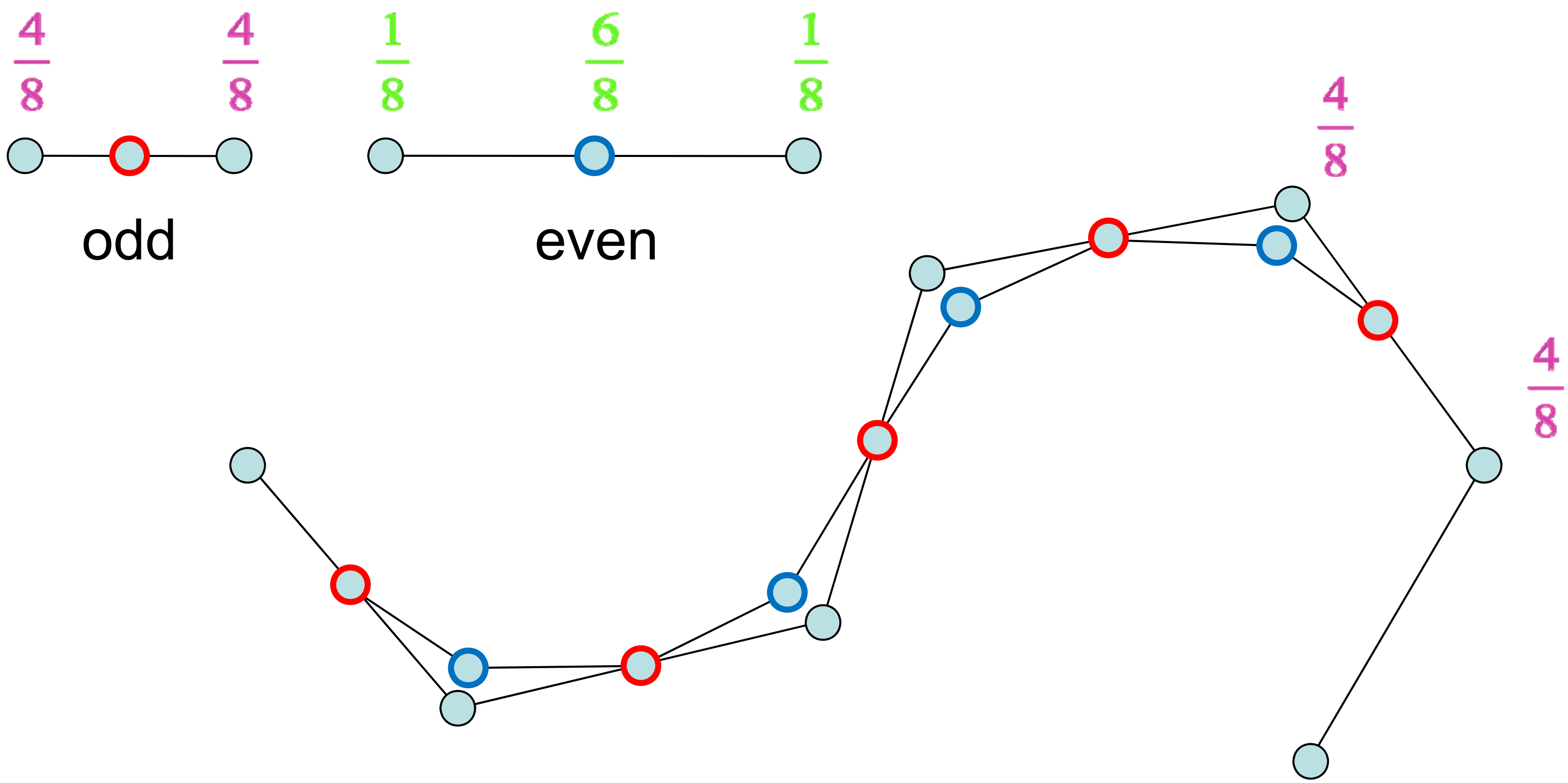
[Stanford CS468 Fall 2010 slides]

Cubic B-Spline



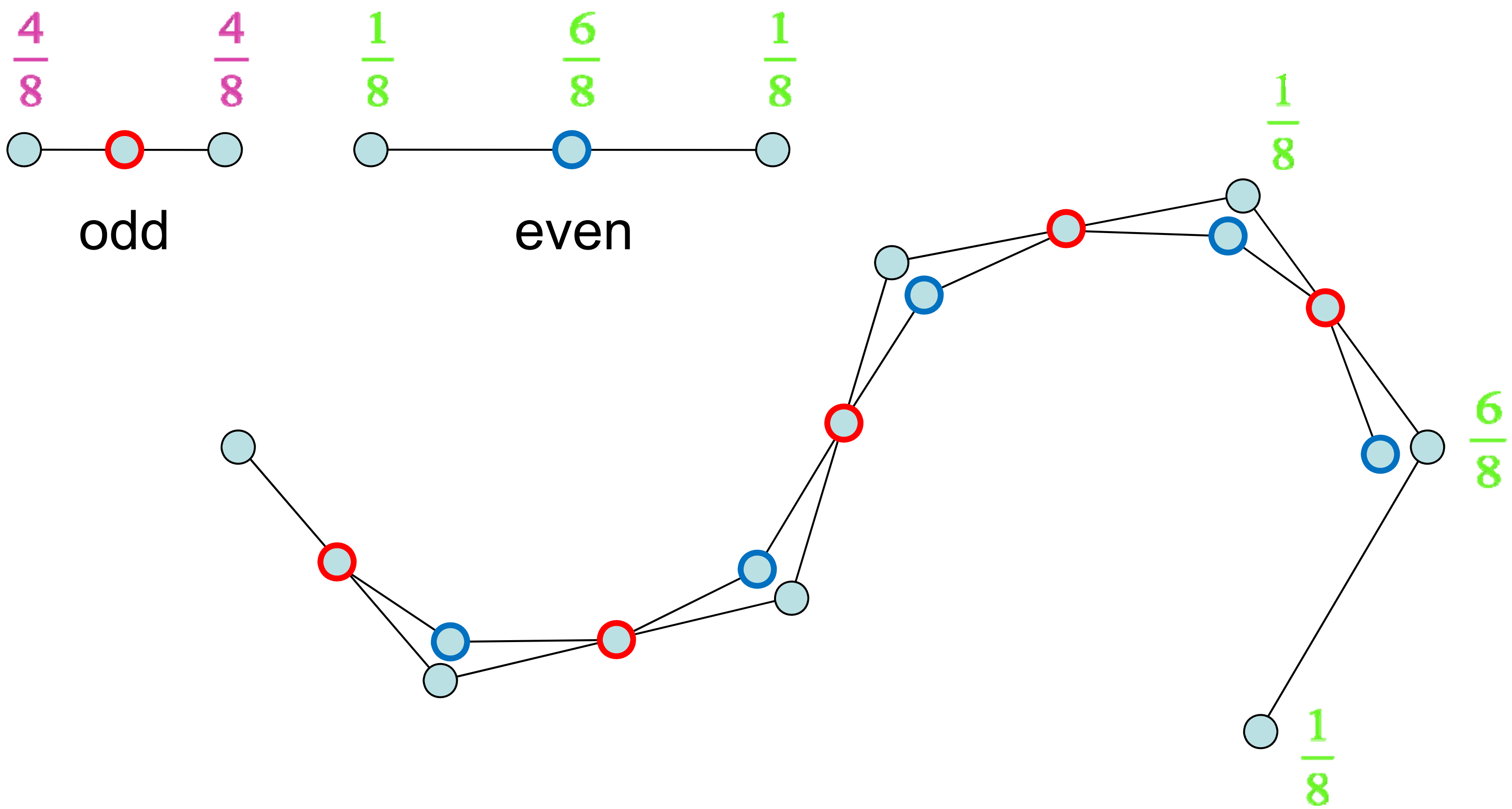
[Stanford CS468 Fall 2010 slides]

Cubic B-Spline



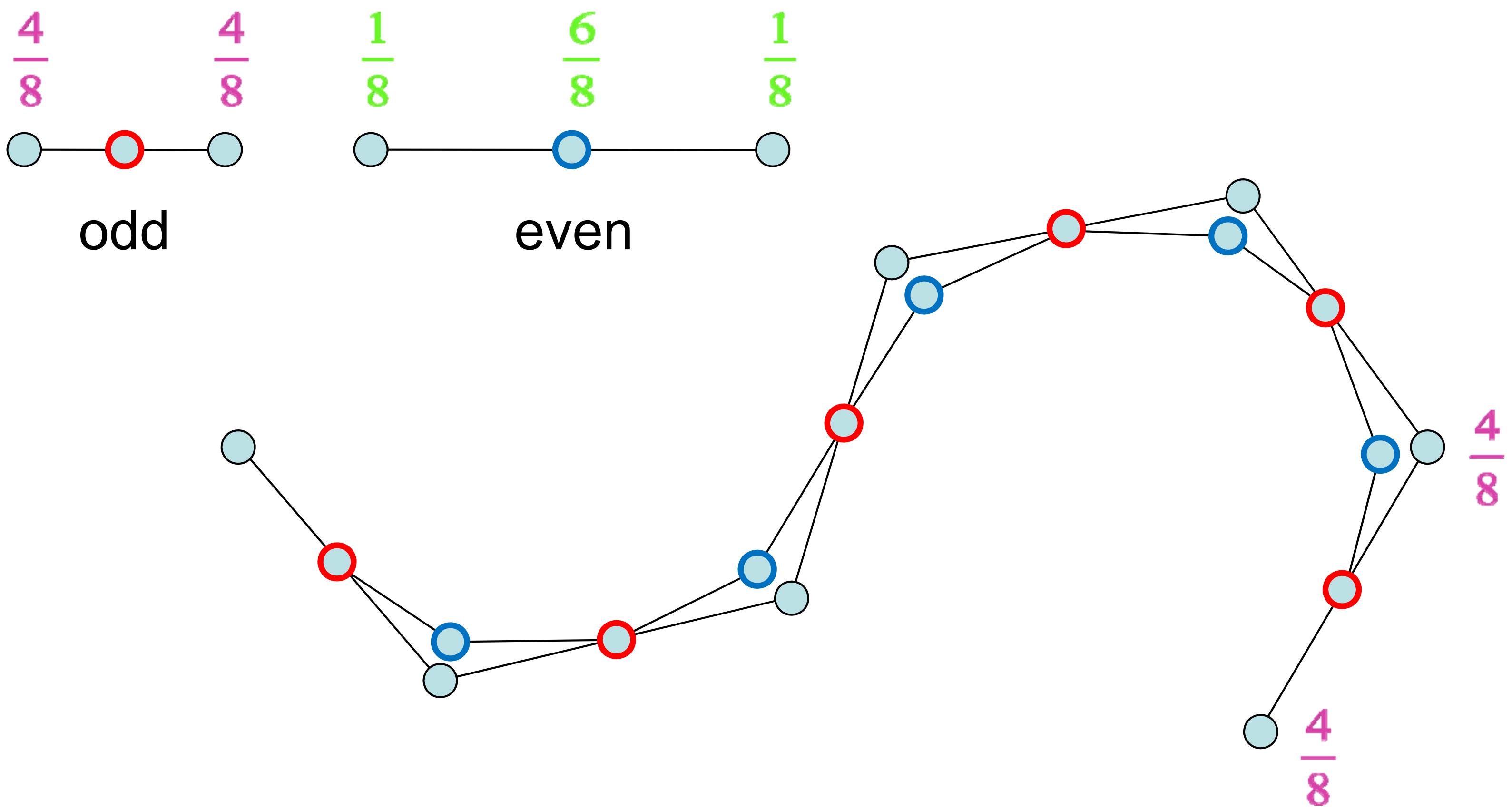
[Stanford CS468 Fall 2010 slides]

Cubic B-Spline



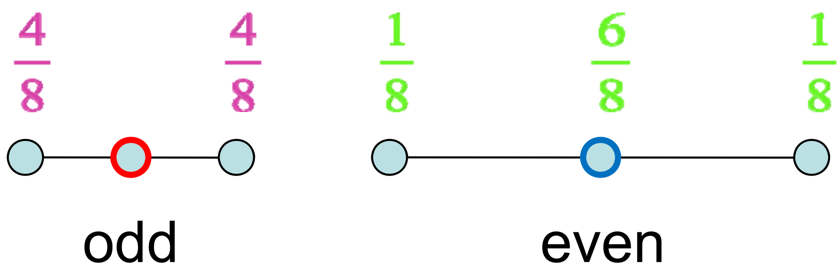
[Stanford CS468 Fall 2010 slides]

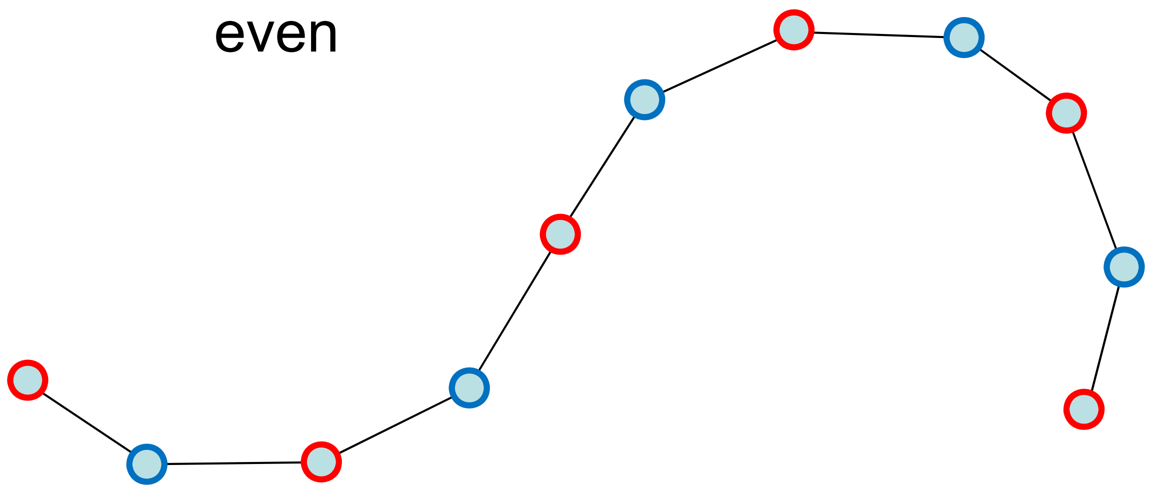
Cubic B-Spline



[Stanford CS468 Fall 2010 slides]

