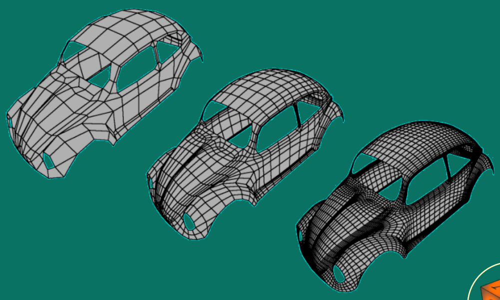
Implementing Subdivision Schemes



Implementing Subdivision Schemes

- data structures for meshes
- elementary operations
- adaptive subdivision
- multi-step rules

Polygonal Meshes

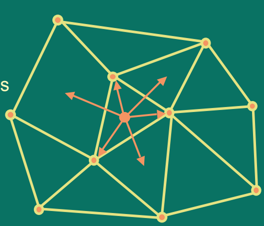
- $M = (\{p_i\},\{T_j\})$
- $p_i \in IR^3$ ... control points / positions
- $T_j \in IN^k$ ... sequence of vertex indices (k-gon)
- access:
  - navigation (topological neighbors)
  - modification (refinement)

Mesh Data Structures

Store topology
- Face based
- Edge based
- Halfedge based

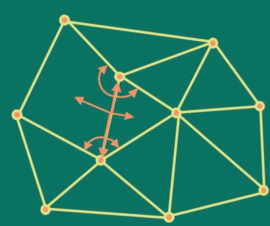
Mesh Data Structures

Store topology
- Face based:
  - n face vertices
  - n neighboring faces
- Edge based
- Halfedge based



Mesh Data Structures

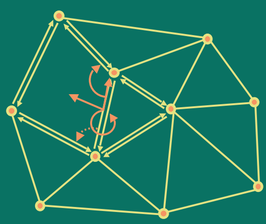
Store topology
- Face based
- Edge based:
  - 2 vertices
  - 2 incident faces
  - 4 edges
- Halfedge based



Mesh Data Structures

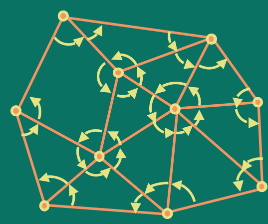
Store topology

- Face based
- Edge based
- Halfedge based:
  - 1 vertex
  - 1 face
  - Next halfedge
  - Opposite halfedge



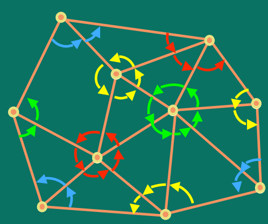
Half-Edge Data Structure

- public class Edge {
    Node A,B;
    Edge Next;
    Edge Prev;
  }



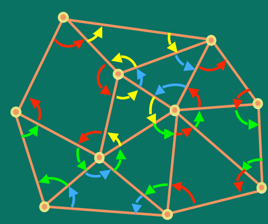
Half-Edge Data Structure

- public class Edge {
    Node A,B;
    Edge Next;
    Edge Prev;
  }



Half-Edge Data Structure

- public class Edge {
    Node A,B;
    Edge Next;
    Edge Prev;
  }

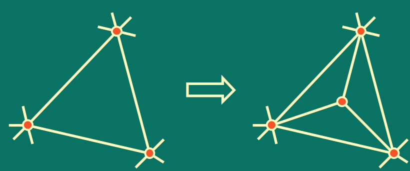


Topological Refinement Operators

- identify simple elementary operations which can be combined
- guarantee consistency of the structure at all intermediate stages
- OpenMesh (www.openmesh.org) provides an open-source implementation

