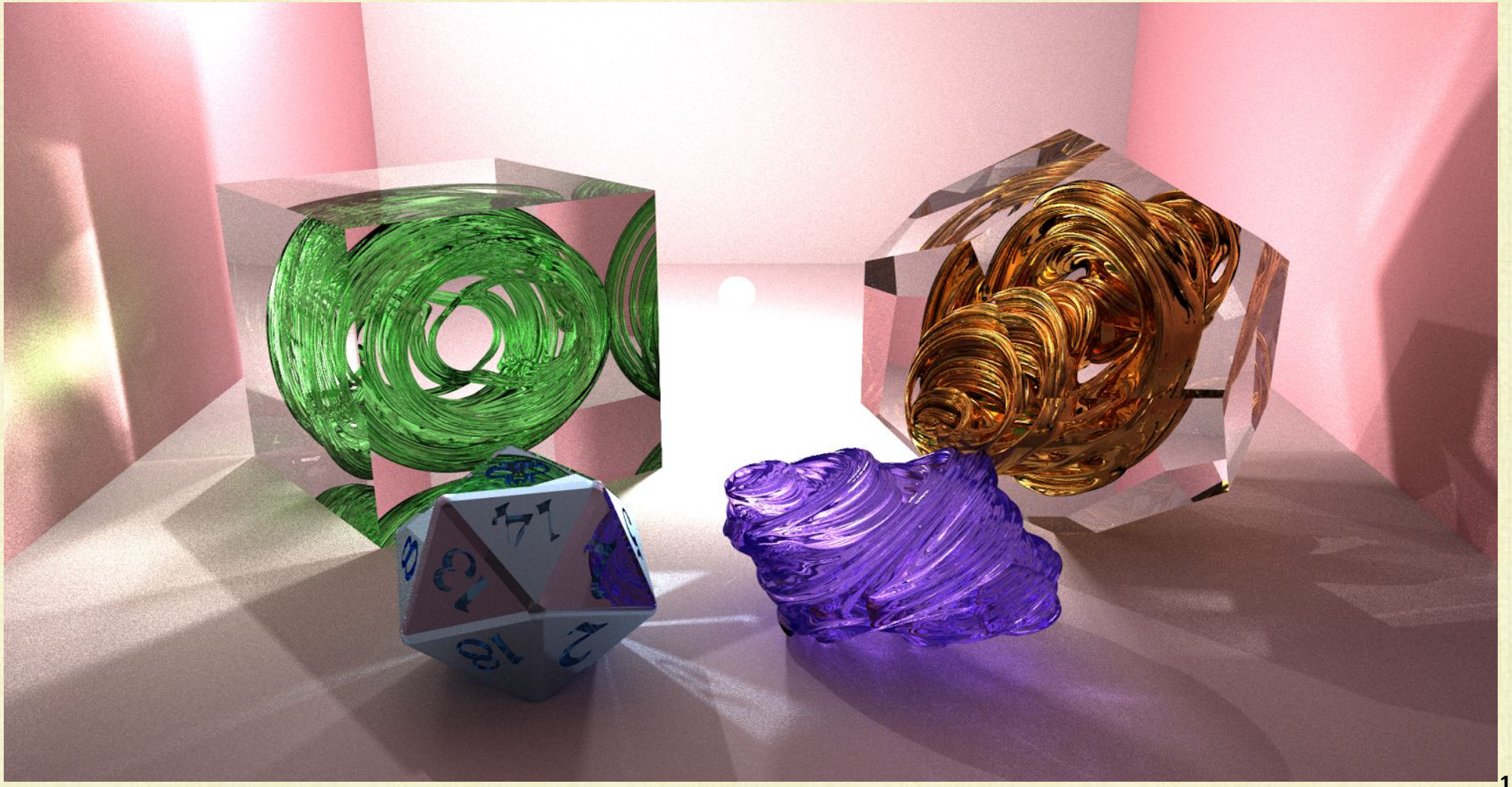
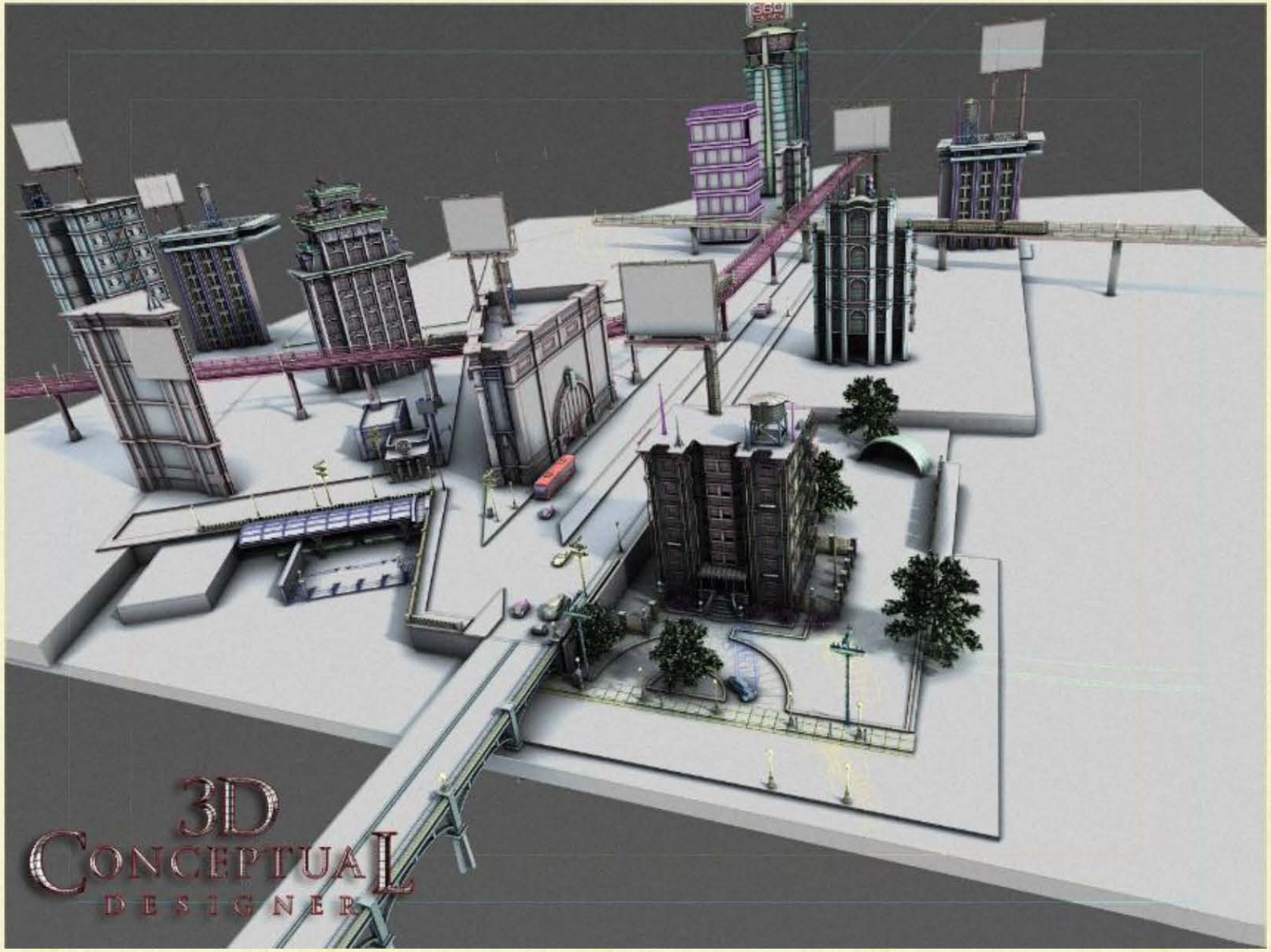
Geometric Modeling

