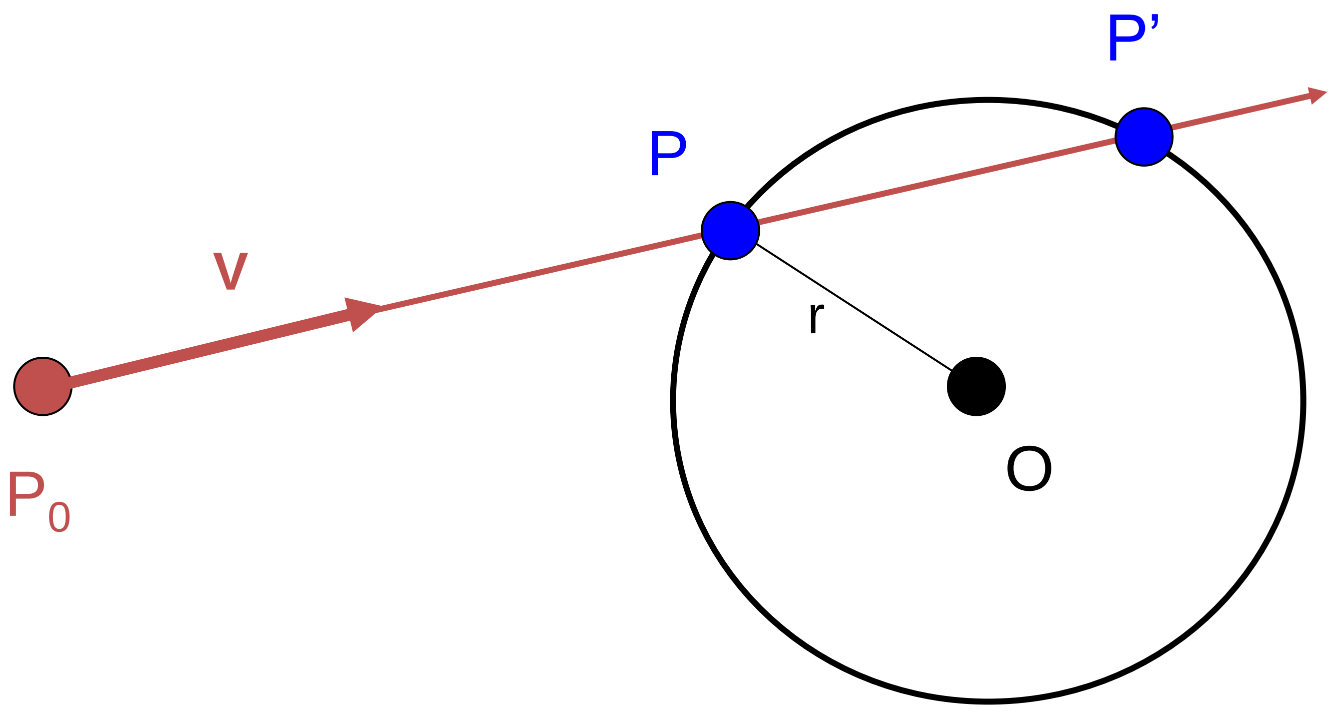
Physically Based Rendering (600.657)

Shapes

Ray-Sphere Intersection

Ray: $P = P_0 + tV$

Sphere: $|P - O|^2 - r^2 = 0$



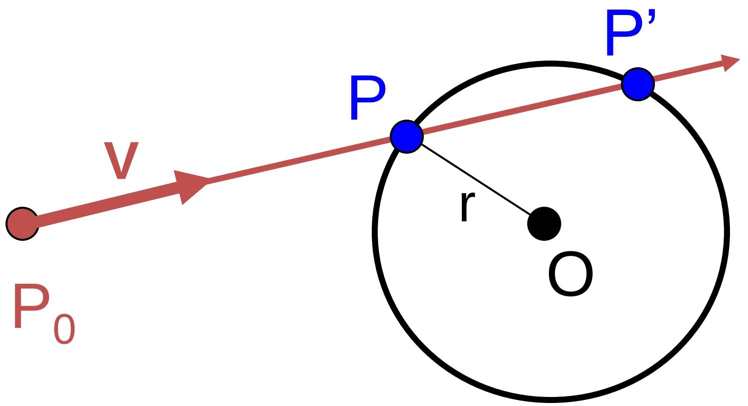
Ray-Sphere Intersection

Ray: $P = P_0 + tV$

Sphere: $|P - O|^2 - r^2 = 0$

Substituting for P, we get:

$$|\mathbf{P_0 + tV} - O|^2 - r^2 = 0$$



Ray-Sphere Intersection

Ray: $P = P_0 + tV$

Sphere: $|P - O|^2 - r^2 = 0$

Substituting for P, we get:

$$|\mathbf{P_0 + tV} - O|^2 - r^2 = 0$$

Solve quadratic equation:

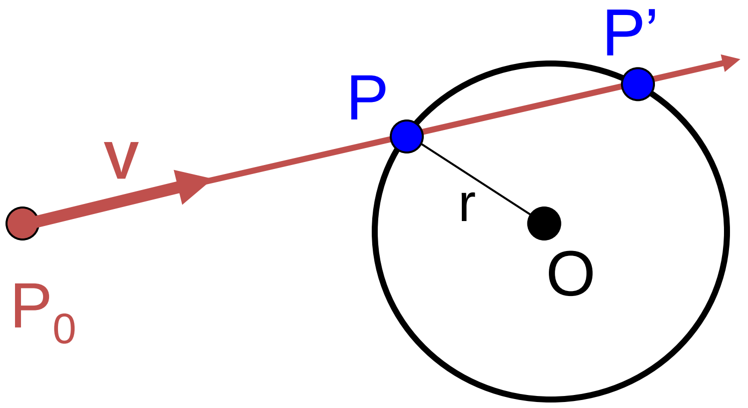
$$at^2 + bt + c = 0$$

where:

$$a = 1$$

$$b = 2\, V \bullet (P_0 - O)$$

$$c = |P_0 - O|^2 - r^2$$



Ray-Sphere Intersection

Ray: $P = P_0 + tV$

Sphere: $|P - O|^2 - r^2 = 0$

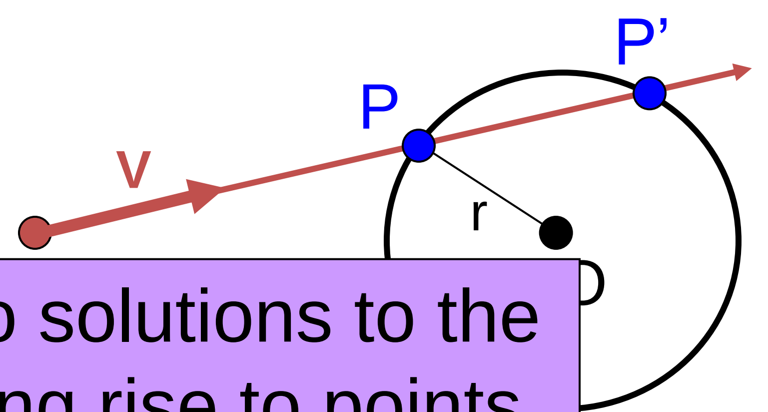
Substituting for P, we get:

$$|\mathbf{P_0 + tV} - O|^2 - r^2 = 0$$

Solve quadratic equation:

$$at^2 + bt + c = 0$$

where:



Generally, there are two solutions to the quadratic equation, giving rise to points P and P'.
You want to return the first (positive) hit.

Ray-Sphere Intersection

Ray: $P = P_0 + tV$
Sphere: $|P - O|^2 - r^2 = 0$

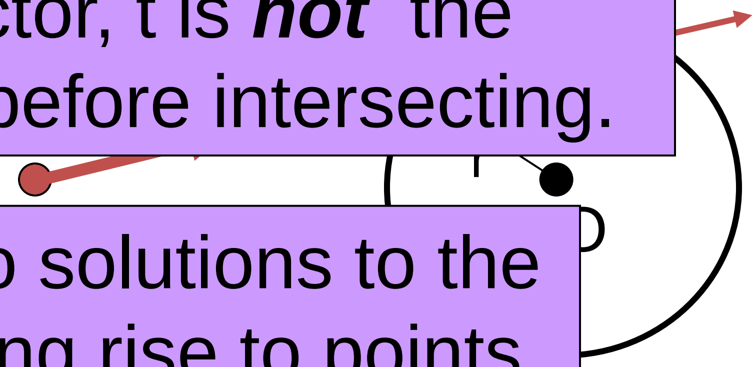
Substituting for P, we get:

$$|\mathbf{P_0 + tV} - O|^2 - r^2 = 0$$

Se

where:

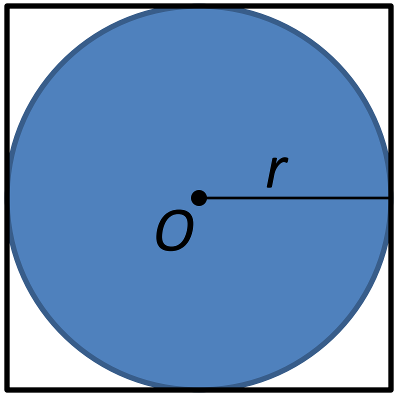
Unless V is a unit-vector, t is ***not*** the distance the ray travels before intersecting.



Generally, there are two solutions to the quadratic equation, giving rise to points P and P'.
You want to return the first (positive) hit.

Bounding the Sphere

- Bounding Box:
  - $[O_x\text{-}r, O_y\text{-}r, O_z\text{-}r]$, $[O_x+r, O_y+r, O_z+r]$

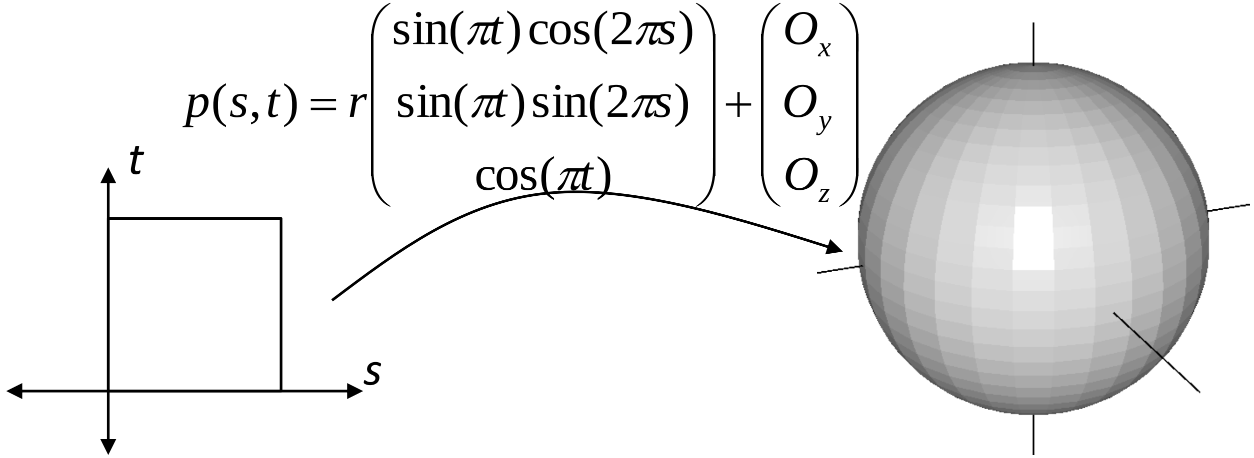


Parameterizing the Sphere

- Parametric Equation:

$$p(s,t) = r\left(\sin(\pi t)\cos(2\pi s), \sin(\pi t)\sin(2\pi s), \cos(\pi t)\right) + \left(O_x, O_y, O_z\right)$$

$$p(s,t) = r\begin{pmatrix} \sin(\pi t)\cos(2\pi s) \\ \sin(\pi t)\sin(2\pi s) \\ \cos(\pi t) \end{pmatrix} + \begin{pmatrix} O_x \\ O_y \\ O_z \end{pmatrix}$$



Parameterizing the Sphere

- Parametric Equation:

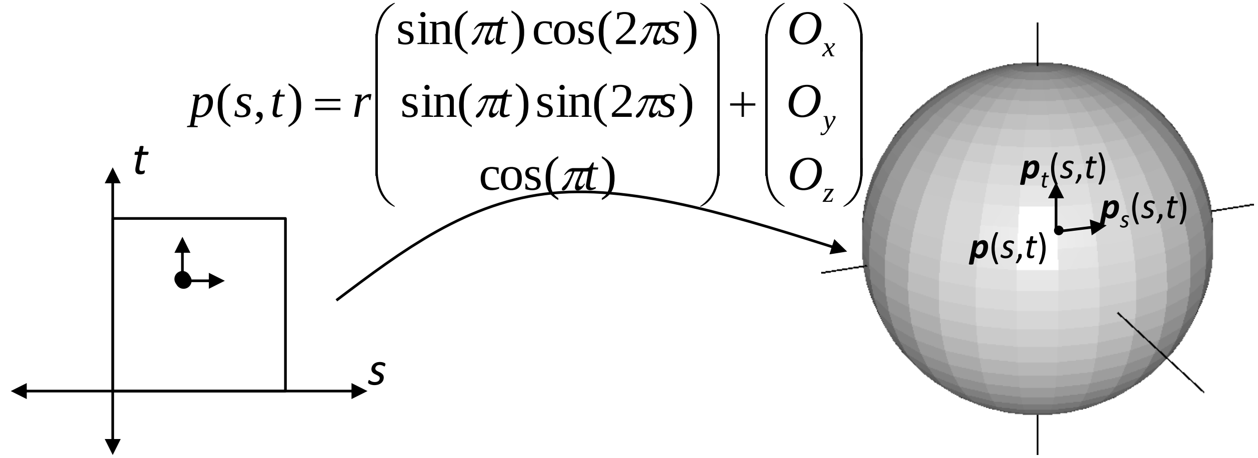
$$p(s,t) = r\left(\sin(\pi t)\cos(2\pi s), \sin(\pi t)\sin(2\pi s), \cos(\pi t)\right) + \left(O_x, O_y, O_z\right)$$

- Partials:

$$\frac{\partial p}{\partial s}(s,t) = 2\pi r\left(-\sin(\pi t)\sin(2\pi s), \sin(\pi t)\cos(2\pi s), 0\right)$$

$$\frac{\partial p}{\partial t}(s,t) = \pi r\left(\cos(\pi t)\cos(2\pi s), \cos(\pi t)\sin(2\pi s), -\sin(\pi t)\right)$$

$$p(s,t) = r\begin{pmatrix} \sin(\pi t)\cos(2\pi s) \\ \sin(\pi t)\sin(2\pi s) \\ \cos(\pi t) \end{pmatrix} + \begin{pmatrix} O_x \\ O_y \\ O_z \end{pmatrix}$$



Parameterizing the Sphere

- Parametric Equation:

$$p(s,t) = r\left(\sin(\pi t)\cos(2\pi s), \sin(\pi t)\sin(2\pi s), \cos(\pi t)\right) + \left(O_x, O_y, O_z\right)$$

