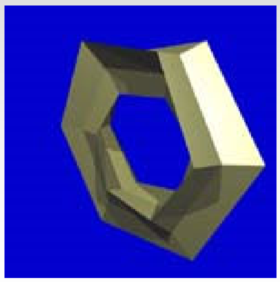
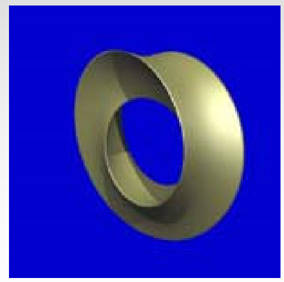
Subdivision Surfaces
or
How to Generate a Smooth Mesh?



?
→



Paper presentations (10% of final grade)

- 15 min presentation + 15min discussion
- Can reuse original author material
  - *Adjust to fit 15min*
  - *Separate core from details*
  - *Can ask me for early feedback (**recommended**)*
- Discussion (50+% of grade)
  - *Summarize/Analize core-ideas/contribution/limitations*
  - *Be ready to answer detailed questions*
- Grade partly based on peer feedback
  - *Upload (private) on piazza after each presentation*
- Everyone (not just presenter) **MUST** read paper
  - *Prepare at least one question/comment for the discussion*

Presentations

- Start 1st week of February
- 2% Bonus for first two presenters

- **Contact me ASAP with your paper/topic ideas**

Grading

10% - Participation

- Regular classroom participation (ask, answer, opine)

- Paper discussions
  - *Reading presented papers (before presentation)*
  - *Be ready to ask at least one question/make a comment*
  - *Goal: Learn critical reading*

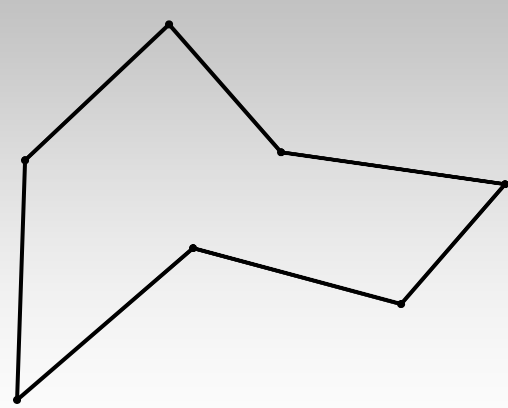
- Peer feedback for presentations (piazza, private)

Subdivision Curves and Surfaces

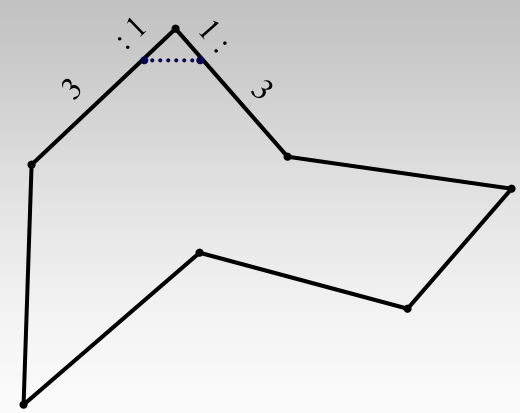
Subdivision – given polyline(2D)/mesh(3D) recursively modify & add vertices to achieve smooth curve/surface