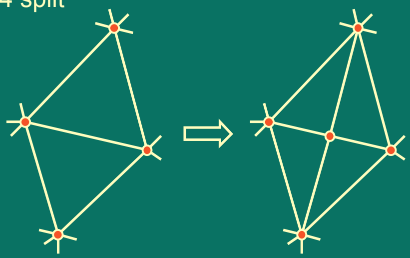
Elementary Operations

- 1-3 split



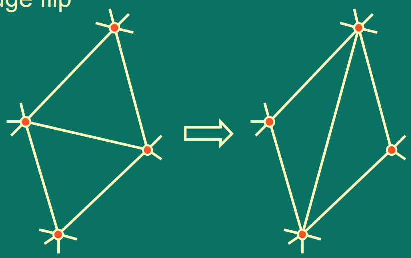
Elementary Operations

- 2-4 split



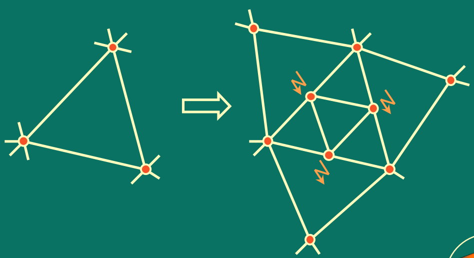
Elementary Operations

- edge flip

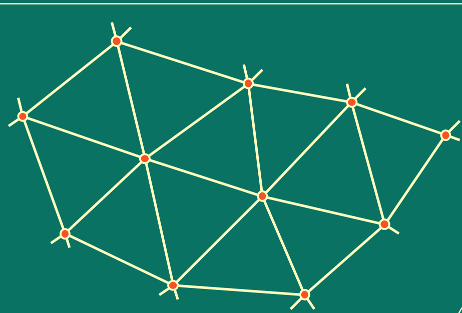


Elementary Operations

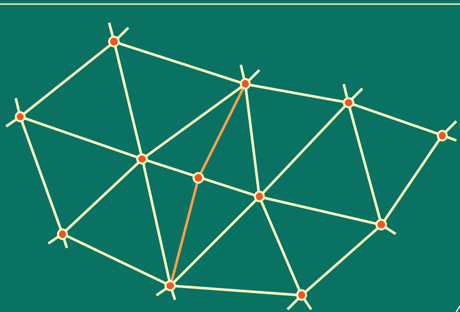
- 1-4 split



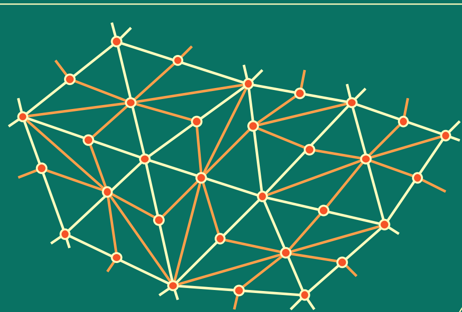
Uniform 1-4 Refinement



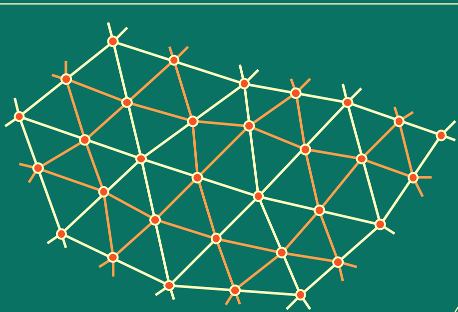
Uniform 1-4 Refinement



Uniform 1-4 Refinement

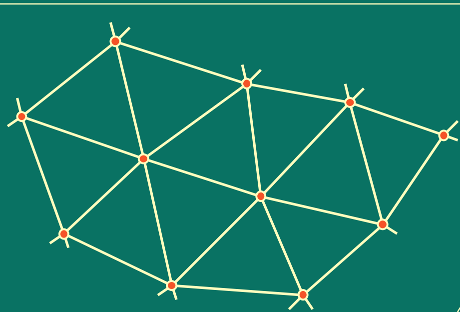


Uniform 1-4 Refinement



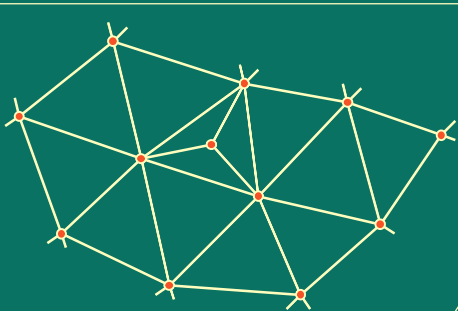
RWTH AACHEN 19 Computer Graphics Group Leif Kobbelt

