# Computer Graphics

Sipos Ágoston
siposagoston@inf.elte.hu

Eötvös Loránd University
Faculty of Informatics

2025-2026. Fall semester

# Table of contents

# Representing surfaces

- Explicit: $z = f(x, y)$
- Implicit: $f(x, y, z) = 0$
- Parametric: $\mathbf{p}(u, v) = \begin{bmatrix} x(u, v) \\ y(u, v) \\ z(u, v) \end{bmatrix}$, where usually $(u, v) \in [a, b] \times [c, d]$

# Surface normal

- The normal vector of the tangent plane at the surface point (a vector perpendicular to the plane)
- Usually a unit vector
- In different forms, the (non-unit length) surface normal is:
  - implicit: $\nabla f(x, y, z) = \begin{bmatrix} f_x(x, y, z) \\ f_y(x, y, z) \\ f_z(x, y, z) \end{bmatrix}$
  - parametric: $\mathbf{n}(u, v) = \mathbf{p}_u(u, v) \times \mathbf{p}_v(u, v)$

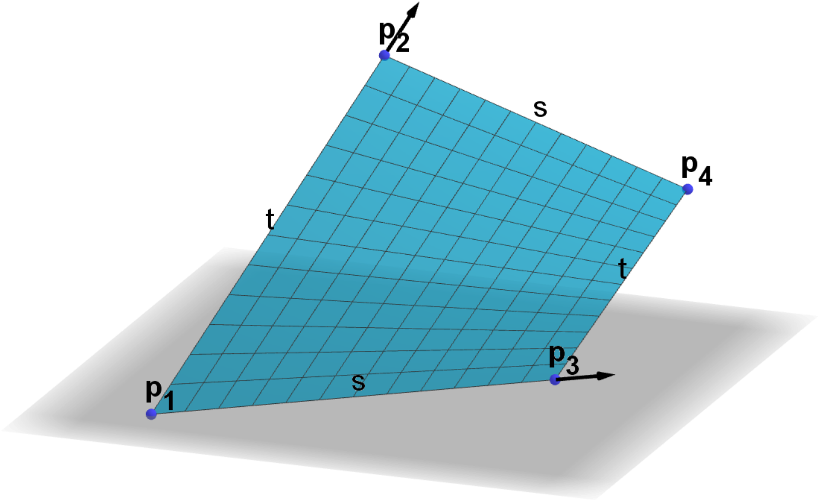
# Bilinear surface

- Let four control points be, $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4 \in \mathbb{E}^3$
- Find a simple parametric surface over $[0, 1] \times [0, 1]$ that interpolates the above four points at the corners
- With three linear interpolations we get a simple surface:

$$\begin{aligned}\mathbf{b}(s, t) = (1 - t)((1 - s)\mathbf{p}_1 + s\mathbf{p}_3) \\ +t((1 - s)\mathbf{p}_2 + s\mathbf{p}_4)\end{aligned}$$

  where $s, t \in [0, 1]$.
- Essentially: we "wrote" two segments into the formula of linear interpolation according to $t$

# Bilinear surface




In general this is not planar.

# Matrix form of curves

- The parametric representation of a degree $n$ polynomial curve in power basis is $\mathbf{r}(u) = \sum_{i=0}^{n} \mathbf{a}_i \cdot u^i,\ \ u \in \mathbb{R}$
- Let's pay attention to that $\mathbf{a}_0 \in \mathbb{E}^3$ and $\mathbf{a}_i \in \mathbb{R}^3, i = 1, 2, .., n$
- The above can also be easily written in matrix form, for example in the case of a cubic integer polynomial

$$\mathbf{r}(u) = \underbrace{[u^3, u^2, u, 1]}_{\mathbf{u}^T} \underbrace{\begin{bmatrix} \mathbf{a}_3 \\ \mathbf{a}_2 \\ \mathbf{a}_1 \\ \mathbf{a}_0 \end{bmatrix}}_{\mathbf{A}}$$