Cubic B-Spline





[Stanford CS468 Fall 2010 slides]

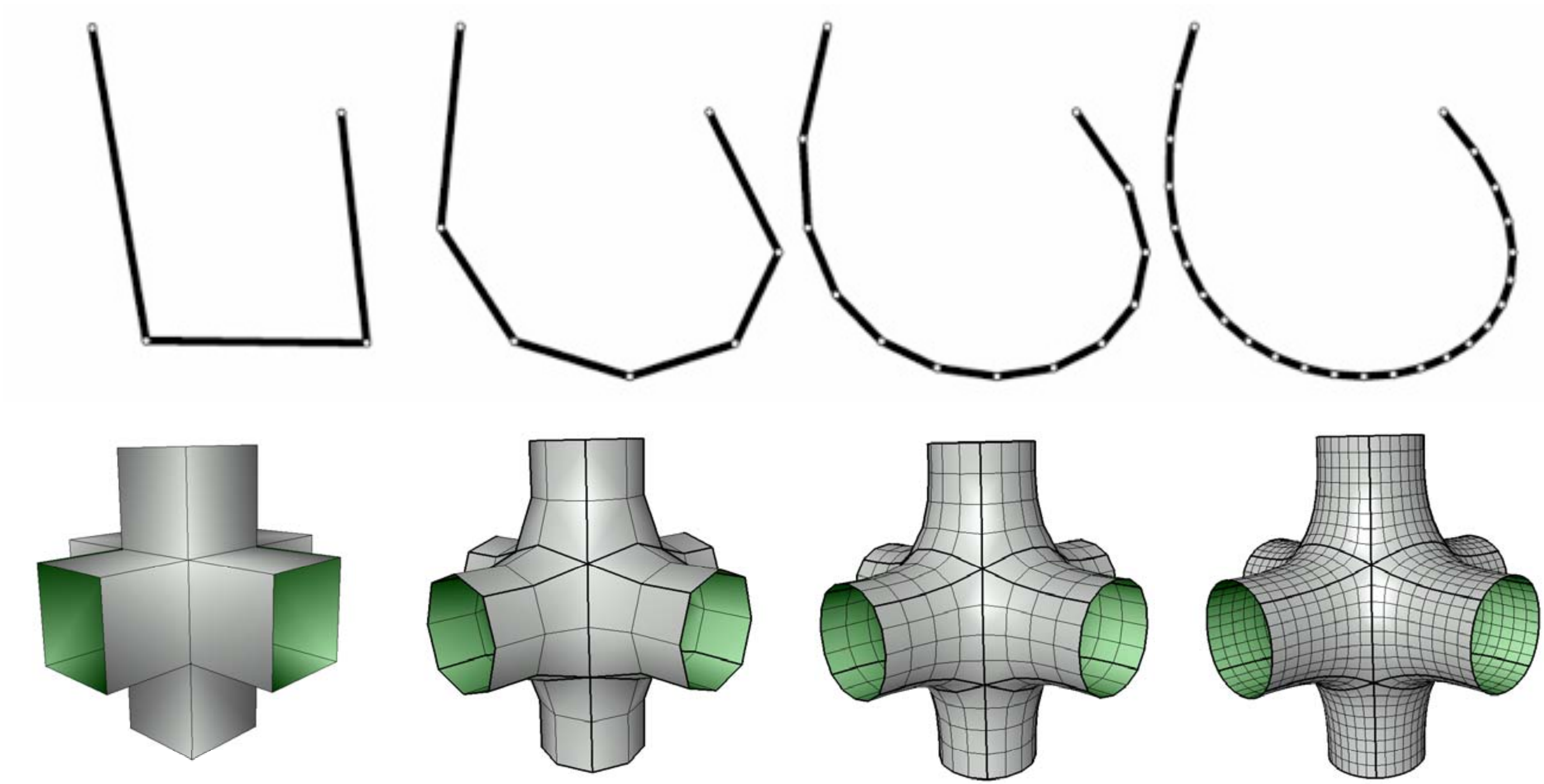
Subdivision curves

- Key idea: let go of the polynomials as the definition of the curve, and let the refinement rule define the curve
- Curve is defined as the *limit of a refinement process*
  - properties of curve depend on the rules
  - some rules make polynomial curves, some don't
  - complexity shifts from implementations to proofs

Playing with the rules

- Once a curve is *defined* using subdivision we can customize its behavior by making exceptions to the rules.
- Example: handle endpoints by simply using the mask [1] at that point.
- Resulting curve *is* a uniform B-spline in the middle, but near the exceptional points it is something different.
  - it might not be a polynomial
  - but it is still linear, still has basis functions
  - the three coordinates of a surface point are still separate

From curves to surfaces



Subdivision surfaces

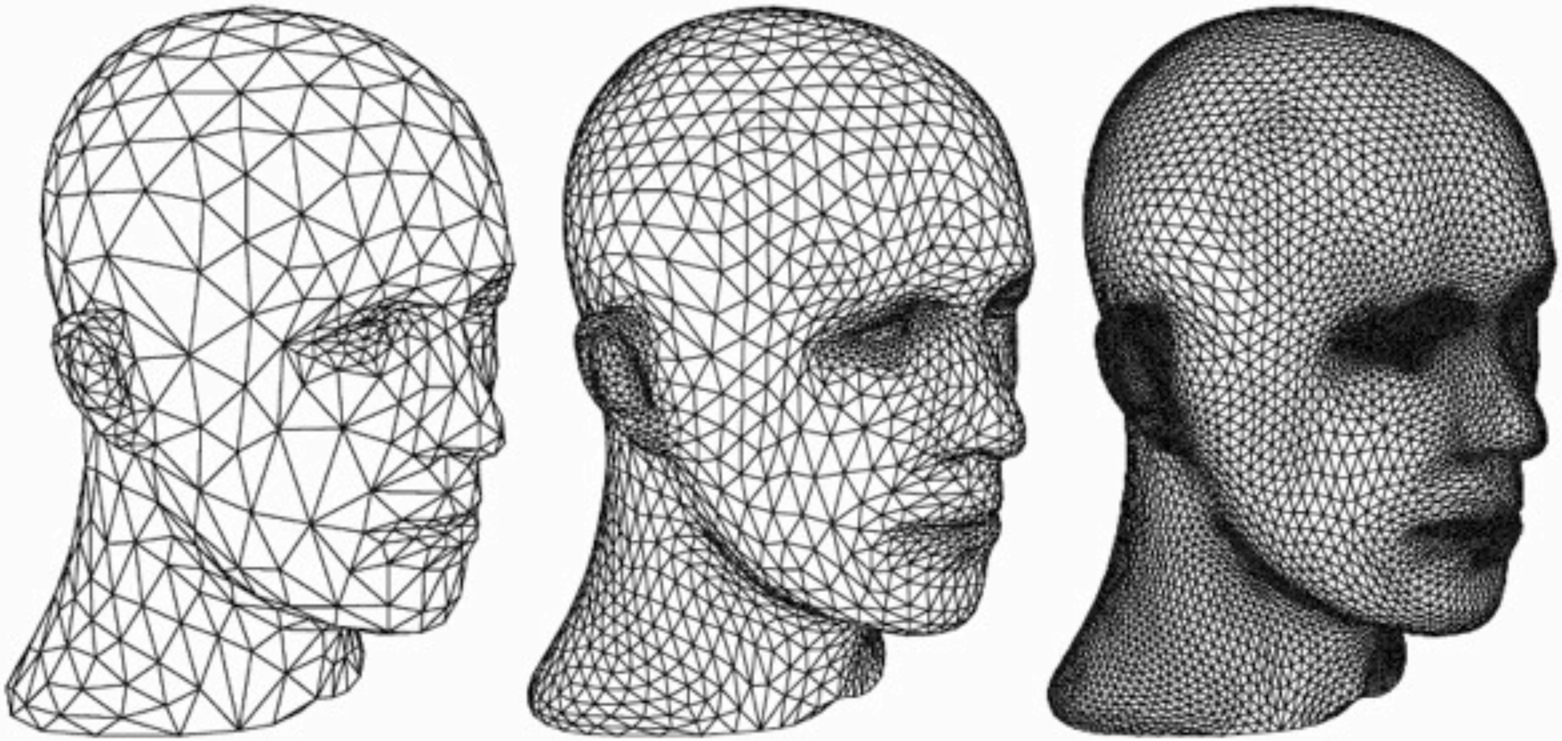


Figure 2.2: *Example of subdivision for a surface, showing 3 successive levels of refinement. On the left an initial triangular mesh approximating the surface. Each triangle is split into 4 according to a particular subdivision rule (middle). On the right the mesh is subdivided in this fashion once again.*

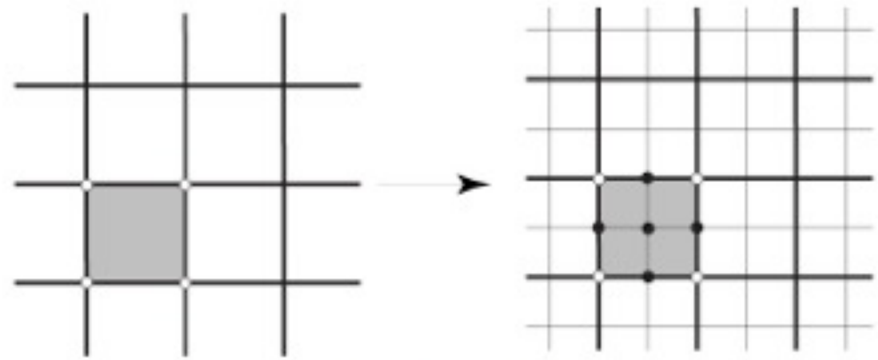
[Schröder & Zorin SIGGRAPH 2000 course 23]

Generalizing from curves to surfaces

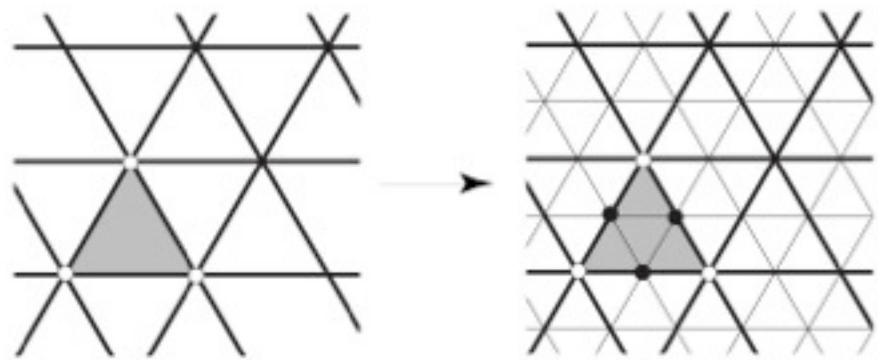
- Two parts to subdivision process
- Subdividing the mesh (computing new topology)
  - For curves: replace every segment with two segments
  - For surfaces: replace every face with some new faces
- Positioning the vertices (computing new geometry)
  - For curves: two rules (one for *odd* vertices, one for *even*)
    - New vertex's position is a weighted average of positions of old vertices that are nearby along the sequence
  - For surfaces: two kinds of rules (still called odd and even)
    - New vertex's position is a weighted average of positions of old vertices that are nearby in the mesh