Uniform √3 Refinement



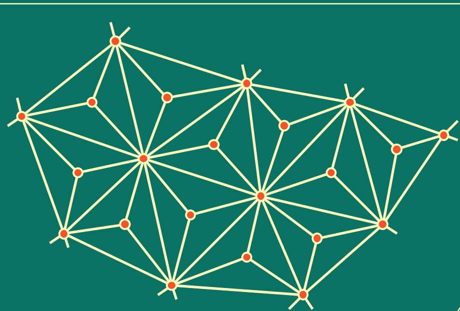
RWTH AACHEN 20 Computer Graphics Group Leif Kobbelt

Uniform √3 Refinement



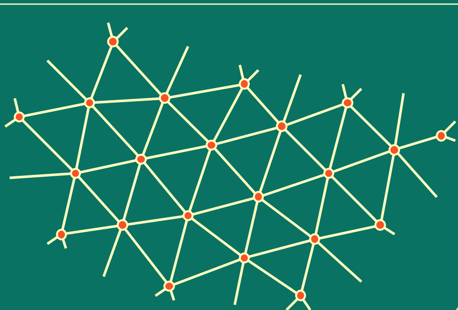
RWTH AACHEN 21 Computer Graphics Group Leif Kobbelt

Uniform √3 Refinement



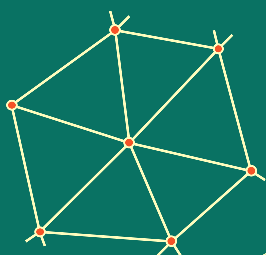
RWTH AACHEN 22 Computer Graphics Group Leif Kobbelt

Uniform √3 Refinement



RWTH AACHEN 23 Computer Graphics Group Leif Kobbelt

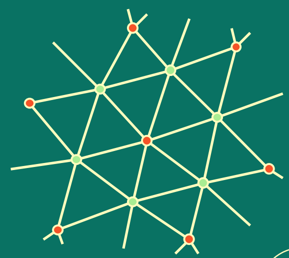
Vertex Positions



RWTH AACHEN 24 Computer Graphics Group Leif Kobbelt

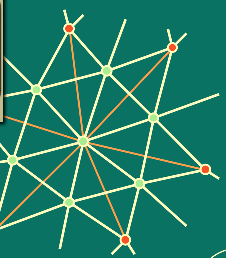
Vertex Positions

Vertex Positions



Vertex Positions

$$\begin{pmatrix} p_0^{n+1}\\ p_1^{n+1}\\ \vdots \\ p_k^{n+1}\end{pmatrix} = \begin{pmatrix} a & b & \cdots & \cdots & b\\ c & d_1 & d_2 & \cdots & d_k \\ c & d_k & d_1 & d_2 & \cdots \\ \vdots & & & \ddots & \\ c & d_2 & \cdots & d_k & d_1\end{pmatrix}\begin{pmatrix} p_0^n \\ p_1^n \\ \vdots \\ p_k^n\end{pmatrix}$$