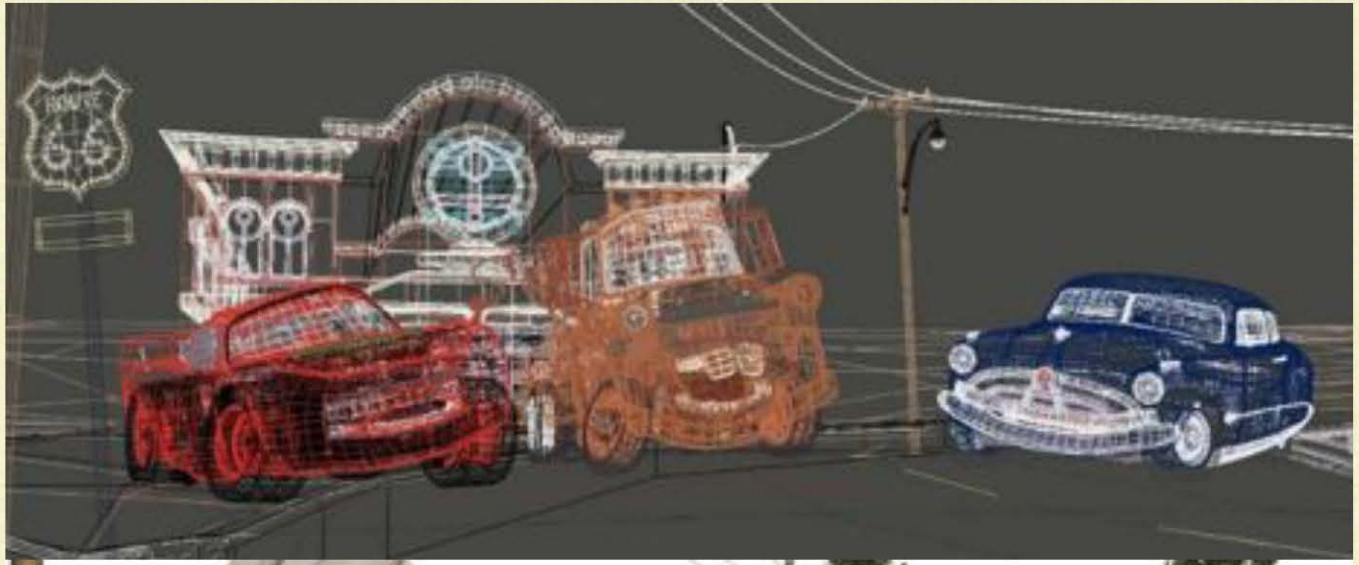


Example: Indoors

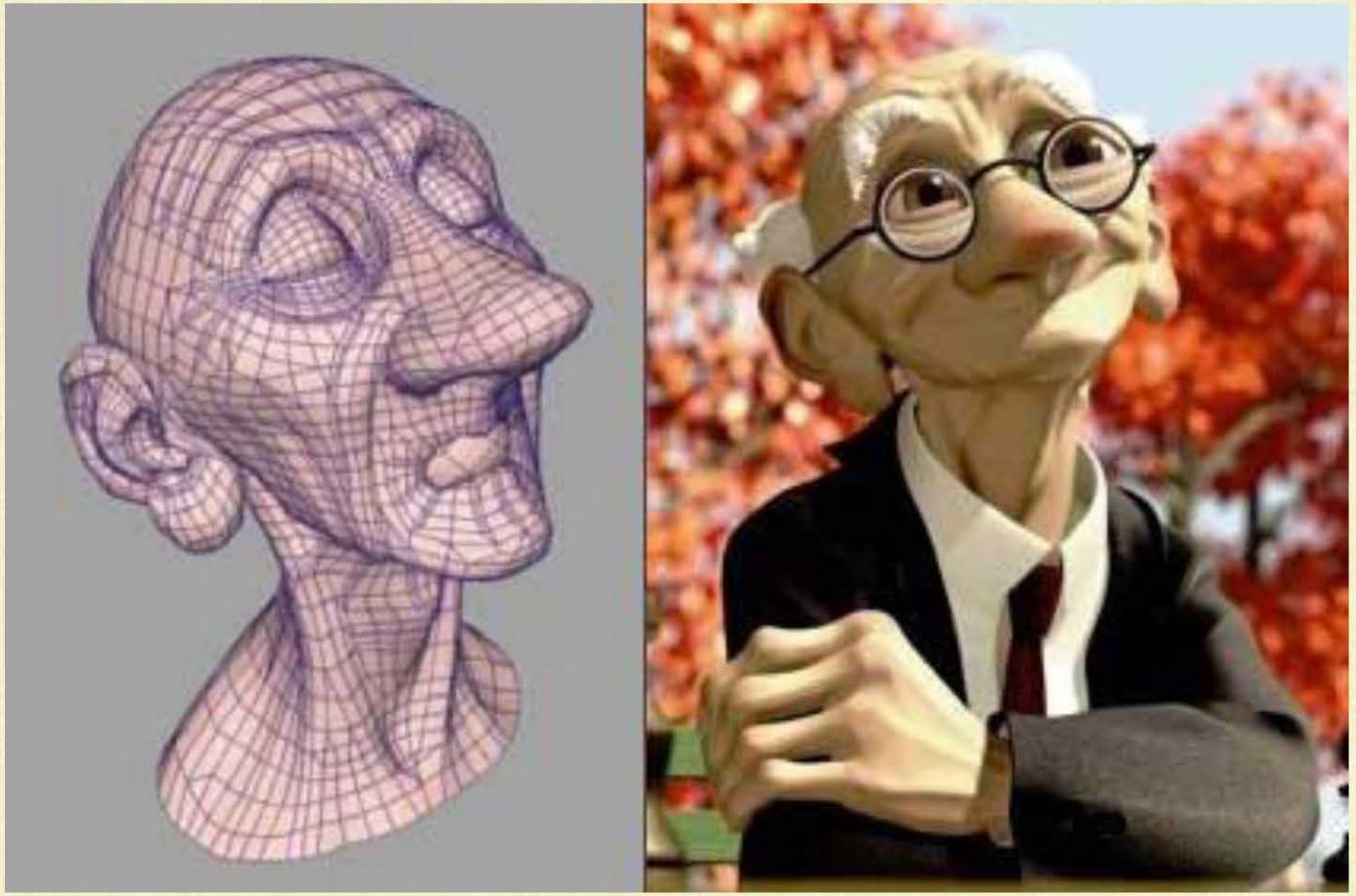
Example: Outdoors

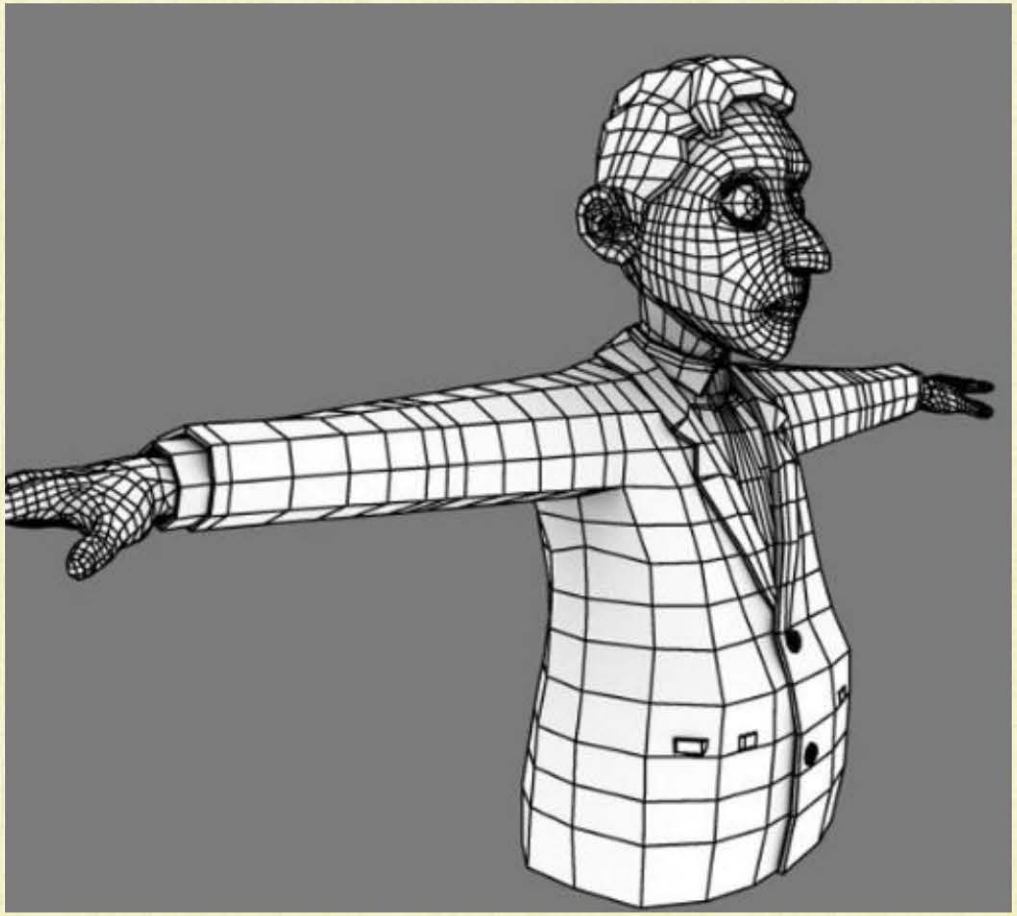


Example: Movies

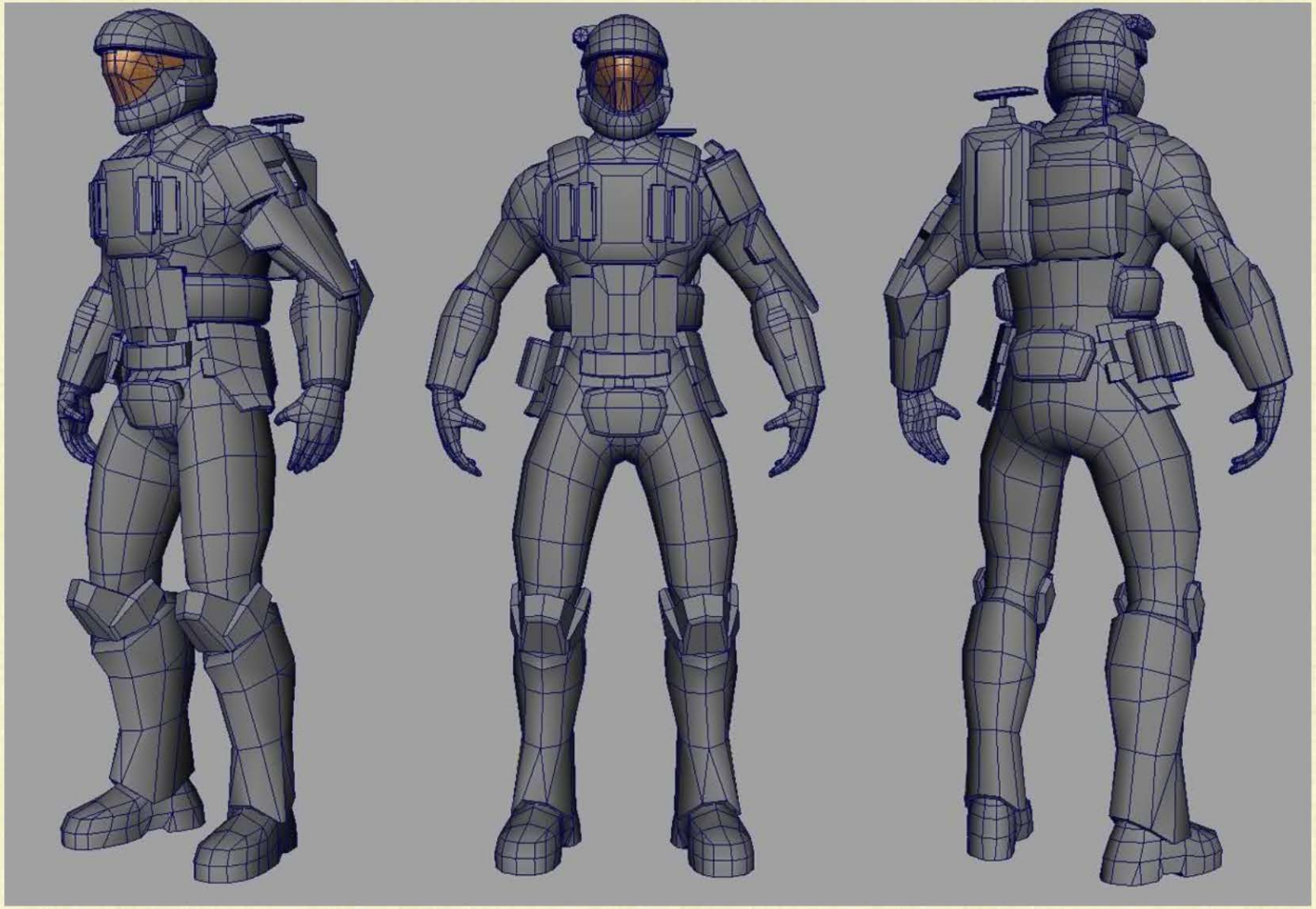