- Each iteration generates smoother + more refined mesh

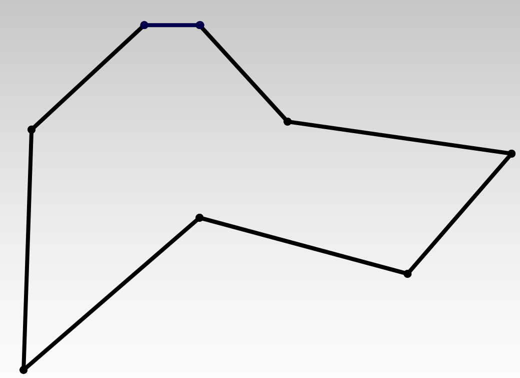
Corner Cutting



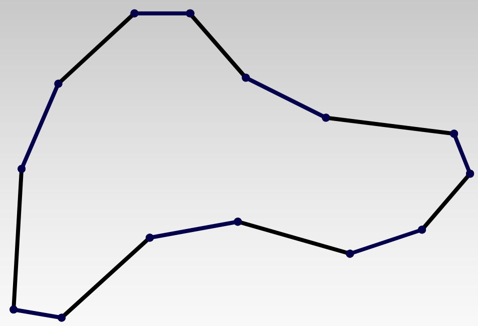
Corner Cutting



Corner Cutting

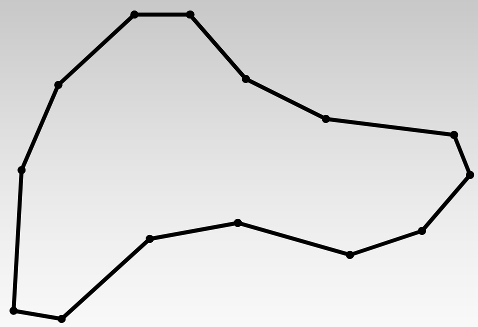


Corner Cutting



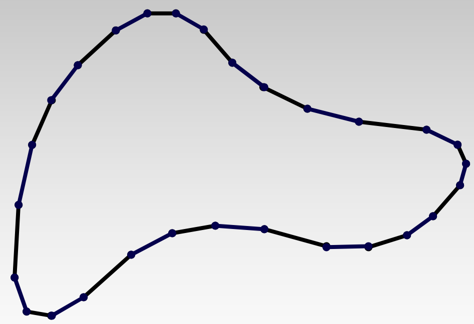
© Alla Sheffer

Corner Cutting

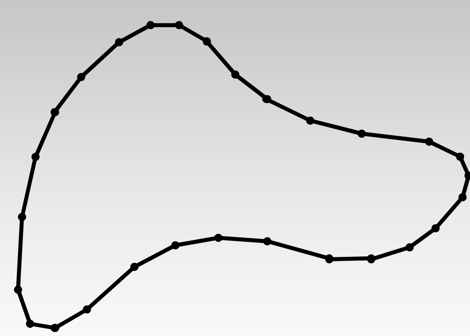


© Alla Sheffer

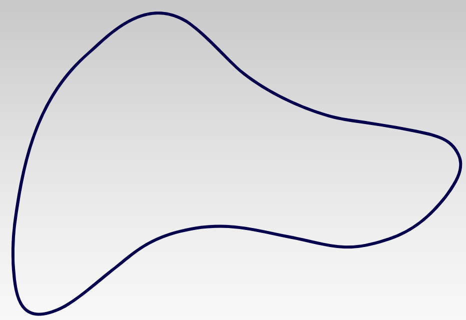
Corner Cutting



Corner Cutting

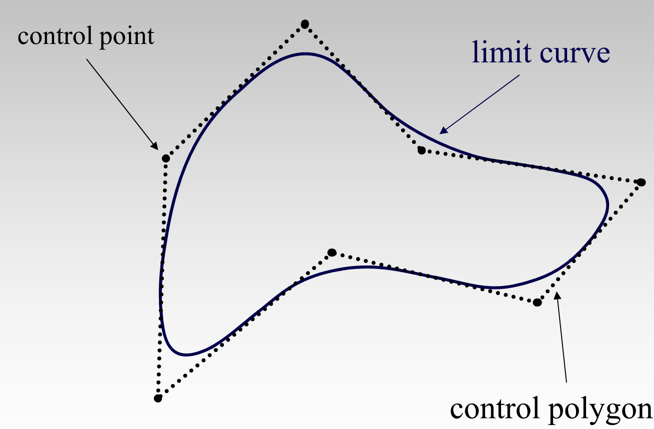


Corner Cutting



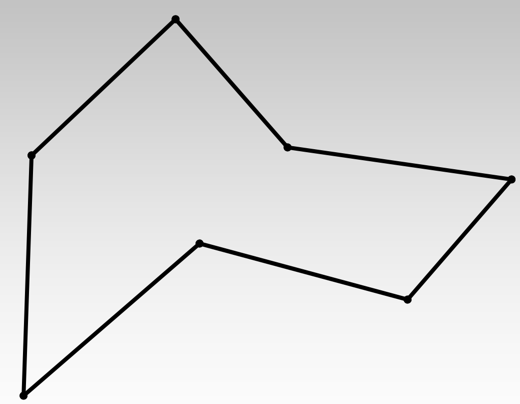
© Alla Sheffer

Corner Cutting



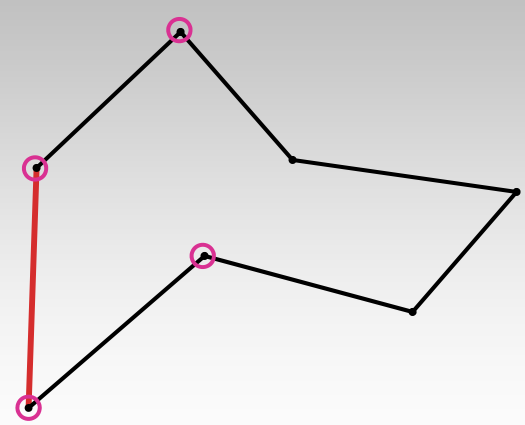
© Alla Sheffer

The 4-point scheme



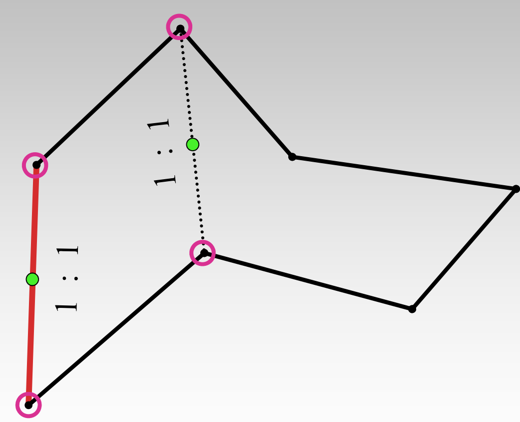
© Alla Sheffer

The 4-point scheme

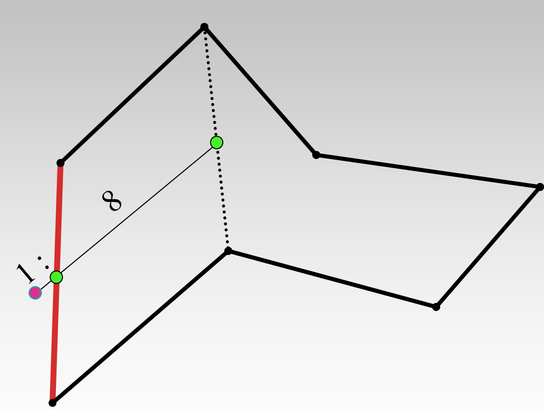


© Alla Sheffer

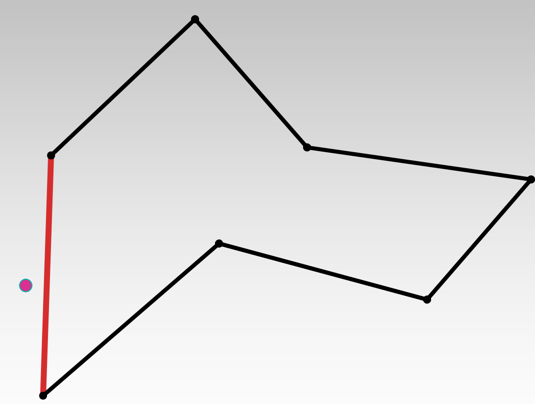
The 4-point scheme