Vertex Positions

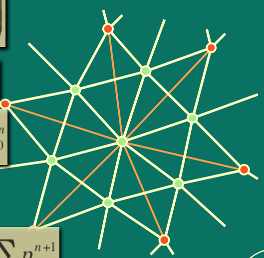
$$\begin{pmatrix} p_0^{n+1}\\ \sum p_i^{n+1}\end{pmatrix} = \begin{pmatrix} a & b \\ kc & d\end{pmatrix}\begin{pmatrix} p_0^n \\ \sum p_i^n\end{pmatrix}$$

$$\sum p_i^{n+1} = kc\, p_0^n + d \sum p_i^n$$

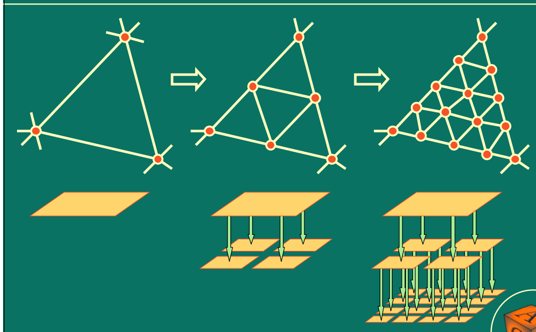
$$\sum p_i^n = \frac{1}{d}\sum p_i^{n+1} - \frac{kc}{d} p_0^n$$

