Computer Animation and Games I

CM50244

Subdivision Surface Modeling

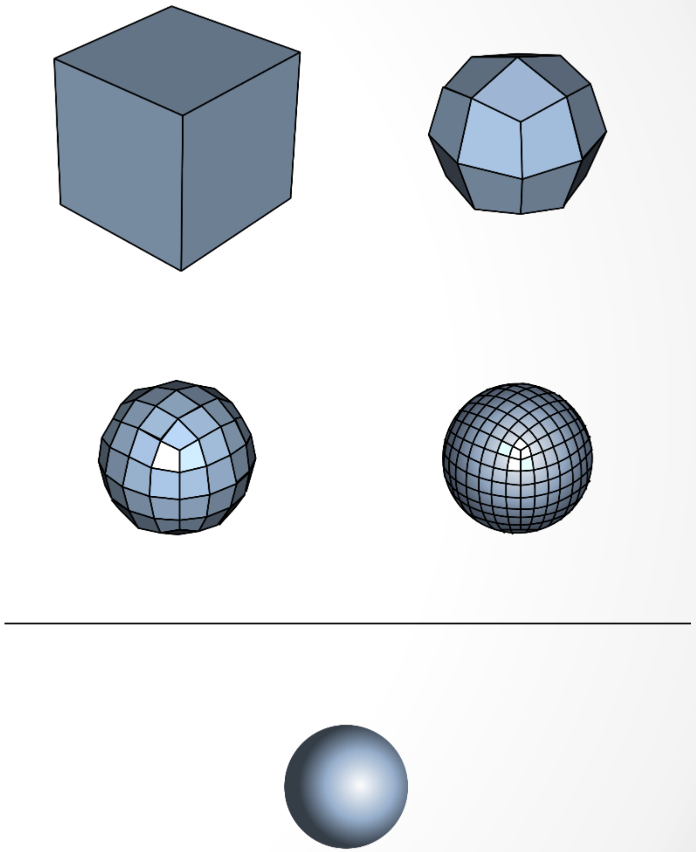
Some slides from Kwang In Kim

Overview

- **What is subdivision?**
- Curve subdivision algorithm
  - Chaiken's algorithm (for curves)
- Surface subdivision algorithms
  - Doo-Sabin algorithm
  - Catmull-Clark algorithm
  - Loop algorithm
- Advantages/disadvantages

What is Subdivision?

- Method of representing a surface using a coarser piecewise polygonal mesh.
- Recursive subdivision leads to better approximations.
- A smooth surface can be calculated in the limit.

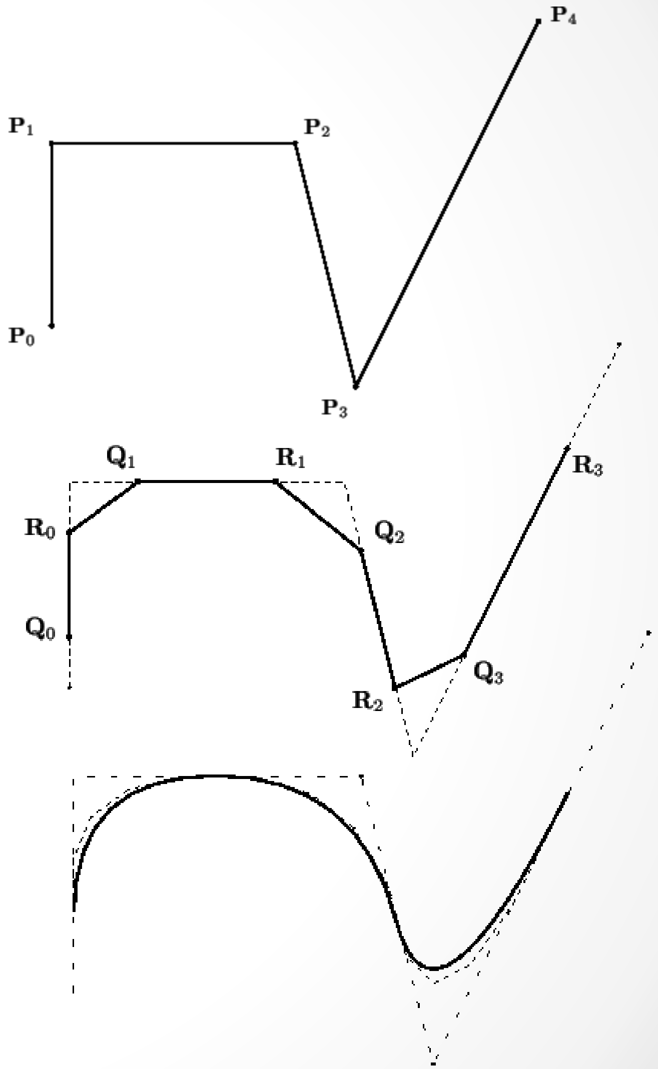


What is subdivision?

Input: polygon or polygonal mesh

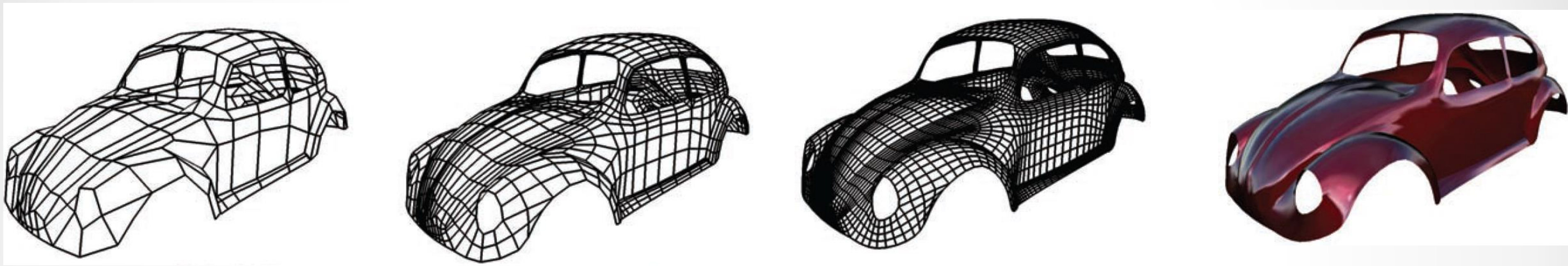
Process: repeatedly refine (subdivide) geometry