Subdivision of meshes

- Quadrilaterals
  - Catmull-Clark 1978
- Triangles
  - Loop 1987



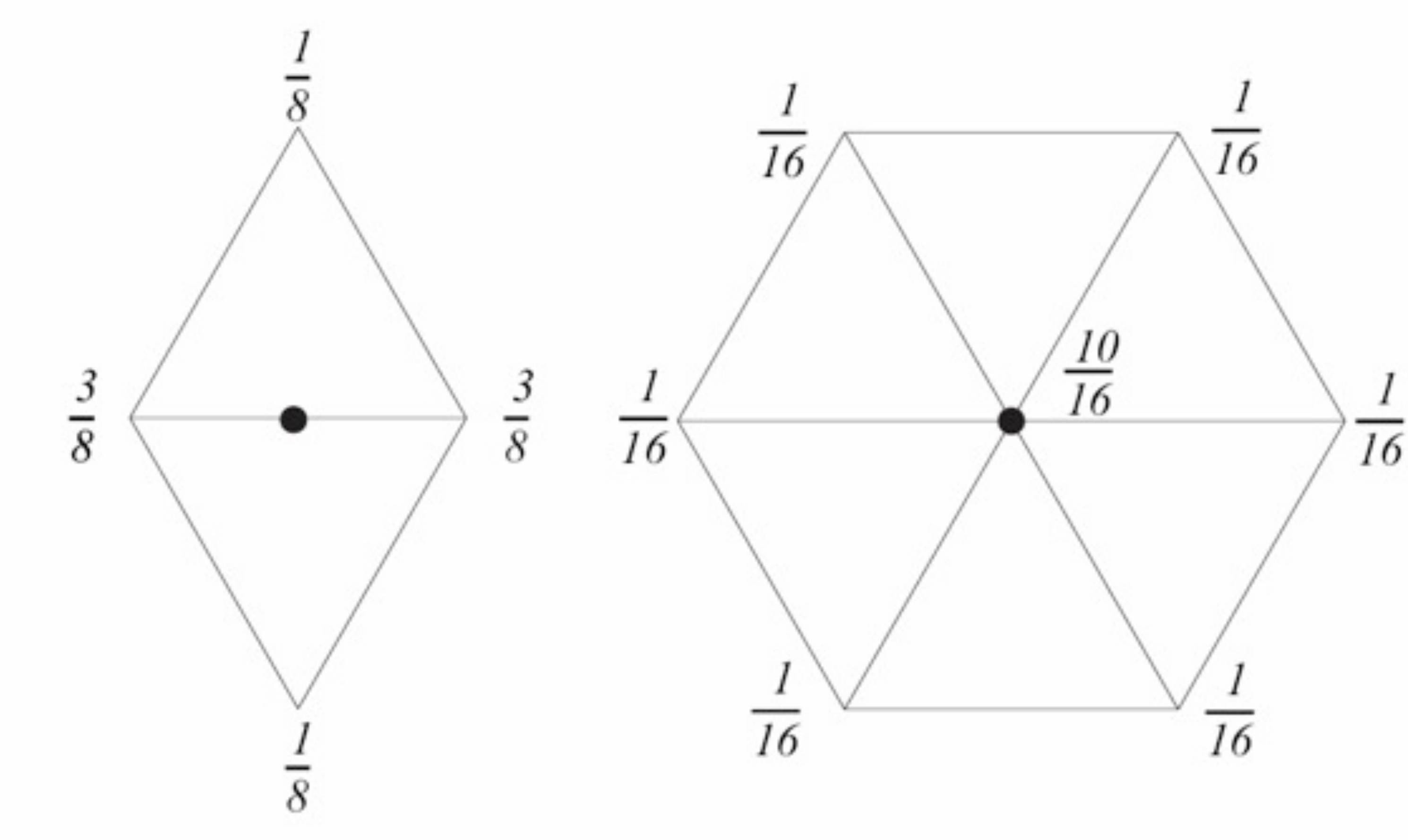
Face split for quads



Face split for triangles

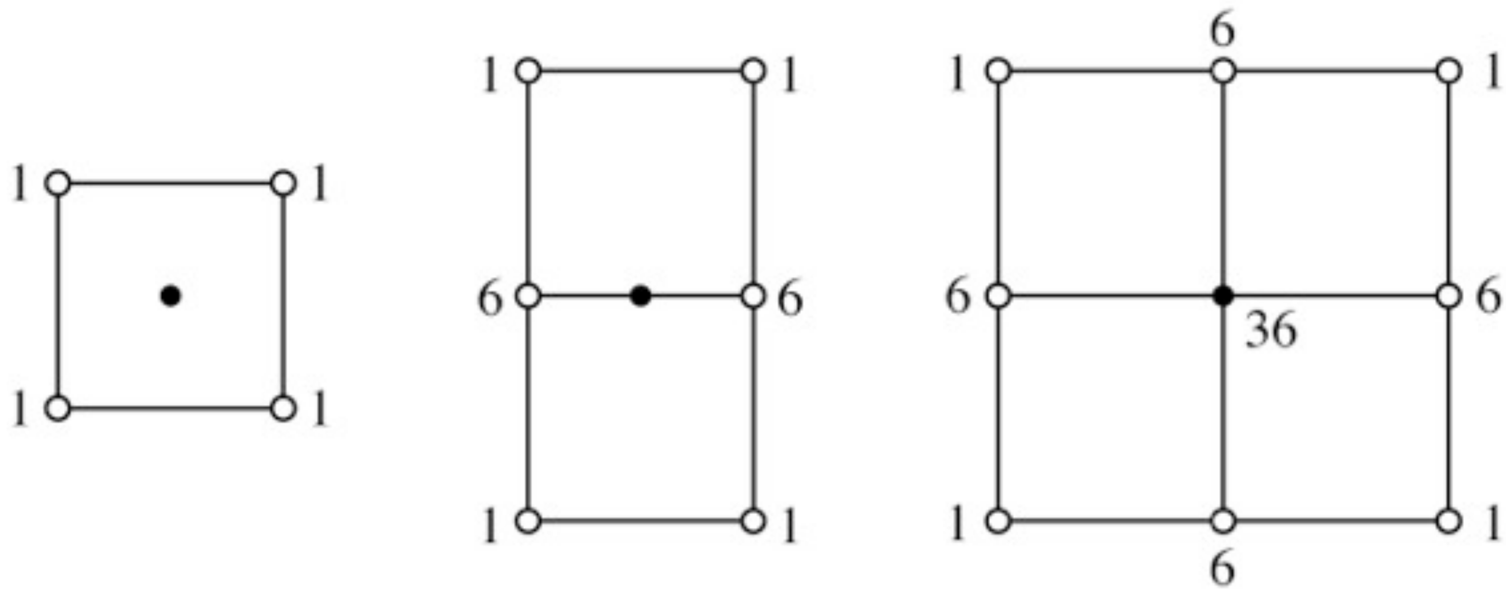
[Schröder & Zorin SIGGRAPH 2000 course 23]

Loop regular rules



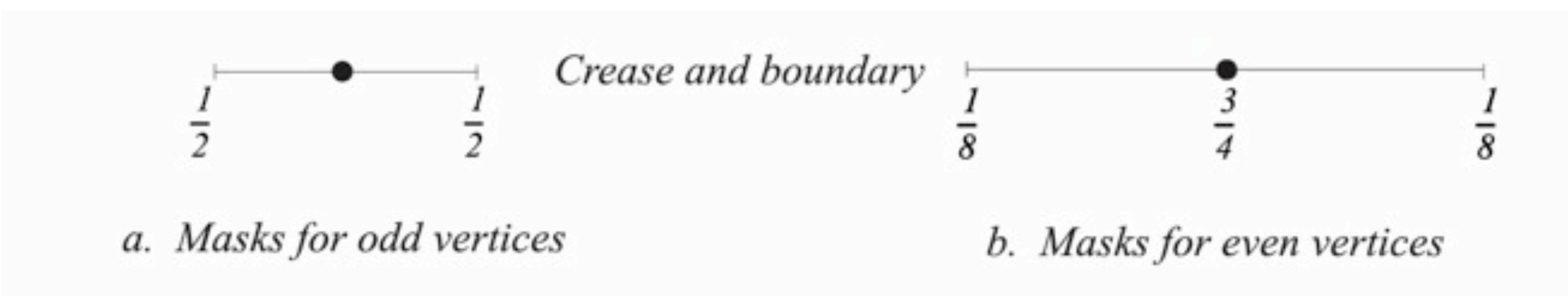
[Schröder & Zorin SIGGRAPH 2000 course 23]

Catmull-Clark regular rules



Creases

- With splines, make creases by turning off continuity constraints
- With subdivision surfaces, make creases by marking edges “sharp”
  - use different rules for vertices with sharp edges
  - these rules produce B-splines that depend only on vertices along crease

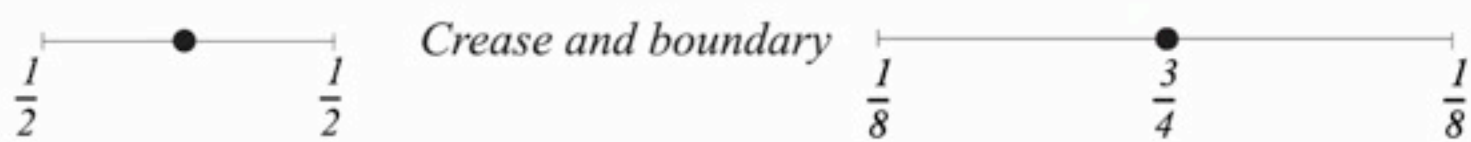


[Schröder & Zorin SIGGRAPH 2000 course 23]

Boundaries

- At boundaries the masks do not work
  - mesh is not manifold; edges do not have two triangles
- Solution: same as crease
  - shape of boundary is controlled only by vertices along boundary

[Schröder & Zorin SIGGRAPH 2000 course 23]

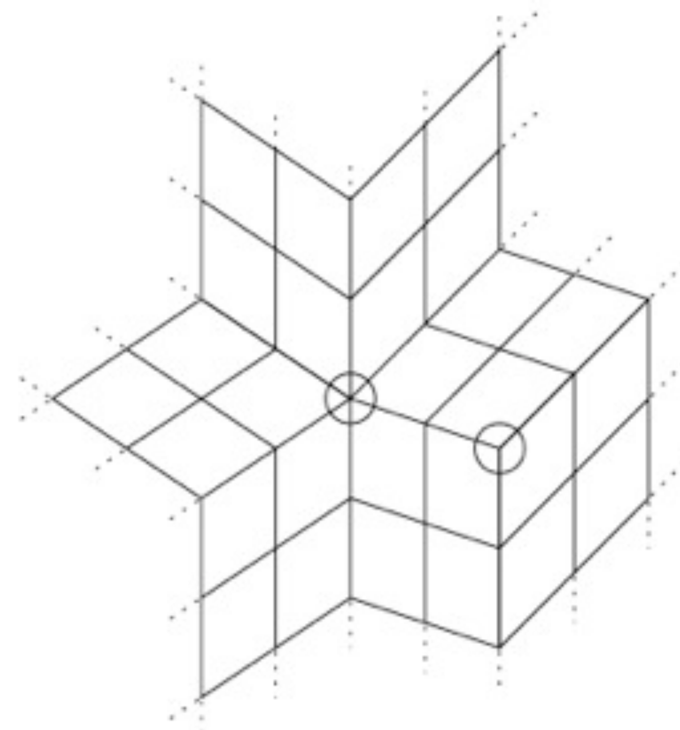


a. Masks for odd vertices

b. Masks for even vertices

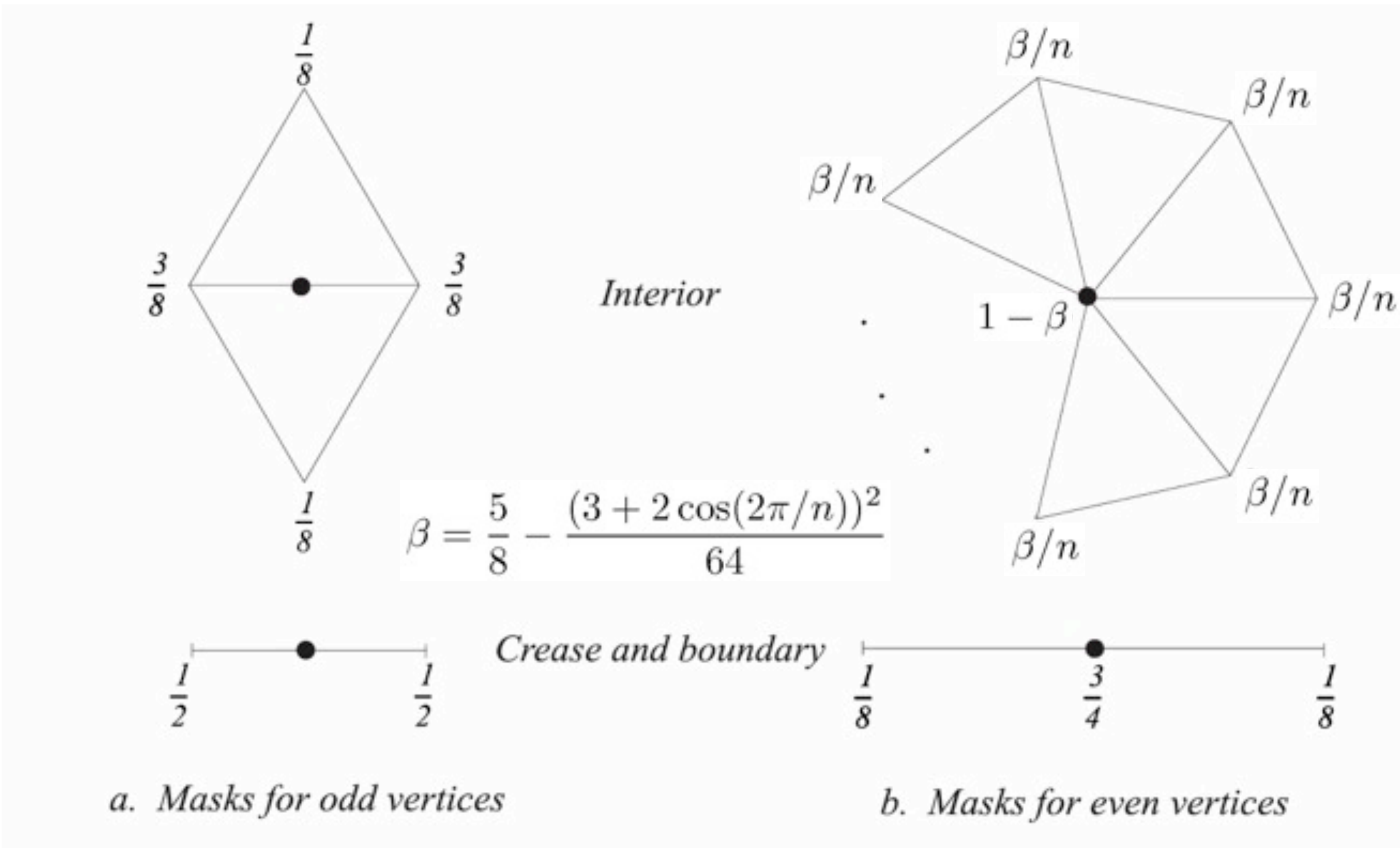
Extraordinary vertices

- Vertices that don't have the "standard" valence
- Unavoidable for most topologies
- Difference from splines
  - treatment of extraordinary vertices is really the only way subdivision surfaces are different from spline patches