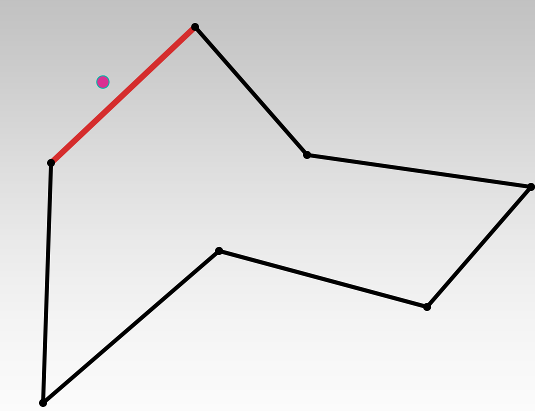
The 4-point scheme



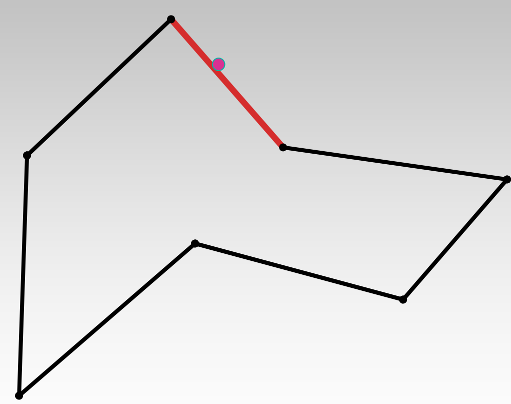
The 4-point scheme



The 4-point scheme

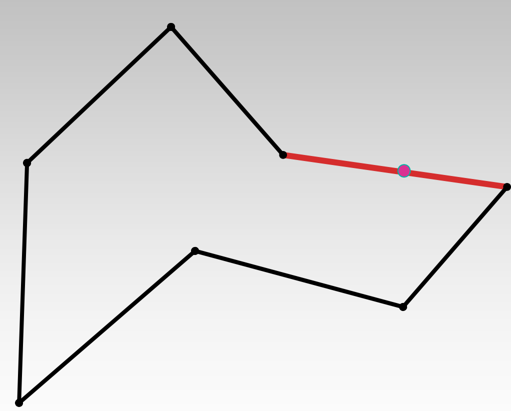


The 4-point scheme



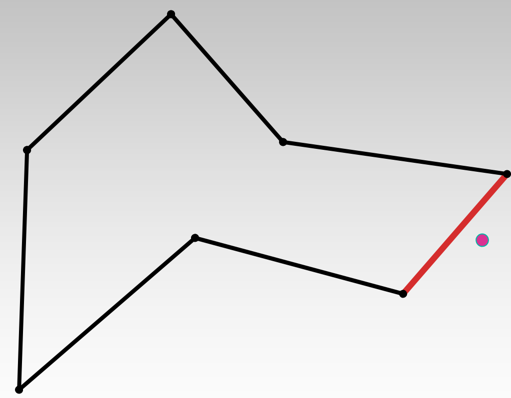
© Alla Sheffer

The 4-point scheme

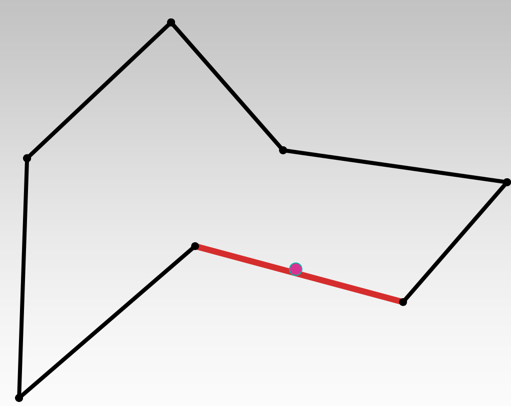


© Alla Sheffer

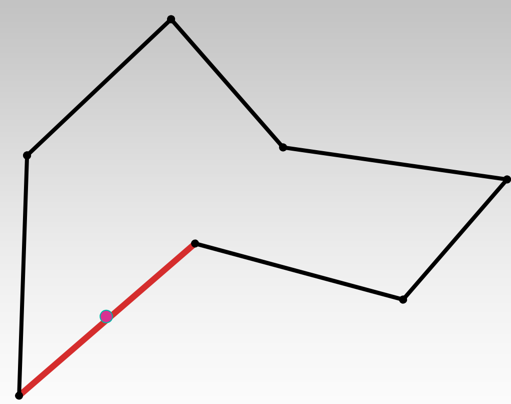
The 4-point scheme



The 4-point scheme

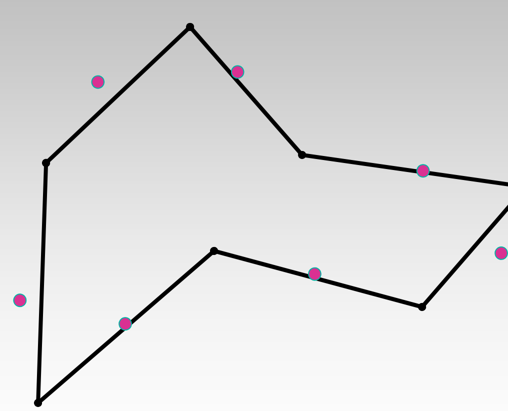


The 4-point scheme



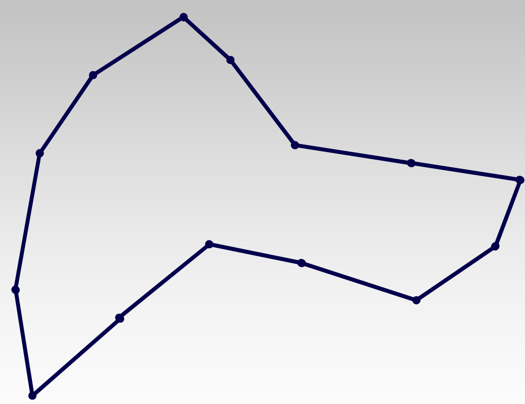
© Alla Sheffer

The 4-point scheme



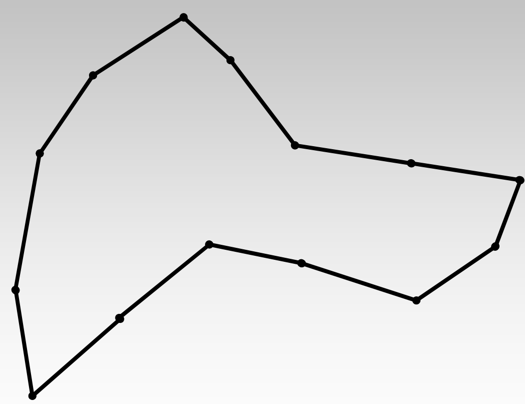
© Alla Sheffer

The 4-point scheme



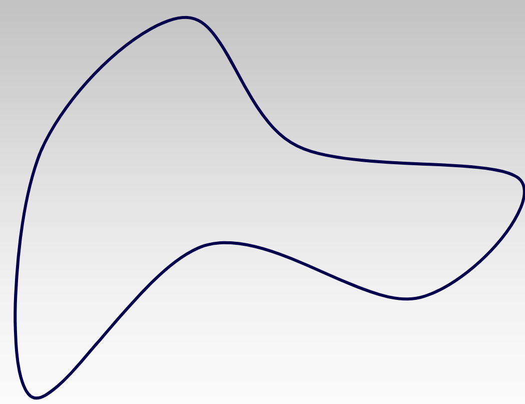
© Alla Sheffer

The 4-point scheme

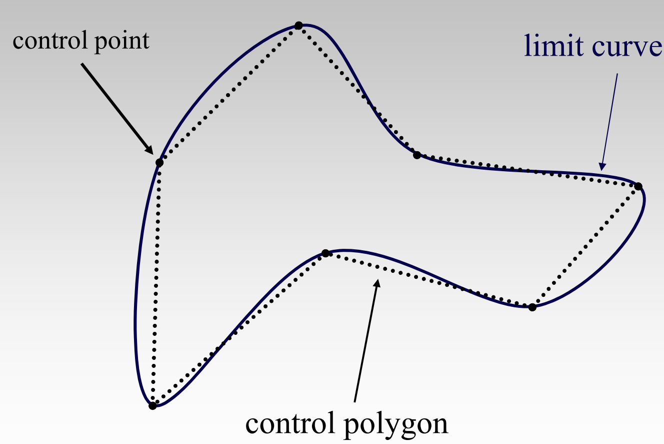


© Alla Sheffer

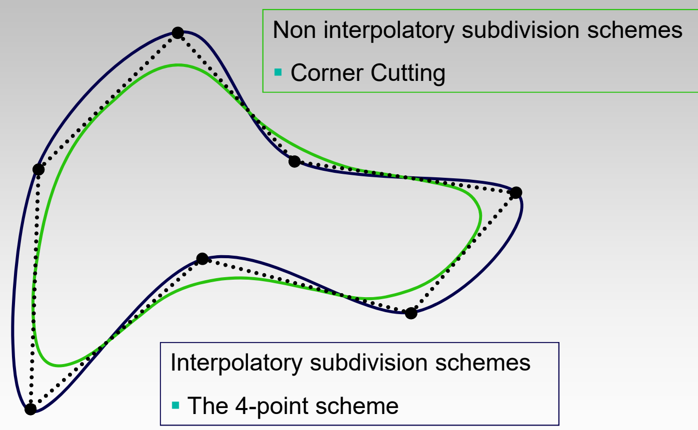
The 4-point scheme



The 4-point scheme



Subdivision curves