Output: “smooth” curve or surface



Subdivision Surface Applications

- Modeling 3D shapes for animation and games

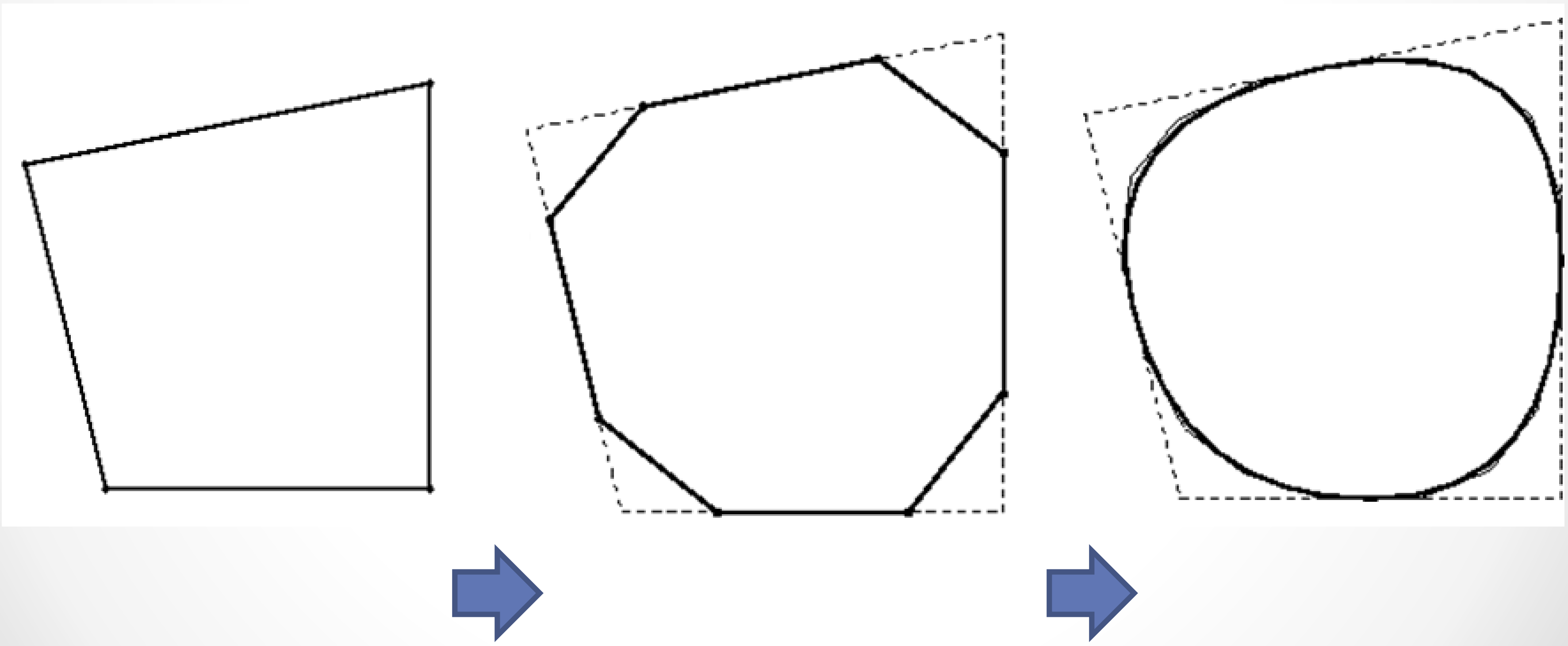


Overview

- What is subdivision?
- **Curve subdivision algorithm**
  - Chaiken's algorithm (for curves)
- Surface subdivision algorithms
  - Doo-Sabin algorithm
  - Catmull-Clark algorithm
  - Loop algorithm
- Advantages/disadvantages.

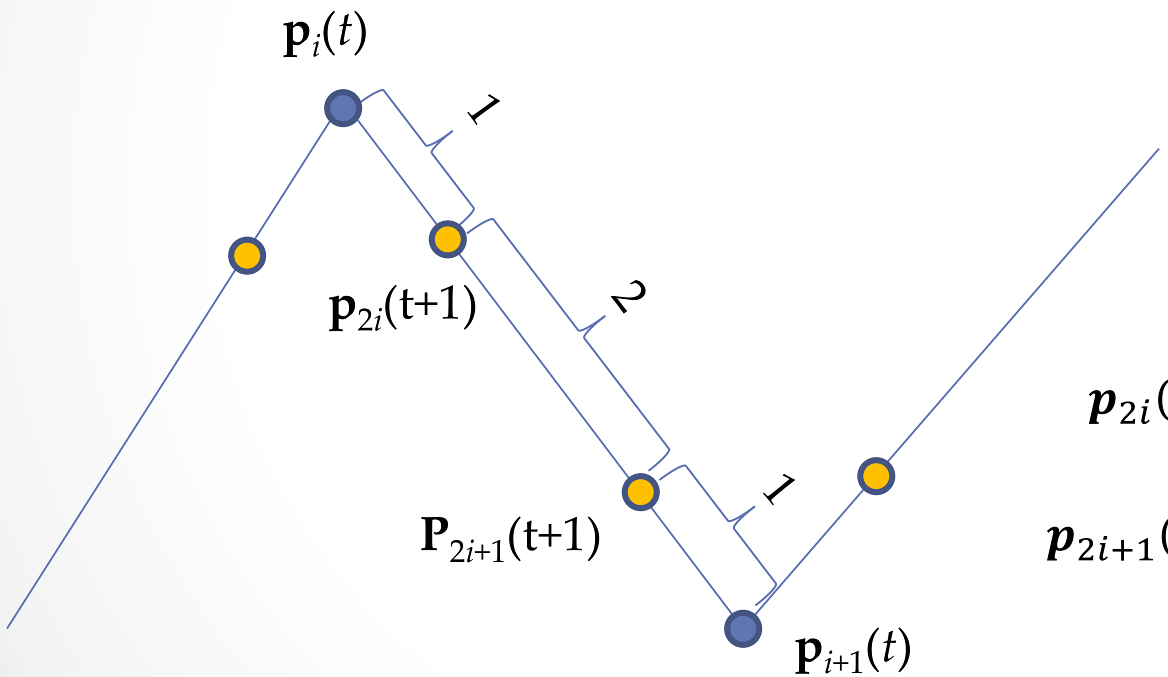
Chaiken's Algorithm

A corner cutting method for high speed curve generation



Chaiken's Algorithm

A corner cutting method for high speed curve generation



$$\boldsymbol{p}_{2i}(t+1) = \frac{3\boldsymbol{p}_i(t) + \boldsymbol{p}_{i+1}(t)}{4}$$

$$\boldsymbol{p}_{2i+1}(t+1) = \frac{\boldsymbol{p}_i(t) + 3\boldsymbol{p}_{i+1}(t)}{4}$$

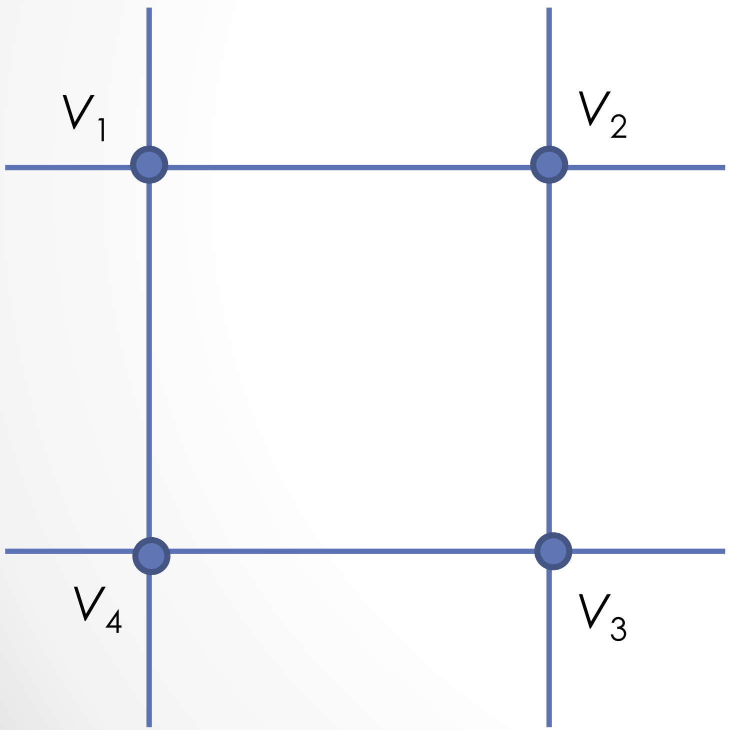
On each edge with ratios 1:2:1

Overview

- What are subdivision surfaces?
- Curve subdivision algorithm
  - Chaiken's algorithm (for curves)
- **Surface subdivision algorithms**
  - Doo-Sabin algorithm
  - Catmull-Clark algorithm
  - Loop algorithm
- Advantages/disadvantages

Doo-Sabin Subdivision

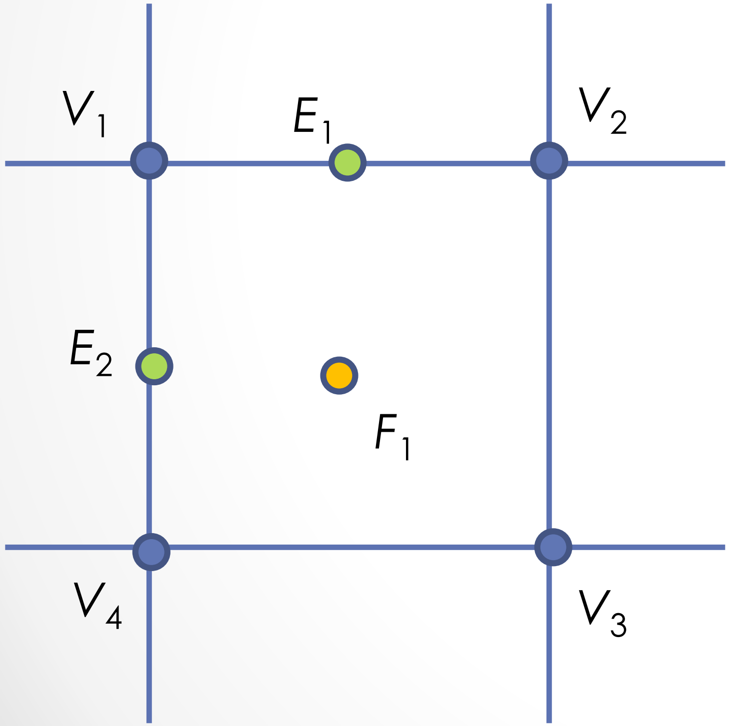
Iteratively generates *n* vertices (for *n*-gons):



A generalization of Chaiken for curves to surfaces

Doo-Sabin Subdivision

Iteratively generates *n* vertices (for *n*-gons):