# Matrix form of curves – example

- The parametric form of $y = x^2$ parabola in power function base

$$\mathbf{p}(u) = \begin{bmatrix} 0 \\ 1 \end{bmatrix} u^2 + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u$$

- And the matrix form

$$\mathbf{p}(u) = [u^2, u, 1] \begin{bmatrix} \mathbf{a}_2 \\ \mathbf{a}_1 \\ \mathbf{a}_0 \end{bmatrix}$$

where

$$\mathbf{a}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \ , \ \mathbf{a}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \ , \ \mathbf{a}_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

# Matrix form of curves in different basis

- If we have a matrix $\mathbf{M}$, which transforms from the power base to another base, then with the curve coordinates $\mathbf{G}$ in the new base, the curve has the following shape:

$$\mathbf{p}(u) = \mathbf{u}^T \underbrace{\mathbf{M} \cdot \mathbf{G}}_{\mathbf{A}}$$

- Here, the result of $\mathbf{u}^T\mathbf{M}$ is the other basis, and $\mathbf{G}$ contains the corresponding data about the curve (e.g. in the case of a Bernstein basis, $\mathbf{G}$ consists of Bézier control points).

# Matrix form of curves in different basis – example

- For example, quadratic Bernstein basis polynomials

$$B_0^2(u) = (1-u)^2 = u^2 - 2u + 1$$
$$B_1^2(u) = 2(1-u)u = -2u^2 + 2u$$
$$B_2^2(u) = u^2$$

- Thus, the matrix form of the transformation from the power base to Bernstein base

$$M = \begin{bmatrix} 1 & -2 & 1 \\ 0 & 2 & -2 \\ 0 & 0 & 1 \end{bmatrix}$$

# Matrix form of curves in different basis – example

- Then the other base from our formula

$$[u^2, u, 1] \cdot \begin{bmatrix} 1 & -2 & 1 \\ 0 & 2 & -2 \\ 0 & 0 & 1 \end{bmatrix} = [B_2^2(u), B_1^2(u), B_0^2(u)]$$

  therefore, the Bézier control points in **G** must be stored in the order $\mathbf{b}_2, \mathbf{b}_1, \mathbf{b}_0$

- Then the form of the curve

$$\mathbf{r}(u) = [u^2, u, 1] \cdot \begin{bmatrix} 1 & -2 & 1 \\ 0 & 2 & -2 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{b}_2 \\ \mathbf{b}_1 \\ \mathbf{b}_0 \end{bmatrix}$$

# *Matrix form of curves in different basis – example

- Cubic Hermite curve's base
  $\mathbf{r}(u) = H_0^3(u)\mathbf{r}_0 + H_3^3(u)\mathbf{r}_1 + H_1^3(u)\mathbf{t}_0 + H_2^3(u)\mathbf{t}_1$

$$H_0^3(u) = 2u^3 - 3u^2 + 1 \ , \ H_1^3(u) = u^3 - 2u^2 + u$$
$$H_2^3(u) = u^3 - u^2 \ , \ H_3^3(u) = -2u^3 + 3u^2$$

- Therefore, the cubic Hermite curve in matrix form:

$$\mathbf{r}(u) = [u^3, u^2, u, 1] \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{r}_0 \\ \mathbf{r}_1 \\ \mathbf{t}_0 \\ \mathbf{t}_1 \end{bmatrix}$$
$$= [H_0^3(u), H_3^3(u), H_1^3(u), H_2^3(u)] \cdot G = \mathbf{h}^T(u) \cdot G$$

# Change of basis

- If we know an array containing the geometric data $\mathbf{G}_1$ of our curve in a base $\mathbf{u}^T\mathbf{M}_1$ and we want to calculate what our curve's coordinates $\mathbf{G}_2$ will be in the $\mathbf{u}^T\mathbf{M}_2$ base, then we need to solve
$$\mathbf{u}^T\mathbf{M}_1\mathbf{G}_1 = \mathbf{u}^T\mathbf{M}_2\mathbf{G}_2$$
system of equations for the unknown variables from $\mathbf{G}_2$
- From this the solution:
$$\mathbf{G}_2 = \mathbf{M}_2^{-1}\mathbf{M}_1\mathbf{G}_1$$