Basic concepts of Subdivision

***Subdivision curve* – limit of recursive subdivision applied to given polygon**

Each iteration

- Increase number of vertices (approximately) * 2

Initial polygon - control polygon

Central questions:

- Convergence: Given a subdivision operator and a control polygon, does the subdivision process converge?
- Smoothness: Does subdivision converge to smooth curve?

Subdivision schemes for surfaces

Each iteration

- Subdivision refines *control net* (mesh)
- Increase number of vertices (approximately) * 4

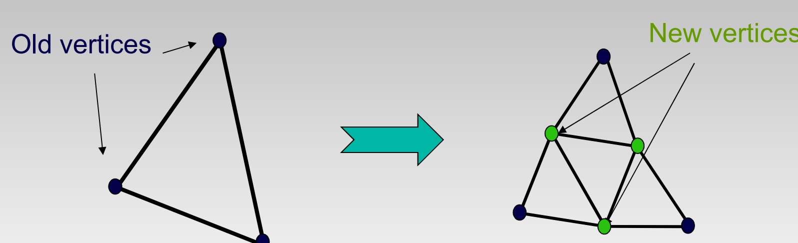
Mesh vertices converge to limit surface

Every subdivision method has:

- Method to generate net topology
- rules to calculate location of new vertices

Triangular subdivision

Defined for triangular meshes (control nets)



Every face replaced by 4 new triangular faces

Two kinds of new vertices:

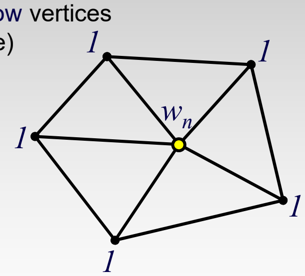
- Green vertices are associated with old edges
- Yellow vertices are associated with old vertices

Loop's scheme

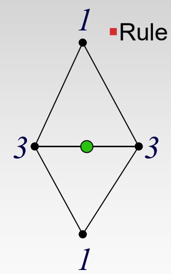
New vertex is weighted average of old vertices

List of weights called subdivision mask or stencil

- Rule for new yellow vertices (n – vertex valence)

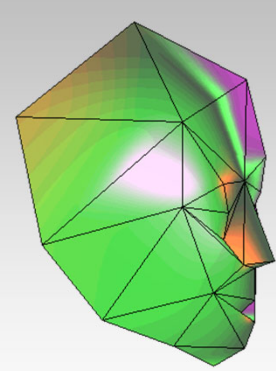


- Rule for new green vertices

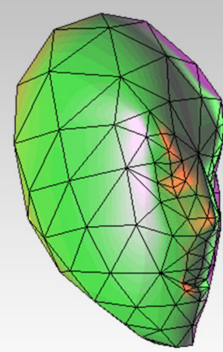


$$w_n = \frac{64n}{40 - (3 + 2\cos(2\pi/n))^2} - n$$

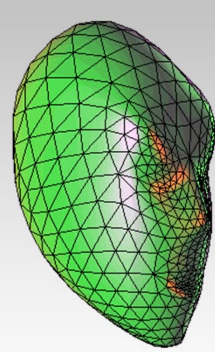
The original control net



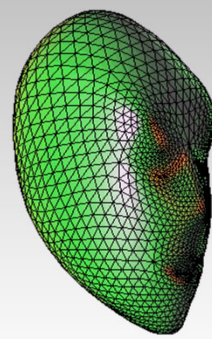
After 1st iteration



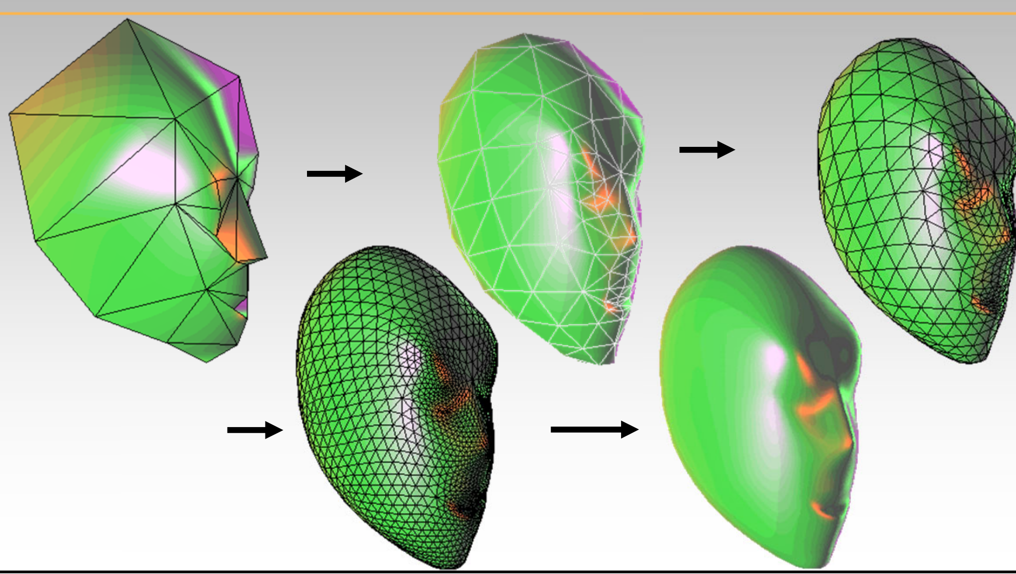
After 2nd iteration



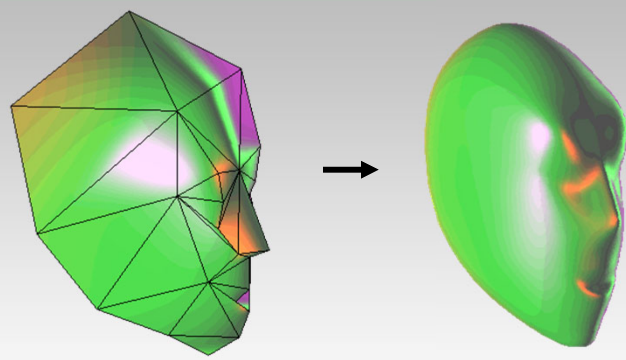
After 3rd iteration



Loop Scheme



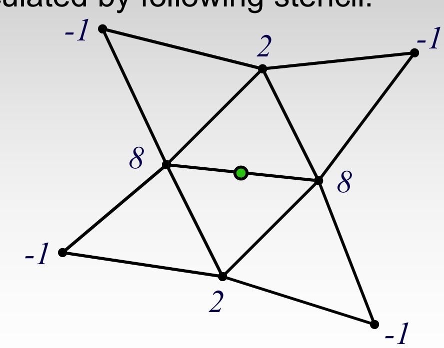
Loop Limit Surface



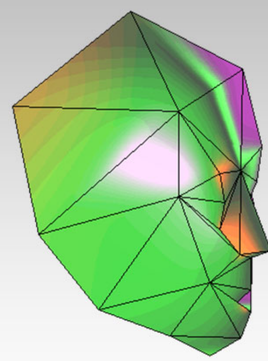
- Limit surfaces of Loop's subdivision is C^2 almost everywhere