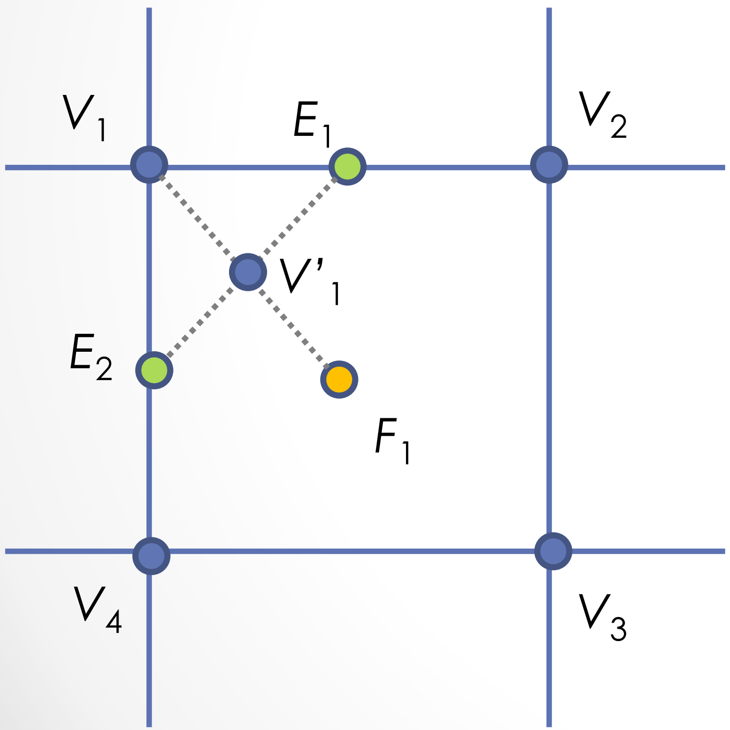


For each vertex V

1. Generate a face point and edge points, e.g., F_1, and E_1 and E_2.

Doo-Sabin Subdivision

Iterativelyates generates *n* vertices (for *n*-gons):

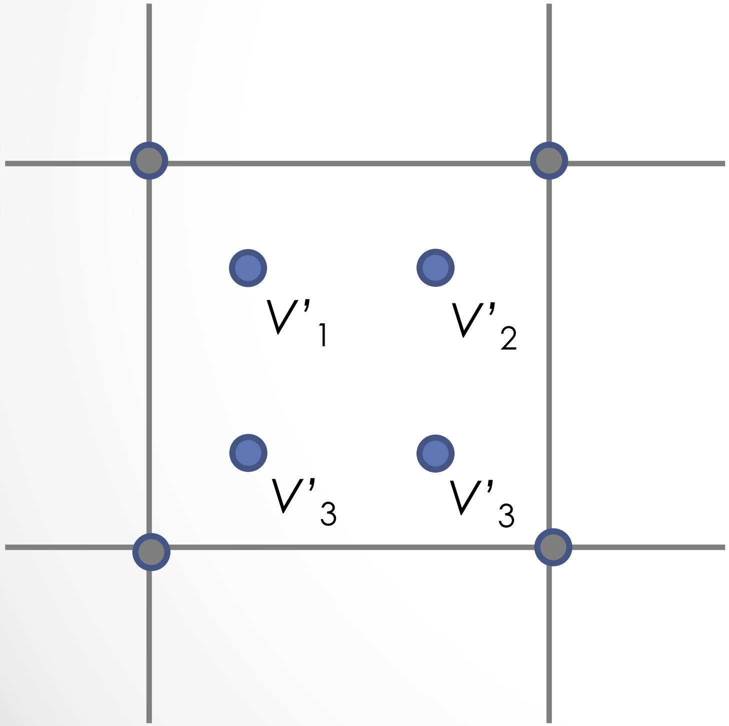


For each vertex V

1. Generate a face point and edge points, e.g., F_1, and E_1 and E_2 (edge points: midpoints of edges).
2. Generate a new vertex V' as the average of the new face and edge points.

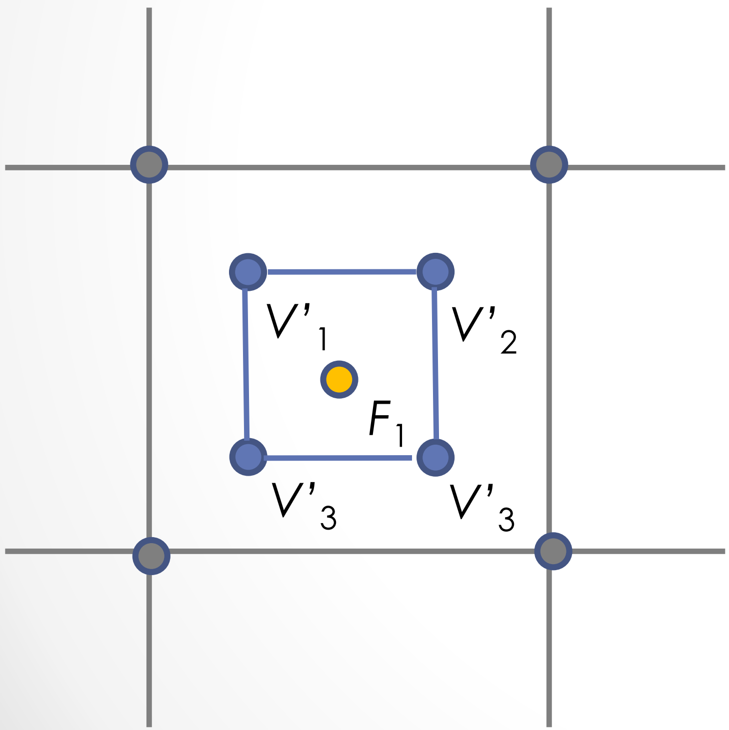
Doo-Sabin Subdivision

Iteratively generates *n* vertices (for *n*-gons):



Doo-Sabin Subdivision

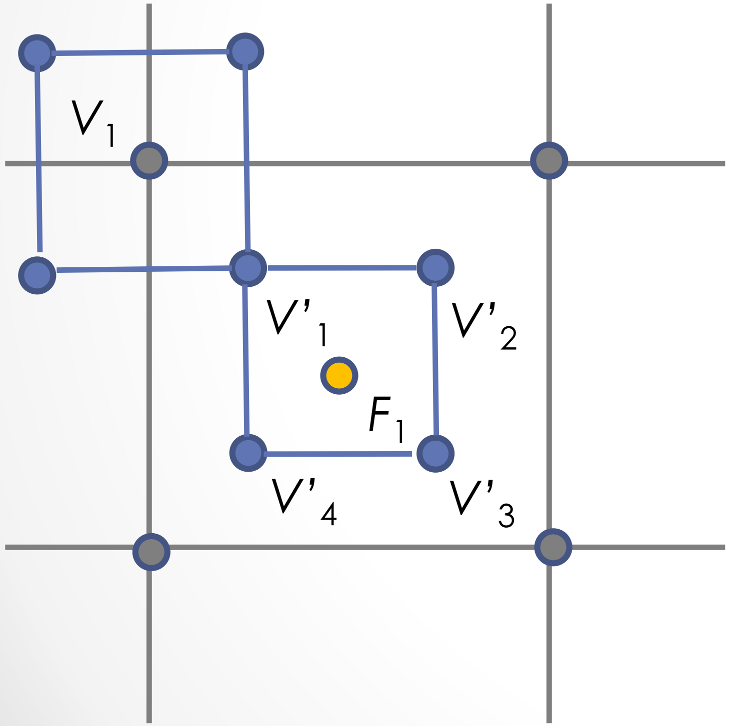
Iteratively generates *n* vertices (for *n*-gons):



3. For each face, connect the new vertex points along edges.

Doo-Sabin Subdivision

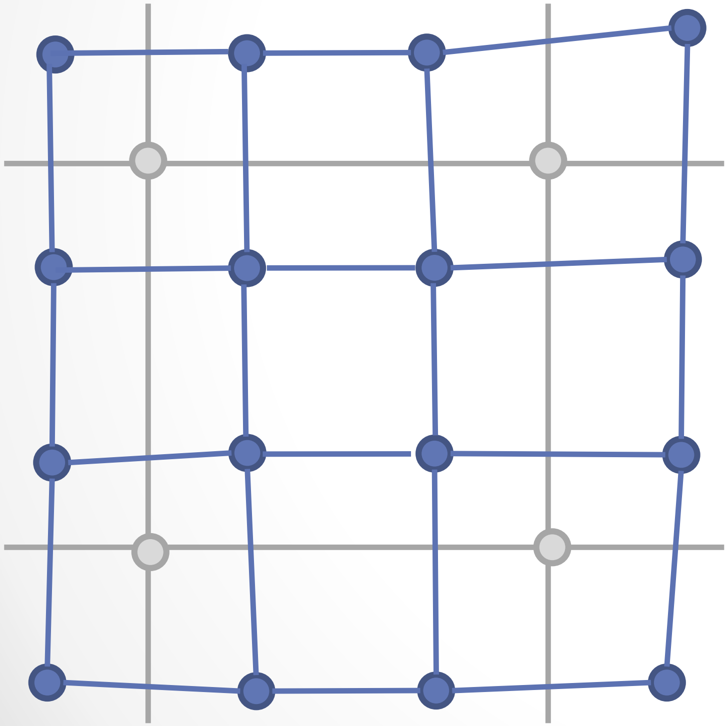
Iteratively generates *n* vertices (for *n*-gons):



3. For each face, connect the new vertex points along edges.
4. For each original vertex (e.g., V_1), connect the new vertices for faces that are adjacent to this vertex.