Example: Movies

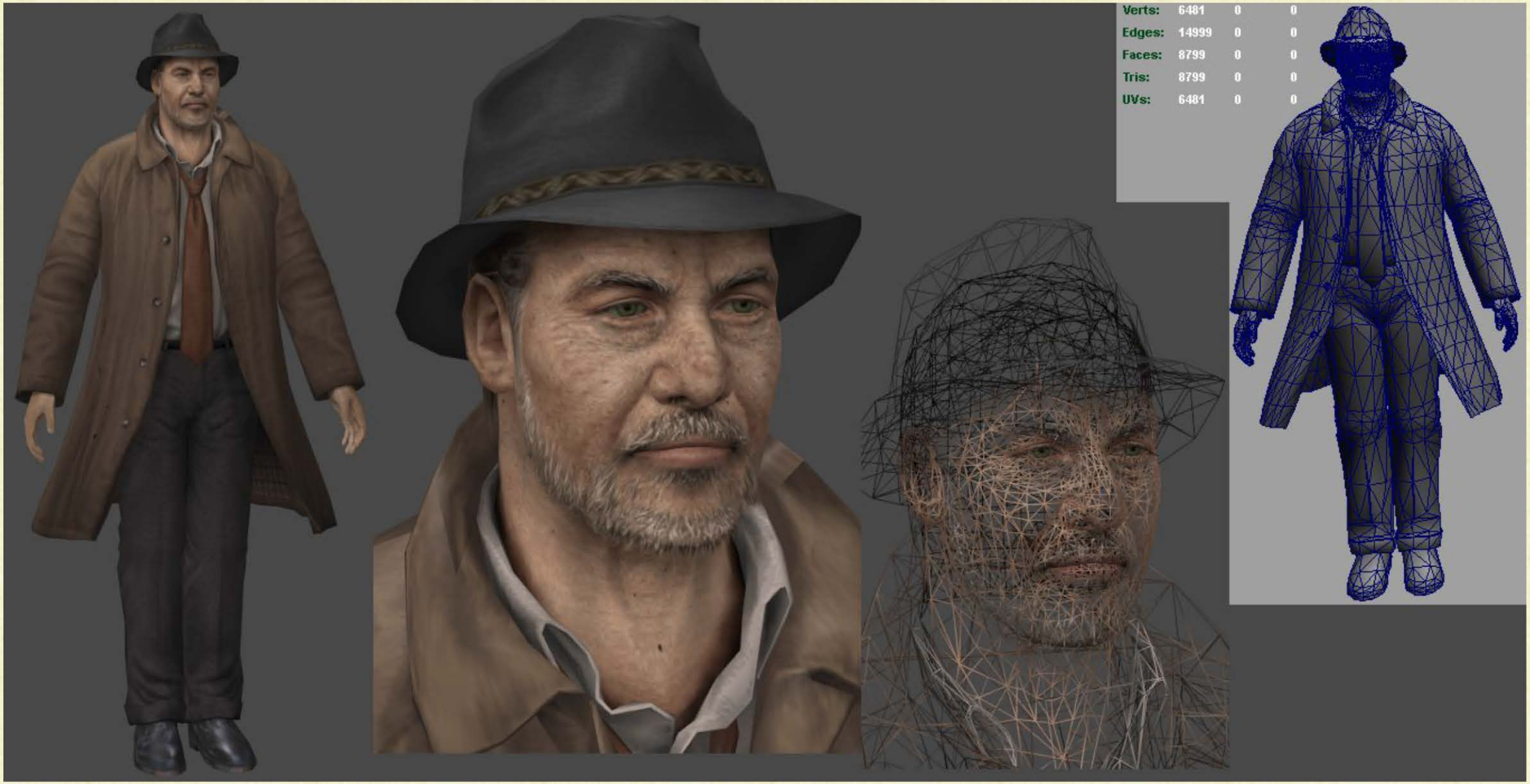




Example: Games

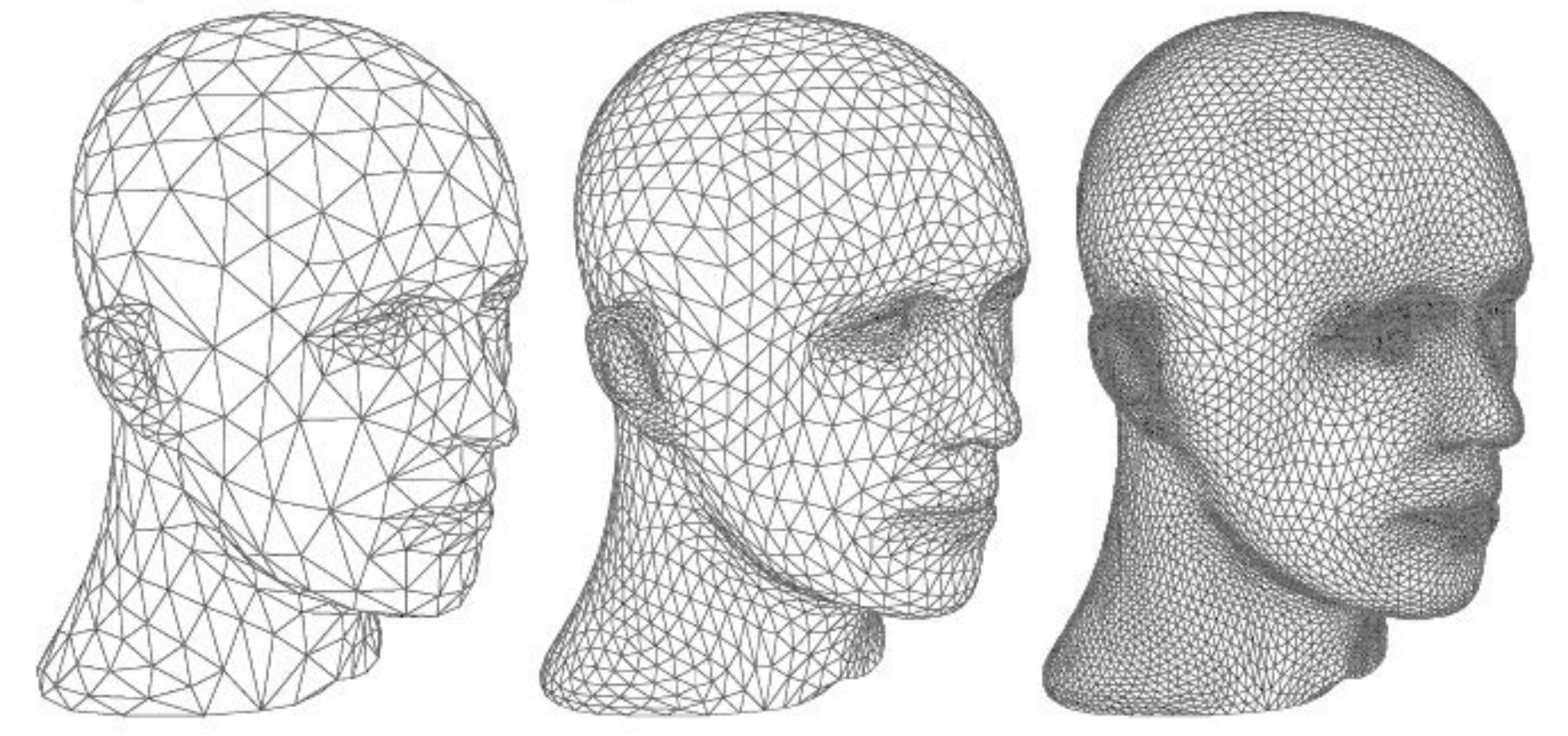


Example: Games



Subdivision

- Procedural algorithm: automatically generate a finer/smoother mesh from a coarser mesh
- Theoretically, a smooth limit surfaces can exist (depending on the algorithm used)
- After (only) a few refinements, additional changes often become too small to see/matter