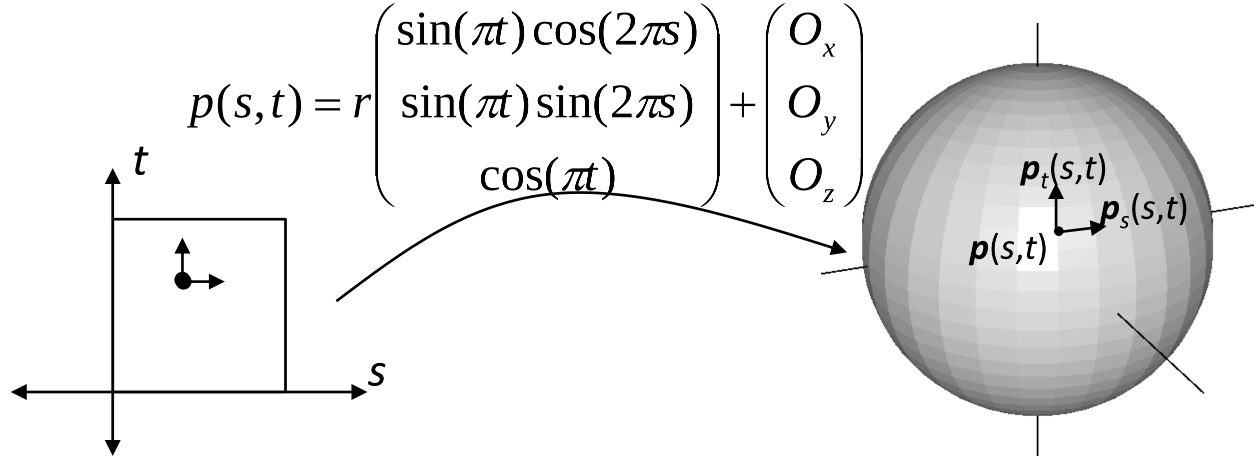
- Partials:

$$\frac{\partial p}{\partial s}(s,t) = 2\pi r\left(-\sin(\pi t)\sin(2\pi s), \sin(\pi t)\cos(2\pi s), 0\right)$$

$$\frac{\partial p}{\partial t}(s,t) = \pi r\left(\cos(\pi t)\cos(2\pi s), \cos(\pi t)\sin(2\pi s), -\sin(\pi t)\right)$$

Why do we care?

$$p(s,t) = r\begin{pmatrix} \sin(\pi t)\cos(2\pi s) \\ \sin(\pi t)\sin(2\pi s) \\ \cos(\pi t) \end{pmatrix} + \begin{pmatrix} O_x \\ O_y \\ O_z \end{pmatrix}$$



Parameterizing the Sphere

<u>Why do we care?</u>

1. Partial derivatives gives us the normal:

$$\vec{n}(s,t) = \frac{\frac{\partial p}{\partial s} \times \frac{\partial p}{\partial t}}{\left\| \frac{\partial p}{\partial s} \times \frac{\partial p}{\partial t} \right\|}$$

Parameterizing the Sphere

Why do we care?

1. Partial derivatives gives us the normal:

$$\vec{n}(s,t)=\frac{\dfrac{\partial p}{\partial s}\times\dfrac{\partial p}{\partial t}}{\left\|\dfrac{\partial p}{\partial s}\times\dfrac{\partial p}{\partial t}\right\|}$$

$$\frac{\partial p}{\partial s}(s,t)=2\pi r\big(-\sin(\pi t)\sin(2\pi s),\sin(\pi t)\cos(2\pi s),0\big)$$

$$\frac{\partial p}{\partial t}(s,t)=\pi r\big(\cos(\pi t)\cos(2\pi s),\cos(\pi t)\sin(2\pi s),-\sin(\pi t)\big)$$

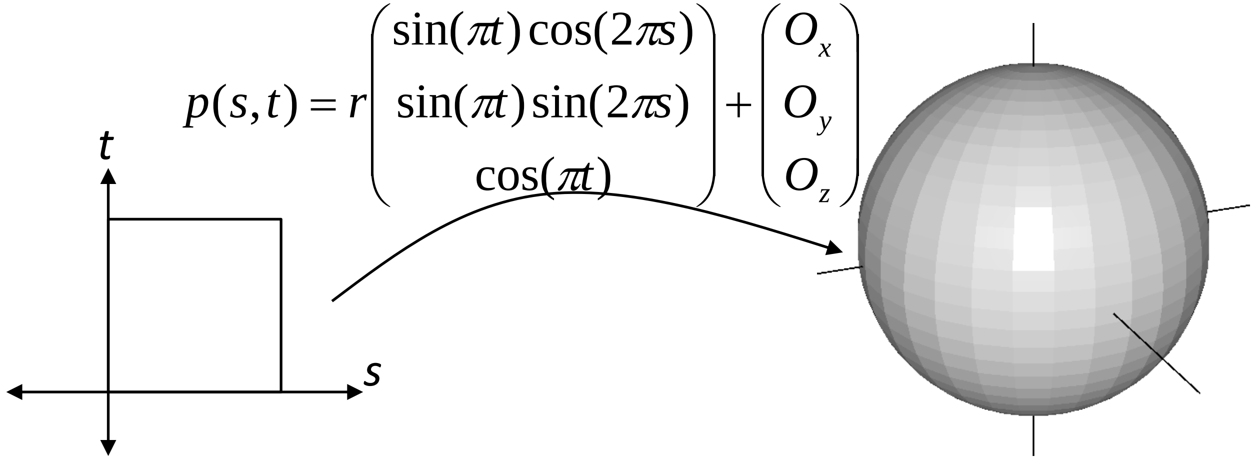
$$\vec{n}(s,t)=-\big(\cos(2\pi s)\sin(\pi t),\sin(2\pi s)\sin(\pi t),\cos(\pi t)\big)$$

Parameterizing the Sphere

<u>Why do we care?</u>

2. Partial derivatives let us compute integrals over the geometry.

$$p(s,t) = r\begin{pmatrix} \sin(\pi t)\cos(2\pi s) \\ \sin(\pi t)\sin(2\pi s) \\ \cos(\pi t) \end{pmatrix} + \begin{pmatrix} O_x \\ O_y \\ O_z \end{pmatrix}$$

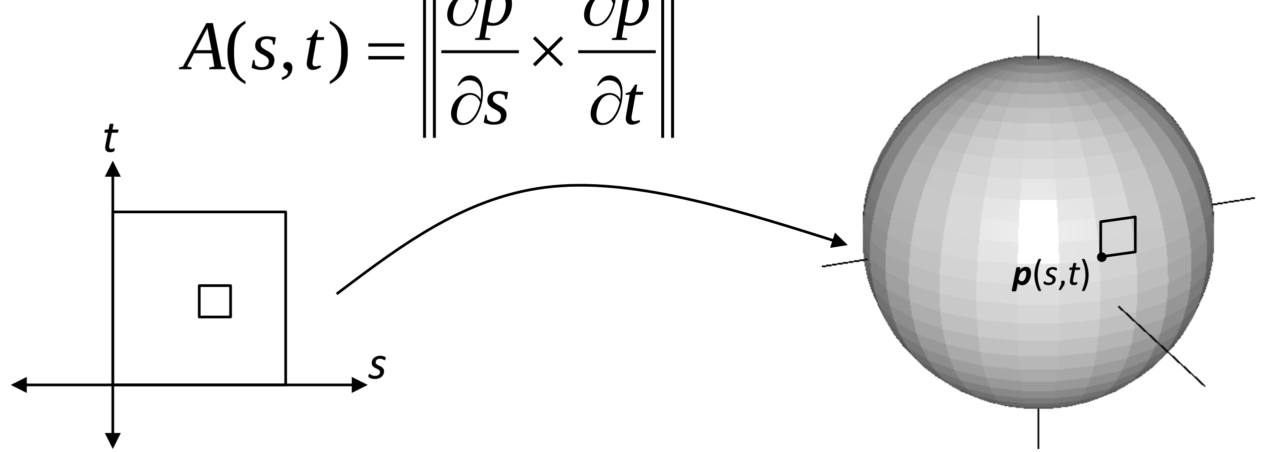


Parameterizing the Sphere

<u>Why do we care?</u>

3. Partial derivatives let us compute integrals over the geometry.
   For a unit square in the parameterization domain, the area of the corresponding patch on the surface is:

$$A(s,t) = \left\| \frac{\partial p}{\partial s} \times \frac{\partial p}{\partial t} \right\|$$



Parameterizing the Sphere

Why do we care?

3. Partial derivatives let us compute integrals over the geometry.
   For a unit square in the parameterization domain, the area of the corresponding patch on the surface is:
   $$A(s,t)=\left\|\frac{\partial p}{\partial s}\times\frac{\partial p}{\partial t}\right\|$$
   So integrals become:
   $$\int_{p\in S} f(p)dp=\int_0^1\int_0^1 f(s,t)A(s,t)dtds$$

Parameterizing the Sphere

<u>Why do we care?</u>

3. Partial derivatives let us compute integrals over the geometry.
   For example, on the sphere we have:

$$\frac{\partial p}{\partial s}(s,t) = 2\pi r\left(-\sin(\pi t)\sin(2\pi s), \sin(\pi t)\cos(2\pi s), 0\right)$$

$$\frac{\partial p}{\partial t}(s,t) = \pi r\left(\cos(\pi t)\cos(2\pi s), \cos(\pi t)\sin(2\pi s), -\sin(\pi t)\right)$$

⬇

$$A(s,t) = 2\pi^2 r^2 \sin(\pi t)$$

Parameterizing the Sphere

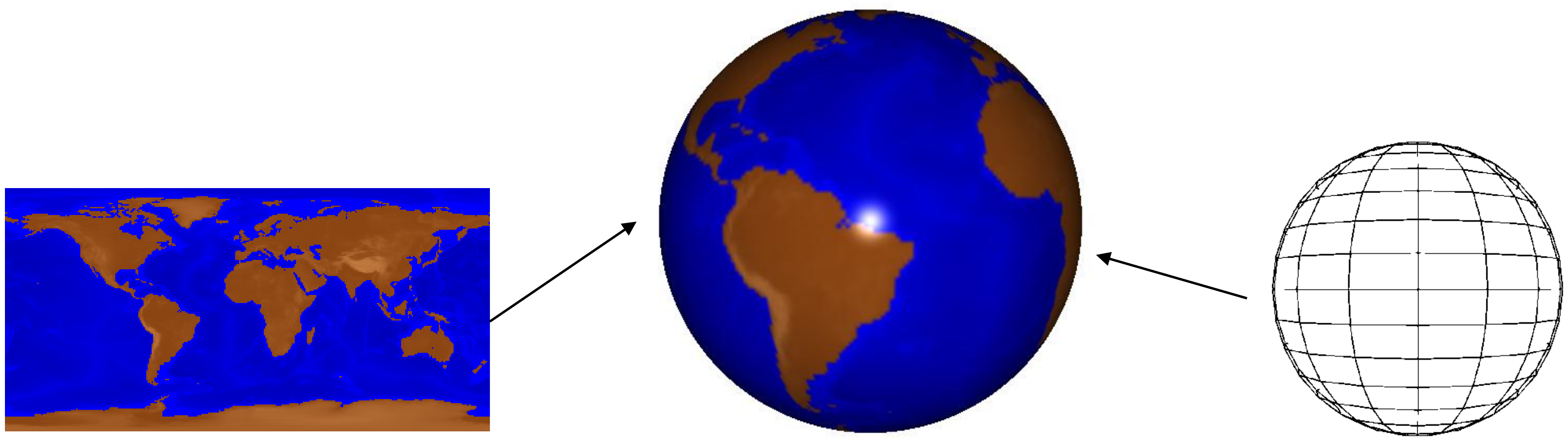
Why do we care?

3. Partial derivatives let us compute integrals over the geometry.
   For example, on the sphere we have:
   $$A(s,t) = 2\pi^2 r^2 \sin(\pi t)$$
   So the area is:
   $$\int_{p\in S^2} 1dp = 2\pi^2 r^2 \int_0^1 \int_0^1 \sin(\pi t)dtds = 4\pi r^2$$

Parameterizing the Sphere

Why do we care?

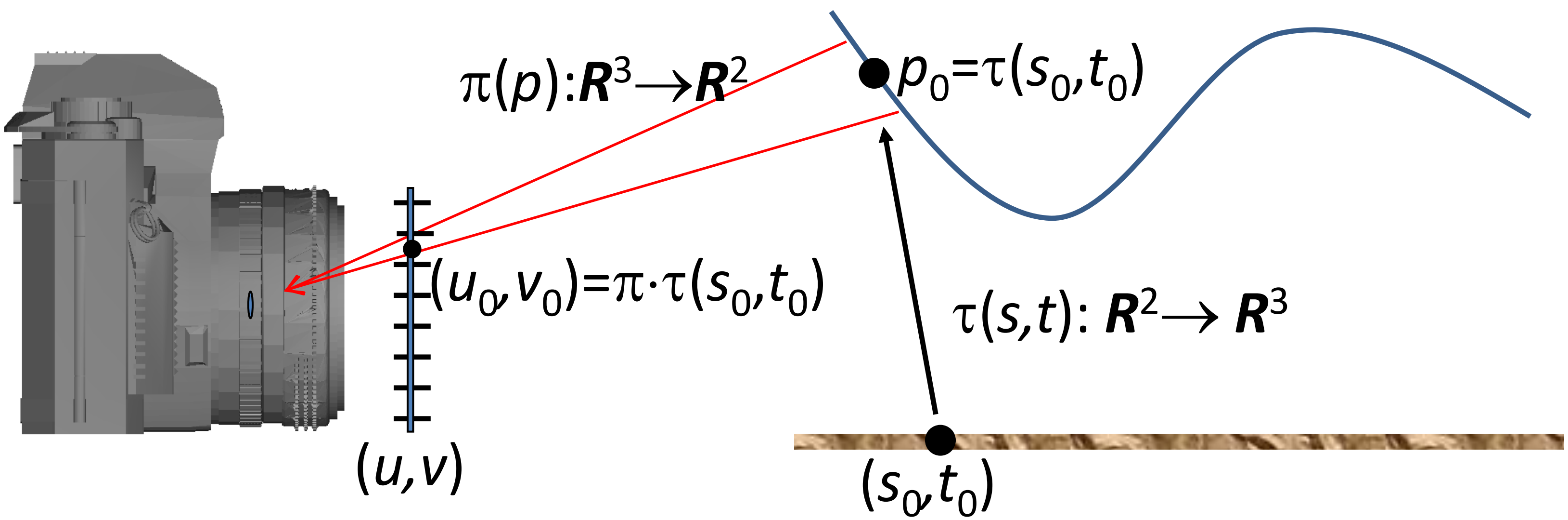
3. Partial derivatives tell us how texture gets distorted when mapped to the film plane.



Parameterizing the Sphere

Why do we care?