Doo-Sabin Subdivision

Iteratively generates *n* vertices (for *n*-gons):

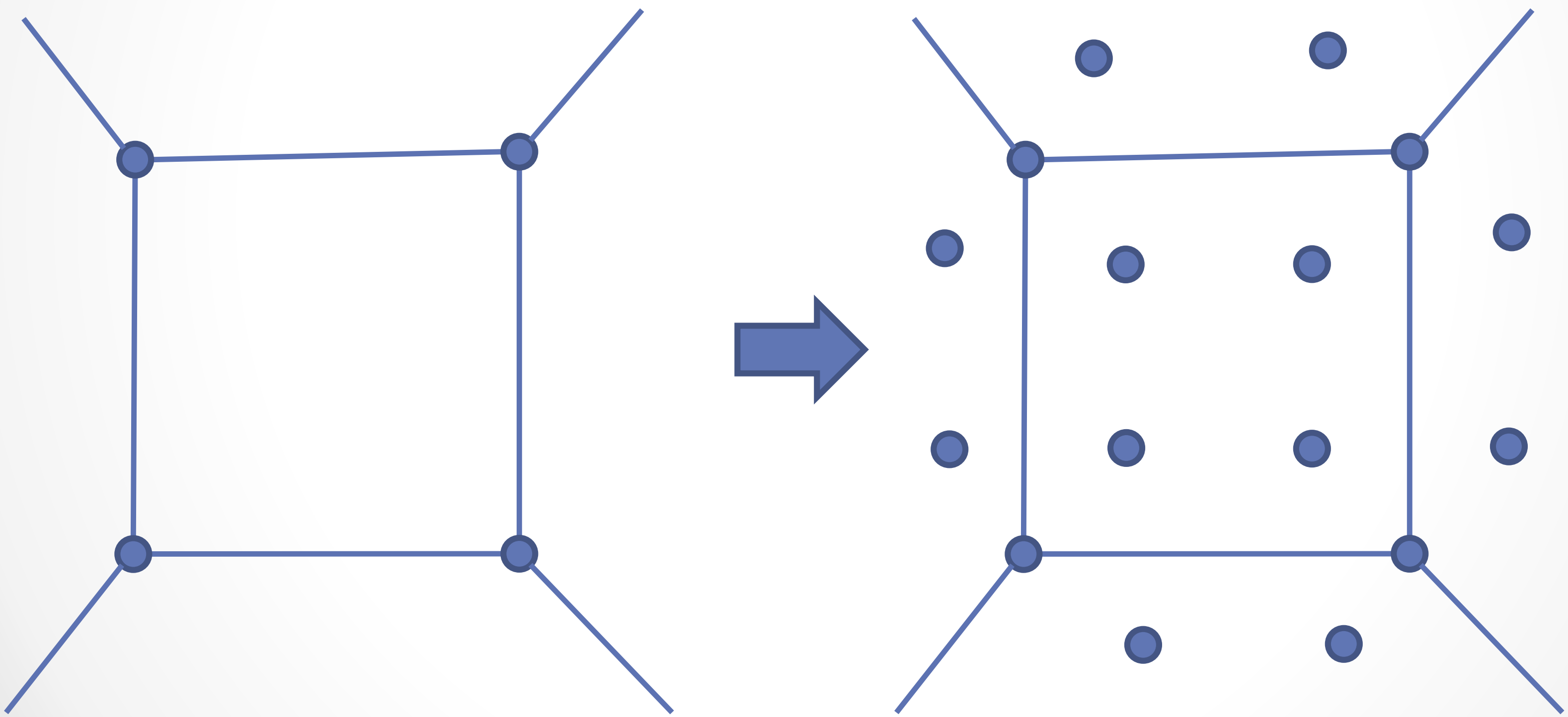


3. For each face, connect the new vertex points along edges.
4. For each original vertex (e.g., V_1), connect the new vertices for faces that are adjacent to this vertex.

Doo-Sabin Subdivision

Iteratively generates *n* vertices (for *n*-gons).

E.g., a side of a cube.