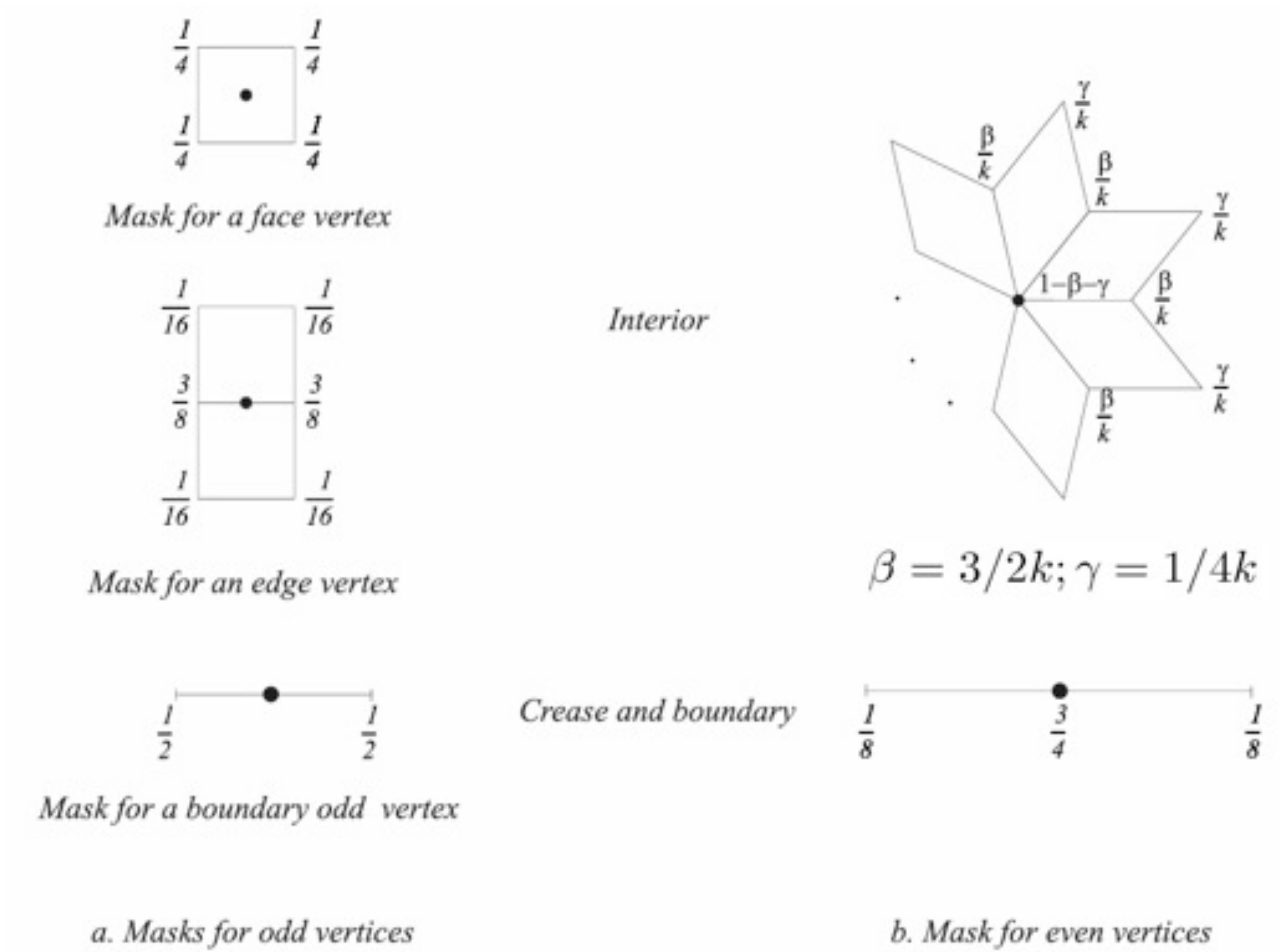
[Schröder & Zorin SIGGRAPH 2000 course 23]

Full Loop rules (triangle mesh)



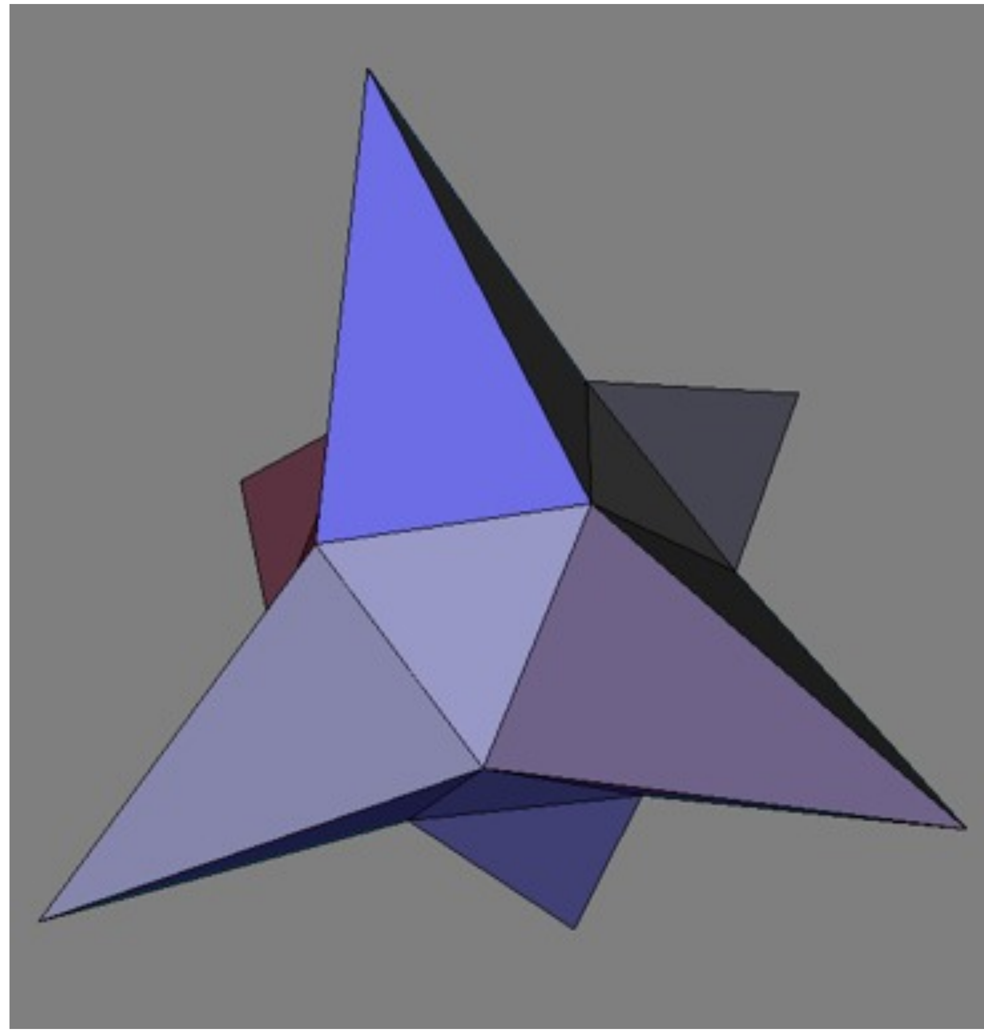
[Schröder & Zorin SIGGRAPH 2000 course 23]

Full Catmull-Clark rules (quad mesh)



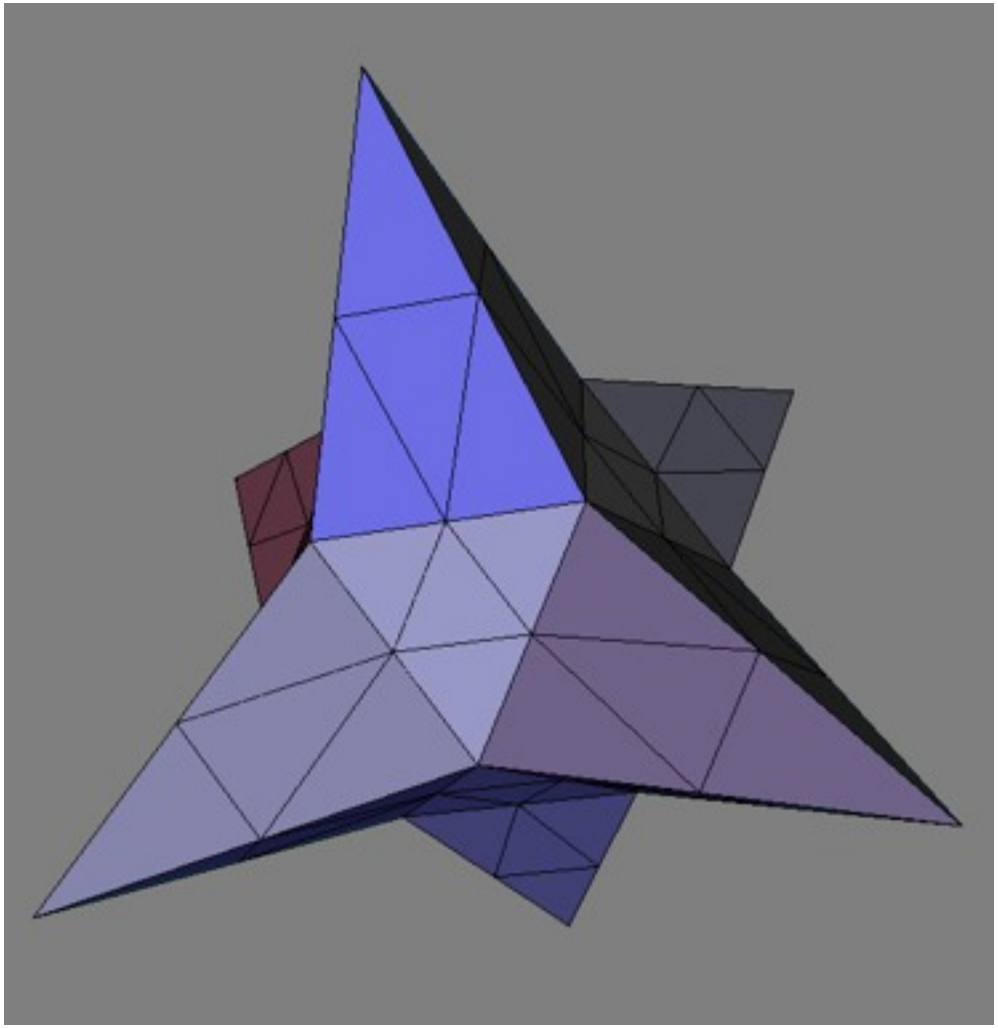
[Schröder & Zorin SIGGRAPH 2000 course 23]