# Change of basis – example

- Let the previous parabola in $\mathbf{p}(u) = [u^2, u, 1] \cdot \begin{bmatrix} \mathbf{a}_2 \\ \mathbf{a}_1 \\ \mathbf{a}_0 \end{bmatrix}$ form, where $\mathbf{a}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \mathbf{a}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \mathbf{a}_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, let's say we want to draw it with Bézier control points in $[0, 1]$ range
- Then we need $\mathbf{b}_0, \mathbf{b}_1, \mathbf{b}_2 \in \mathbb{E}^3$ Bézier control points
- We need to solve

$$[u^2, u, 1] \begin{bmatrix} \mathbf{a}_2 \\ \mathbf{a}_1 \\ \mathbf{a}_0 \end{bmatrix} = [u^2, u, 1] \begin{bmatrix} 1 & -2 & 1 \\ 0 & 2 & -2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{b}_2 \\ \mathbf{b}_1 \\ \mathbf{b}_0 \end{bmatrix}$$

# Change of basis – example

- That is, the Bezier control points are

$$\begin{bmatrix} \mathbf{b}_2 \\ \mathbf{b}_1 \\ \mathbf{b}_0 \end{bmatrix} = \begin{bmatrix} 1 & -2 & 1 \\ 0 & 2 & -2 \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{a}_2 \\ \mathbf{a}_1 \\ \mathbf{a}_0 \end{bmatrix}$$

  after the inverse

$$\begin{bmatrix} \mathbf{b}_2 \\ \mathbf{b}_1 \\ \mathbf{b}_0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & \frac{1}{2} & 1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{a}_2 \\ \mathbf{a}_1 \\ \mathbf{a}_0 \end{bmatrix}$$

# Change of basis – example

- Thus, the required Bézier control points are:

$$\mathbf{b}_0 = \mathbf{a}_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\mathbf{b}_1 = \mathbf{a}_0 + \frac{1}{2}\mathbf{a}_1 = \begin{bmatrix} \frac{1}{2} \\ 0 \end{bmatrix}$$

$$\mathbf{b}_2 = \mathbf{a}_0 + \mathbf{a}_1 + \mathbf{a}_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

- Check: the $\mathbf{b}_0, \mathbf{b}_1, \mathbf{b}_2$ are in fact points (point + vector)
- HW: calculate the Bézier control point of the parabola in $[-2, 2]$!

# Matrix form of the surface

- The parametric representation of a polynomial surface in power basis is $\mathbf{r}(u,v) = \sum_{i=0}^{m} \sum_{j=0}^{n} \mathbf{a}_{ij} u^i v^j,\ \ u, v \in \mathbb{R}$
- Let's pay attention to that, $\mathbf{a}_{00} \in \mathbb{E}^3$ and $\mathbf{a}_{ij} \in \mathbb{R}^3,\ (ij) \neq (00)$
- In case of a surface which is cubic in both parameter directions

$$\mathbf{r}(u,v) = \underbrace{[u^3, u^2, u, 1]}_{\mathbf{u}^T} \underbrace{\begin{bmatrix} \mathbf{a}_{33} & \mathbf{a}_{32} & \mathbf{a}_{31} & \mathbf{a}_{30} \\ \mathbf{a}_{23} & \mathbf{a}_{22} & \mathbf{a}_{21} & \mathbf{a}_{20} \\ \mathbf{a}_{13} & \mathbf{a}_{12} & \mathbf{a}_{11} & \mathbf{a}_{10} \\ \mathbf{a}_{03} & \mathbf{a}_{02} & \mathbf{a}_{01} & \mathbf{a}_{00} \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} v^3 \\ v^2 \\ v \\ 1 \end{bmatrix}}_{\mathbf{v}} = \mathbf{u}^T \mathbf{A} \mathbf{v}$$