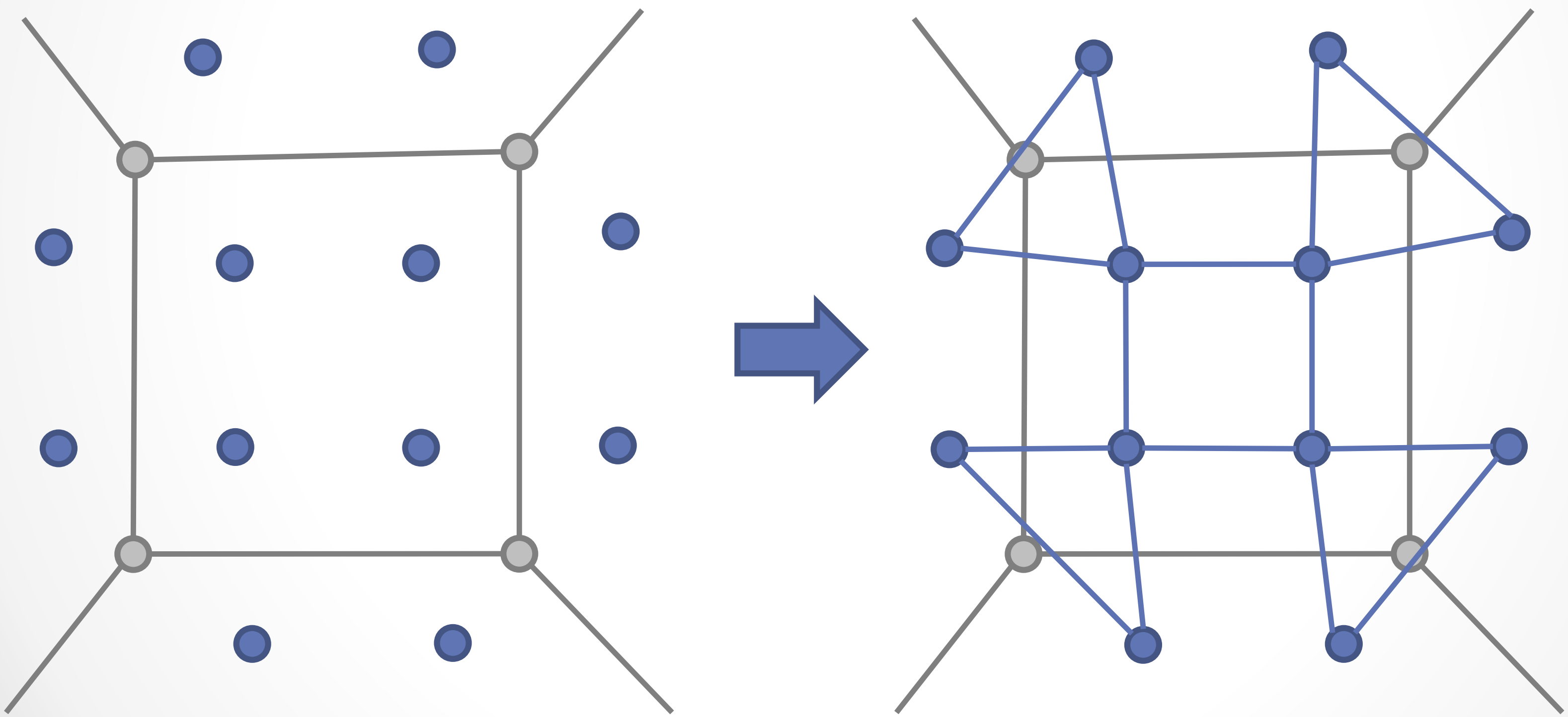


E.g., a side of a cube

Doo-Sabin Subdivision

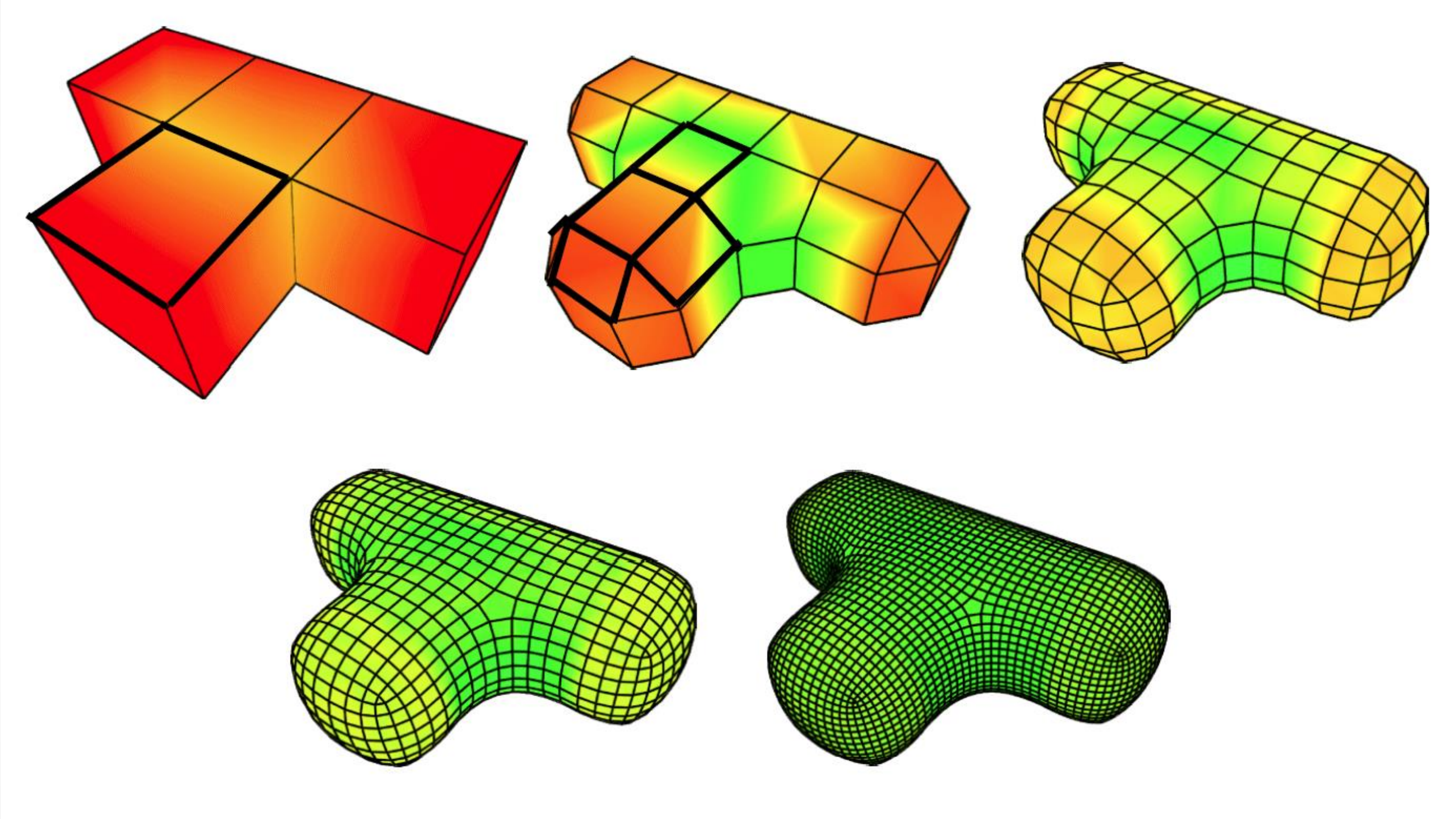
Iteratively generates *n* vertices (for *n*-gons).

E.g., a side of a cube.



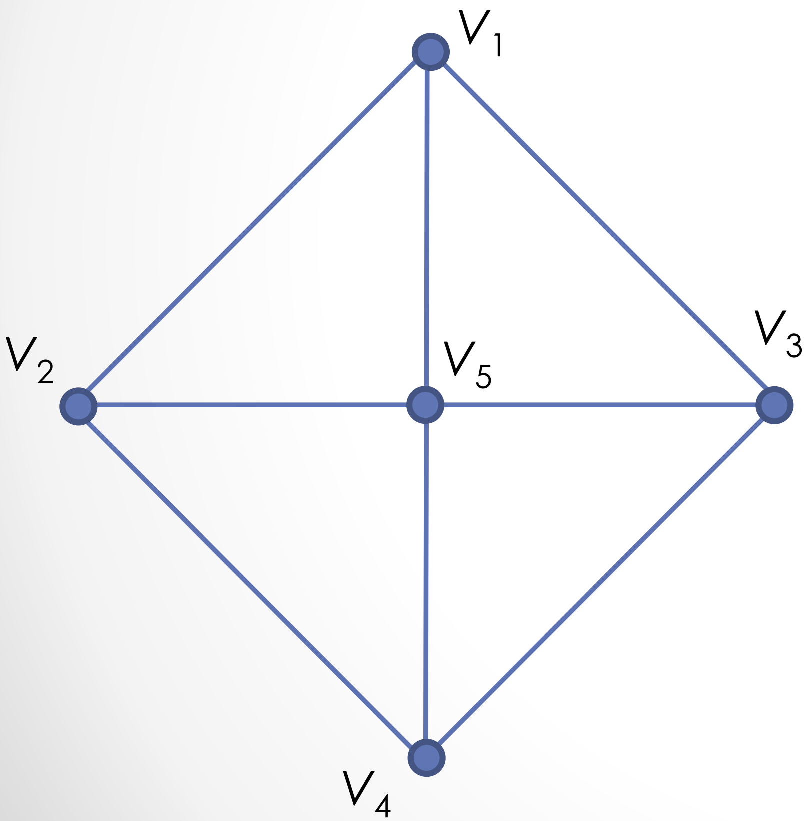
E.g., a side of a cube

Doo-Sabin Subdivision



- All vertices have valence four.
- Triangular facets in the corners: become extraordinary points in the limit (C0 continuous only).

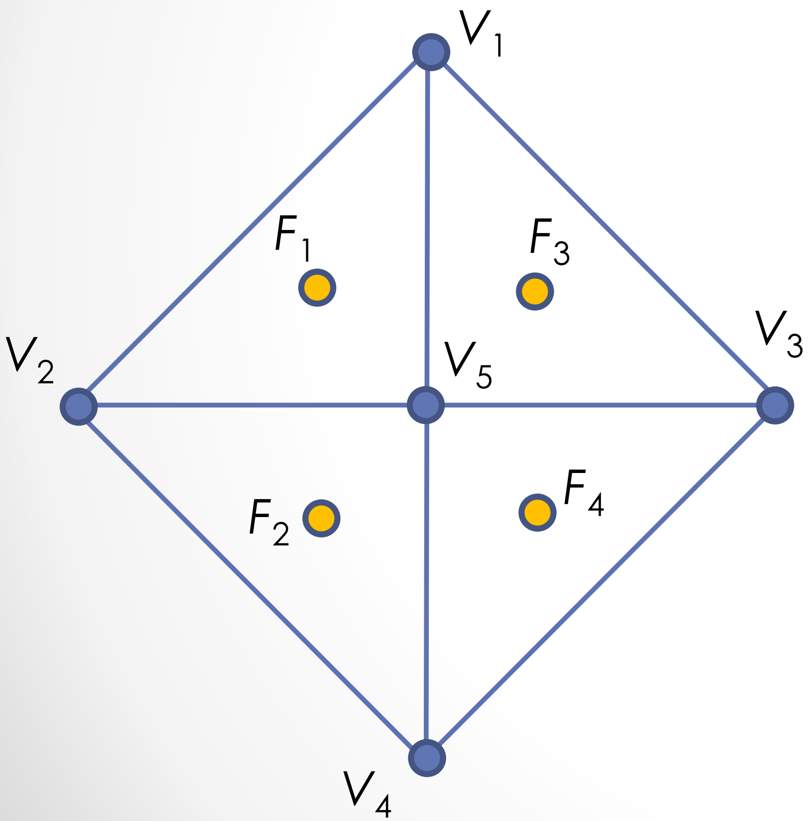
Catmull-Clark Subdivision



Iteratively add three types of points

- Face points *F*.
- Edge points *E*.
- Vertex points *V*.