Loop Subdivision Example



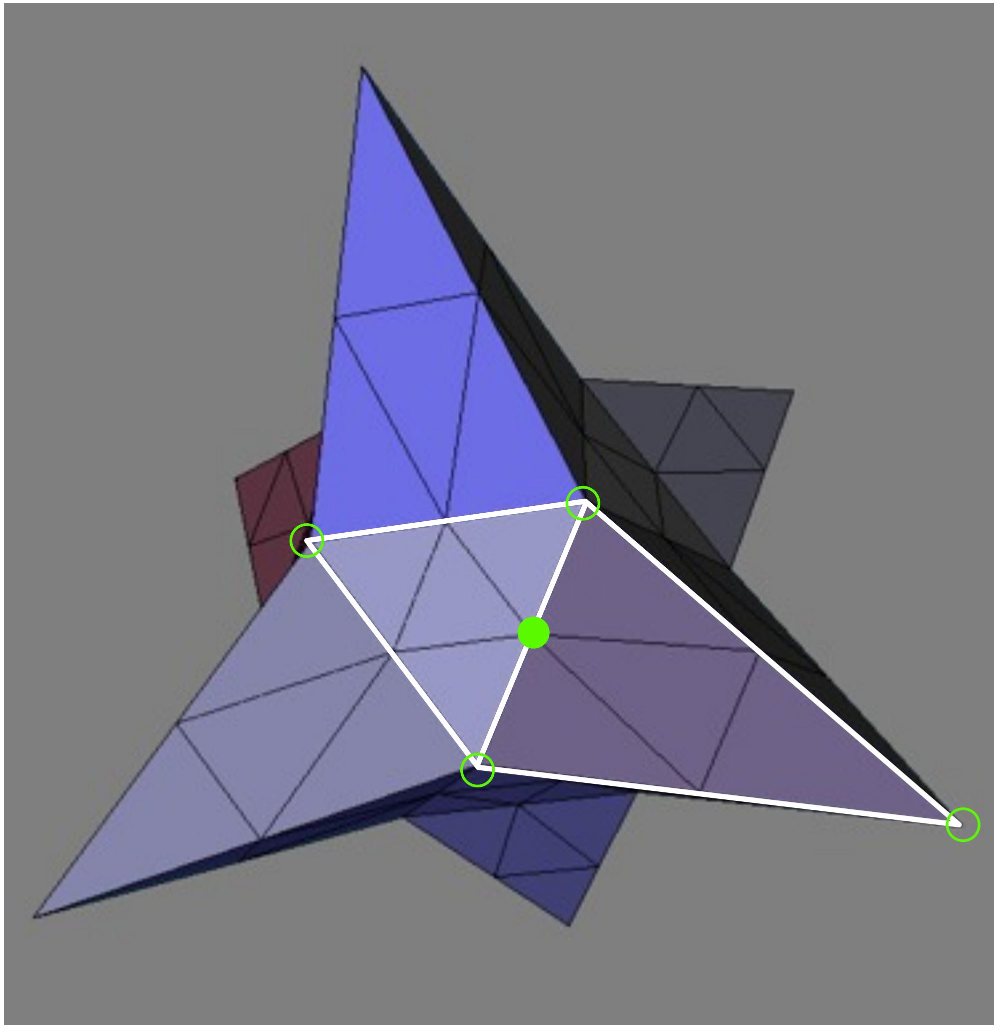
control polyhedron

Loop Subdivision Example



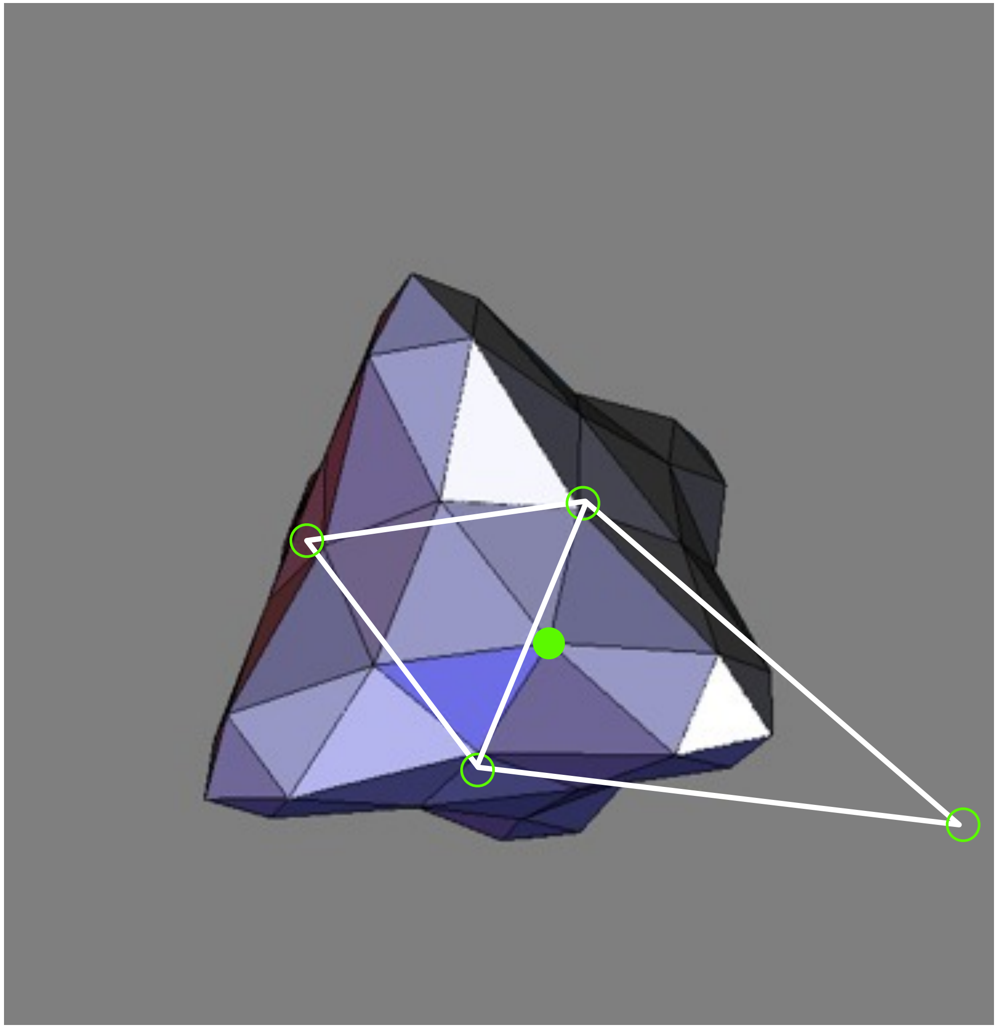
refined
control polyhedron

Loop Subdivision Example



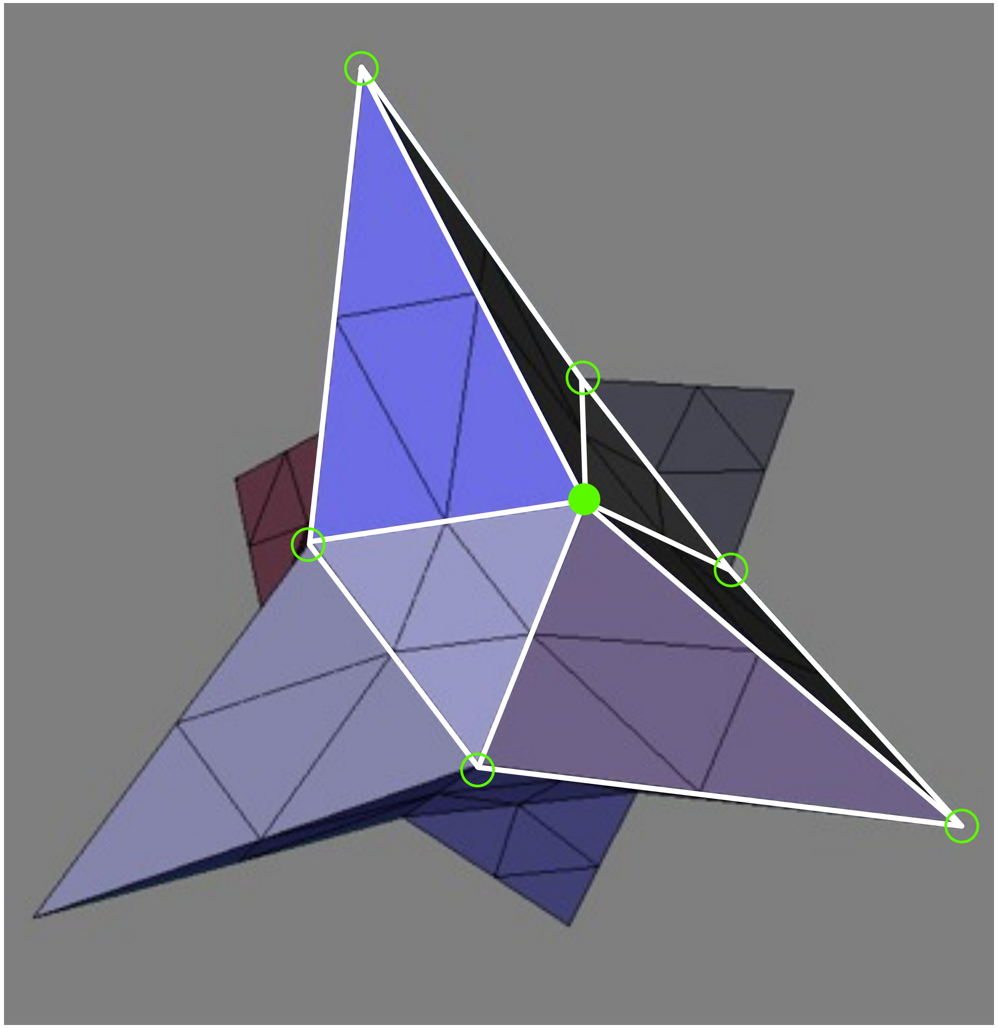
odd
subdivision mask

Loop Subdivision Example



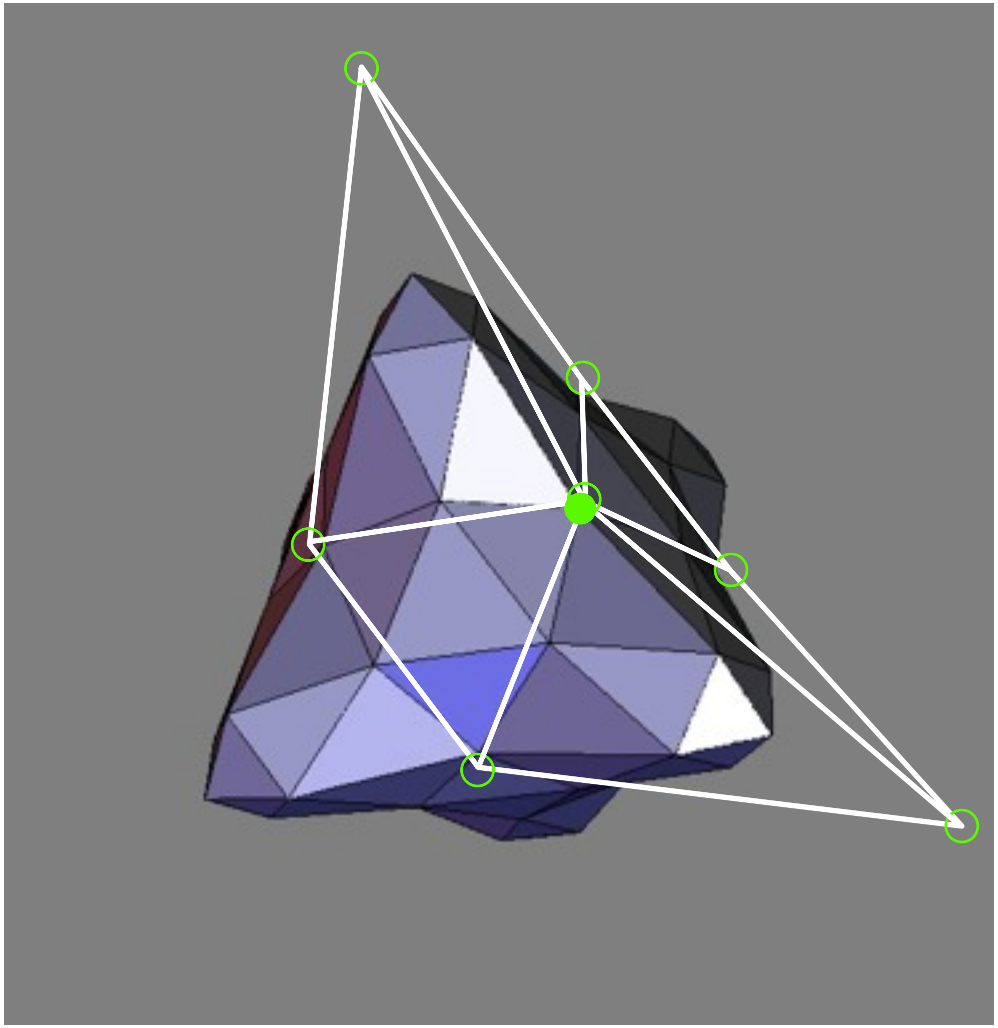
subdivision level 1

Loop Subdivision Example



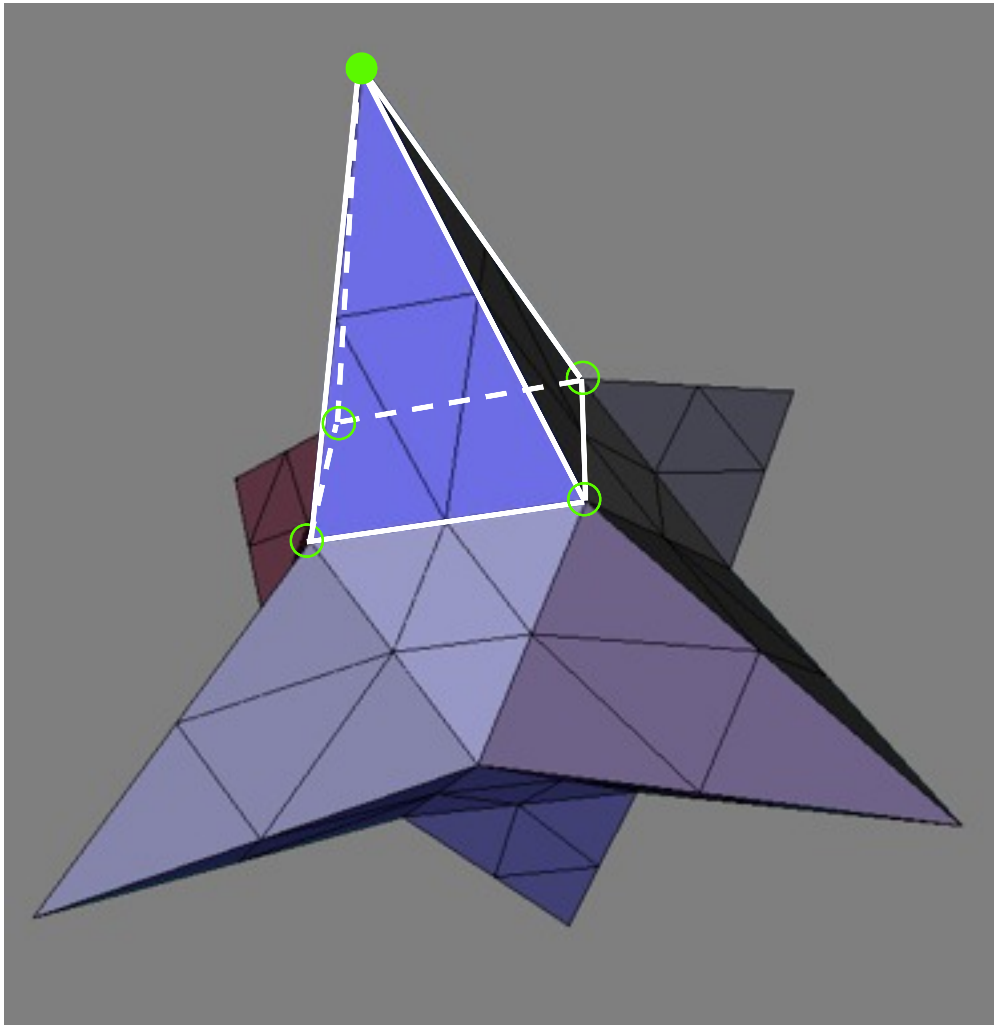
even
subdivision mask
(ordinary vertex)