$$p_0^{n+1} = a\, p_0^n + b \sum p_i^n$$

$$p_0^{n+1} = \frac{ad - kbc}{d} p_0^n + \frac{b}{d}\sum p_i^{n+1}$$

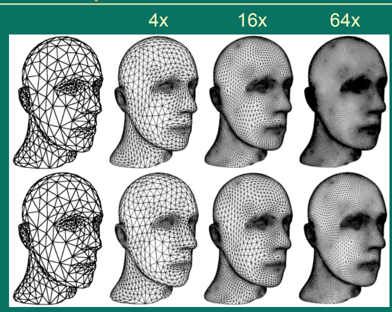


Hierarchical Data Structures

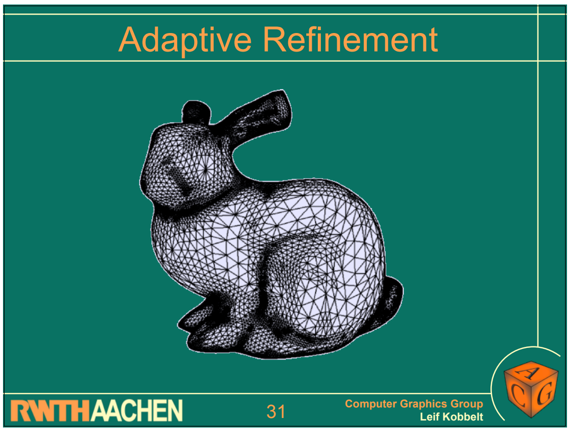


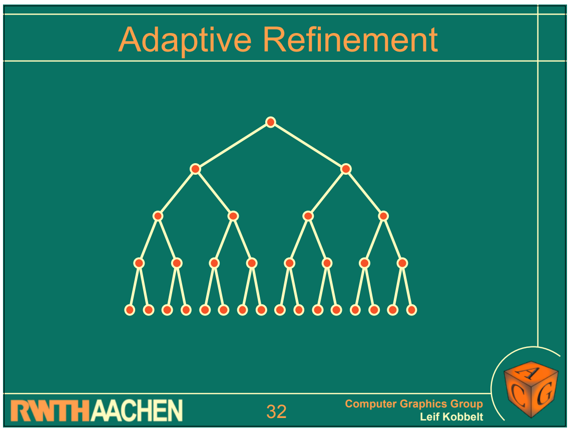
Exponential Growth

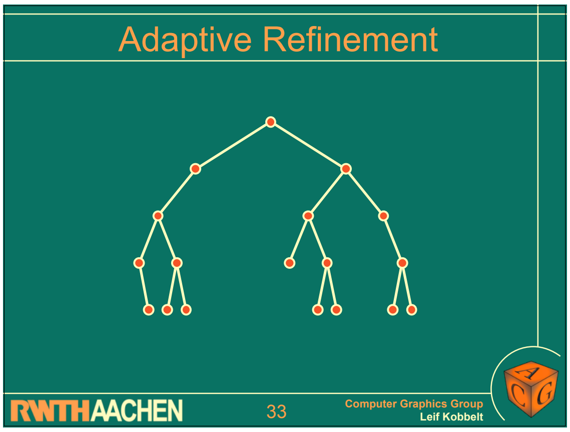
4x 16x 64x

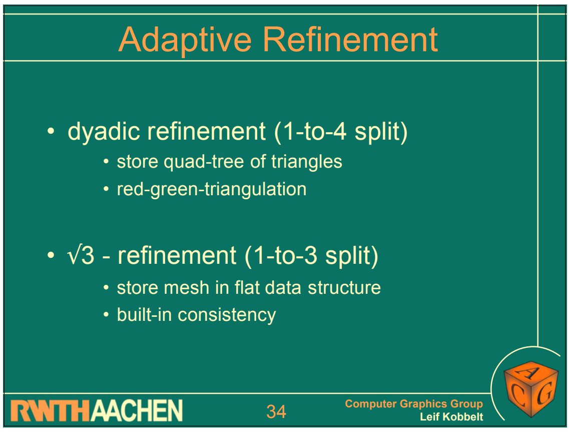


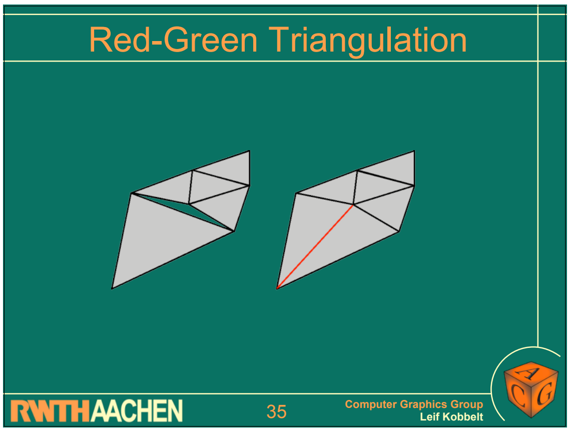
3x 9x 27x

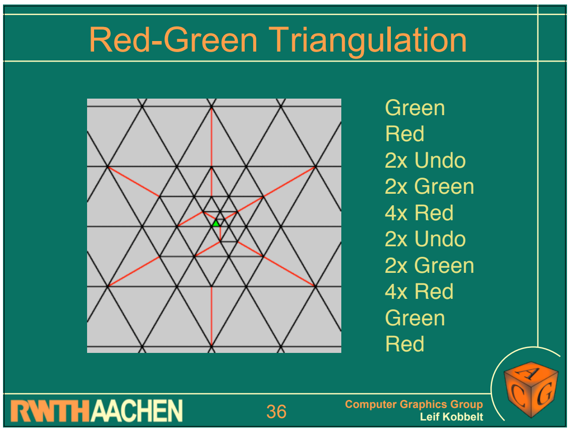

Adaptive Refinement
RWTHAACHEN
31
Computer Graphics Group
Leif Kobbelt


Adaptive Refinement
RWTHAACHEN
32
Computer Graphics Group
Leif Kobbelt

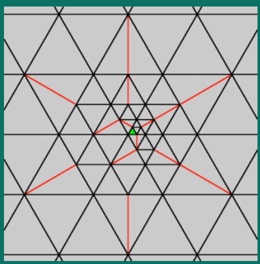

Adaptive Refinement
RWTHAACHEN
33
Computer Graphics Group
Leif Kobbelt


Adaptive Refinement
• dyadic refinement (1-to-4 split)
• store quad-tree of triangles
• red-green-triangulation
• √3 - refinement (1-to-3 split)
• store mesh in flat data structure
• built-in consistency
RWTHAACHEN
34
Computer Graphics Group
Leif Kobbelt


Red-Green Triangulation
RWTHAACHEN
35
Computer Graphics Group
Leif Kobbelt


Red-Green Triangulation
Green
Red
2x Undo
2x Green
4x Red
2x Undo
2x Green
4x Red
Green
Red
RWTHAACHEN
36
Computer Graphics Group
Leif Kobbelt