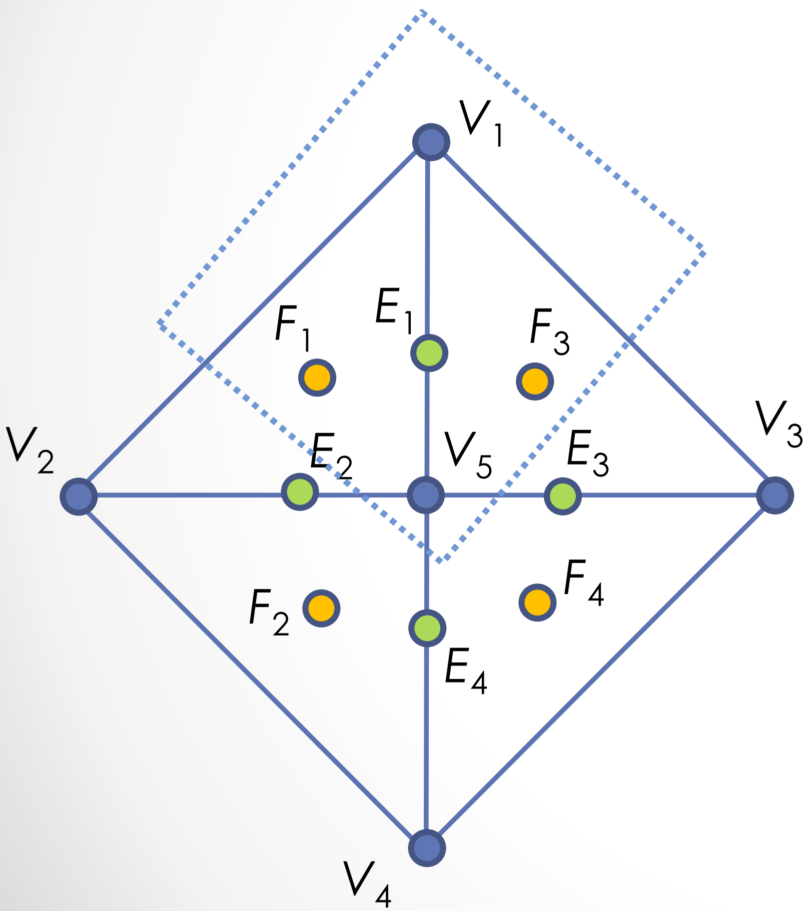
Catmull-Clark Subdivision



1. Add new *face points*: average of the original points in each face, e.g., $F_1 = (V_1 + V_2 + V_5)/3$.

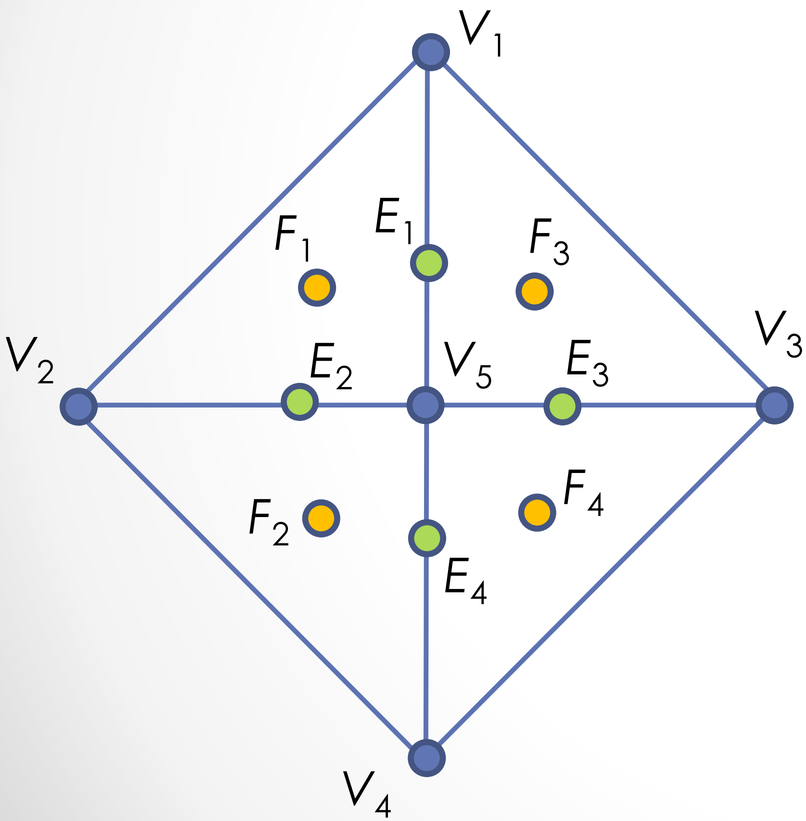
A face point is the centroid of a face.

Catmull-Clark Subdivision



1. Add new *face points*: average of the original points in each face, e.g., $F_1 = (V_1 + V_2 + V_5)/3$.
2. Add new edge points: average of the original end points + two face neighbors, e.g., $E_1 = (V_1 + V_5 + F_1 + F_3)/4$.

Catmull-Clark Subdivision



3. Add new *vertex points*: For each V (with n incident edges),

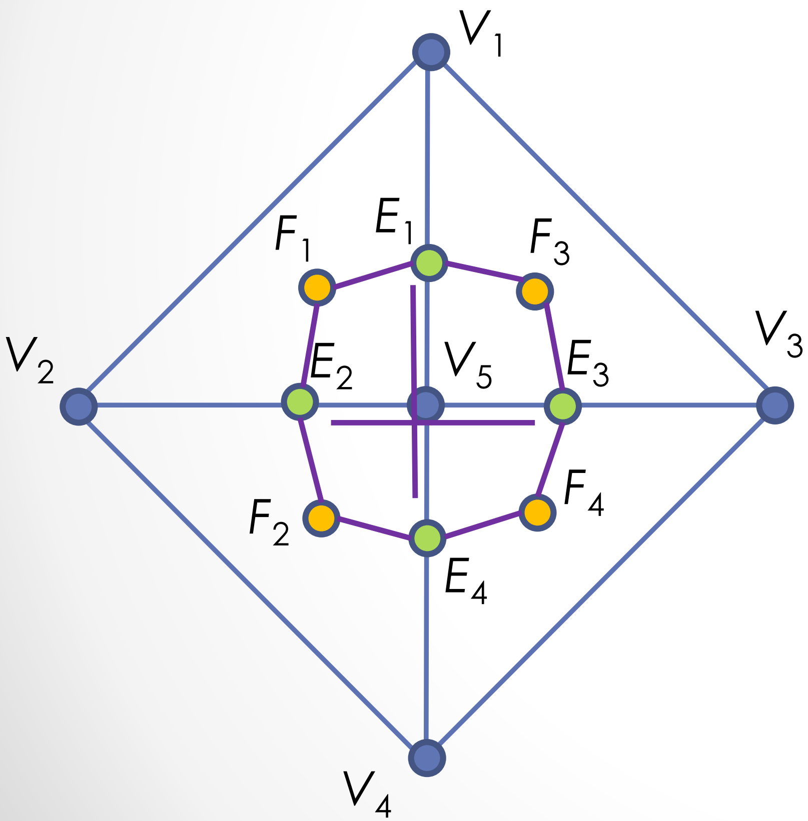
$$V' = \frac{(Q + 2R + (n-3)V)}{n}$$

Q: average of new face points for faces adjacent to V

R: average of midpoints of n edges,.

e.g.,$V'_5 = V_5$.

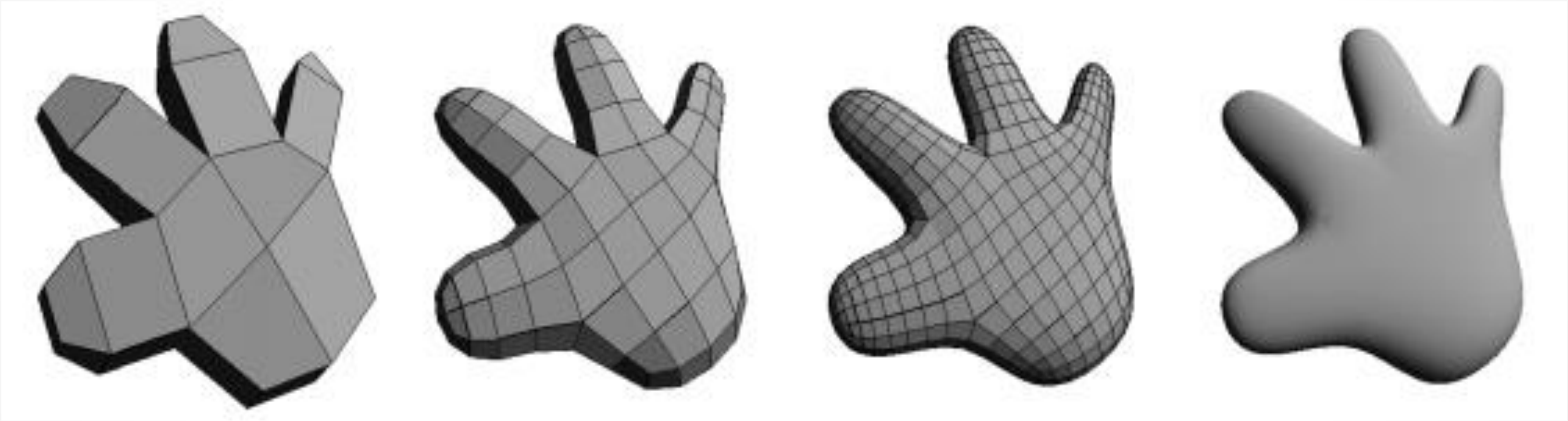
Catmull-Clark Subdivision



4. Connect each new face point *F* to new edge points on the boundary of *F*.
5. Connect each new vertex point V to new neighboring edge points