Loop Subdivision Example



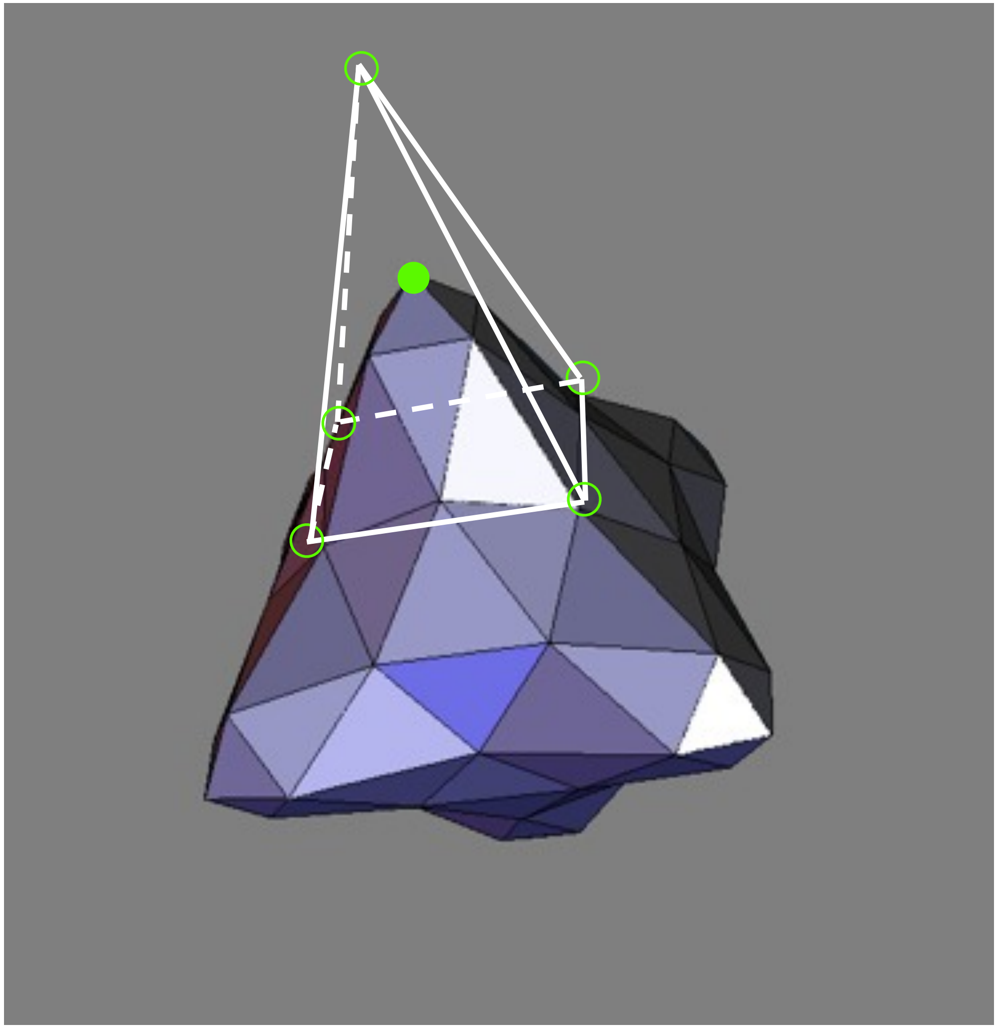
subdivision level 1

Loop Subdivision Example



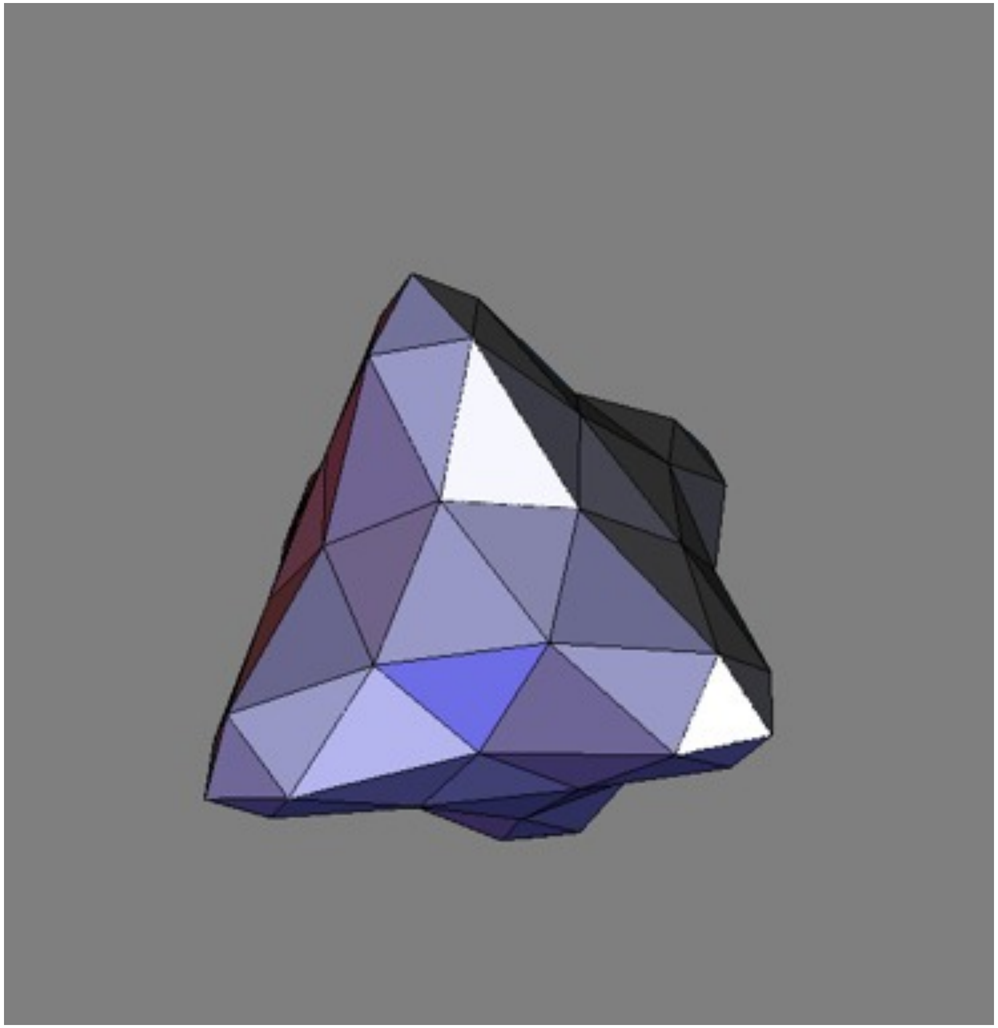
even
subdivision mask
(extraordinary vertex)