# Surfaces in different basis

- Similar to what we saw with the curves, writing down the matrix form of a surface in a general basis gives the following:

  $$\mathbf{r}(u,v) = \mathbf{u}^T \cdot \mathbf{M} \cdot \mathbf{G} \cdot \mathbf{N}^T \cdot \mathbf{v} \ ,$$

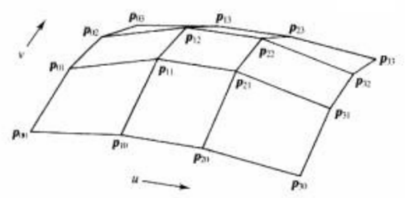
  where $\mathbf{M}, \mathbf{N}$ are the $m$ and $n$-th power $\rightarrow$ general basis conversion matrices

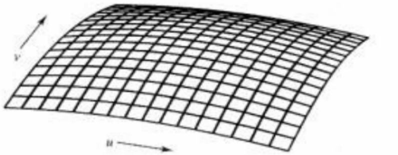
# Bézier surface

- The form of an $n \times m$-th degree Bézier surface defined by $\mathbf{b}_{ij} \in \mathbb{E}^3,\ i = 0, .., n, j = 0, .., m$ control polygon

$$\mathbf{b}(u, v) = \sum_{j=0}^{m} \sum_{i=0}^{n} B_i^n(u) B_j^m(v) \mathbf{b}_{ij}$$

where $u, v \in [0, 1]$.





# *Bézier surface – derivatives

- Using what we learned with curves, at $u$ parameter direction:

$$\begin{aligned}\partial_u \mathbf{b}(u,v) &= \sum_{j=0}^{m} \partial_u \Big(\sum_{i=0}^{n} B_i^n(u)\mathbf{b}_{ij}\Big) B_j^m(v) \\ &= n \sum_{j=0}^{m} \sum_{i=0}^{n-1} \Delta^{1,0}\mathbf{b}_{ij} B_i^{n-1}(u) B_j^m(v)\end{aligned}$$

where $\Delta^{1,0}\mathbf{b}_{i,j} = \mathbf{b}_{i+1,j} - \mathbf{b}_{i,j}$

# *Bézier surface – derivatives

- Same with $v$

$$\partial_v \mathbf{b}(u,v) = m \sum_{j=0}^{m-1} \sum_{i=0}^{n} \Delta^{0,1} \mathbf{b}_{ij} B_i^n(u) B_j^{m-1}(v)$$

where $\Delta^{0,1}\mathbf{b}_{i,j} = \mathbf{b}_{i,j+1} - \mathbf{b}_{i,j}$

# *Bézier surface – derivatives

- We call the plane spanned by the partial derivatives at the given surface point, *tangent plane*
- Its normal – which is also the normal of our surface at the point corresponding to the given parameter values – is $[\partial_u \mathbf{p} \times \partial_v \mathbf{p}]_0$, which can be nicely described in the vertices with the control points

# *Bézier surface – derivatives

Using what you learned with the curves, the derivatives generally look like this:

$$\partial_{u^r,v^s}\mathbf{b}(u,v) = \frac{m!n!}{(n-r)!(m-s)!}\sum_{j=0}^{m-s}\sum_{i=0}^{n-r}\Delta^{r,s}\mathbf{b}_{ij}B_i^{n-r}(u)B_j^{m-s}(v)$$

where

$$\begin{aligned}\Delta^{i,j}\mathbf{b}_{i,j} &= \Delta^{i-1,j}\mathbf{b}_{i+1,j} - \Delta^{i-1,j}\mathbf{b}_{i,j} \\ &= \Delta^{i,j-1}\mathbf{b}_{i,j+1} - \Delta^{i,j-1}\mathbf{b}_{i,j}\end{aligned}$$

# Spline surface

- We join together lower degree surface pieces (patches)
- Pay attention to the continuity of the connections
- More on this: Geometric Modelling, Surface and Body modeling - MSc.