3. Partial derivatives tell us how texture gets distorted when mapped to the film plane.

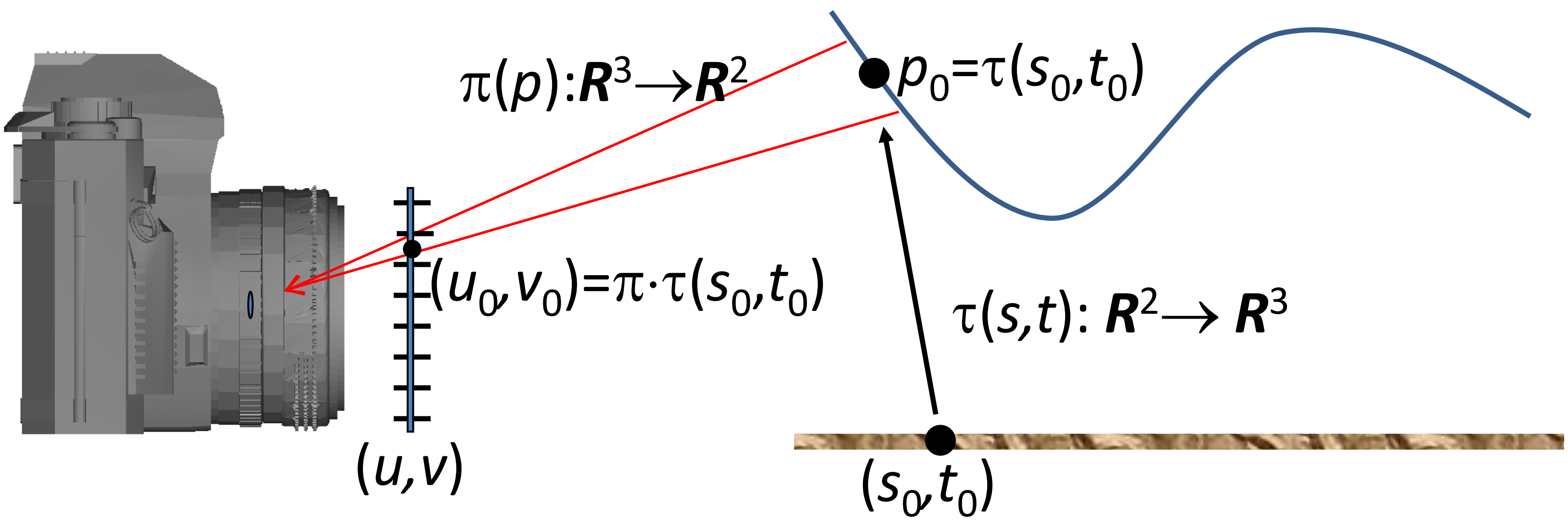


Parameterizing the Sphere

<u>Why do we care?</u>

3. Partial derivatives tell us how texture gets distorted when mapped to the film plane.

$$(\pi \circ \tau)(s_0+\delta_s, t_0+\delta_t) \approx (\pi \circ \tau)(s_0, t_0) + d(\pi \circ \tau)\begin{pmatrix}\delta_s \\ \delta_t\end{pmatrix}$$



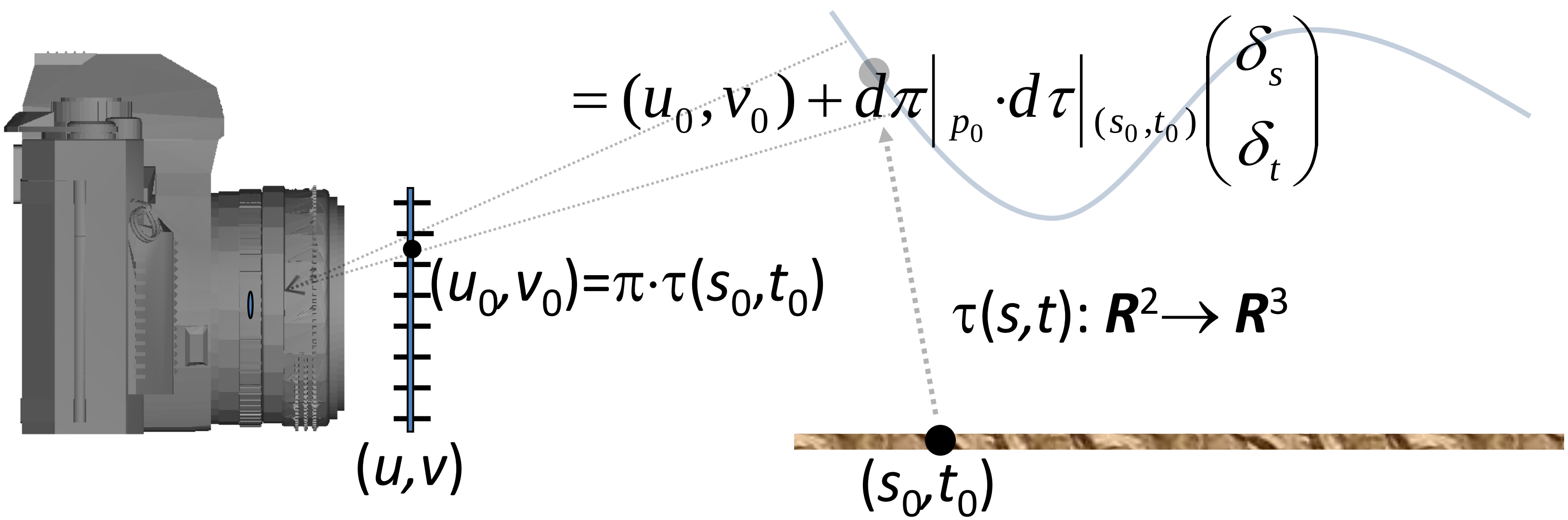
Parameterizing the Sphere

<u>Why do we care?</u>

3. Partial derivatives tell us how texture gets distorted when mapped to the film plane.

$$(\pi \circ \tau)(s_0+\delta_s, t_0+\delta_t) \approx (\pi \circ \tau)(s_0, t_0) + d(\pi \circ \tau)\Big|_{(s_0,t_0)} \begin{pmatrix} \delta_s \\ \delta_t \end{pmatrix}$$

$$= (u_0, v_0) + d\pi\Big|_{p_0} \cdot d\tau\Big|_{(s_0,t_0)} \begin{pmatrix} \delta_s \\ \delta_t \end{pmatrix}$$



Parameterizing the Sphere

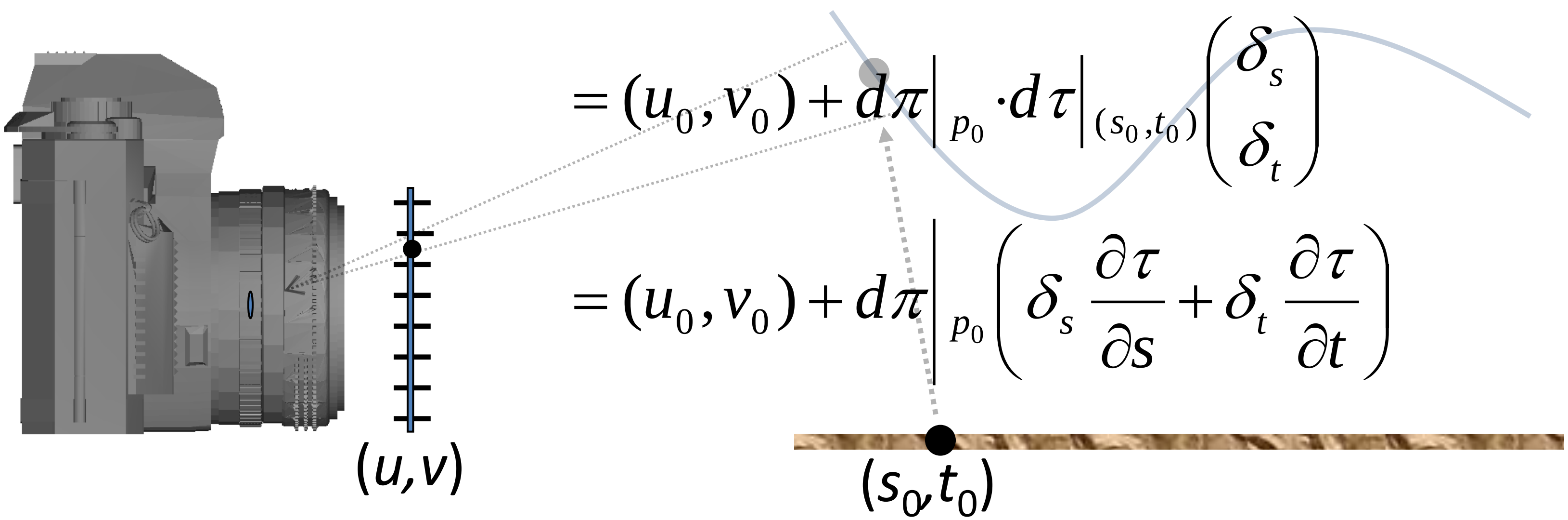
<u>Why do we care?</u>

3. Partial derivatives tell us how texture gets distorted when mapped to the film plane.

$$(\pi \circ \tau)(s_0+\delta_s, t_0+\delta_t) \approx (\pi \circ \tau)(s_0,t_0) + d(\pi \circ \tau)\Big|_{(s_0,t_0)} \begin{pmatrix} \delta_s \\ \delta_t \end{pmatrix}$$

$$= (u_0, v_0) + d\pi\Big|_{p_0} \cdot d\tau\Big|_{(s_0,t_0)} \begin{pmatrix} \delta_s \\ \delta_t \end{pmatrix}$$

$$= (u_0, v_0) + d\pi\Big|_{p_0} \left(\delta_s \frac{\partial \tau}{\partial s} + \delta_t \frac{\partial \tau}{\partial t} \right)$$



Parameterizing the Sphere

<u>Why do we care?</u>

3. F

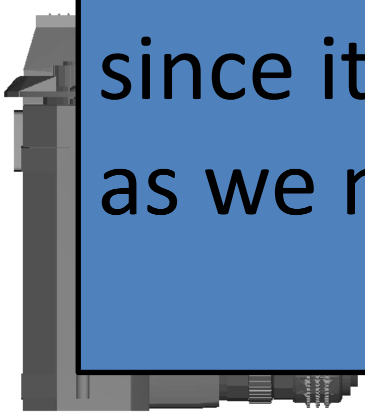
c

(π

In practice, we are more interested in the equation:

$$(\pi \circ \tau)^{-1}(y_0+\delta_u, v_0+\delta_v) \approx (s_0,t_0) + \left(d(\pi\circ\tau)\big|_{(s_0,t_0)}\right)^{-1}\begin{pmatrix}\delta_u\\ \delta_v\end{pmatrix}$$

since it tells us how texture changes as we move in the film-plane.



Parameterizing the Sphere

Why do we care?

3. F

c

(π

In practice, we are more interested in the equation:

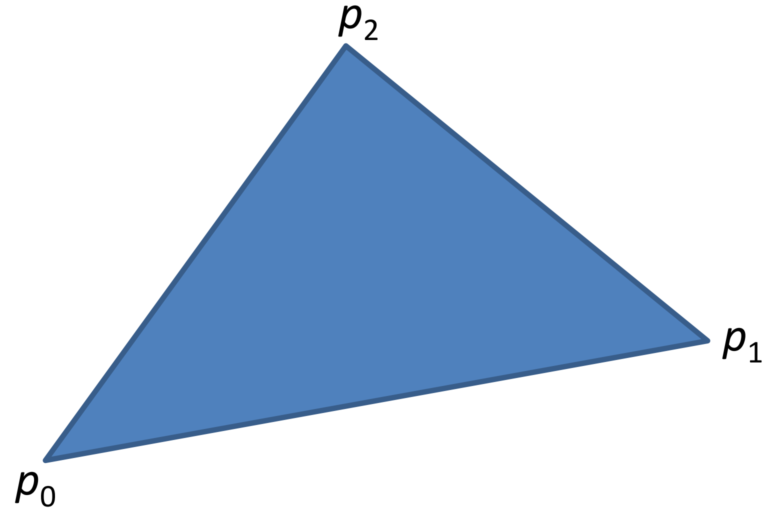
$$(\pi \circ \tau)^{-1}(y_0+\delta_u, v_0+\delta_v) \approx (s_0,t_0) + \left(d(\pi\circ\tau)\big|_{(s_0,t_0)}\right)^{-1}\begin{pmatrix}\delta_u\\ \delta_v\end{pmatrix}$$

This is good for anti-aliasing textures. To anti-alias textures reflected off the surface, we also need to know how the normals change on the reflecting surface.

Triangles

Described by three vertices:

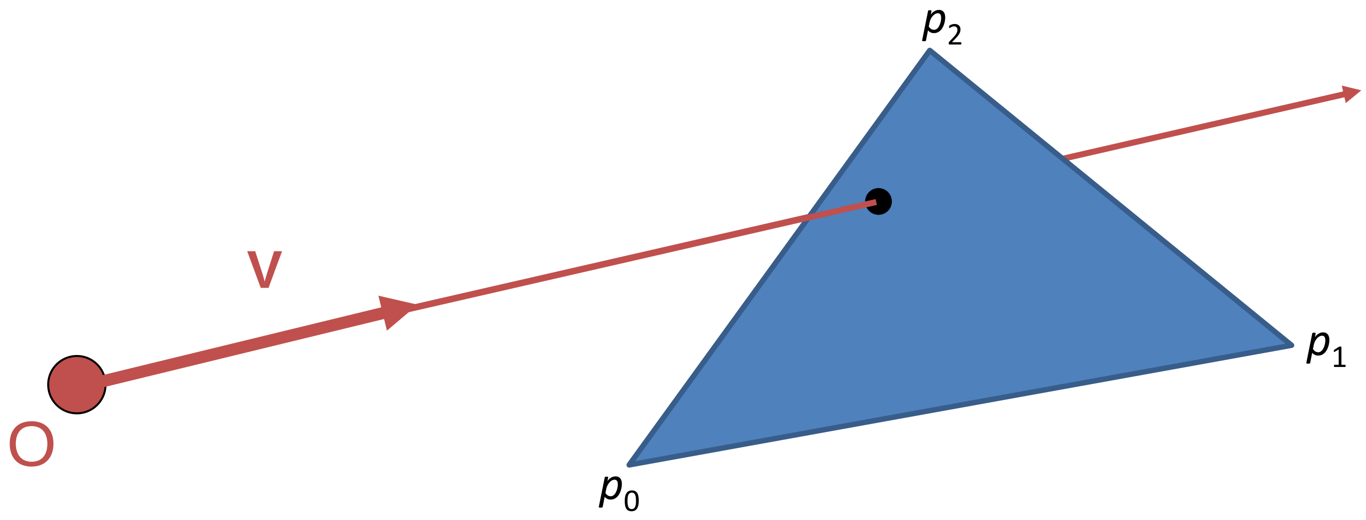
- Position: $\{p_i\}$
- Parameter values: $\{(u_i,v_i)\}$
- Normals: $\{n_i\}$