Subdivision Curves

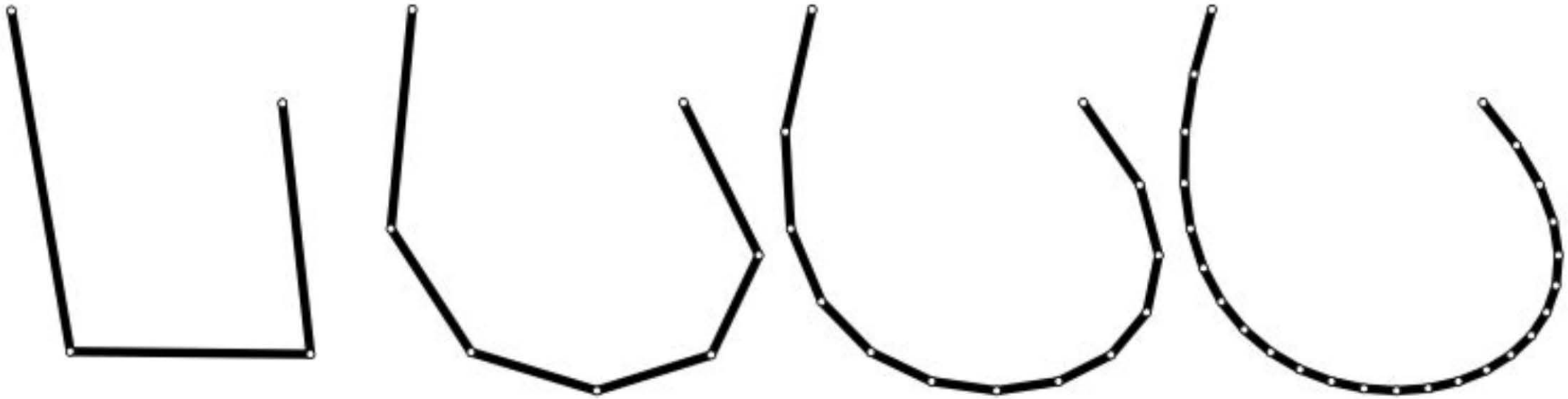
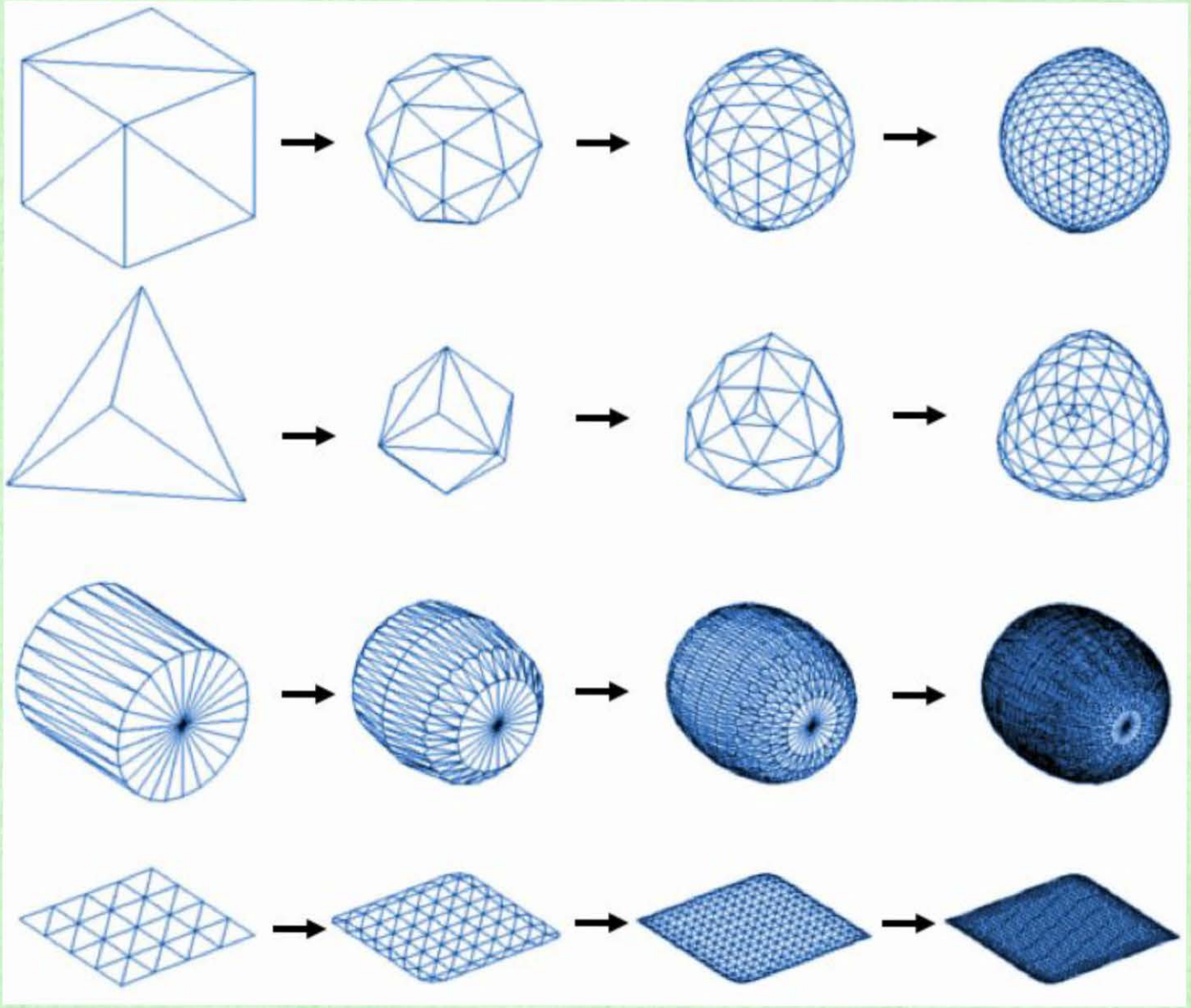


Figure 2.1: *Example of subdivision for curves in the plane. On the left 4 points connected with straight line segments. To the right of it a refined version: 3 new points have been inserted "inbetween" the old points and again a piecewise linear curve connecting them is drawn. After two more steps of subdivision the curve starts to become rather smooth.*

Subdivision Surfaces



Charles Loop

Smooth Subdivision Surfaces Based on Triangles

by

Charles Teorell Loop

A thesis submitted to the faculty of
the University of Utah
in partial fulfillment of the requirements for the degree of

Master of Science

Department of Mathematics

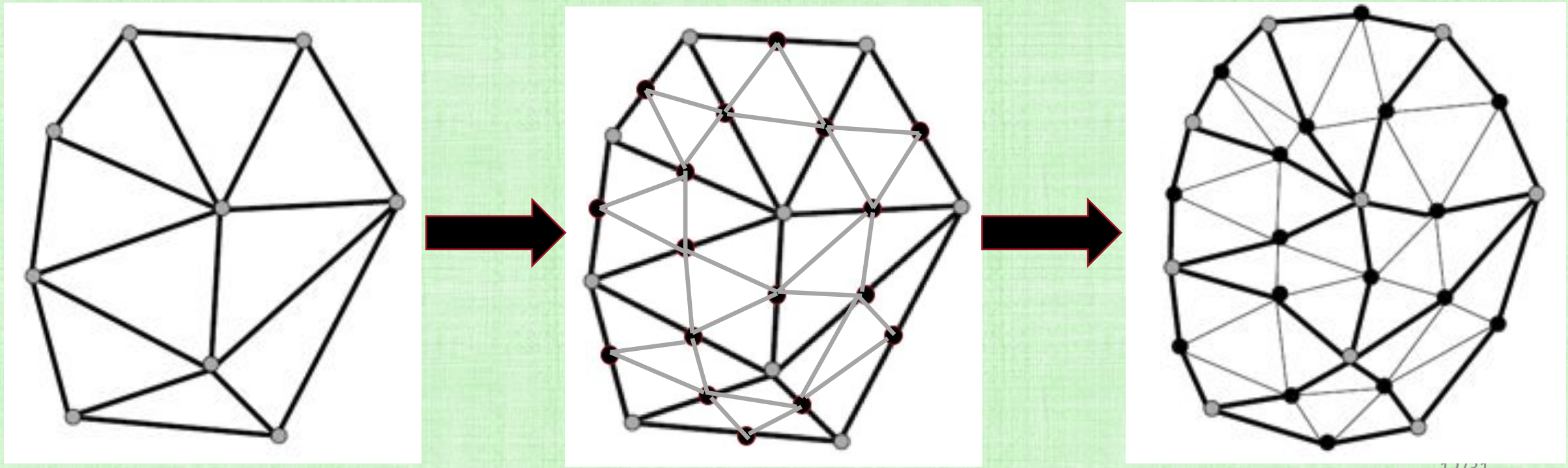
The University of Utah

August 1987

2203 citations (and counting) MS thesis!