Loop Subdivision Example



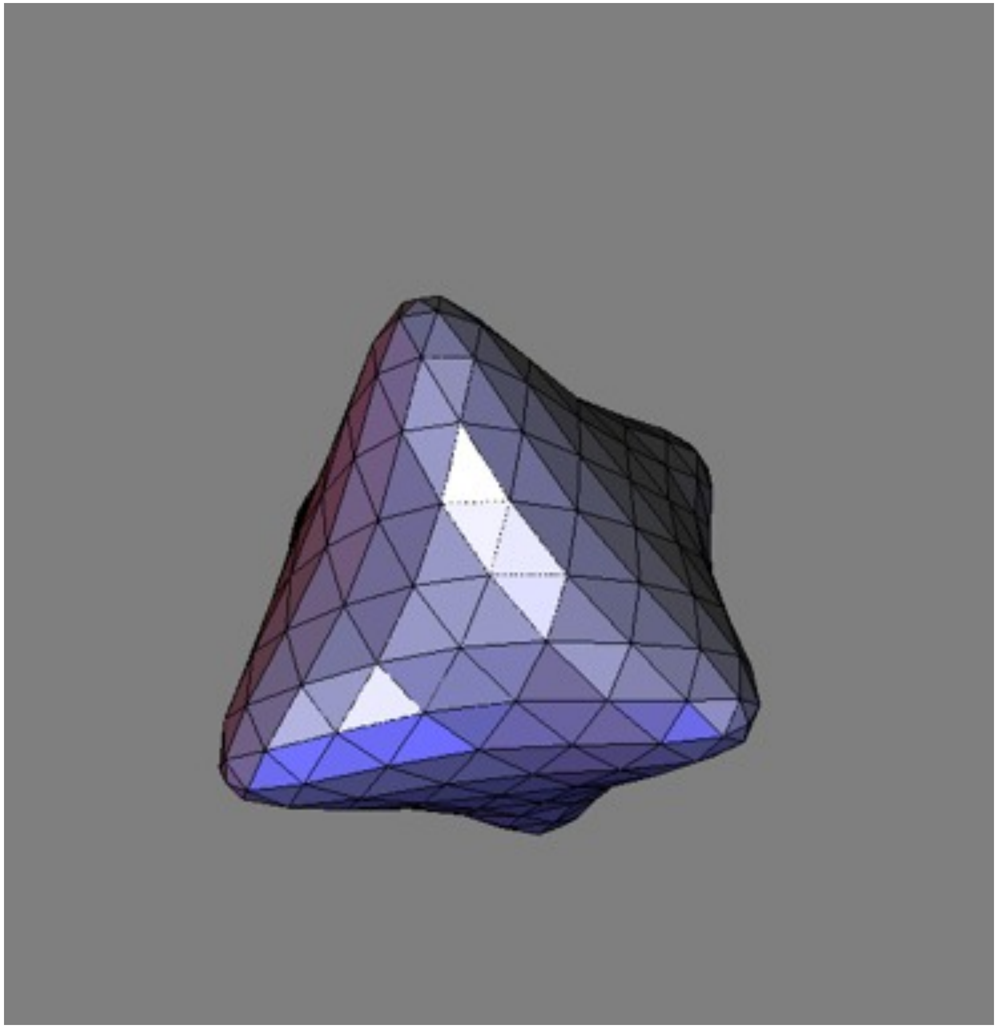
subdivision level 1

Loop Subdivision Example



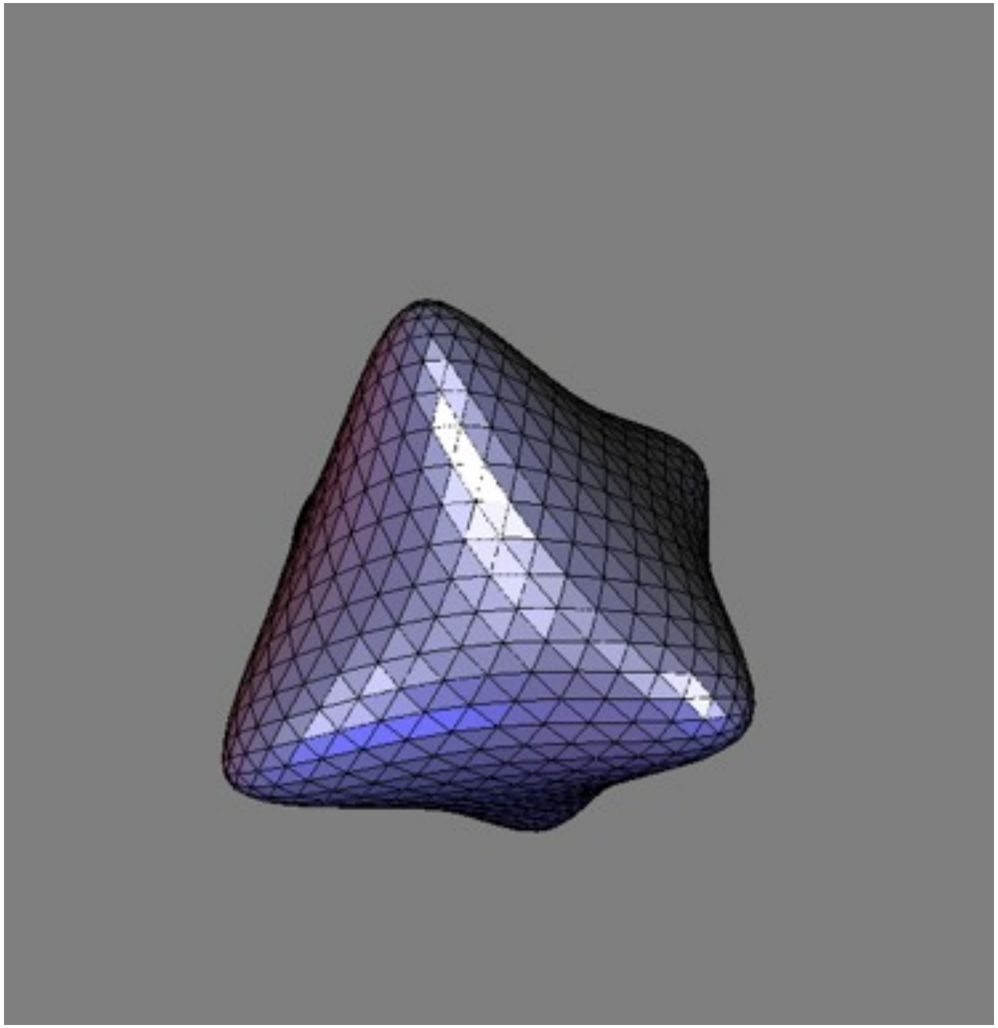
subdivision level 1

Loop Subdivision Example



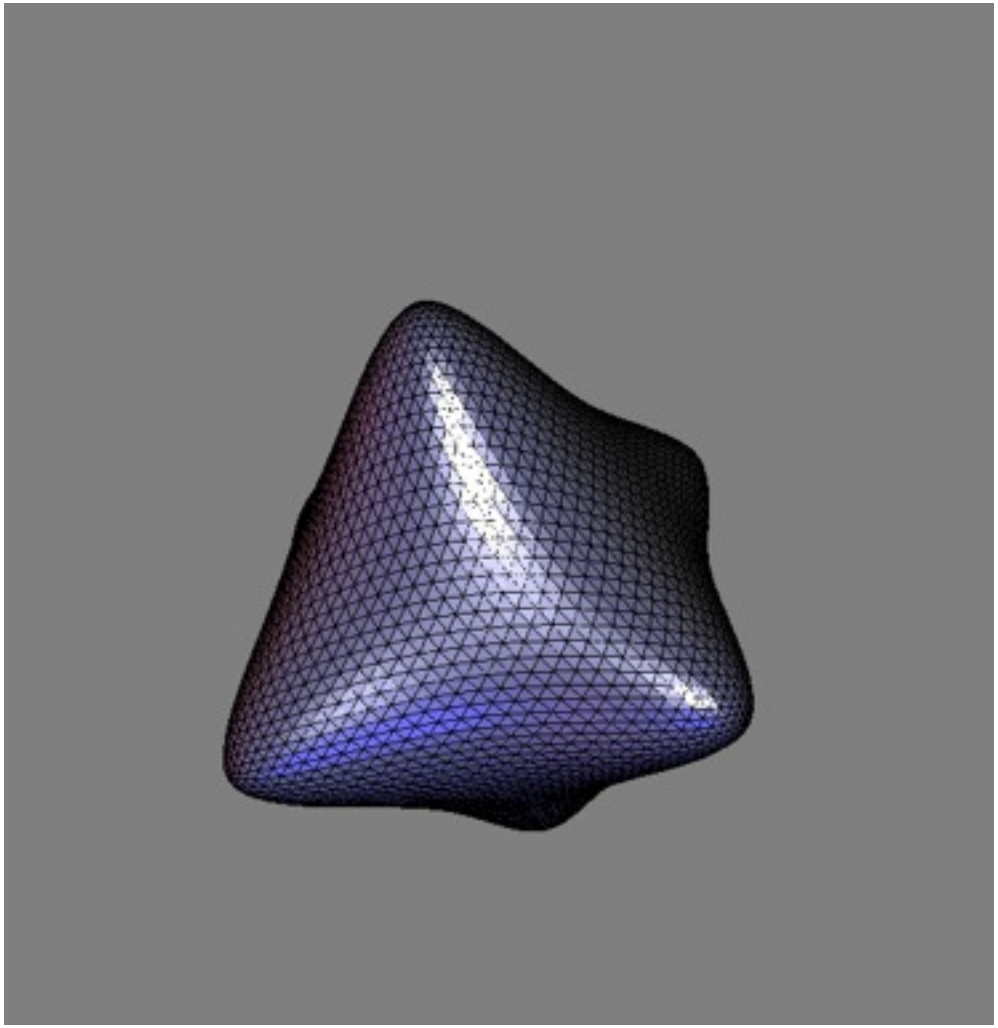
subdivision level 2

Loop Subdivision Example



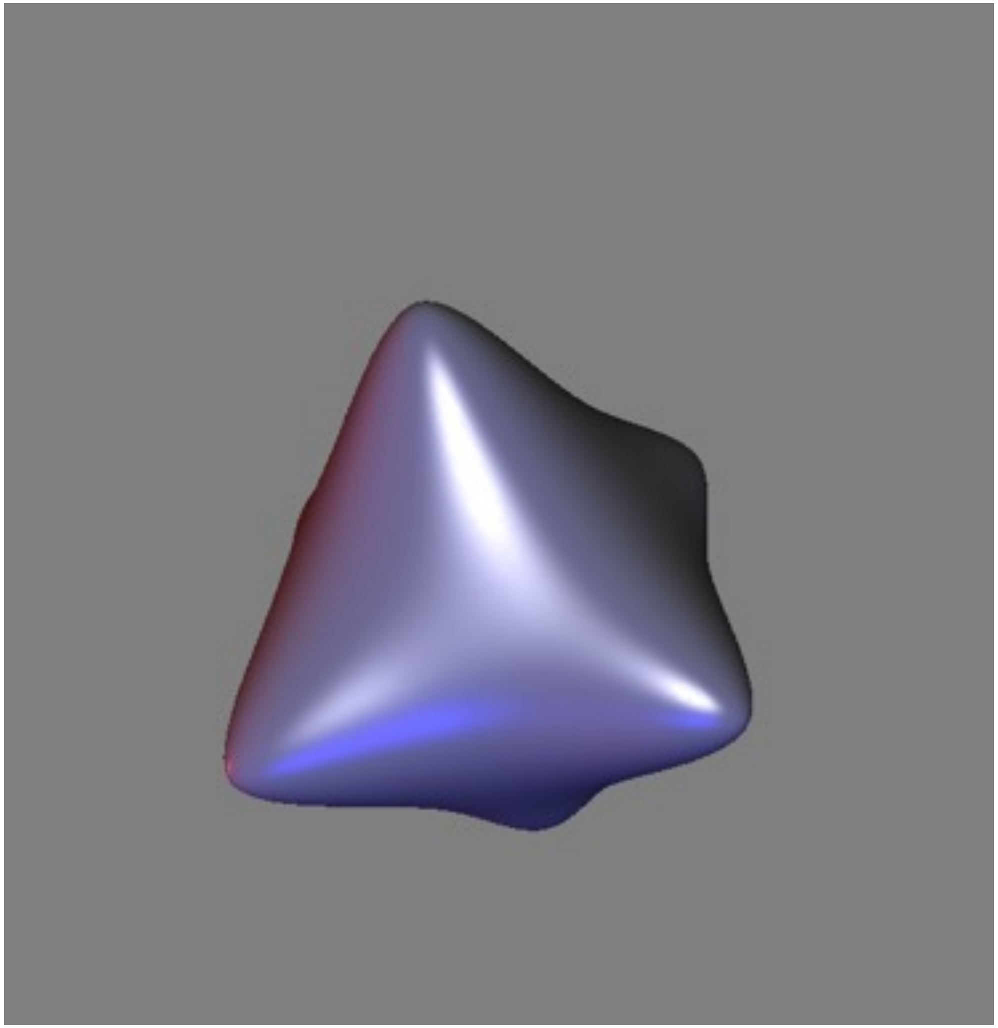
subdivision level 3

Loop Subdivision Example



subdivision level 4

Loop Subdivision Example

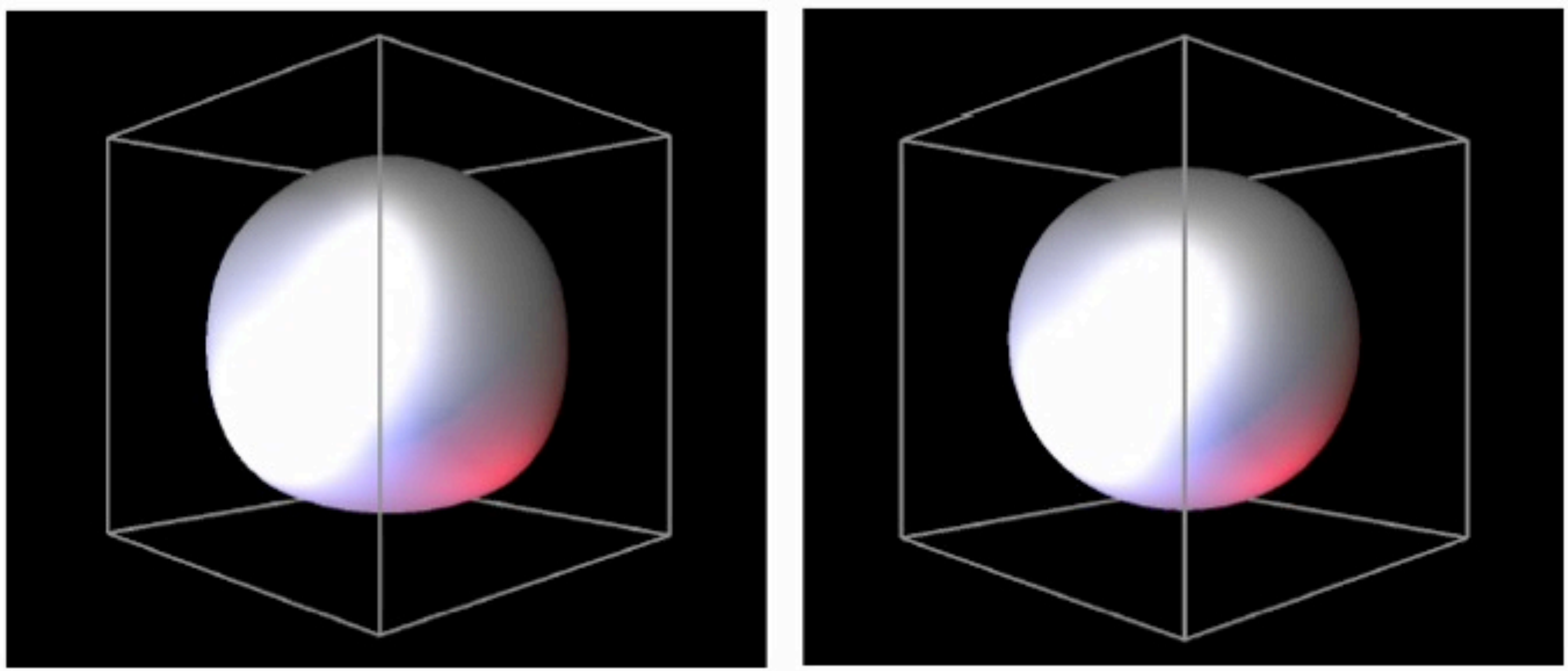


limit surface

Relationship to splines

- In regular regions, behavior is identical
- At extraordinary vertices, achieve C^1
  - near extraordinary, different from splines
- Linear everywhere
  - mapping from parameter space to 3D is a linear combination of the control points
  - “emergent” basis functions per control point
    - match the splines in regular regions
    - “custom” basis functions around extraordinary vertices

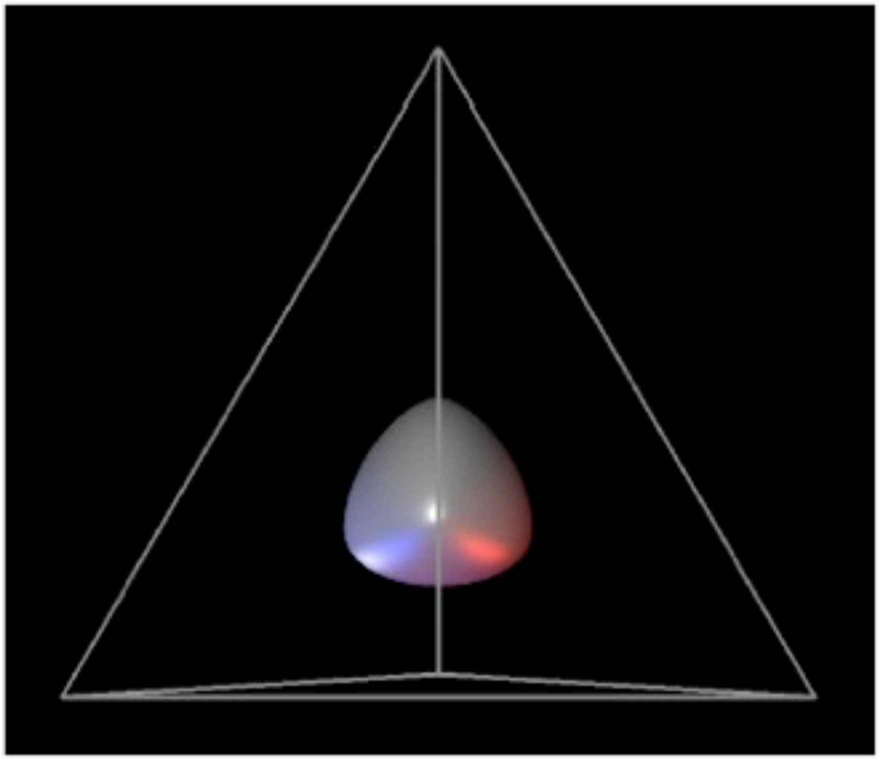
Loop vs. Catmull-Clark



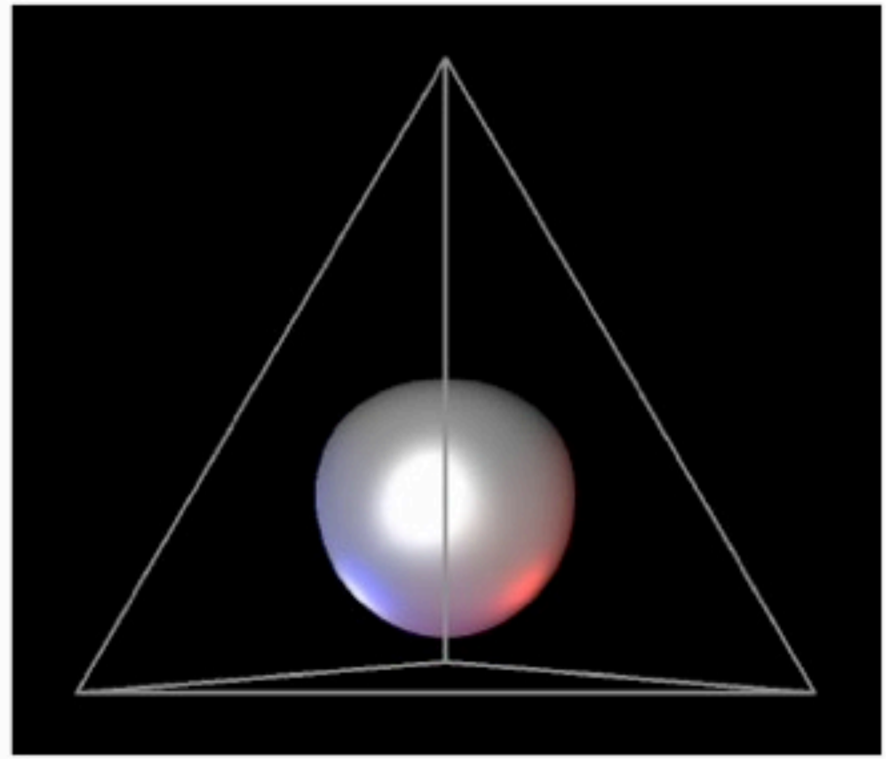
Loop *Catmull-Clark*

[Schröder & Zorin SIGGRAPH 2000 course 23]