Red-Green Triangulation



Green

2x Green

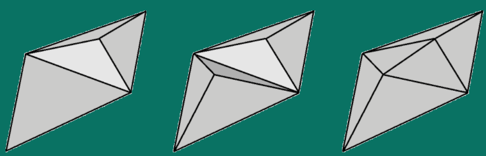
2x Green

Green
14x Red

Red-Green Triangulation

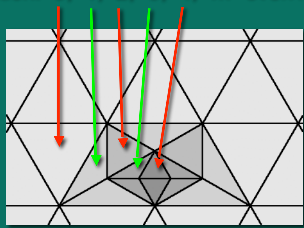
```
Split(T)
    for i = 1,2,3
        if level(T->neighbor[i]) < level(T)
            Split(T->neighbor[i])
    Quadrisect(T)
    Crack fixing …
```


√3 - Refinement

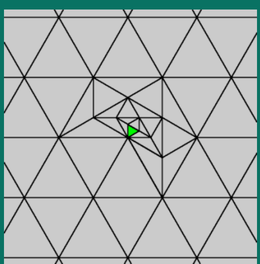


√3 - Refinement

Face generation: 0, 1, 2, 3, 4, … even / odd



√3 - Refinement



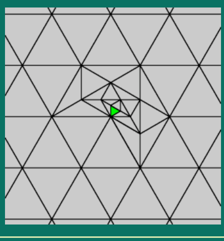
Center
Flip
Center
Flip
Center
Flip

√3 - Refinement

```
Split(T)
    if generation(T) = even
        center-split(T)
        flip edges (if possible)
    else
        if generation(T->opposite) = odd
            Split(T->opposite)
        Split(T->opposite)
```

Adaptive Subdivision

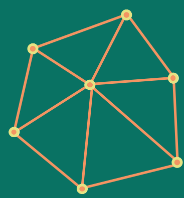
- vertex positions change from level to level
- direct access to $P_{i,l}$



$P_{i,\infty}$

Multi-Step Rules

$$\begin{pmatrix} p_0^{n+1} \\ p_1^{n+1} \\ \vdots \\ p_k^{n+1} \end{pmatrix} = \begin{pmatrix} a & b & \cdots & \cdots & b \\ c & d_1 & d_2 & \cdots & d_k \\ c & d_k & d_1 & d_2 & \cdots \\ \vdots & & & \ddots & \\ c & d_2 & \cdots & d_k & d_1 \end{pmatrix} \begin{pmatrix} p_0^{n} \\ p_1^{n} \\ \vdots \\ p_k^{n} \end{pmatrix}$$



Multi-Step Rules

$$\begin{pmatrix} p_0^{n+1} \\ \sum p_i^{n+1} \end{pmatrix} = \begin{pmatrix} a & b \\ k\,c & \sum d_i \end{pmatrix} \begin{pmatrix} p_0^{n} \\ \sum p_i^{n} \end{pmatrix}$$

Multi-Step Rules

$$\begin{pmatrix} p_0^{n+1} \\ \frac{1}{k}\sum p_i^{n+1} \end{pmatrix} = \begin{pmatrix} a & 1-a \\ c & 1-c \end{pmatrix} \begin{pmatrix} p_0^{n} \\ \frac{1}{k}\sum p_i^{n} \end{pmatrix}$$