Butterfly Scheme

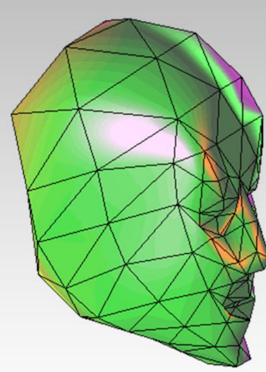
- Interpolatory scheme
- New yellow vertices inherit location of old vertices
- New green vertices calculated by following stencil:



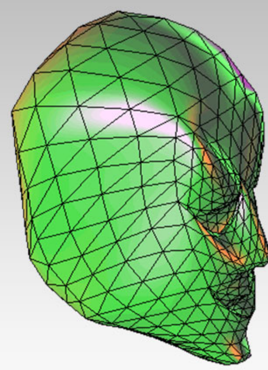
The original control net



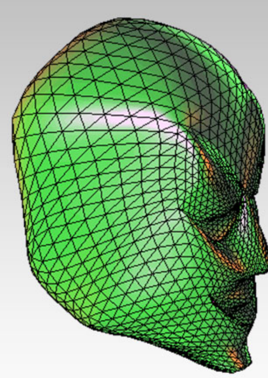
After 1st iteration



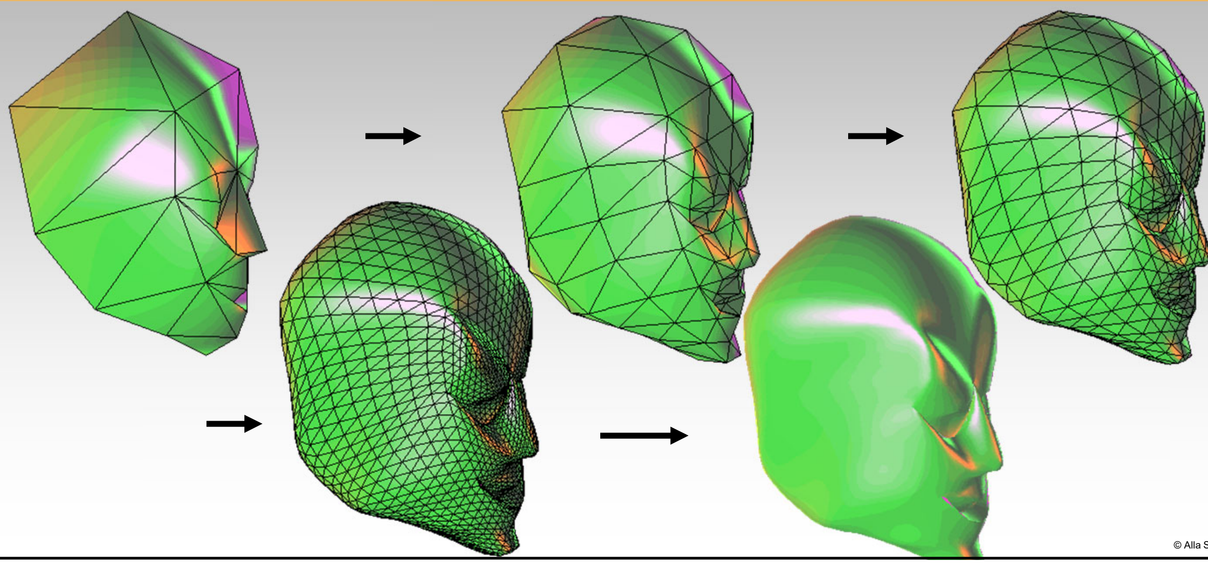
After 2nd iteration



After 3rd iteration

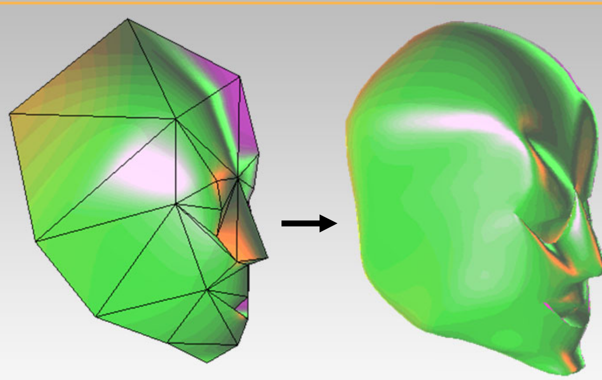


Butterfly Scheme



© Alla Sheffer

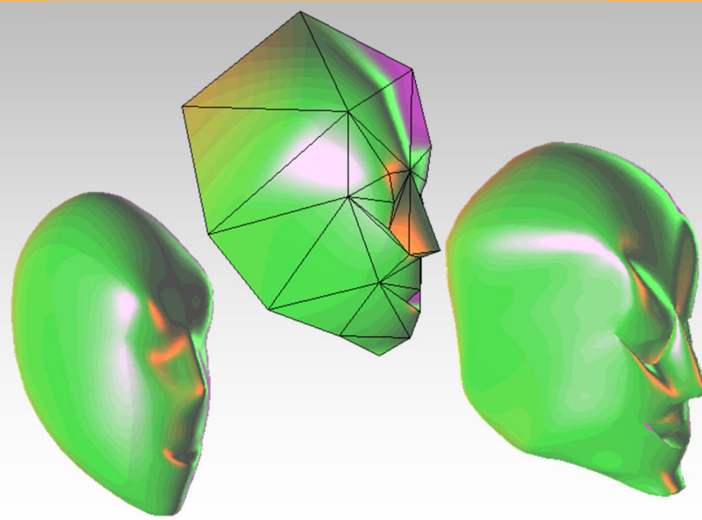
Butterfly limit surface



- Limit surfaces of Butterfly subdivision are C^1, but do not have second derivative