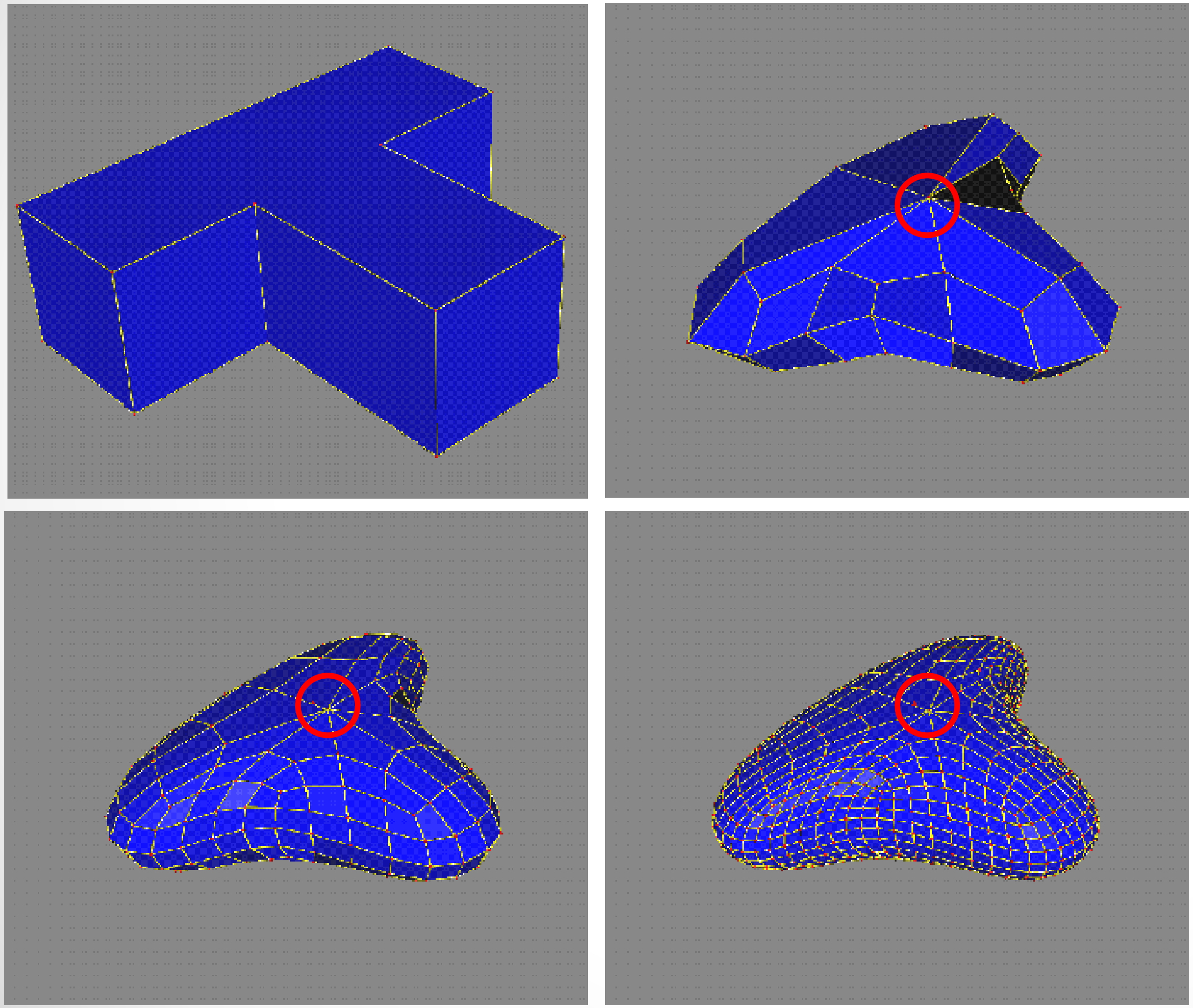
Catmull-Clark Subdivision

Examples



[Suzuki, *Journal of the Japan Society of Mechanical Engineers*, 2001]

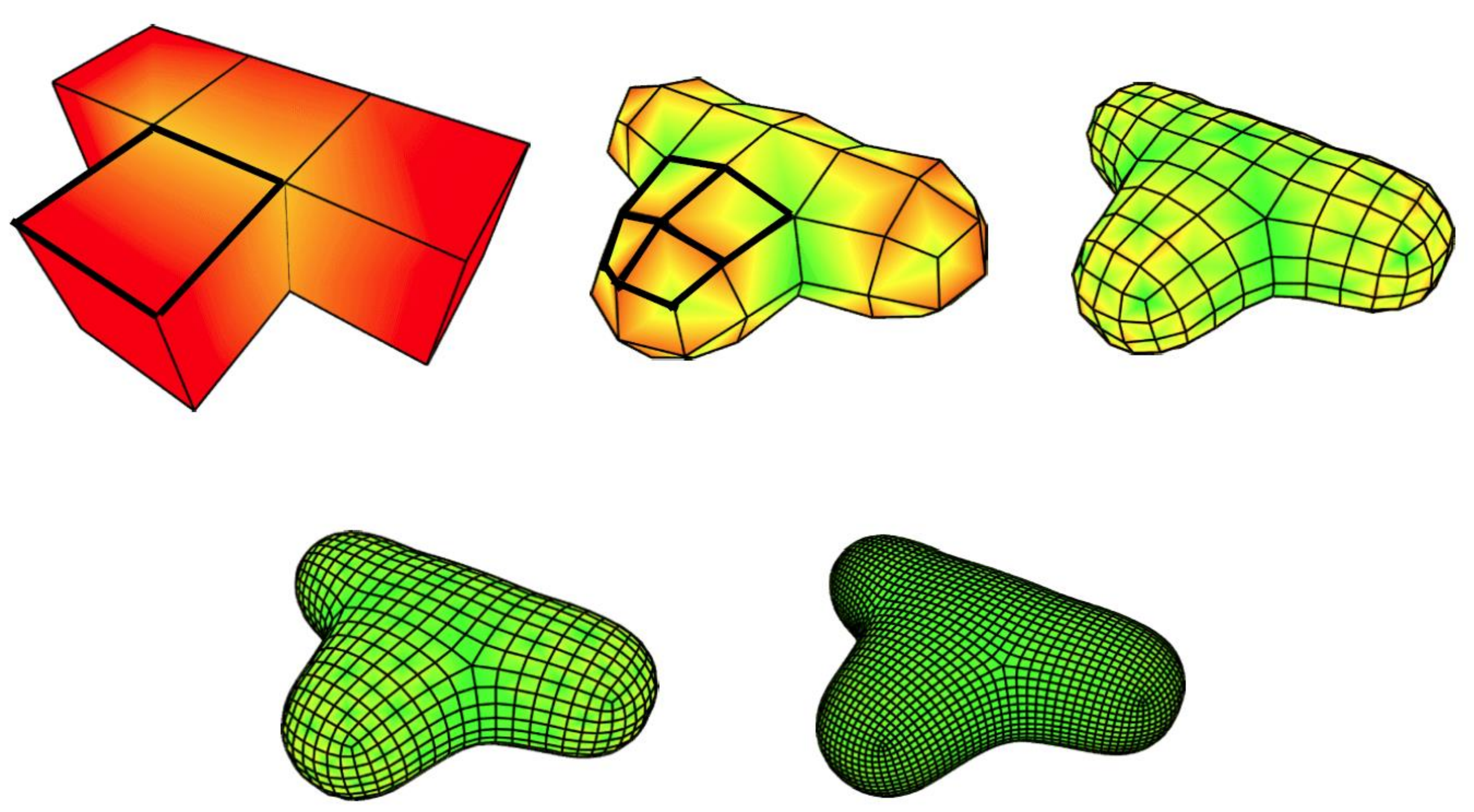
Catmull-Clark Subdivision



Extra-ordinary points

- Valence $\neq 4$.
- Less smooth.