Multi-Step Rules

$$\begin{pmatrix} p_0^{n+1} \\ \frac{1}{k}\sum p_i^{n+1} \end{pmatrix} = \begin{pmatrix} a & 1-a \\ c & 1-c \end{pmatrix} \begin{pmatrix} p_0^{n} \\ \frac{1}{k}\sum p_i^{n} \end{pmatrix}$$

$$\begin{pmatrix} p_0^{n+1} \\ \frac{1}{k}\sum p_i^{n+1} \end{pmatrix} = \frac{c}{1-a+c} \begin{pmatrix} 1 & \frac{1-a}{c} \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & a-c \end{pmatrix} \begin{pmatrix} 1 & \frac{1-a}{c} \\ 1 & -1 \end{pmatrix} \begin{pmatrix} p_0^{n} \\ \frac{1}{k}\sum p_i^{n} \end{pmatrix}$$

$$\begin{pmatrix} p_0^{n+m} \\ \frac{1}{k}\sum p_i^{n+m} \end{pmatrix} = \frac{c}{1-a+c} \begin{pmatrix} 1 & \frac{1-a}{c} \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & (a-c)^m \end{pmatrix} \begin{pmatrix} 1 & \frac{1-a}{c} \\ 1 & -1 \end{pmatrix} \begin{pmatrix} p_0^{n} \\ \frac{1}{k}\sum p_i^{n} \end{pmatrix}$$

Limit Rules

$$\begin{pmatrix} p_0^{\infty} \\ \frac{1}{k}\sum p_i^{\infty} \end{pmatrix} = \frac{c}{1-a+c} \begin{pmatrix} 1 & \frac{1-a}{c} \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & \frac{1-a}{c} \\ 1 & -1 \end{pmatrix} \begin{pmatrix} p_0^{n} \\ \frac{1}{k}\sum p_i^{n} \end{pmatrix}$$

$$p_0^{\infty} = \frac{c}{1-a+c}\, p_0^{n} + \frac{1-a}{1-a+c}\, \frac{1}{k}\sum p_i^{n}$$

Limit Rules

$$\begin{pmatrix} p_0^\infty \\ \frac{1}{k}\sum p_i^\infty \end{pmatrix} = \frac{c}{1-a+c} \begin{pmatrix} 1 & \frac{1-a}{c} \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & \frac{1-a}{c} \\ 1 & -1 \end{pmatrix} \begin{pmatrix} p_0^n \\ \frac{1}{k}\sum p_i^n \end{pmatrix}$$

$$p_0^\infty = \frac{c}{1-a+c}\, p_0^n + \frac{1-a}{1-a+c}\, \frac{1}{k}\sum p_i^n$$

$$\frac{1}{k}\sum p_i^n = \frac{c}{a-1}\, p_0^n + \frac{a-1-c}{a-1}\, p_0^\infty$$

Multi-Step Rules

$$\begin{pmatrix} p_0^{n+m} \\ \frac{1}{k}\sum p_i^{n+m} \end{pmatrix} = \frac{c}{1-a+c} \begin{pmatrix} 1 & \frac{1-a}{c} \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & (a-c)^m \end{pmatrix} \begin{pmatrix} 1 & \frac{1-a}{c} \\ 1 & -1 \end{pmatrix} \begin{pmatrix} p_0^n \\ \frac{1}{k}\sum p_i^n \end{pmatrix}$$

$$\begin{pmatrix} p_0^n \\ \frac{1}{k}\sum p_i^n \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ \frac{c}{a-1} & \frac{a-1-c}{a-1} \end{pmatrix} \begin{pmatrix} p_0^n \\ p_0^\infty \end{pmatrix}$$

$$p_0^{n+m} = (a-c)^m\, p_0^n + \left[1-(a-c)^m\right] p_0^\infty$$

Implementation

- each vertex stores P_i
  - its „birth“ position $P_{i,n}$
  - its limit position $P_{i,\infty}$
- access by `[x,y,z] = pos(i,m)`

$$p_0^m = (a-c)^{m-n}\, p_0^n + \left[1-(a-c)^{m-n}\right] p_0^\infty$$

Implementing Subdivision Schemes

- data structures for meshes
  - face-, edge, half-edge-based
  - hierarchical (quad-trees)
- elementary operations
  - 1-3, 2-4, edge-flip
  - uniform refinement
- adaptive subdivision
  - red-green
  - √3
- multi-step rules