# Intersection with ray

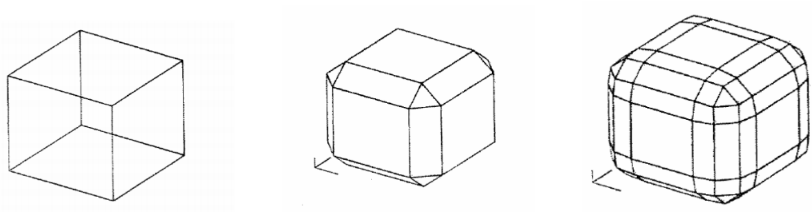
- For example, when we select with the mouse, it may be necessary to find the intersection of our ray $\mathbf{p}(t) = \mathbf{p}_0 + t\mathbf{v}$ and the surface $\mathbf{b}(u, v)$
- So we need to solve the following system of equations: $\mathbf{p}(t) = \mathbf{b}(u, v)$, where the unknowns are $t, u, v$ (let's pay attention to their restrictions!)
- multi variable Newton, etc. ...

# Subdivision

SIGGRAPH Subdivision tutorial for those interested in the topic:
`http://www.mrl.nyu.edu/publications/subdiv-course2000/`

# Subdivision surfaces – Doo-Sabin

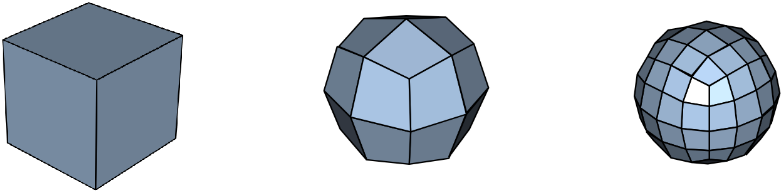
Vertex split algorithm



# Subdivision surfaces – Catmull-Clark

Face split algorithm

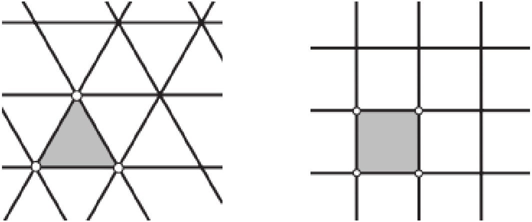
2001: Catmull received an Oscar ”for significant advancements to the field of motion picture rendering as exemplified in Pixar’s RenderMan”



# Concepts – schema mesh-type

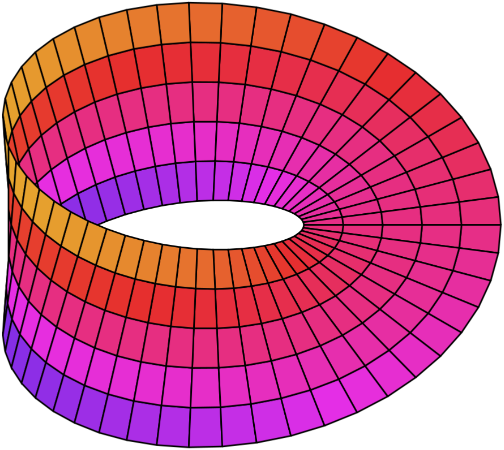
- Most subdivision schemes are based on some regular subdivision/refinement scheme
- When we talk about the mesh type of a scheme, we mean this parent scheme
- In a plane we can cover points in a regular grid with regular triangles, squares, or regular hexagons.
- Accordingly, we call a scheme *triangle-*, *quadrilateral-* or *hexagon-based* (in practice, the latter is rare)

# Mesh-type

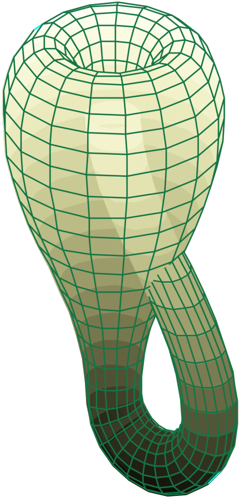


- Be careful: you cannot cover everything "aesthetically" (joining along whole edges) with 6-regular triangle or 4-regular quadrilateral mesh without degenerate cases!
- The above regular topologies can be used to describe the infinite plane, the infinite cylindrical surface, or surfaces with topology like the torus's
- For example, surfaces with topology like the sphere's cannot be covered