© Alla Sheffer

Comparison



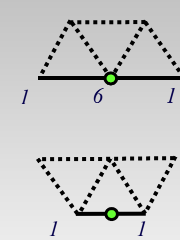
Boundaries & Features: Loop

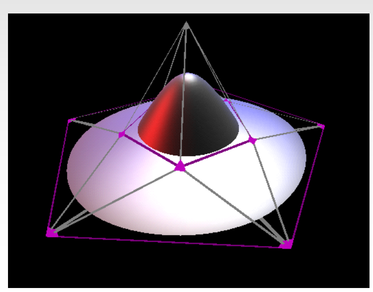
Boundary vertices – depend ONLY on other boundary vertices

Corner vertices left in place

- Sometimes modified rules for corner neighbors

Treat features (creases) like boundaries





Scheme Zoo

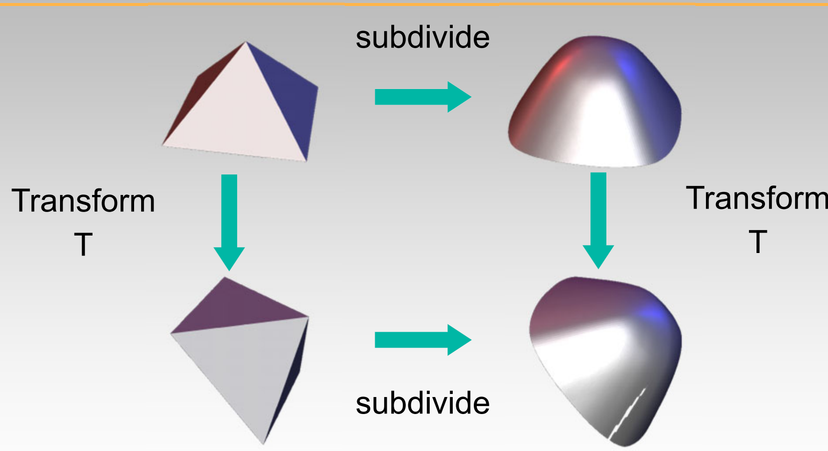
More schemes:

- Catmul-Clark
- Kobbelt
- Duo-Sabin
- …

Proving scheme works:

- **Convergence**
- Degree of continuity
- Affine invariance

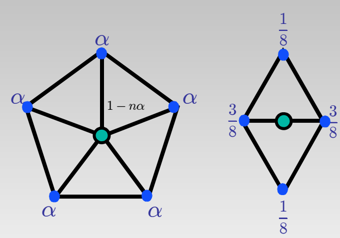
Affine Invariance



- Coefficients of masks sum to 1 –weighted average

Affine Invariance

Coefficients of masks must sum to 1



$$p = \sum a_i p_i$$

displacement

$$\sum a_i (p_i + \boxed{t}) = \underbrace{\left(\sum a_i\right)}_{1} \boxed{t} + p$$

Analysis of Subdivision

Test:

- Convergence
- Smoothness

Goals:

- Help to choose the rules
- Ensure that all surfaces have desired properties

Plan

- Define subdivision surfaces
- Relate properties to coefficients

Subdivision Matrix

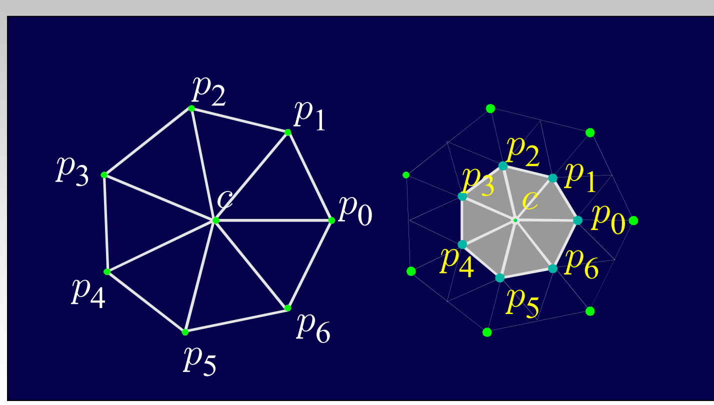
Relate control on finer level to coarser level

Useful for

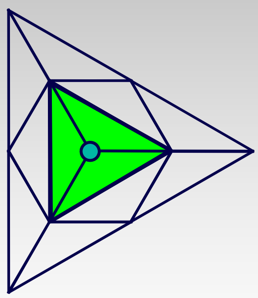
- Analysis of properties
  - *smoothness*
  - *affine invariance*
- Formulas for normals
- Explicit evaluation of surfaces at arbitrary points of the domain

Controls of K-Sided Patch

Simplest case - consider only 1-ring

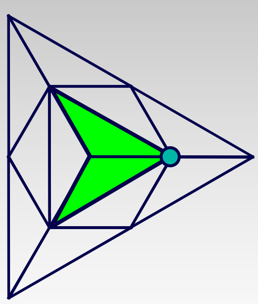


Subdivision Matrix



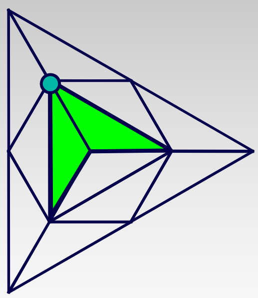
$$\left(\begin{array}{c|ccc} 7/16 & 3/16 & 3/16 & 3/16 \\ \hline 3/8 & 3/8 & 1/8 & 1/8 \\ 3/8 & 1/8 & 3/8 & 1/8 \\ 3/8 & 1/8 & 1/8 & 3/8 \end{array}\right)$$