Catmull-Clark Subdivision

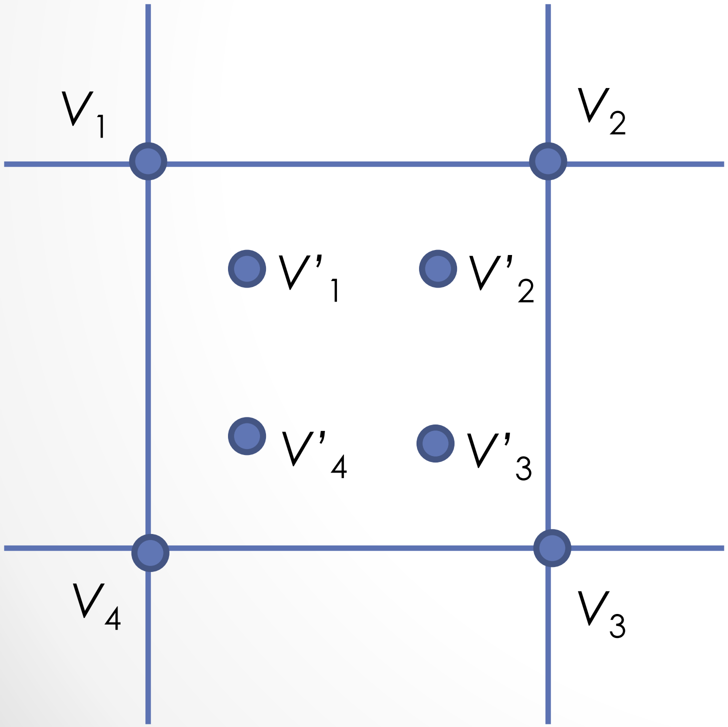


Extraordinary point if valence is not equal to 4 (C1 continuous only)

Convex Combinations

D-S and C-C use convex combinations:

Each new point is a weighted combination of existing points (weights total to 1).



E.g., in D-S, $V'_i = \sum_{j=1}^{n} w_{ij} V_j$

$\Rightarrow \sum_{j=1}^{n} w_{ij} = 1$

can be verified for quad, general n-gon also true

Convex Combinations

D-S and C-C use convex combinations:

Each new point is a weighted combination of existing points (weights total to 1).

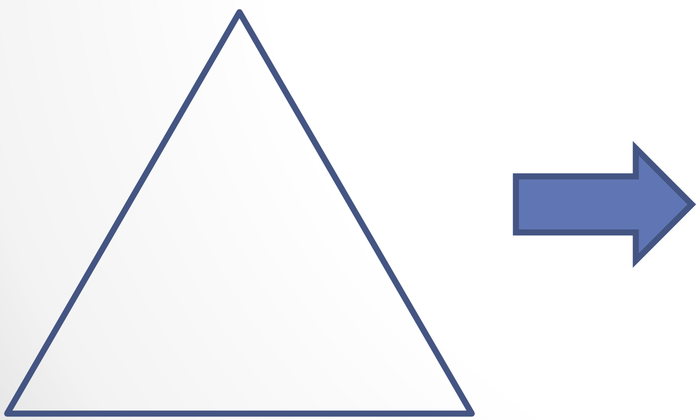
Guarantees

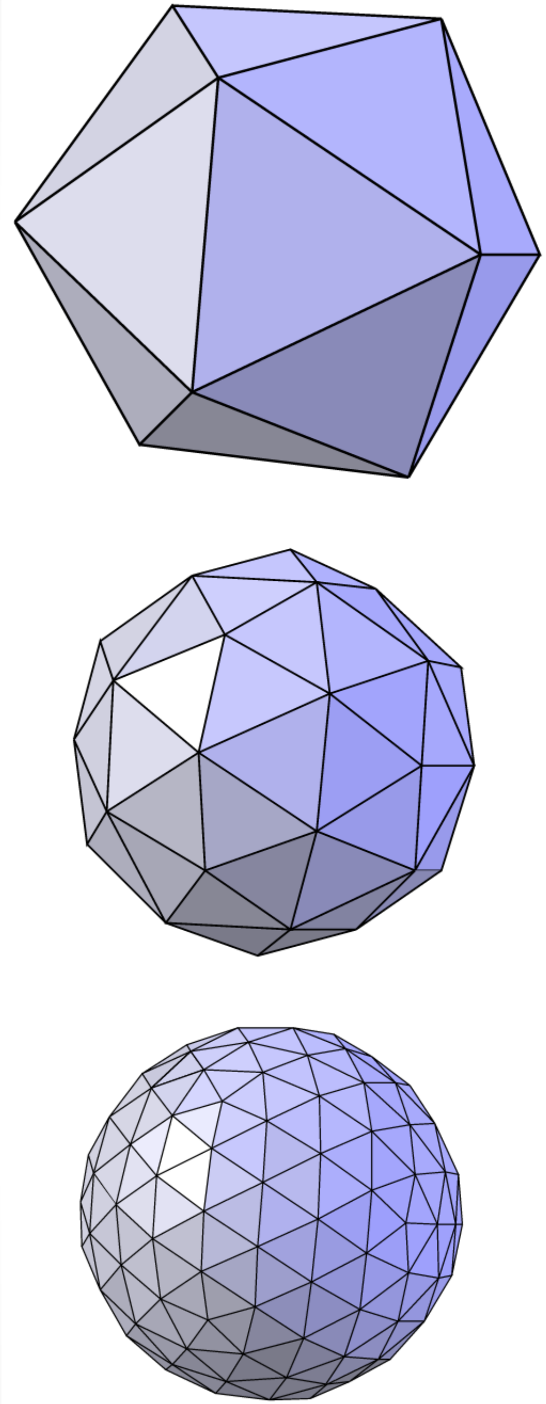
- New points in convex hull of old
- Local control
- Affinely invariant:

  affine trans. then subdivision = subdivision then affine trans.

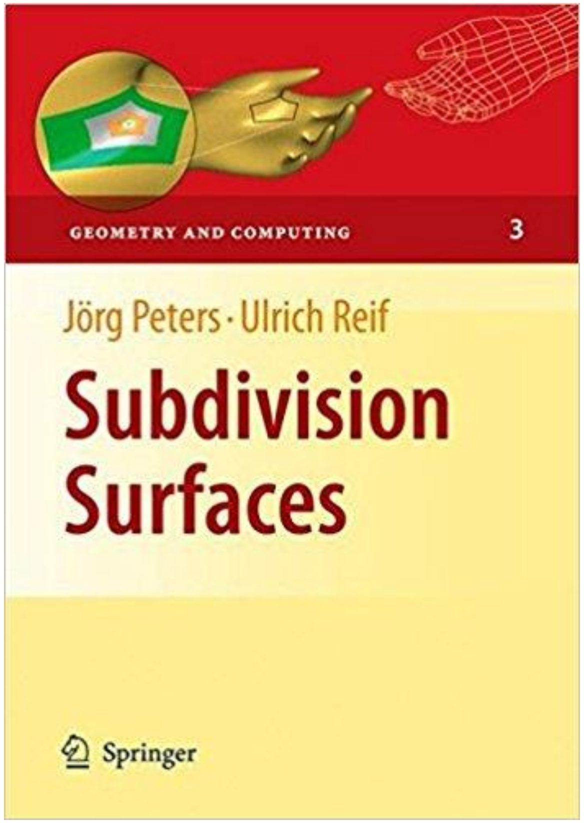
Loop Subdivision

- Named after Charles Loop.
- Applies to triangles.
- Split each triangle into 4 triangles.
- Can form a sphere from an icosahedron.





More Algorithms and Proofs



Overview

- What is subdivision?
- Curve subdivision algorithm
  - Chaiken’s algorithm (for curves)
- Surface subdivision algorithms
  - Doo-Sabin algorithm
  - Catmull-Clark algorithm
  - Loop algorithm
- **Advantages/disadvantages**

Pros and Cons

- Pros
  - Easy to make complex geometry with arbitrary topology
  - Supports multiresolution and efficient rendering/processing
- Cons
  - Precision difficult to specify in general (cartoon characters would be ok)

Used in Disney's *A Bug's Life*, *Finding Nemo*, and *The Incredibles*.

References

- Ken Joy's lecture notes: subdivision
  http://graphics.cs.ucdavis.edu/~joy/GeometricModelingLectures/Unit-9/Unit9.html
  [Geometric modeling lectures: http://graphics.cs.ucdavis.edu/~joy/GeometricModelingLectures/]

- Steve Marschner's lecture slides
  http://www.cs.cornell.edu/courses/Cs4620/2013fa/lectures/18subdivision.pdf

- NYU Media Lab's subdivision project web
  http://www.mrl.nyu.edu/projects/subdivision/

- Caltech Multi-Res Modeling Group's online demo
  http://www.multires.caltech.edu/teaching/demos/java/chaikin.htm