# Mesh-type – Möbius



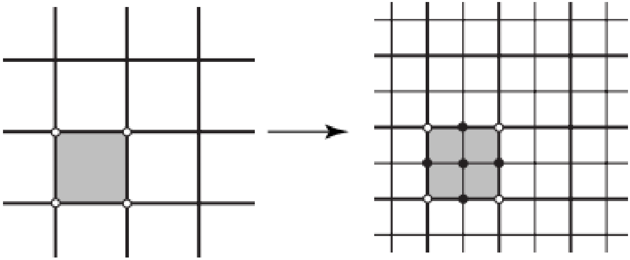
# Mesh-type – Klein bottle



# Concepts – face-split (primal)

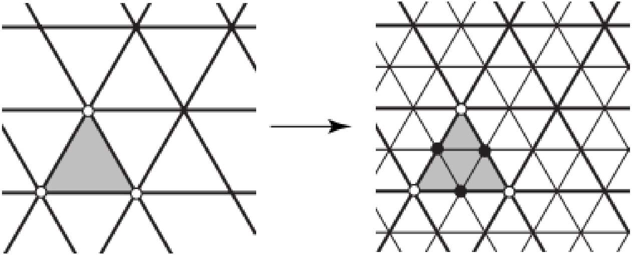
- Each face corresponding to its mesh type is divided into four
- We keep the vertices of the mesh from the previous step (but we can change their position – if we don't change them, we are talking about an interpolation scheme)
- We insert new vertices on each edge (thus splitting them in two)
- In the case of quadrilateral-based schemes, we also derive a new vertex from the face

# Face-split on 4-regular mesh



*Face split for quads*

# Face-split on 6-regular mesh



*Face split for triangles*

# Concepts – face-split

Even vertices:

- In face-split schemes, the vertices of the coarser resolution mesh that correspond to the vertices of the finer mesh
- White on the previous figure

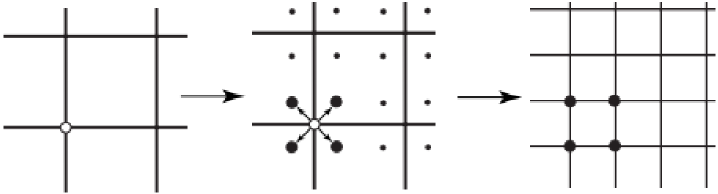
Odd vertices:

- Newly created vertices that do not correspond to any vertices from the previous refinement level
- Black on the previous figure

# Concepts – vertex-split (dual)

- In this case, a new vertex is created from each vertex for each of the faces neighboring the original vertex
- A new face is directly derived from the old face
- Along the edges we get new faces (connecting new vertices, which are created from the endpoints of an edge, across the two faces that are divided by that edge)
- Instead of the old vertices, we get a new face with new vertices.

# Vertex-split on 4-regular mesh



*Vertex split for quads*

# Concepts – face- and vertex-split

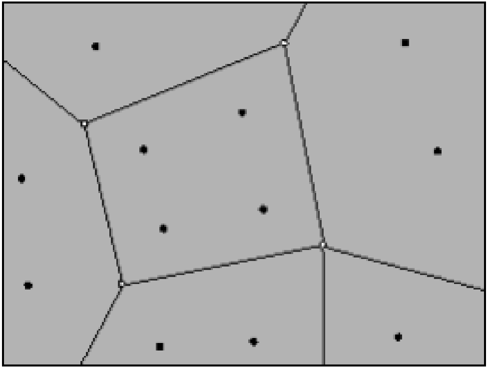
- On a regular quadrilateral mesh, in both cases the new mesh will be 4-regular $\rightarrow$ maintains the topology!
- Pay attention: with regular triangle meshes, after vertex-split, we also get triangles, quadrilaterals and hexagons!