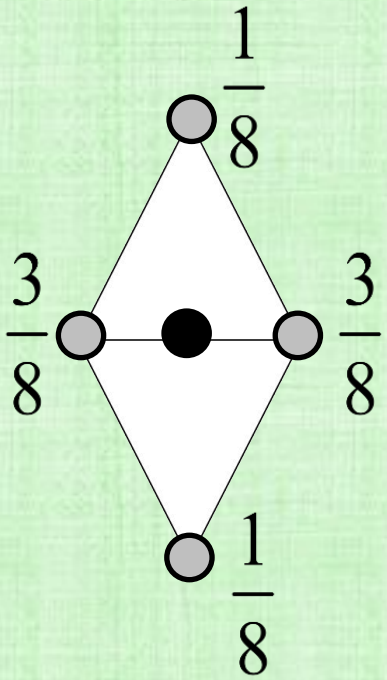
Loop Subdivision

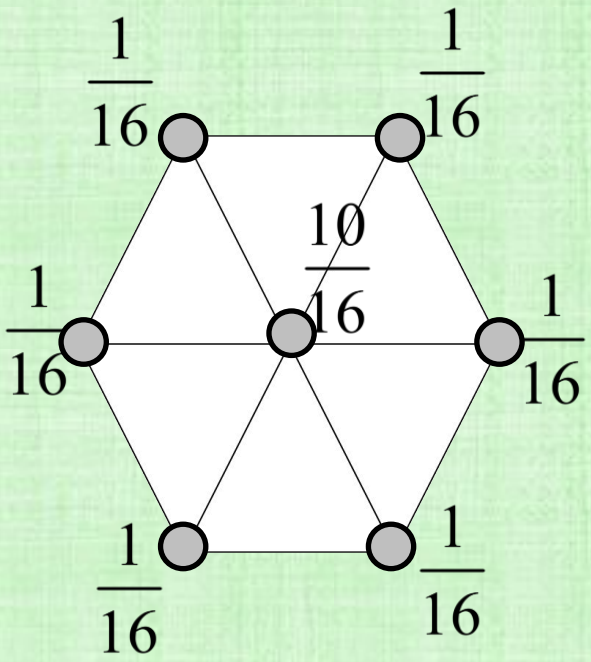
- Subdivide each triangle into 4 sub-triangles, and move both the old/new vertices
- Repeat (if desired)
- Generates a C^2 continuous limit surface almost everywhere (except at some extraordinary vertices where the limit is only C^1)



Move Old/New Vertices

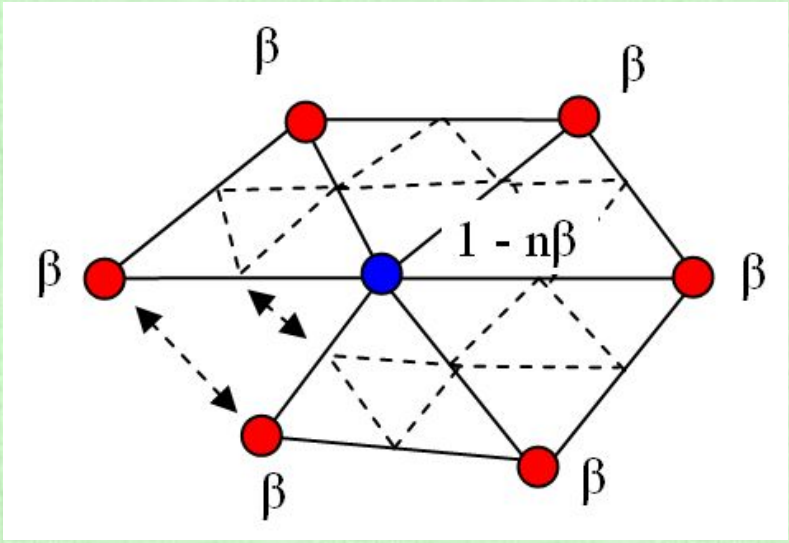
- Perturb the position of each new vertex (black) using a weighted average of the four nearby original vertices (grey)
- Perturb the position of each regular original vertex (grey) using a weighted average of the six adjacent original vertices (grey)
- Repeat (if desired)