Triangle Intersection

Ray: $P = O + tV$

Triangle: $(1-b_1-b_2)p_0+b_1p_1+b_2p_2$ with $0 \leq b_1,b_2 \leq 1$

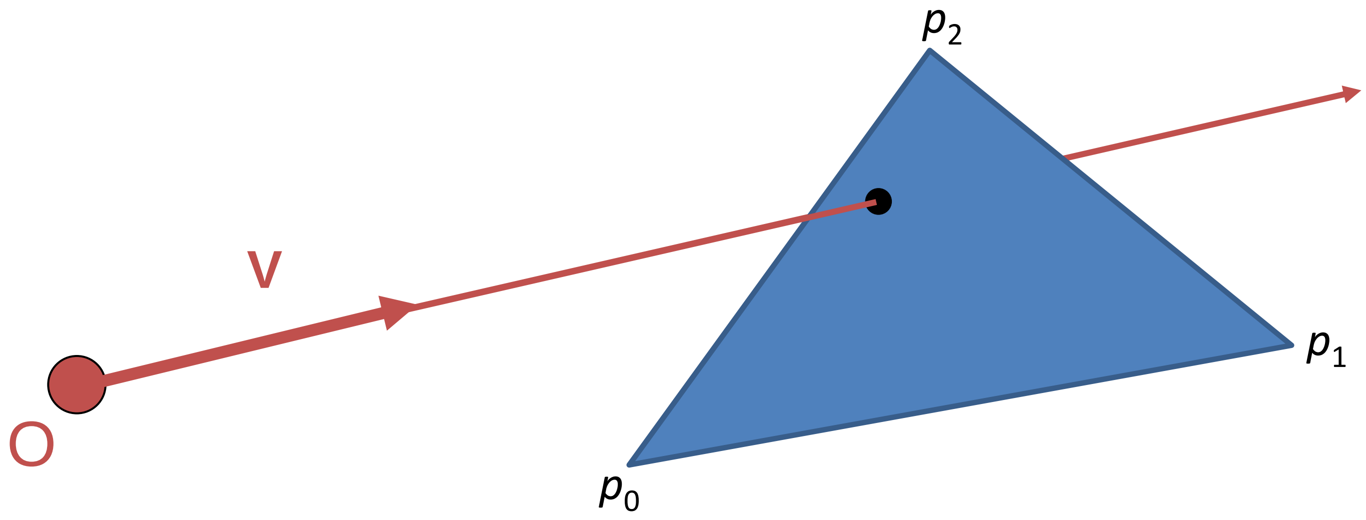


Triangle Intersection

Ray: P = O + tV

Triangle: $(1-b_1-b_2)p_0+b_1p_1+b_2p_2$ with $0 \le b_1,b_2 \le 1$

$$\begin{pmatrix} -v_x & p_{1x}-p_{0x} & p_{2x}-p_{0x} \\ -v_y & p_{1y}-p_{0y} & p_{2y}-p_{0y} \\ -v_z & p_{1z}-p_{0z} & p_{2z}-p_{0z} \end{pmatrix} \begin{pmatrix} t \\ b_1 \\ b_2 \end{pmatrix} = \begin{pmatrix} O_x - p_{0x} \\ O_y - p_{0y} \\ O_z - p_{0z} \end{pmatrix}$$



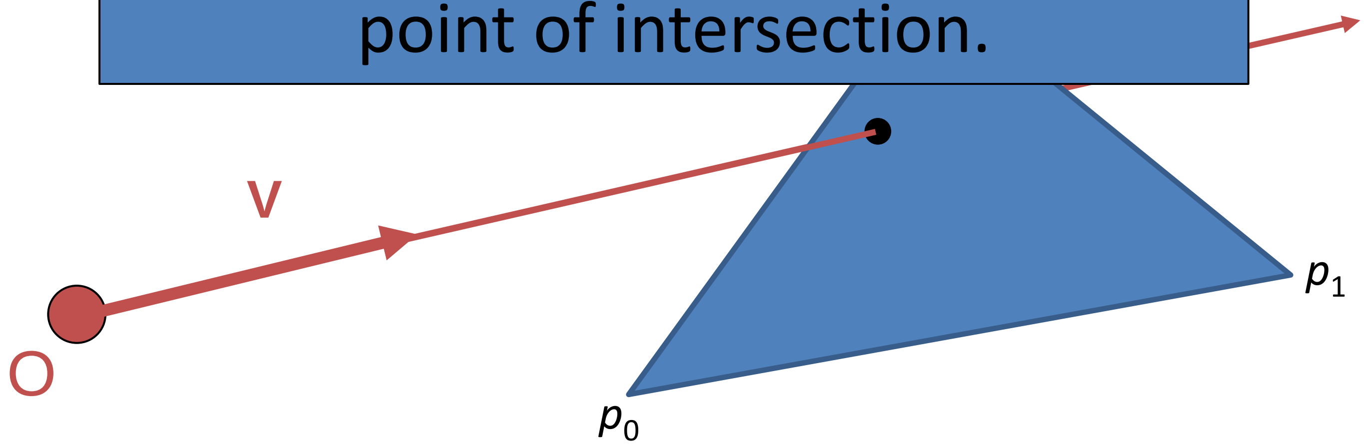
Triangle Intersection

Ray: P = O + tV

Triangle: $(1-b_1-b_2)p_0+b_1p_1+b_2p_2$ with $0 \leq b_1,b_2 \leq 1$

$$\begin{pmatrix} -v_x & p_{1x}-p_{0x} & p_{2x}-p_{0x} \\ -v_y & p_{1y}-p_{0y} & p_{2y}-p_{0y} \end{pmatrix} \begin{pmatrix} t \\ b_1 \end{pmatrix} = \begin{pmatrix} O_x - p_{0x} \\ O_y - p_{0y} \end{pmatrix}$$

Note that $(1-b_1-b_2)$, b_1, and b_2 are the barycentric coordinates of the point of intersection.



Triangle Intersection

Ray: P = O + tV

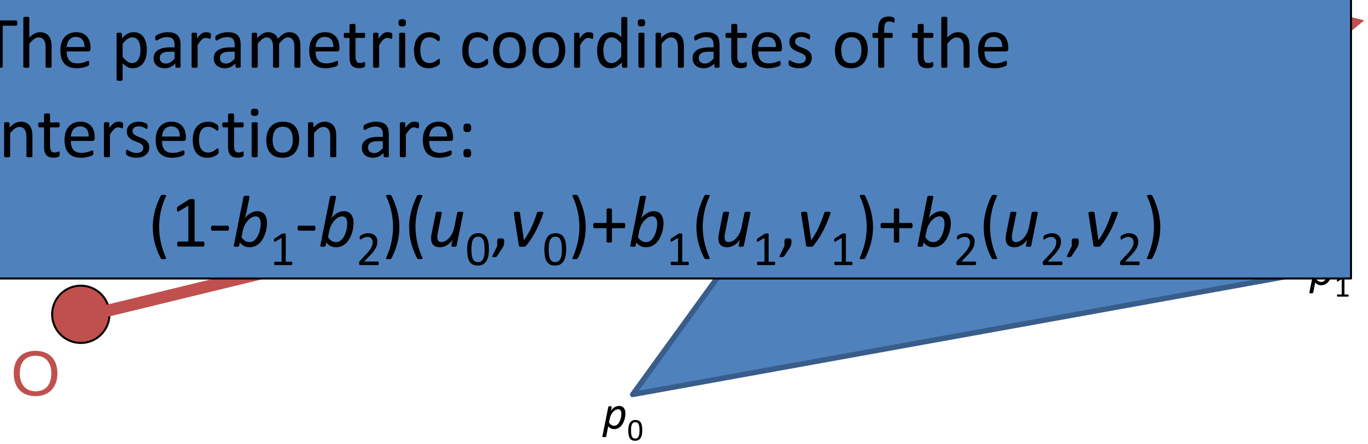
Triangle: $(1\text{-}b_1\text{-}b_2)p_0+b_1p_1+b_2p_2$ with $0 \leq b_1,b_2 \leq 1$

$$\begin{pmatrix} -v_x & p_{1x}-p_{0x} & p_{2x}-p_{0x} \\ -v_y & p_{1y}-p_{0y} & p_{2y}-p_{0y} \end{pmatrix}\begin{pmatrix} t \\ b_1 \end{pmatrix} = \begin{pmatrix} O_x - p_{0x} \\ O_y - p_{0y} \end{pmatrix}$$

Note that $(1\text{-}b_1\text{-}b_2)$, b_1, and b_2 are the barycentric coordinates of the

The parametric coordinates of the intersection are:

$$(1\text{-}b_1\text{-}b_2)(u_0,v_0)+b_1(u_1,v_1)+b_2(u_2,v_2)$$

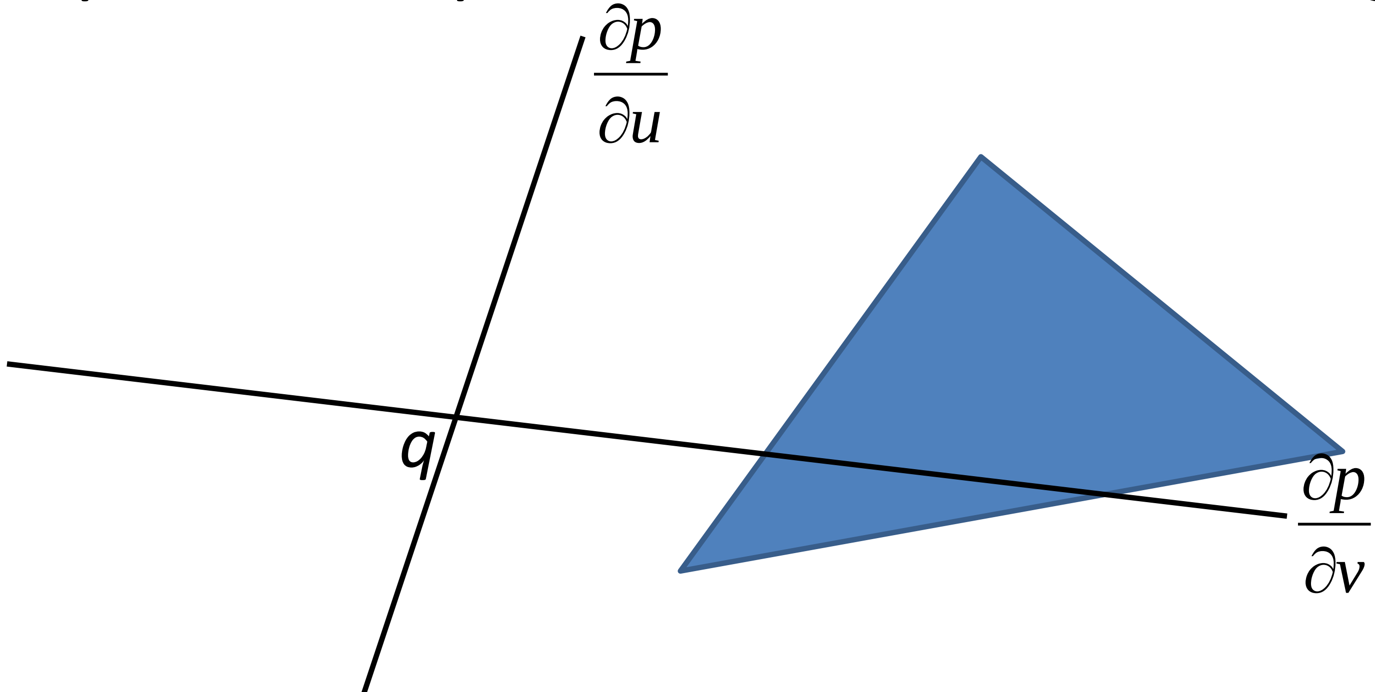


Parameterizing the Triangle

We expect a parameterization of the triangle that is linear:

$$p(u,v) = q + u\frac{\partial p}{\partial u} + v\frac{\partial p}{\partial v}$$

with $\partial p/\partial u$ and $\partial p/\partial v$ constant on the triangle.



Parameterizing the Triangle

We expect a parameterization of the triangle that is linear:

$$p(u,v) = q + u\frac{\partial p}{\partial u} + v\frac{\partial p}{\partial v}$$

with $\partial p/\partial u$ and $\partial p/\partial v$ constant on the triangle.

Given the parameter values $\{(u_i,v_i)\}$ at the three corners, q, $\partial p/\partial u$ and $\partial p/\partial v$ satisfy:

$$p_i = q + u_i\frac{\partial p}{\partial u} + v_i\frac{\partial p}{\partial v}$$

Parameterizing the Triangle

We expect a parameterization of the triangle that is linear:

$$p(u,v) = q + u\frac{\partial p}{\partial u} + v\frac{\partial p}{\partial v}$$

with $\partial p/\partial u$ and $\partial p/\partial v$ constant on the triangle.

Given the parameter values $\{(u_i,v_i)\}$ at the three corners, q, $\partial p/\partial u$ and $\partial p/\partial v$ satisfy:

$$p_i = q + u_i\frac{\partial p}{\partial u} + v_i\frac{\partial p}{\partial v}$$

$$p_i - p_1 = (u_i - u_1)\frac{\partial p}{\partial u} + (v_i - v_1)\frac{\partial p}{\partial v}$$

Parameterizing the Triangle

$$p_i - p_1 = (u_i - u_1)\frac{\partial p}{\partial u} + (v_i - v_1)\frac{\partial p}{\partial v}$$

We obtain the partials by solving:

$$\begin{pmatrix} (p_2 - p_1)_{x/y/z} \\ (p_3 - p_1)_{x/y/z} \end{pmatrix} = \begin{pmatrix} u_2 - u_1 & v_2 - v_1 \\ u_3 - u_1 & v_3 - v_1 \end{pmatrix} \begin{pmatrix} \left(\frac{\partial p}{\partial u}\right)_{x/y/z} \\ \left(\frac{\partial p}{\partial v}\right)_{x/y/z} \end{pmatrix}$$

Shading the Triangle

In addition to the true geometry (e.g. constant normal) there is also the shading geometry defined by the normals:

- ~~The shading normal is the normal of the triangle.~~

Shading the Triangle

In addition to the true geometry (e.g. constant normal) there is also the shading geometry defined by the normals:

- ~~The shading normal is the normal of the triangle.~~
- ~~The (shading) change in normal is zero.~~

Shading the Triangle

In addition to the true geometry (e.g. constant normal) there is also the shading geometry defined by the normals:

– The shading normal is:

$$n = (1 - b_1 - b_2) n_0 + b_1 n_1 + b_2 n_2$$

Shading the Triangle

In addition to the true geometry (e.g. constant normal) there is also the shading geometry defined by the normals:

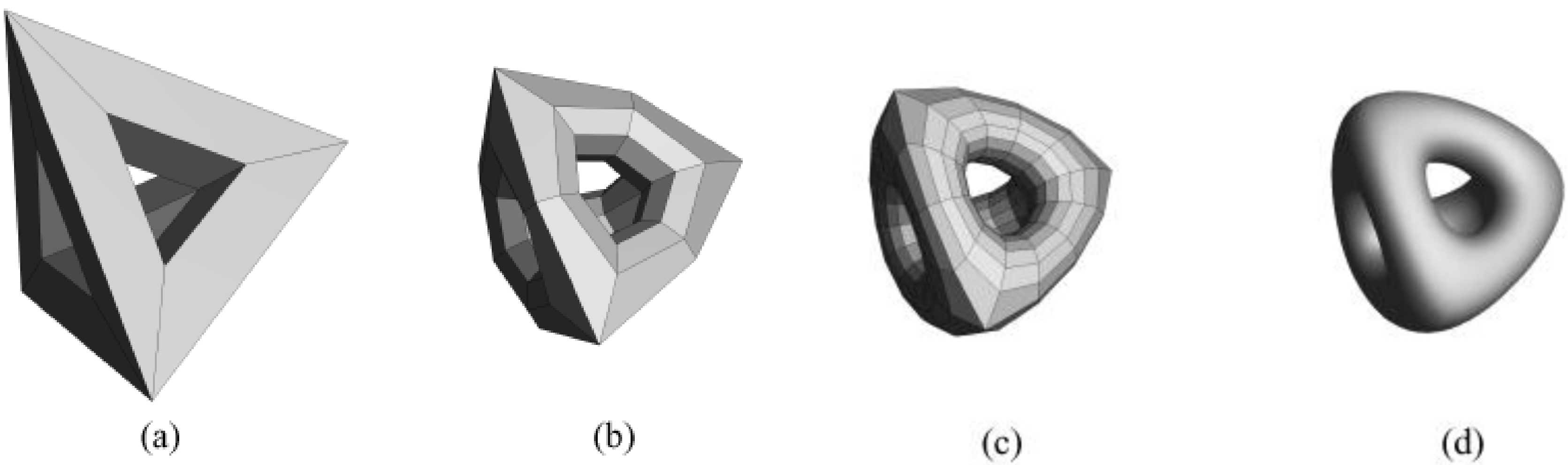
– The shading normal is:

$$n=(1-b_1-b_2)n_0+b_1n_1+b_2n_2$$

– The (shading) change in normal can be obtained by fitting a linear interpolant to the vertex normals and taking the derivative.