# Doo-Sabin

Original article:
`https://web.archive.org/web/20110707175713/http://trac2.assembla.com/DooSabinSurfaces/export/12/trunk/docs/Doo%201978%20Subdivision%20algorithm.pdf`

Short description: `http://www.cs.unc.edu/~dm/UNC/COMP258/LECTURES/Doo-Sabin.pdf`

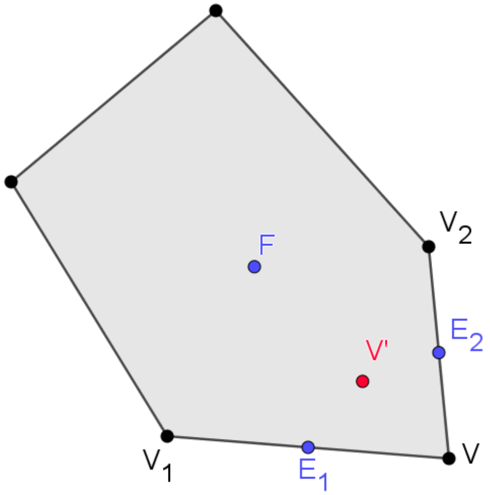
# Doo-Sabin – calculating new points



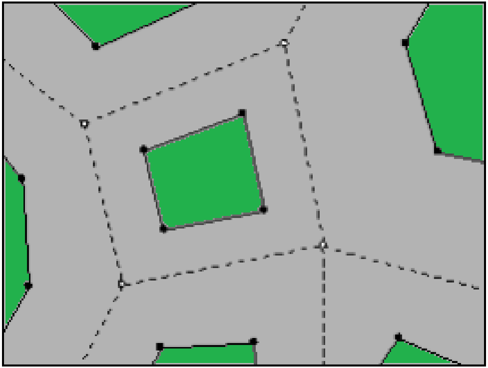
# Doo-Sabin – calculating new points

- Vertex split algorithm: for every face and for each of its vertices we calculate a new vertex
- Let the given vertex be $V \in \mathbb{E}^3$, its two neighbours on the face $V_1$ and $V_2 \in \mathbb{E}^3$, and the centroid of the face $F \in \mathbb{E}^3$
- Then the two edge vertices are $E_1 = \frac{1}{2}V + \frac{1}{2}V_1$ and $E_2 = \frac{1}{2}V + \frac{1}{2}V_2$
- The new vertex is $V' = \frac{1}{4}V + \frac{1}{4}E_1 + \frac{1}{4}E_2 + \frac{1}{4}F$