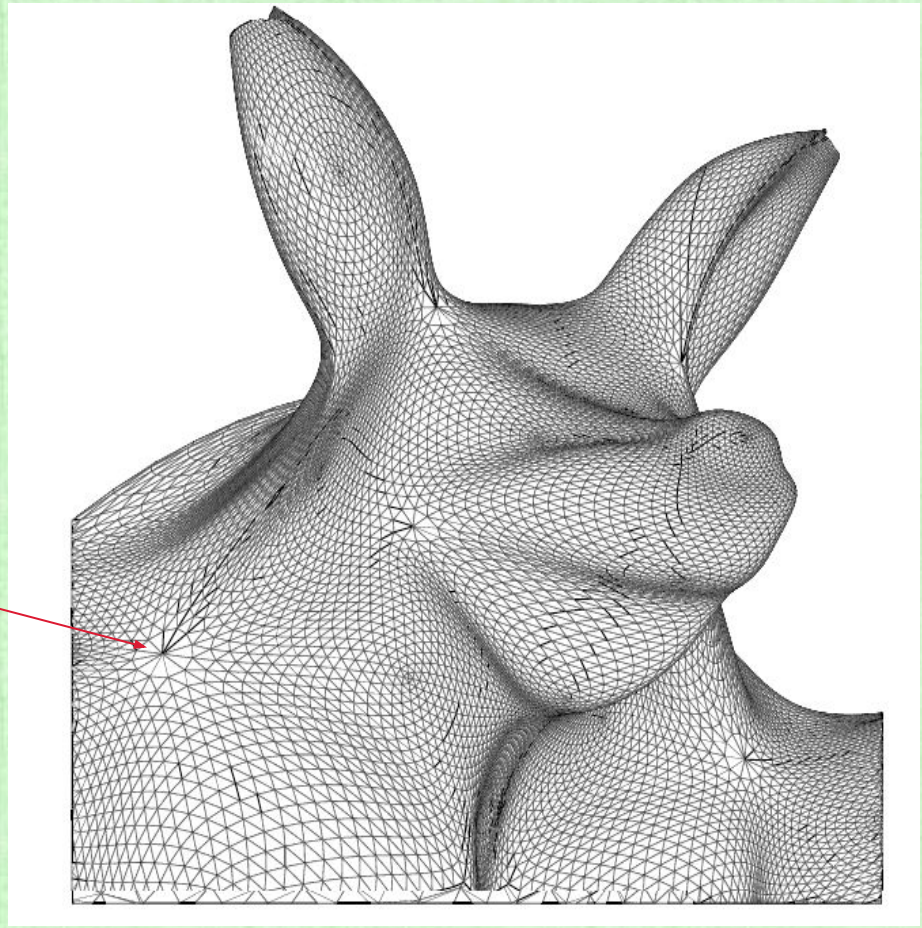




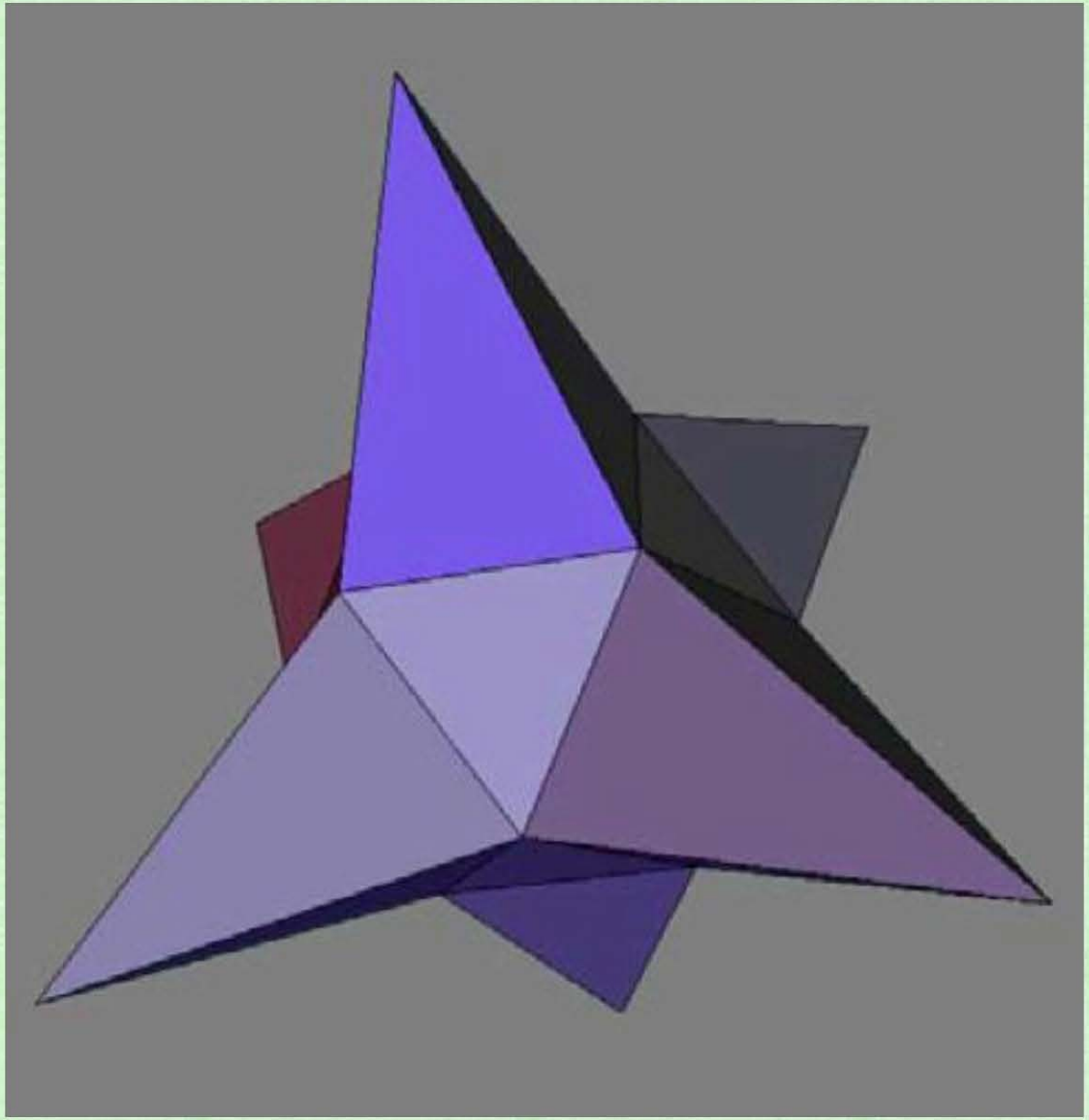
Extraordinary Points

- Most vertices are regular (degree 6)
- If a mesh is topologically equivalent to a sphere, not all vertices can have degree 6
- At extraordinary points, Warren weights can be used to generate a smooth surface (i.e., the tangent plane is continuous)
  - When the number of points is $n = 3$, $\beta = \frac{3}{16}$; else $\beta = \frac{3}{8n}$