Subdivision Surfaces

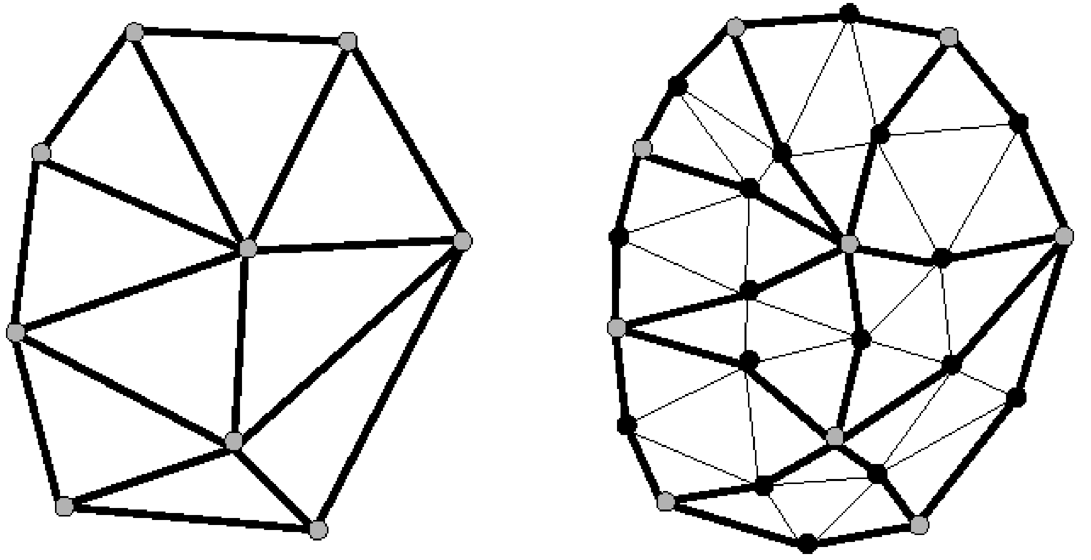
- Coarse mesh & subdivision rule
  - Define smooth surface as limit of sequence of refinements



Zorin & Schroeder
SIGGRAPH 99
Course Notes

Key Questions

- How to subdivide the mesh?
  - Aim for properties like smoothness



Zorin & Schroeder
SIGGRAPH 99
Course Notes

General Subdivision Scheme

- How to subdivide the mesh?

  Two parts:

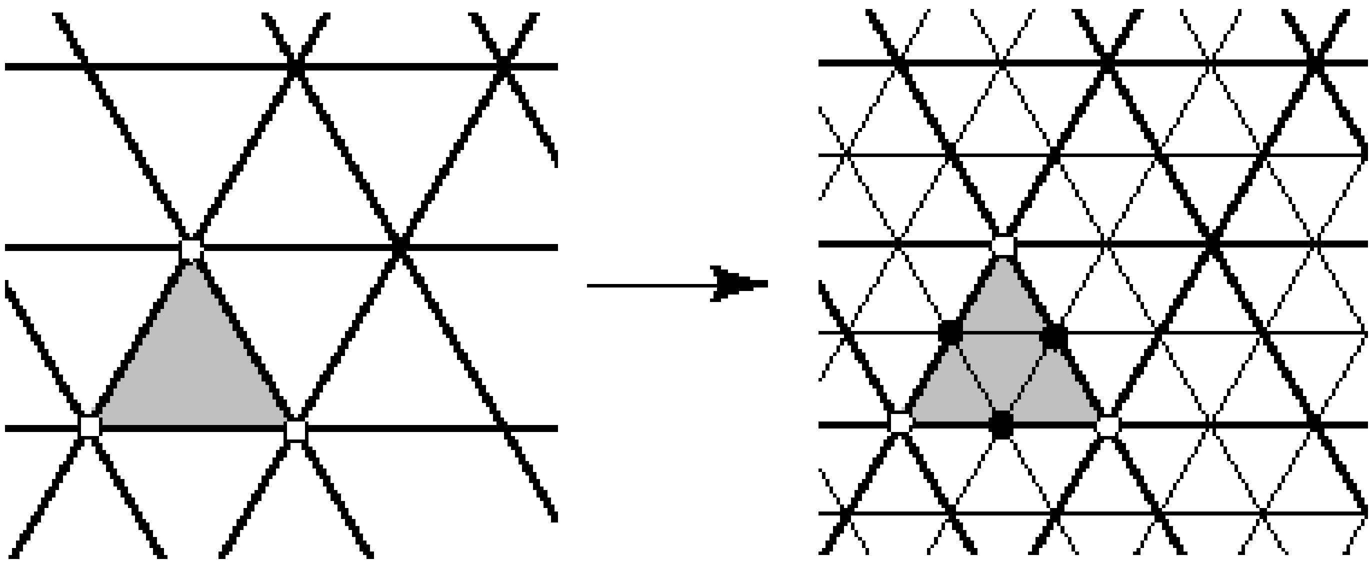
  - Refinement:
    - Add new vertices and connect (topological)
  - Smoothing:
    - Move vertex positions (geometric)

Loop Subdivision Scheme

- How to subdivide the mesh?

  Refinement:

  - Subdivide each triangle into 4 triangles by splitting each edge and connecting new vertices



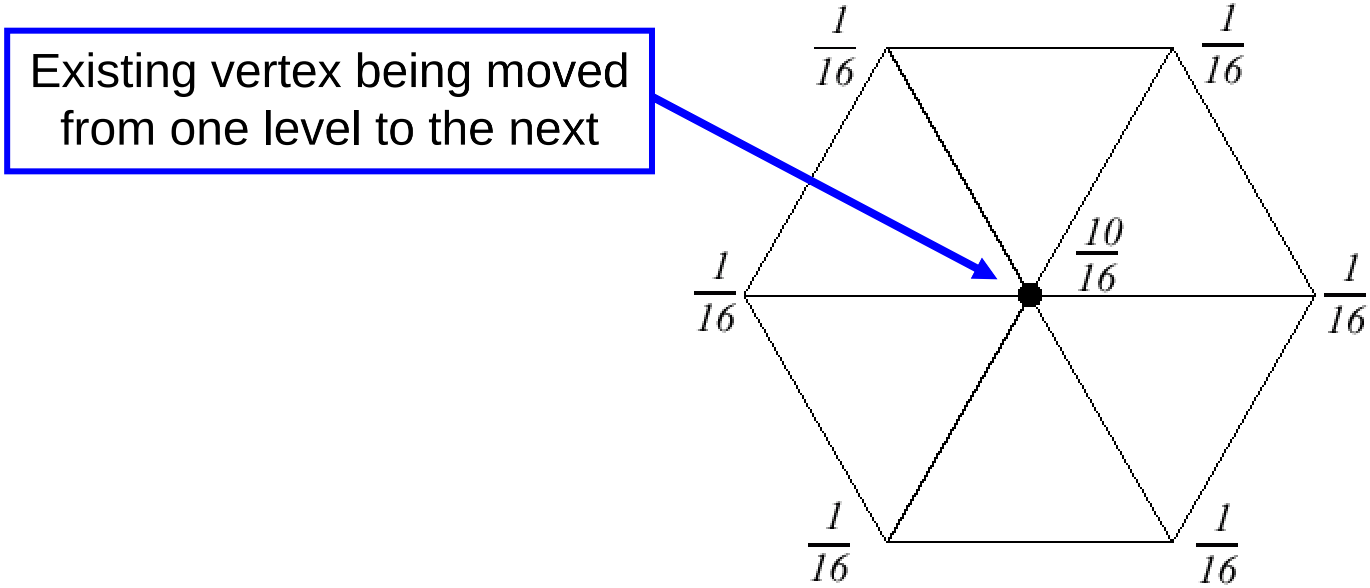
Loop Subdivision Scheme

- How to subdivide the mesh:

  Refinement

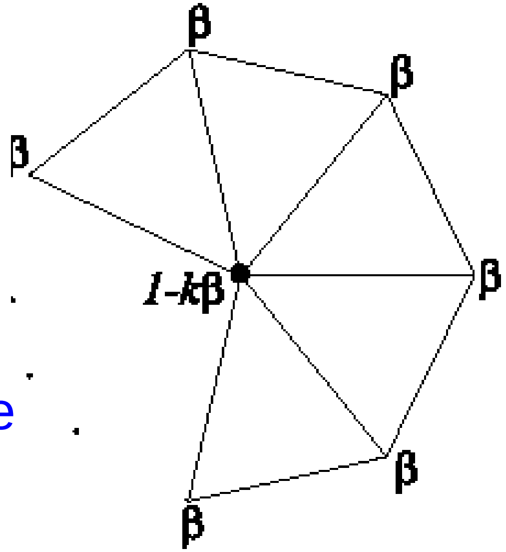
  Smoothing:

  - <u>Existing Vertices</u>: Choose *new* location as weighted average of *original* vertex and its neighbors



Loop Subdivision Scheme

- General rule for moving existing *interior vertices*:

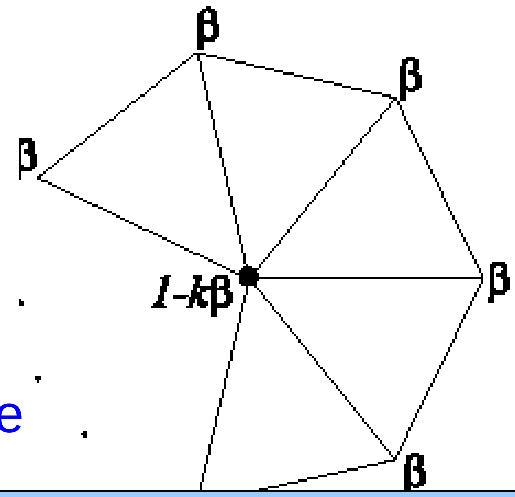


What about vertices that have more
Or less than 6 neighboring faces?

New_position = (1 - $k\beta$)original_position + sum(β * *each_original_vertex*)

Loop Subdivision Scheme

- General rule for moving existing *interior vertices*:



What about vertices that have more
Or less than 6 neighboring faces?

New ... *tex*)

<u>0≤ β ≤1/k</u>:

• As β increases, the contribution from adjacent vertices plays a more important role.

Where do existing vertices move?

- How to choose β?
  - Analyze properties of limit surface
  - Interested in continuity of surface and smoothness
  - Involves calculating eigenvalues of matrices
    - Original Loop

$$\beta = \frac{1}{k}\left(\frac{5}{8} - \left(\frac{3}{8} + \frac{1}{4}\cos\frac{2\pi}{k}\right)^2\right)$$

    - Warren

$$\beta = \begin{cases} \frac{3}{8k} & n > 3 \\ \frac{3}{16} & n = 3 \end{cases}$$

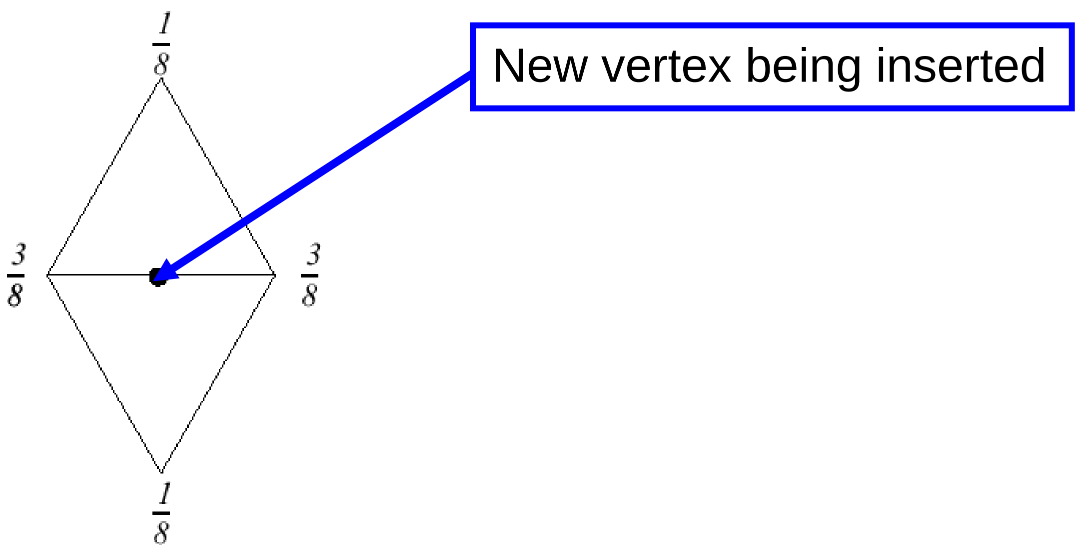
Loop Subdivision Scheme

- How to subdivide the mesh:

  Refinement

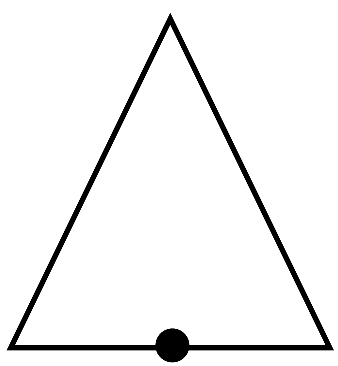
  Smoothing:

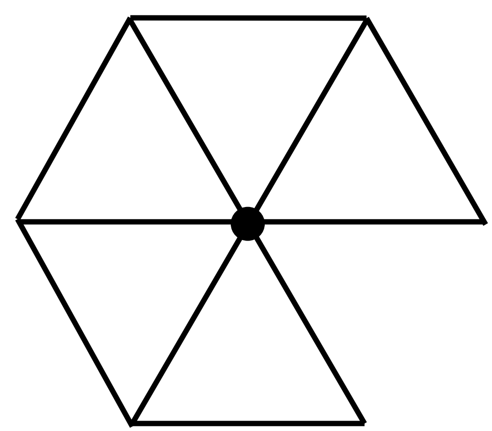
  - <u>Inserted Vertices</u>: Choose location as weighted average of *original* vertices in local neighborhood



Boundary Cases?