Subdivision Matrix



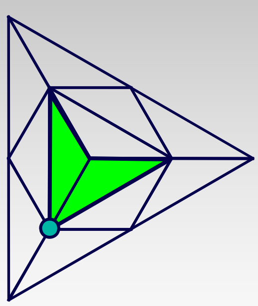
$$\left(\begin{array}{c|ccc} 7/16 & 3/16 & 3/16 & 3/16 \\ \hline 3/8 & 3/8 & 1/8 & 1/8 \\ 3/8 & 1/8 & 3/8 & 1/8 \\ 3/8 & 1/8 & 1/8 & 3/8 \end{array}\right)$$

Subdivision Matrix



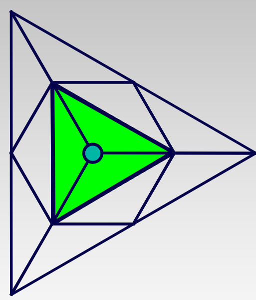
7/16	3/16	3/16	3/16
3/8	3/8	1/8	1/8
3/8	1/8	3/8	1/8
3/8	1/8	1/8	3/8

Subdivision Matrix



7/16	3/16	3/16	3/16
3/8	3/8	1/8	1/8
3/8	1/8	3/8	1/8
3/8	1/8	1/8	3/8

Eigen Values



$$\left(\begin{array}{c|ccc} 7/16 & 3/16 & 3/16 & 3/16 \\ \hline 3/8 & 3/8 & 1/8 & 1/8 \\ 3/8 & 1/8 & 3/8 & 1/8 \\ 3/8 & 1/8 & 1/8 & 3/8 \end{array}\right)$$

Eigenvalues
1
0.25
0.25
0.0625

Eigen Decomposition

Diagonalize subdivision matrix

- eigenvectors $x_i,\ i = 0..N$
- eigenvalues $\lambda_i,\ i = 0..N$

- p : vector of points in a neighborhood

$$p = \sum_i a_i x_i$$

(N+1)-vector of 3D points

3D vector

Eigenvectors

"Good" case:

- $\lambda_0 = 1$ & $|\lambda_i| < 1, i = 1, \ldots, n-1$

$$S^m p = a_0 x_0 + \lambda_1^m a_1 x_1 + \lambda_2^m a_2 x_2 + \lambda_3^m a_3 x_3 + \ldots$$

limit position

tangent vectors

- can make a_0 zero by moving control points (by affine invariance)

Subdominant Eigenvectors

Next higher order terms

- assume

$$\lambda = \lambda_1 = \lambda_2 > |\lambda_3|$$

- move control points so that $a_0 = 0$

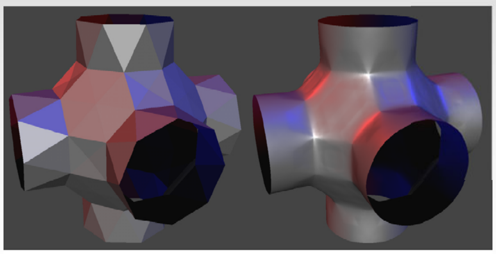
$$\frac{1}{\lambda^m} S^m p = (a_1 x_1 + a_2 x_2) + \left(\frac{\lambda_3}{\lambda}\right)^m x_3 + \ldots$$

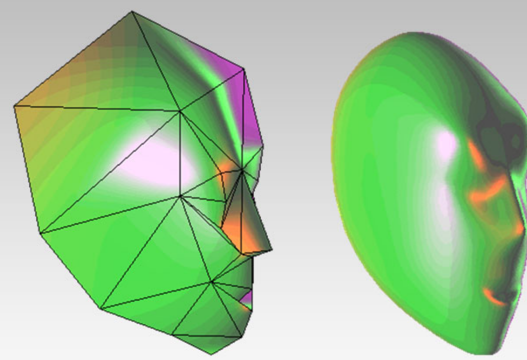
subdominant eigenvectors

vanishes

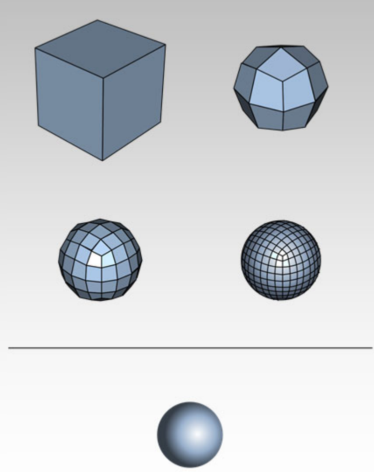
Subdivision Drawbacks

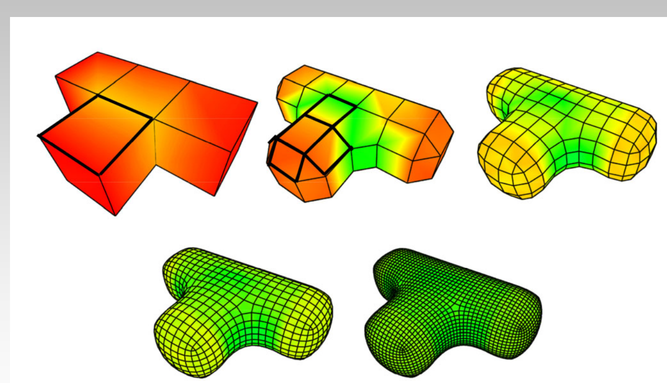
Not always intuitive
Can have artifacts
Hard to control


Initial mesh — Butterfly scheme interpolation



Quad: Catmull-Clark & Doo-Sabin





Properties

Works best on regular connectivity (valence 6)

Easy to implement

- **Efficiency is another matter**
- Use Eigen-analysis for exact computation

Local support – operations depend on local data

- Order does not matter
- Data structure support

Allow LOD

- View mesh at any level as control mesh
  - *Apply geometric modifications*

Continuous

- Scheme dependent

Geri's Game