# Doo-Sabin

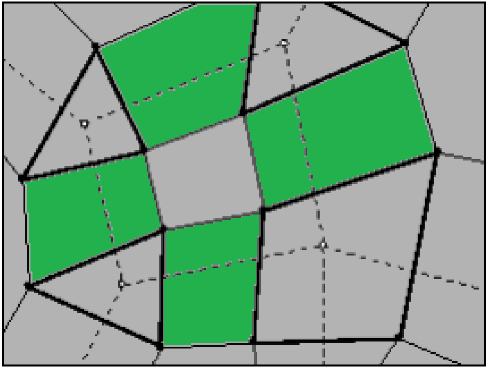


# Doo-Sabin – faces from faces



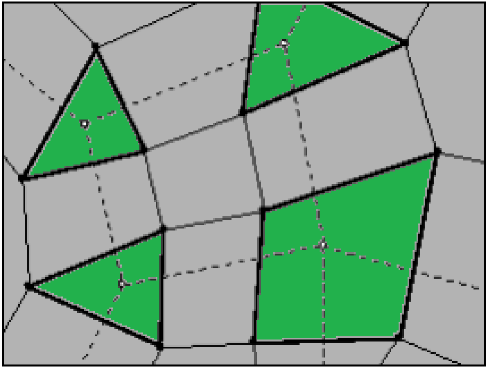
Number of sides equals the original

# Doo-Sabin – faces from edges



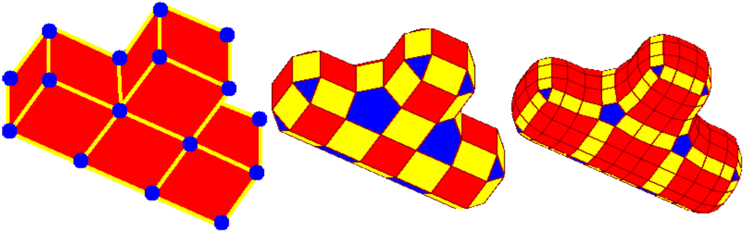
Always a quadrilateral

# Doo-Sabin – faces from vertices



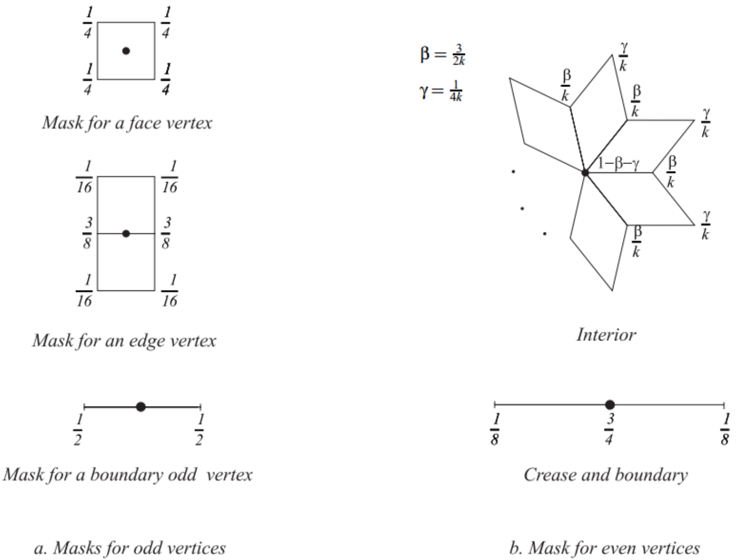
Number of sides equals the valency of the vertex

# Doo-Sabin



Be careful, the resulting polygons may not be planar!

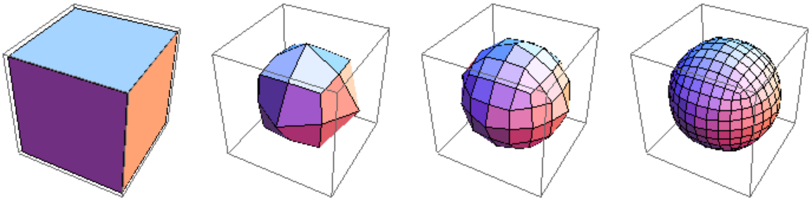
# Catmull-Clark



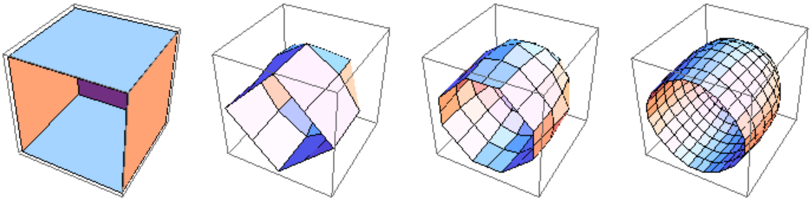
# Catmull-Clark

- Face split algorithm, defined for quadrilateral meshes
- The new vertex positions are the result of the weighted average of the vertices of the neighboring faces. The weights are shown in the previous figure.
- A new face vertex is the centroid of the face
- A new edge vertex takes into account the two endpoints of the original edge with a weight of $\frac{3}{8}$, and the other vertices on the same face with a weight of $\frac{1}{16}$
- For even vertices the neighboring vertices have a larger weight than the further ones. The value $k$ indicated in the figure is the number of faces adjacent to the vertex.
- We can also handle the edge of the mesh (*boundary*), and in the same way, if you want to leave a sharp edge in the model (*crease*)

# Catmull-Clark



# Catmull-Clark – boundary



# Catmull-Clark – crease