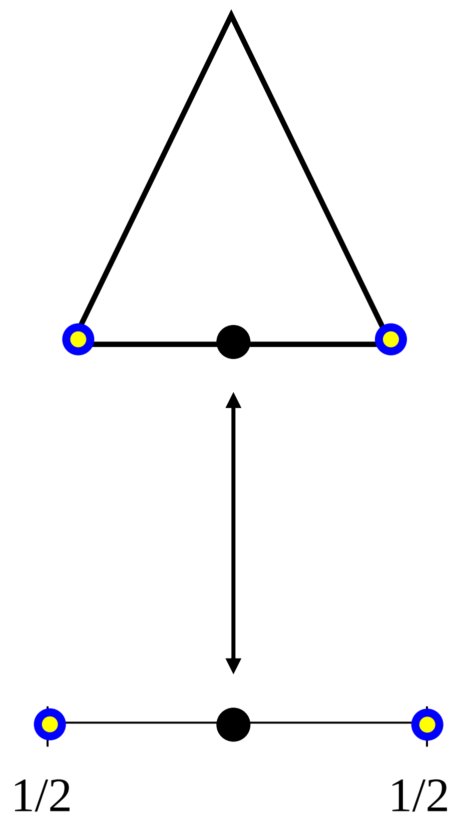
- What about *extraordinary vertices* and *boundary edges?*:
  - Existing vertex adjacent to a missing triangle
  - New vertex bordered by only one triangle

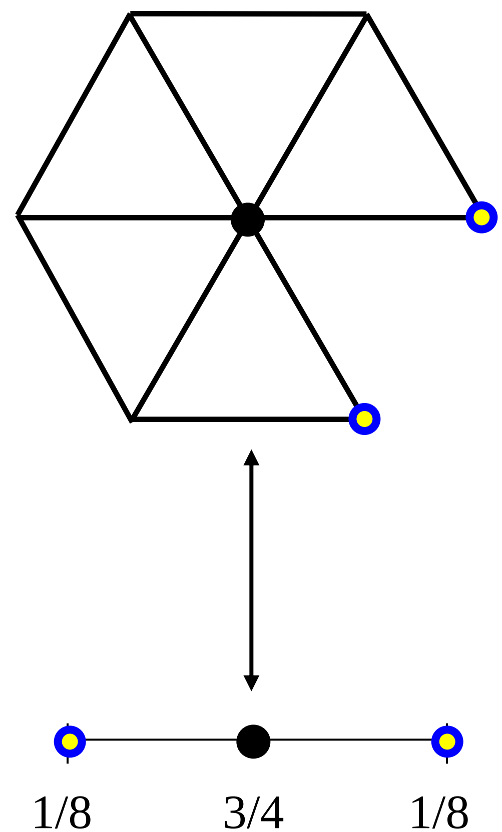




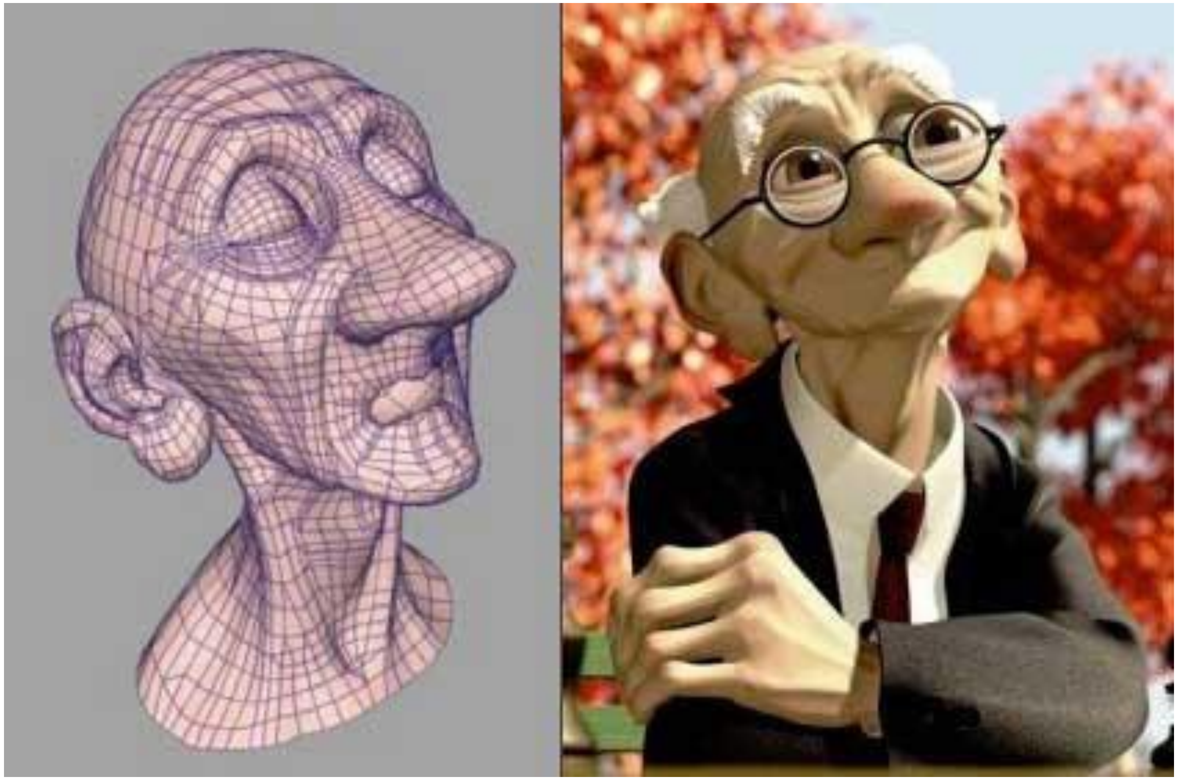
Boundary Cases?

- Rules for *extraordinary vertices* and *boundaries*:



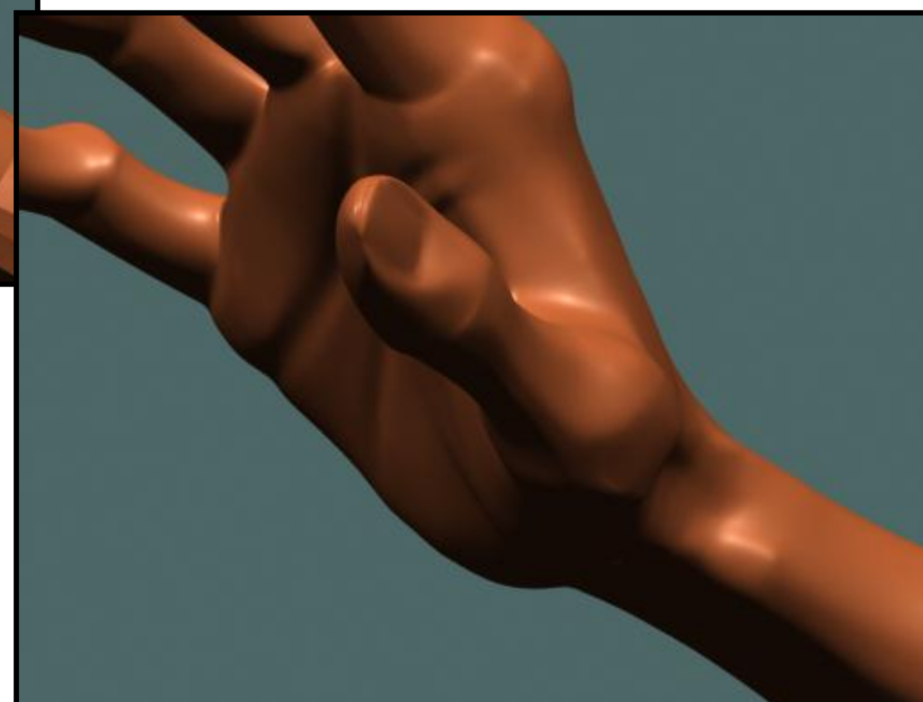


Loop Subdivision Scheme



Pixar

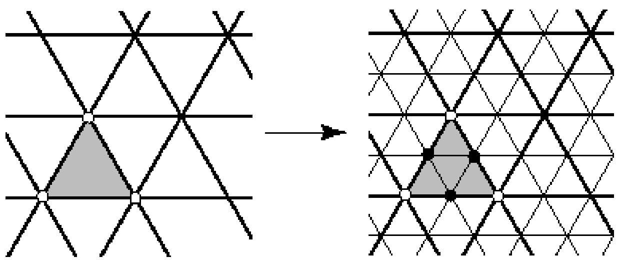
Loop Subdivision Scheme



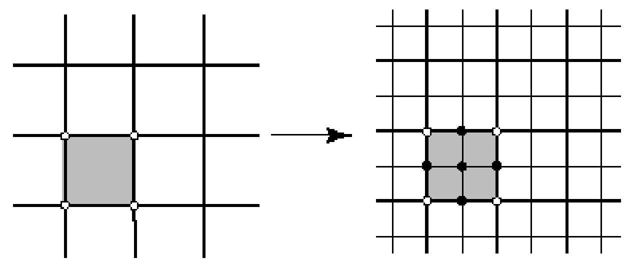
Zorin & Schroeder
SIGGRAPH 99
Course Notes

Subdivision Schemes

- There are different subdivision schemes
  - Different methods for refining topology
  - Different rules for positioning vertices



Face split for triangles



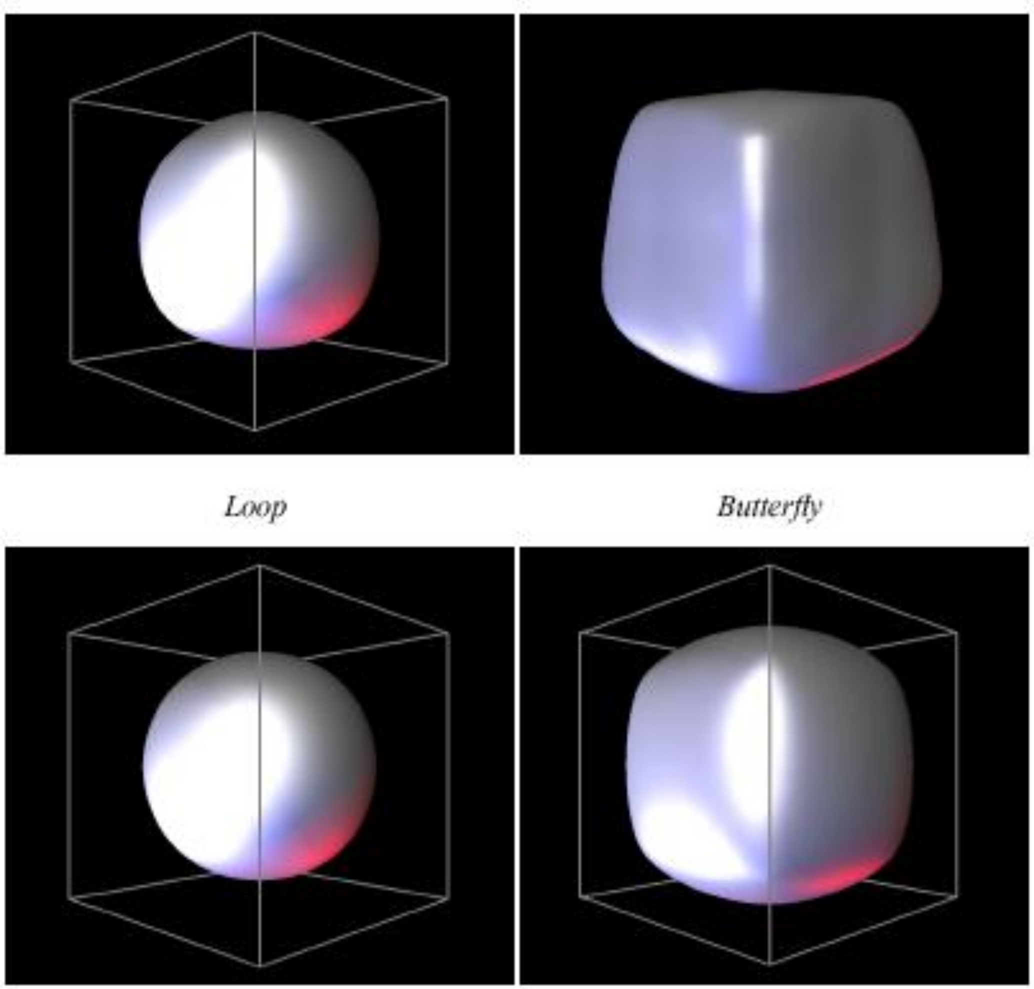
Face split for quads

Face split		
	Triangular meshes	*Quad. meshes*
Approximating	Loop (C^2)	Catmull-Clark (C^2)
Interpolating	Mod. Butterfly (C^1)	Kobbelt (C^1)

Vertex split
Doo-Sabin, Midedge (C^1)
Biquartic (C^2)

Zorin & Schroeder, SIGGRAPH 99 , Course Notes

Subdivision Schemes



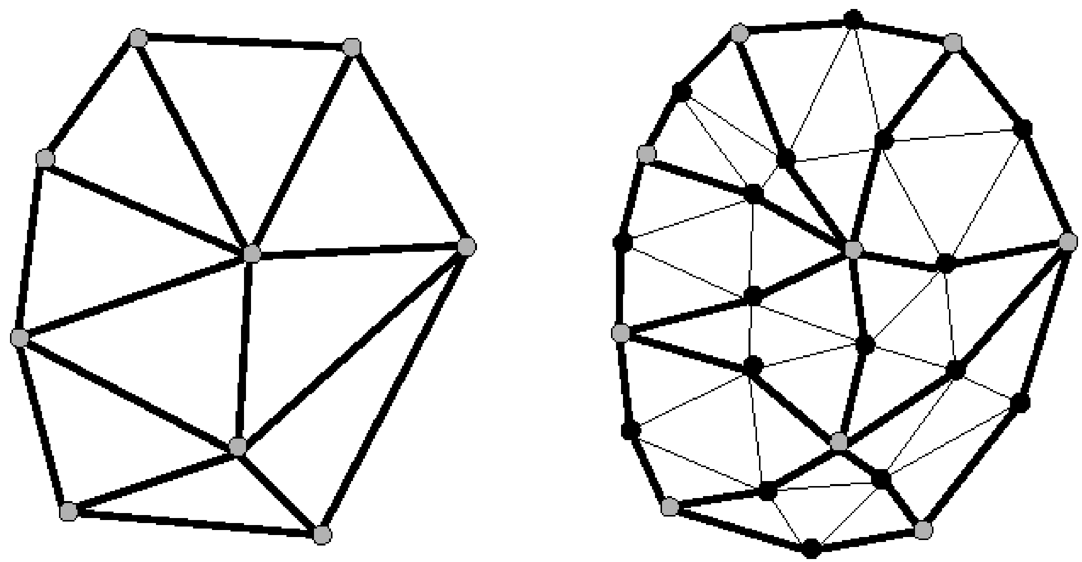
Loop *Butterfly*

Catmull-Clark *Doo-Sabin*

Zorin & Schroeder
SIGGRAPH 99
Course Notes

Key Questions

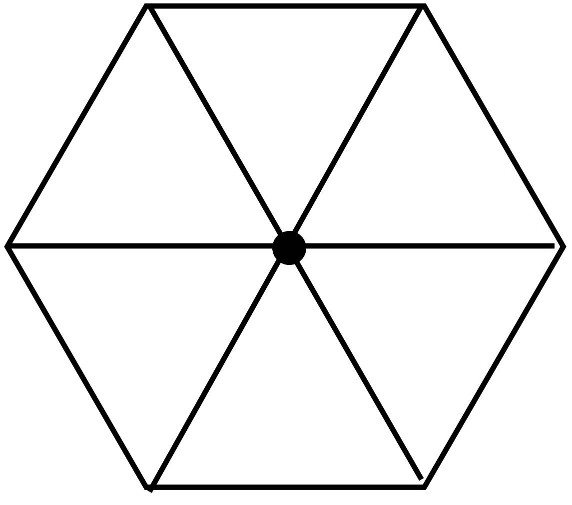
- How to refine the mesh?
  - **Aim for properties like smoothness**



Subdivision Smoothness

To determine the smoothness of the subdivision:

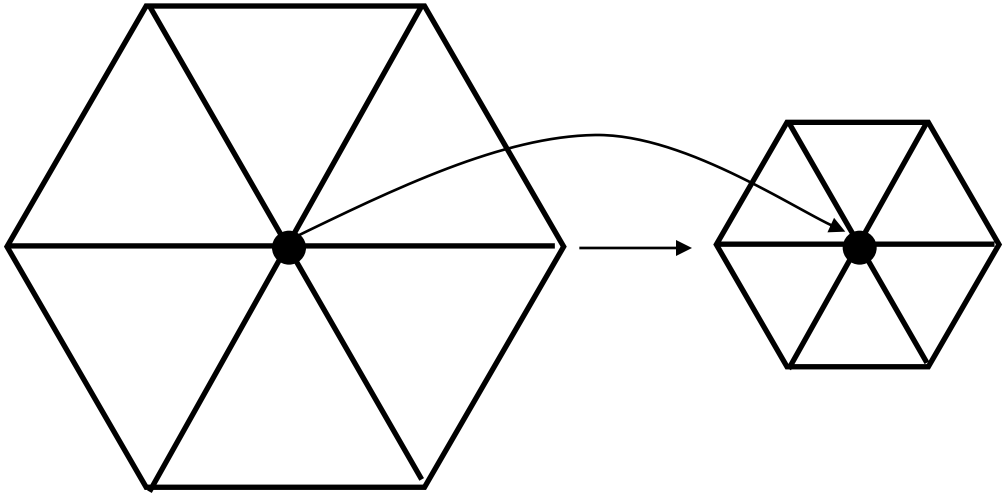
- Repeatedly apply the subdivision scheme
- Look at the neighborhood in the limit.