Loop vs. Catmull-Clark



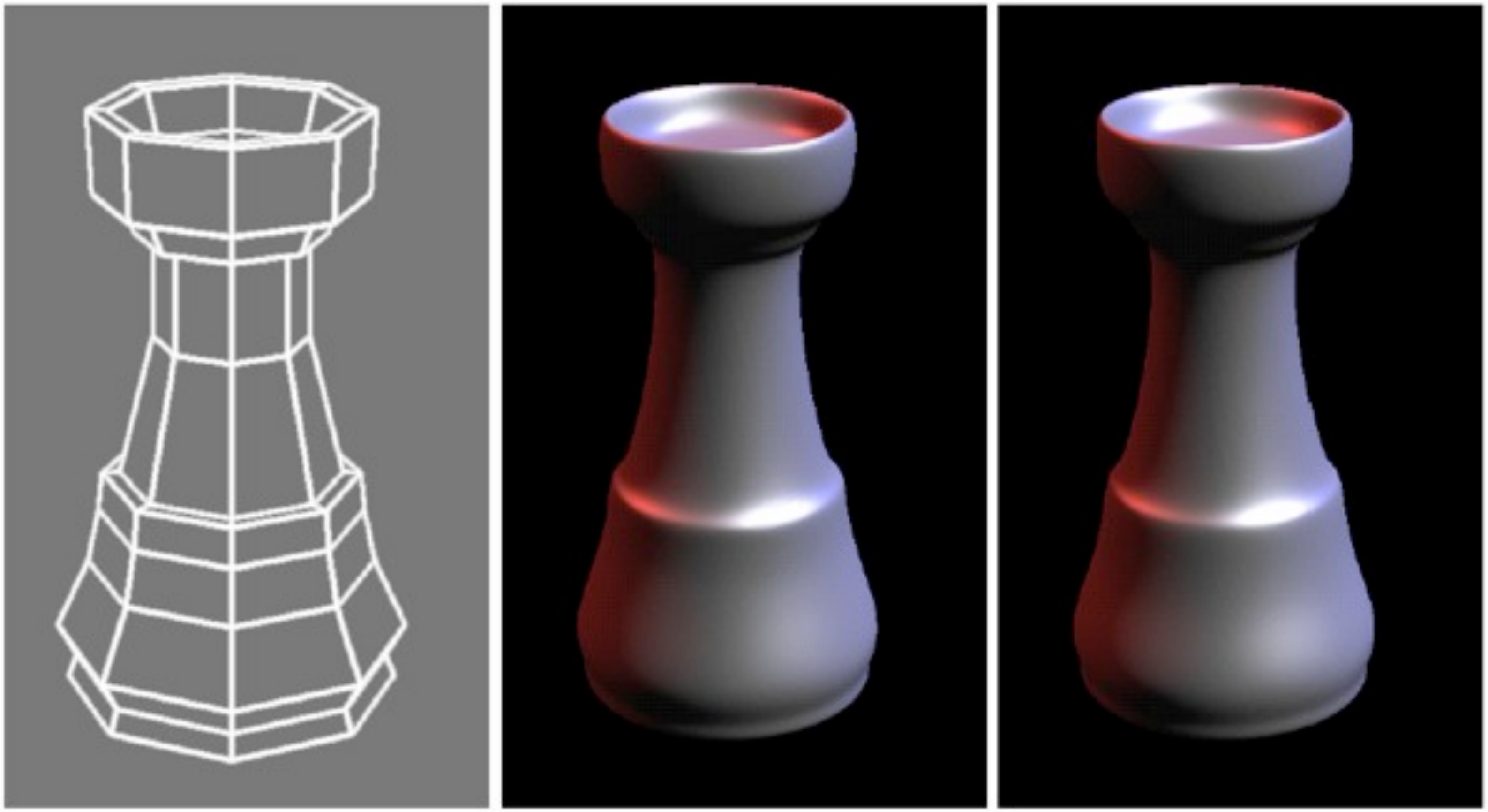
Loop



Catmull-Clark

[Schröder & Zorin SIGGRAPH 2000 course 23]

Loop vs. Catmull-Clark

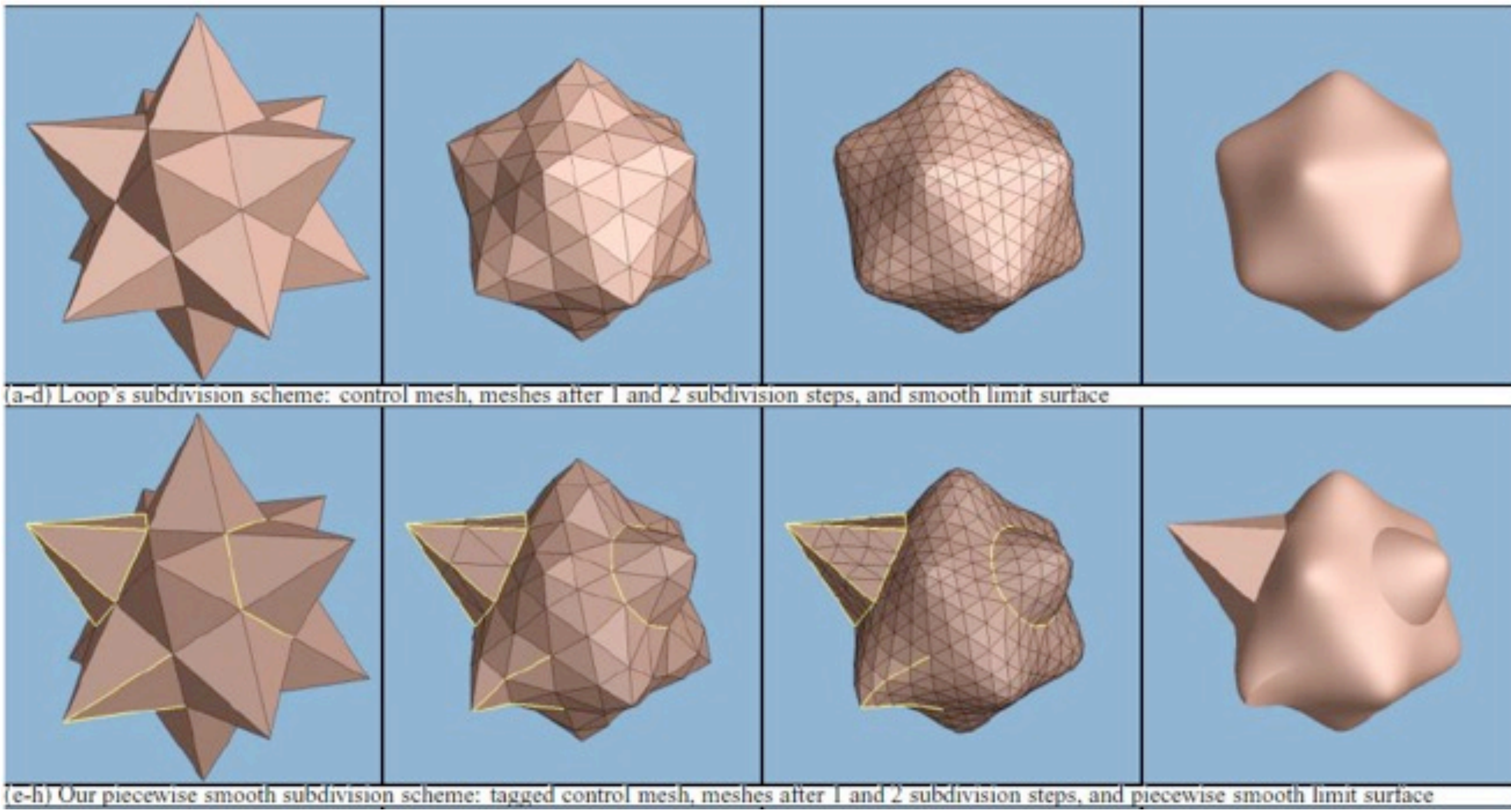


Loop
(after splitting faces)

Catmull-Clark

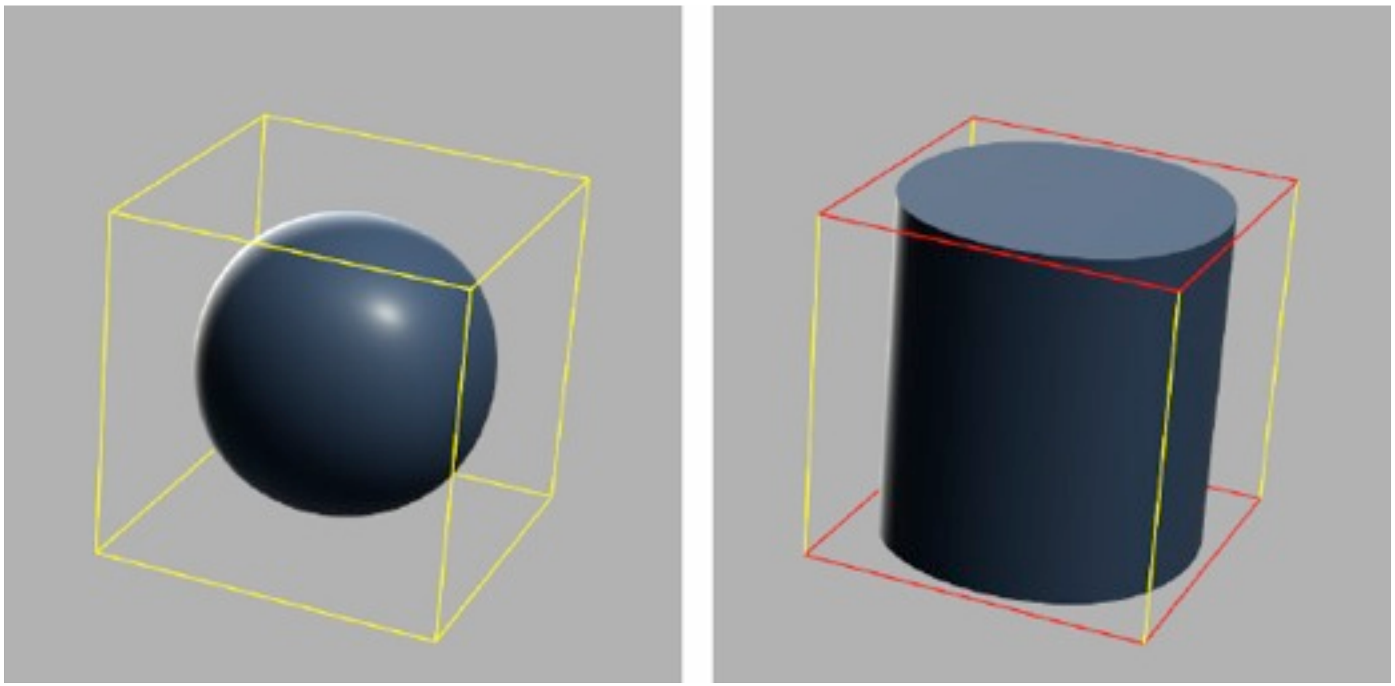
[Schröder & Zorin SIGGRAPH 2000 course 23]

Loop with creases



[Hugues Hoppe]

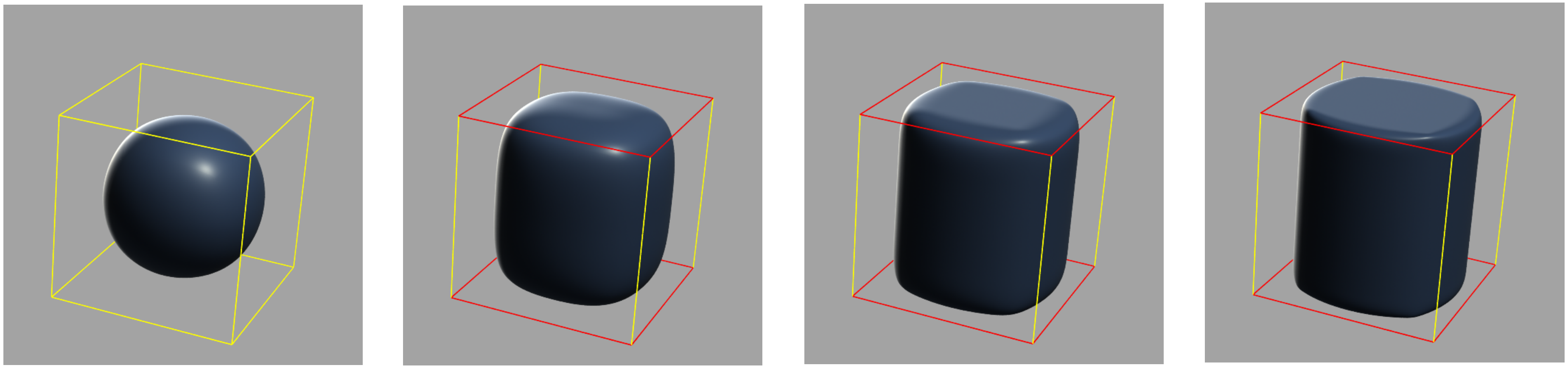
Catmull-Clark with creases



[DeRose et al. SIGGRAPH 1998]

Variable sharpness creases

- Idea: subdivide for a few levels using the crease rules, then proceed with the normal smooth rules.
- Result: a soft crease that gets sharper as we increase the number of levels of sharp subdivision steps



sharpness 0 sharpness 1 sharpness 2 sharpness 3

Geri's Game

- Pixar short film to test subdivision in production
  - Catmull-Clark (quad mesh) surfaces
  - complex geometry
  - extensive use of creases
  - subdivision surfaces to support cloth dynamics

[DeRose et al. SIGGRAPH 1998]