Subdivision Smoothness

To determine the smoothness of the subdivision:

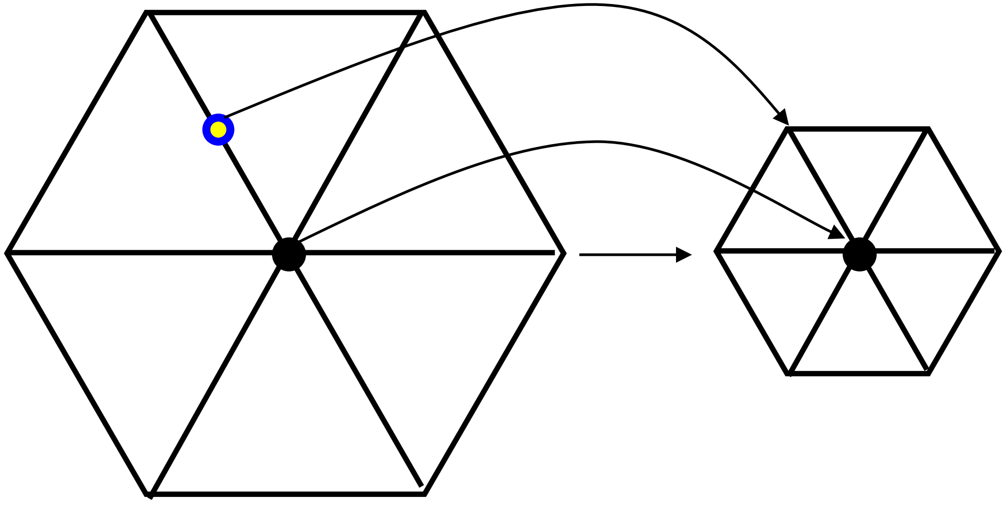
- Repeatedly apply the subdivision scheme
- Look at the neighborhood in the limit.



Subdivision Smoothness

To determine the smoothness of the subdivision:

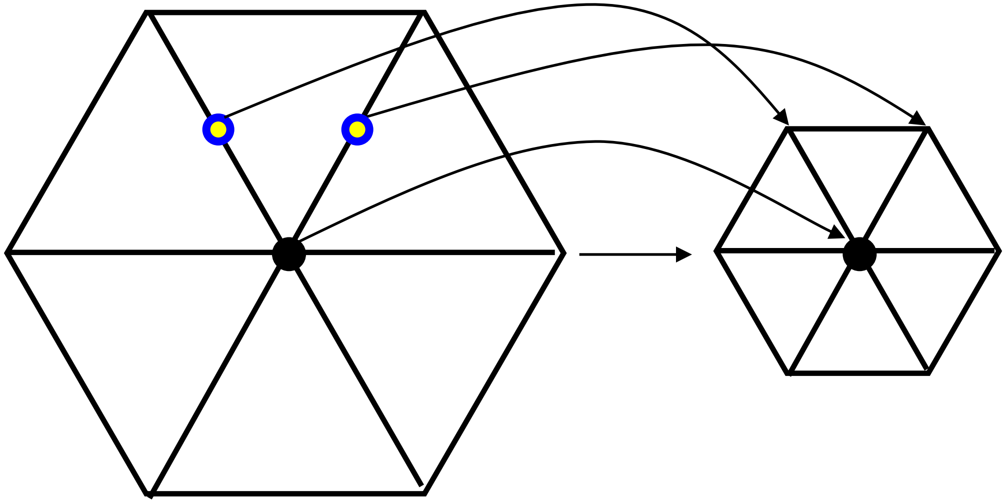
- Repeatedly apply the subdivision scheme
- Look at the neighborhood in the limit.



Subdivision Smoothness

To determine the smoothness of the subdivision:

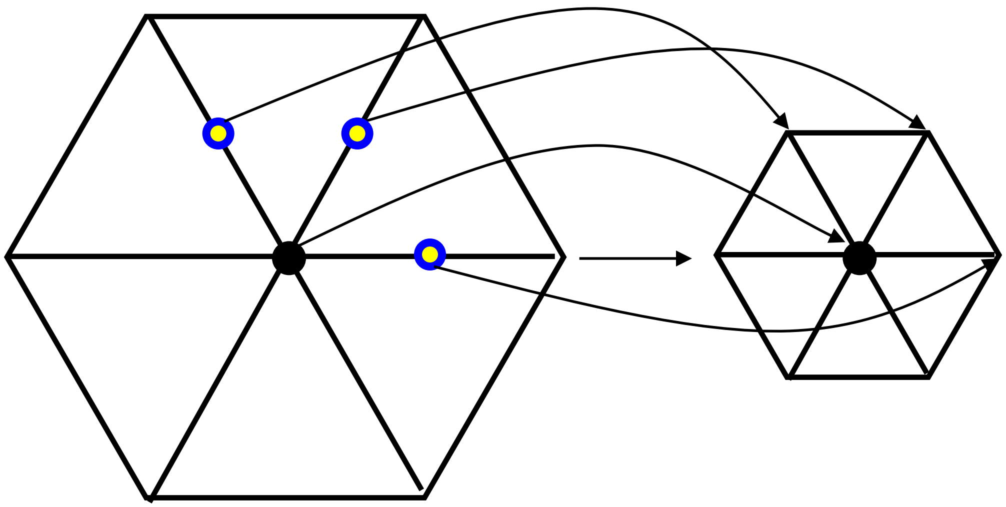
- Repeatedly apply the subdivision scheme
- Look at the neighborhood in the limit.



Subdivision Smoothness

To determine the smoothness of the subdivision:

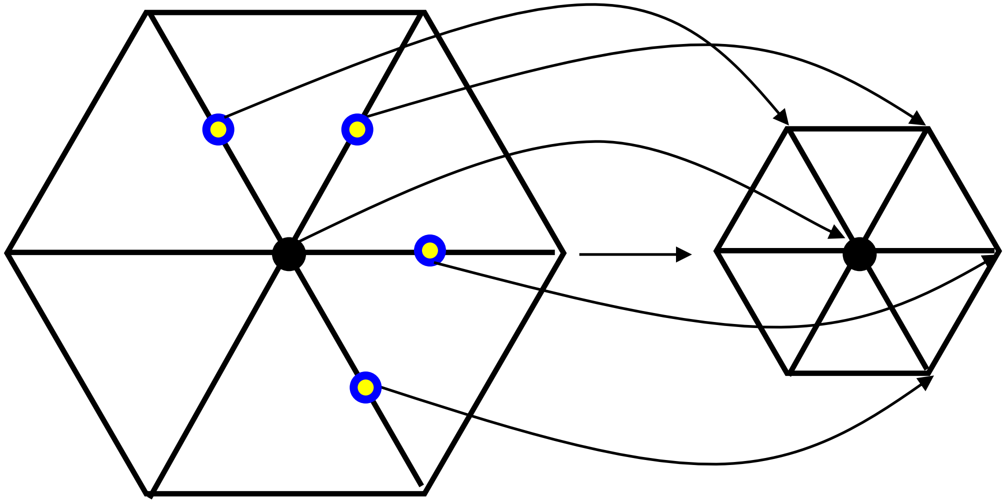
- Repeatedly apply the subdivision scheme
- Look at the neighborhood in the limit.



Subdivision Smoothness

To determine the smoothness of the subdivision:

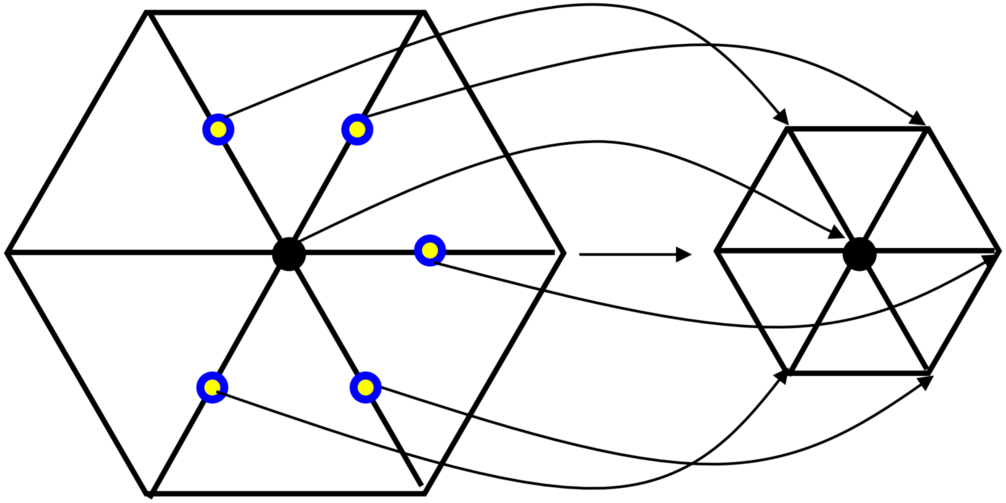
- Repeatedly apply the subdivision scheme
- Look at the neighborhood in the limit.



Subdivision Smoothness

To determine the smoothness of the subdivision:

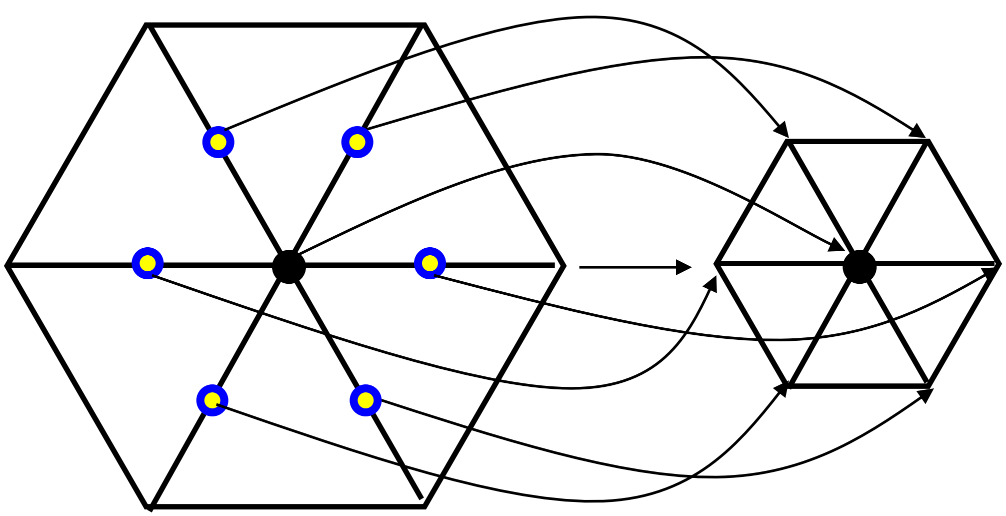
- Repeatedly apply the subdivision scheme
- Look at the neighborhood in the limit.



Subdivision Smoothness

To determine the smoothness of the subdivision:

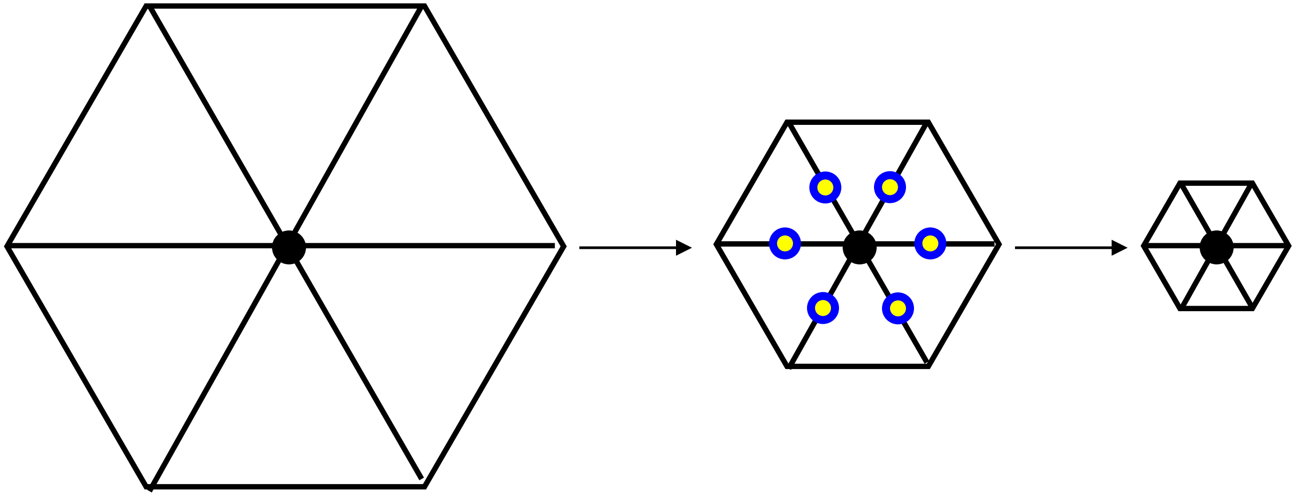
- Repeatedly apply the subdivision scheme
- Look at the neighborhood in the limit.



Subdivision Smoothness

To determine the smoothness of the subdivision:

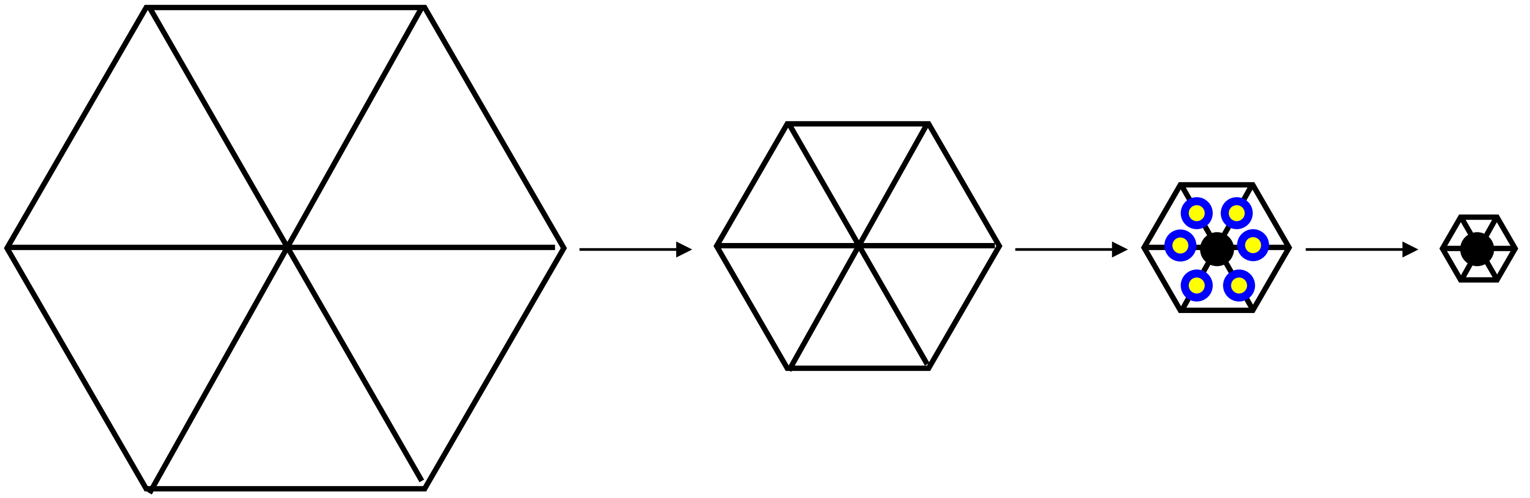
- Repeatedly apply the subdivision scheme
- Look at the neighborhood in the limit.



Subdivision Smoothness

To determine the smoothness of the subdivision:

- Repeatedly apply the subdivision scheme
- Look at the neighborhood in the limit.

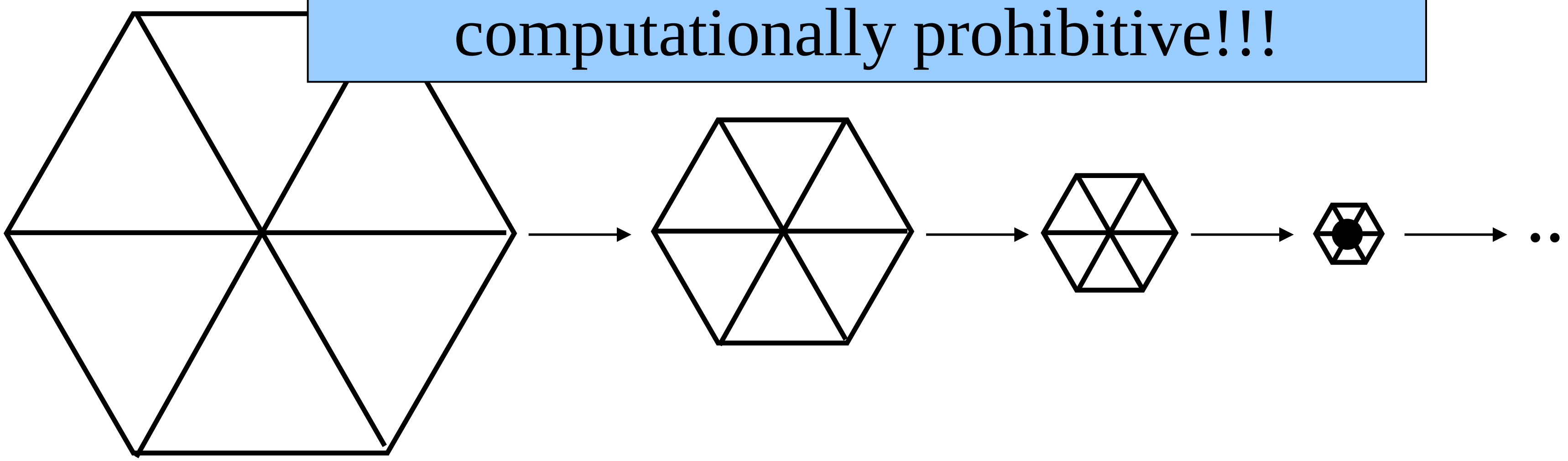


Subdivision Smoothness

To determine the smoothness of the subdivision:

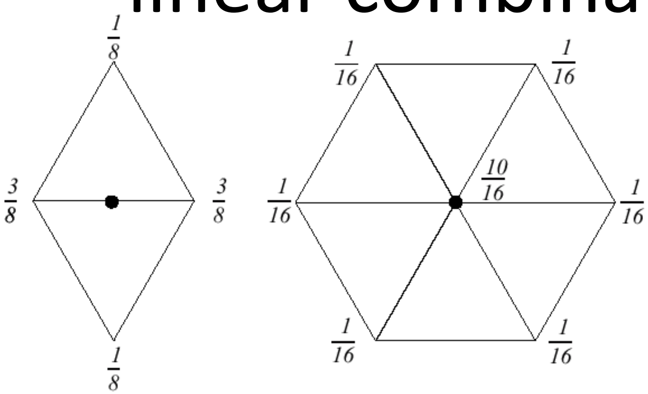
- Repeatedly apply the subdivision scheme
- Look a

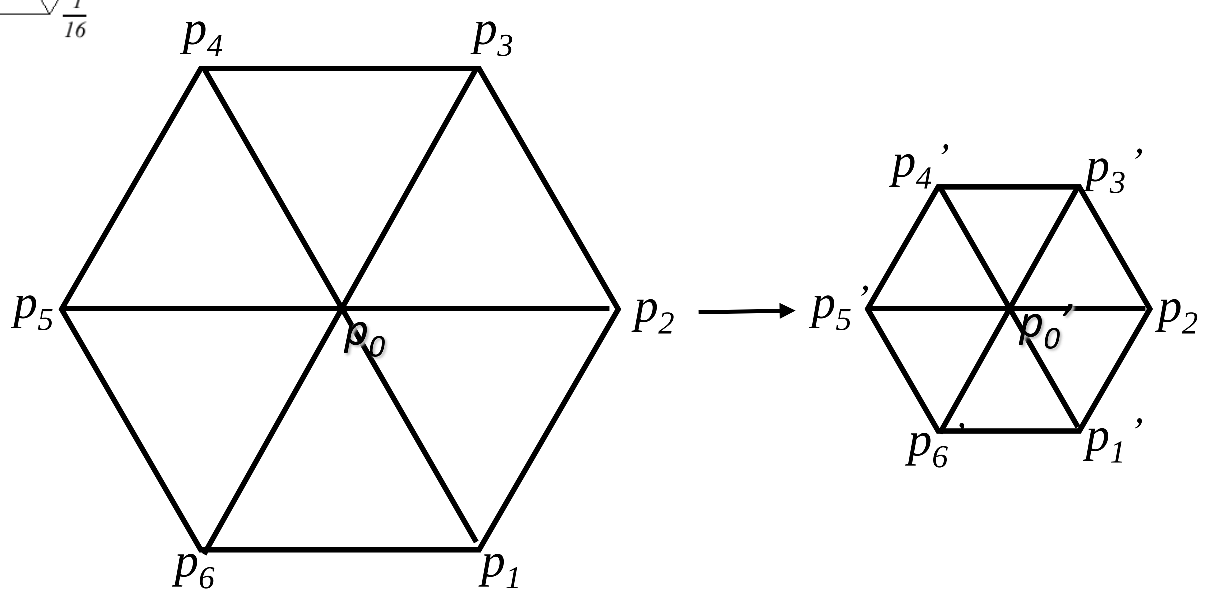
Computing infinitely many iterations is computationally prohibitive!!!



Subdivision Matrix

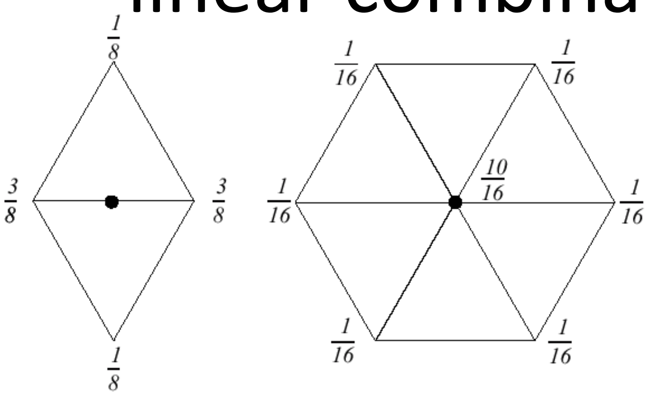
- Compute the new positions/vertices as a linear combination of previous ones.





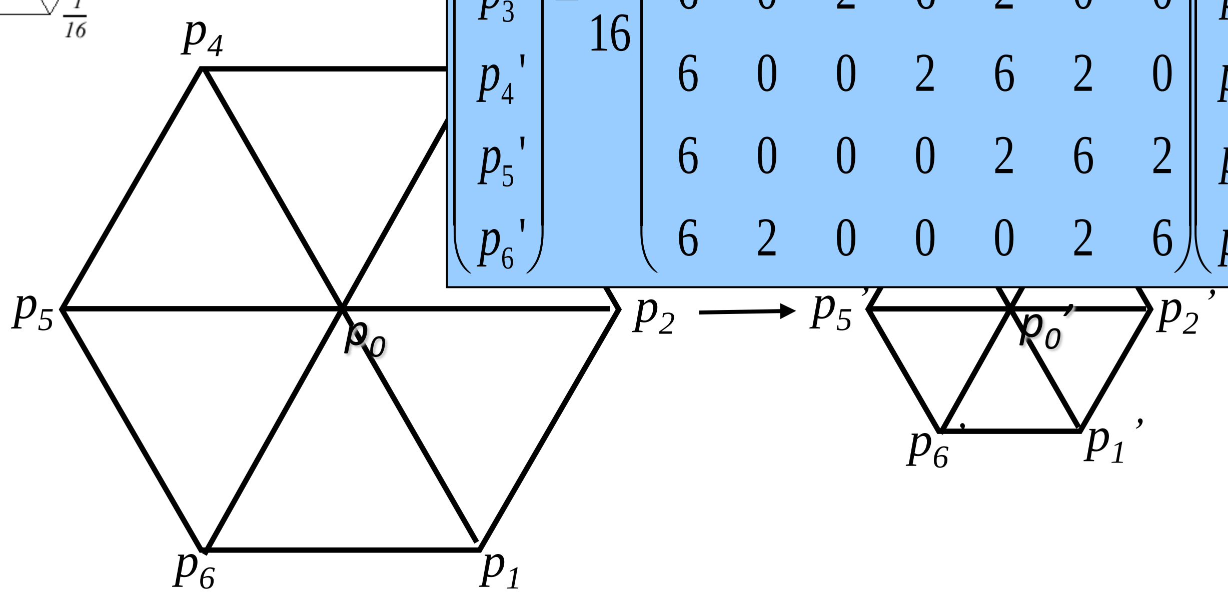
Subdivision Matrix

- Compute the new positions/vertices as a linear combination of p



Subdivision Matrix

$$\begin{pmatrix} p_0' \\ p_1' \\ p_2' \\ p_3' \\ p_4' \\ p_5' \\ p_6' \end{pmatrix} = \frac{1}{16} \begin{pmatrix} 10 & 1 & 1 & 1 & 1 & 1 & 1 \\ 6 & 6 & 2 & 0 & 0 & 0 & 2 \\ 6 & 2 & 6 & 2 & 0 & 0 & 0 \\ 6 & 0 & 2 & 6 & 2 & 0 & 0 \\ 6 & 0 & 0 & 2 & 6 & 2 & 0 \\ 6 & 0 & 0 & 0 & 2 & 6 & 2 \\ 6 & 2 & 0 & 0 & 0 & 2 & 6 \end{pmatrix} \begin{pmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \end{pmatrix}$$



Subdivision Matrix

- Compute the new positions/vertices as a linear combination of previous ones.
- To find the limit position of p_0, repeatedly apply the subdivision matrix.

$$\begin{pmatrix} p_0^{(n)} \\ p_1^{(n)} \\ p_2^{(n)} \\ p_3^{(n)} \\ p_4^{(n)} \\ p_5^{(n)} \\ p_6^{(n)} \end{pmatrix} = \left[\frac{1}{16} \begin{pmatrix} 10 & 1 & 1 & 1 & 1 & 1 & 1 \\ 2 & 2 & 6 & 0 & 0 & 0 & 6 \\ 2 & 6 & 2 & 6 & 0 & 0 & 0 \\ 2 & 0 & 6 & 2 & 6 & 0 & 0 \\ 2 & 0 & 0 & 6 & 2 & 6 & 0 \\ 2 & 0 & 0 & 0 & 6 & 2 & 6 \\ 2 & 6 & 0 & 0 & 0 & 6 & 2 \end{pmatrix} \right]^n \begin{pmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \end{pmatrix}$$

Subdivision Matrix

- Compute the new positions/vertices as a linear combination of previous ones.
- To find the limit position of p_0, repeatedly apply the subdivision matrix.
- Use eigen-value decomposition to compute the n^{th} power of the matrix efficiently.

$$\begin{pmatrix} p_0^{(n)} \\ p_1^{(n)} \\ p_2^{(n)} \\ p_3^{(n)} \\ p_4^{(n)} \\ p_5^{(n)} \\ p_6^{(n)} \end{pmatrix} = \left[\frac{1}{16} \begin{pmatrix} 10 & 1 & 1 & 1 & 1 & 1 & 1 \\ 2 & 2 & 6 & 0 & 0 & 0 & 6 \\ 2 & 6 & 2 & 6 & 0 & 0 & 0 \\ 2 & 0 & 6 & 2 & 6 & 0 & 0 \\ 2 & 0 & 0 & 6 & 2 & 6 & 0 \\ 2 & 0 & 0 & 0 & 6 & 2 & 6 \\ 2 & 6 & 0 & 0 & 0 & 6 & 2 \end{pmatrix} \right]^n \begin{pmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \end{pmatrix}$$

Subdivision Matrix

- Compute the new positions/vertices as a linear combination of previous ones.
- To find the limit position of p_0, repeatedly apply the subdivision matrix.
- Use eigen-value

$$\begin{pmatrix} p_0^{(n)} \end{pmatrix} = \left[\begin{pmatrix} 10 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \right]^n \begin{pmatrix} p_0 \end{pmatrix}$$

If, after a change of basis we have $M=A^{-1}DA$, where D is a diagonal matrix, then:

$$M^n=A^{-1}D^nA,$$

Since D is diagonal, raising D to the n-th power just amounts to raising each of the diagonal entries of D to the n-th power.