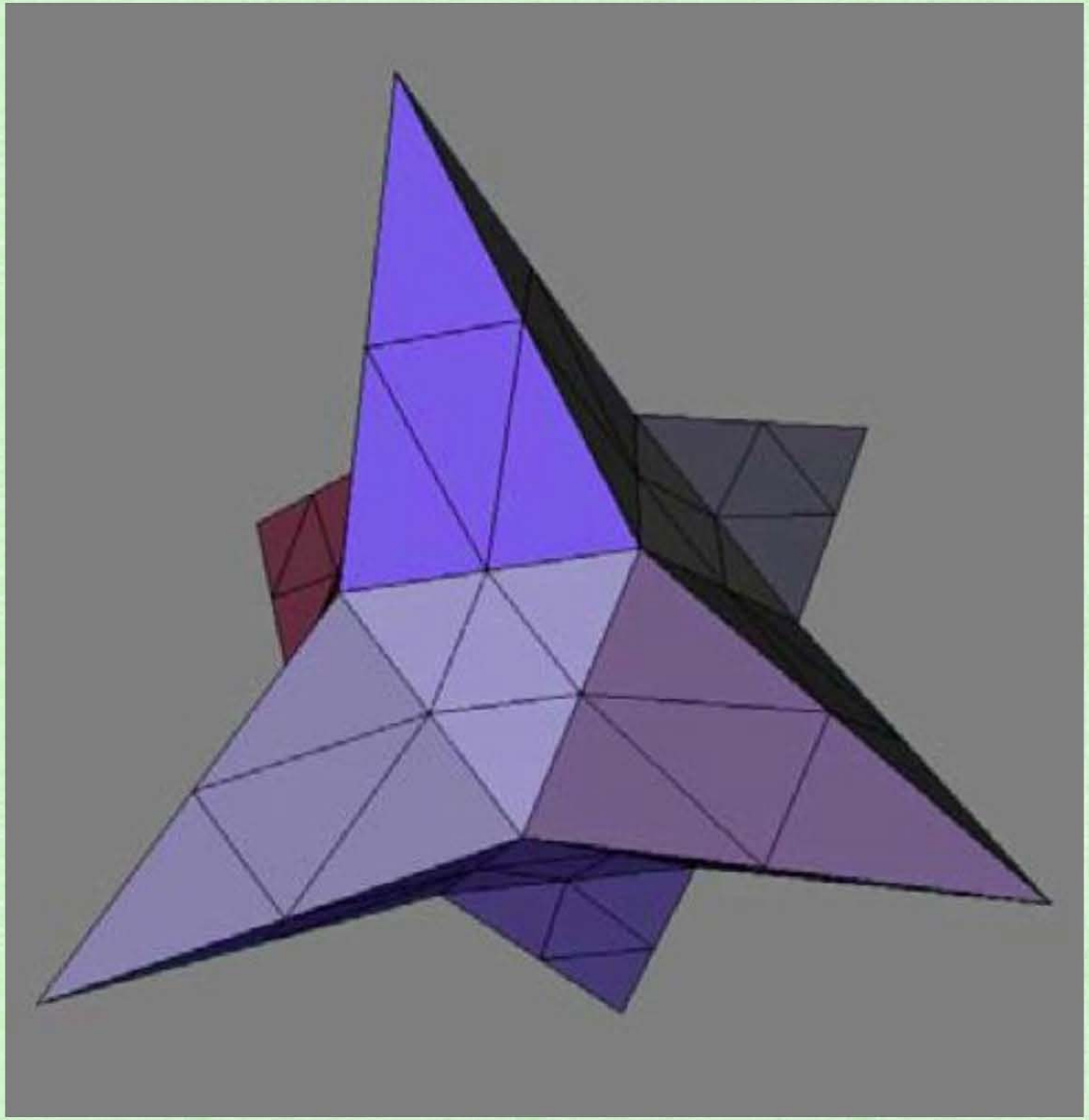




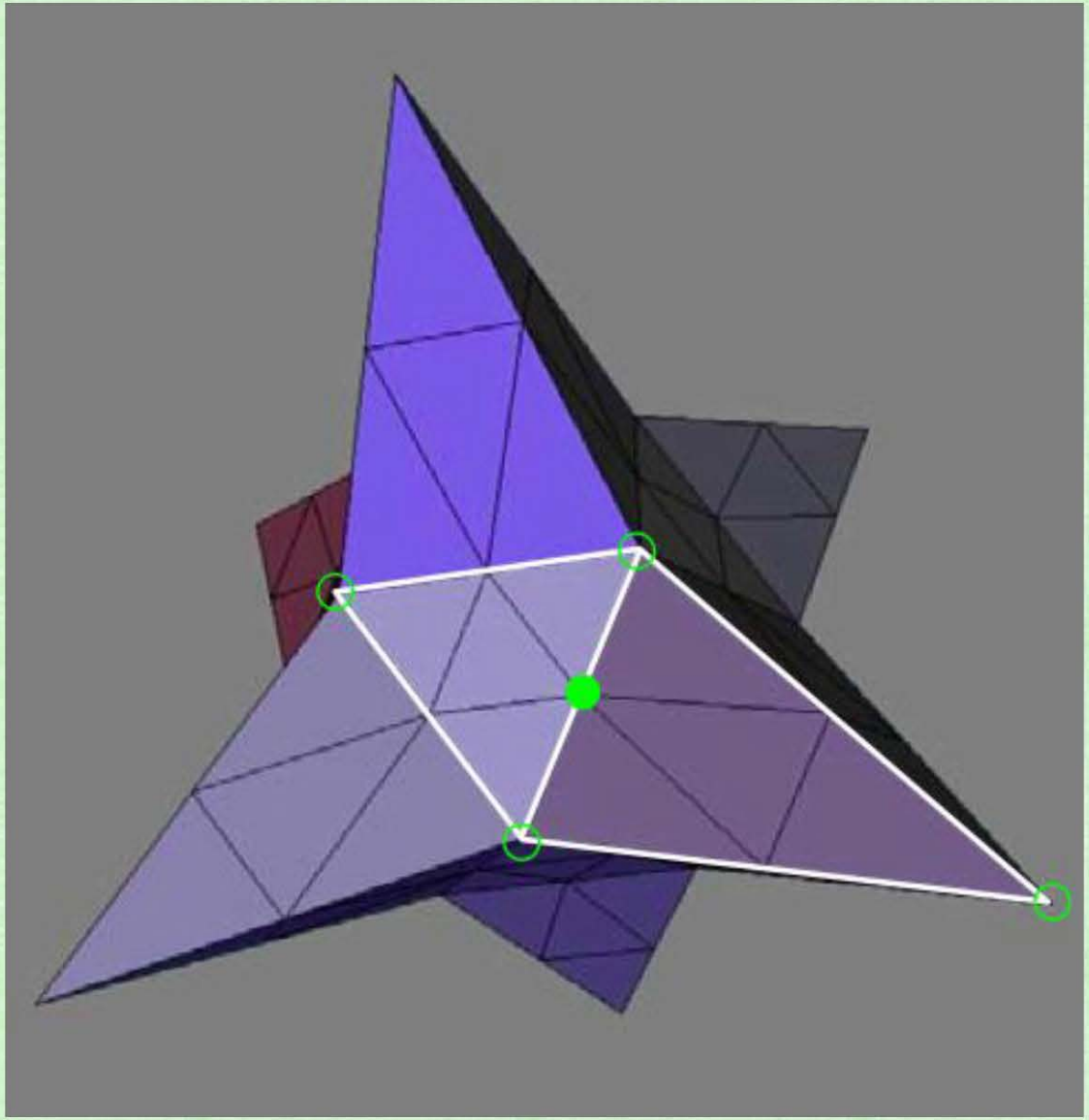
Initial Mesh



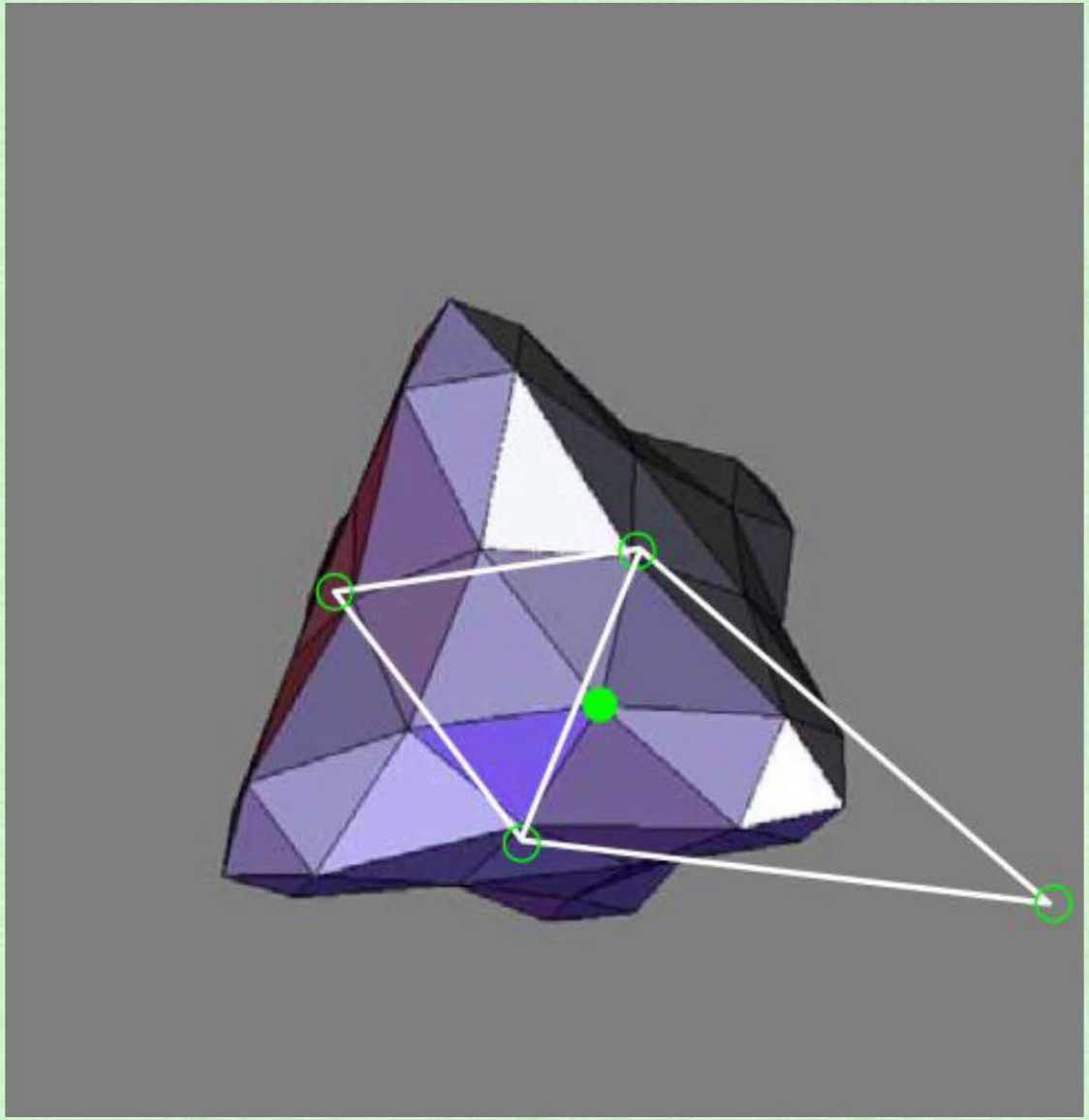
Add New Vertices



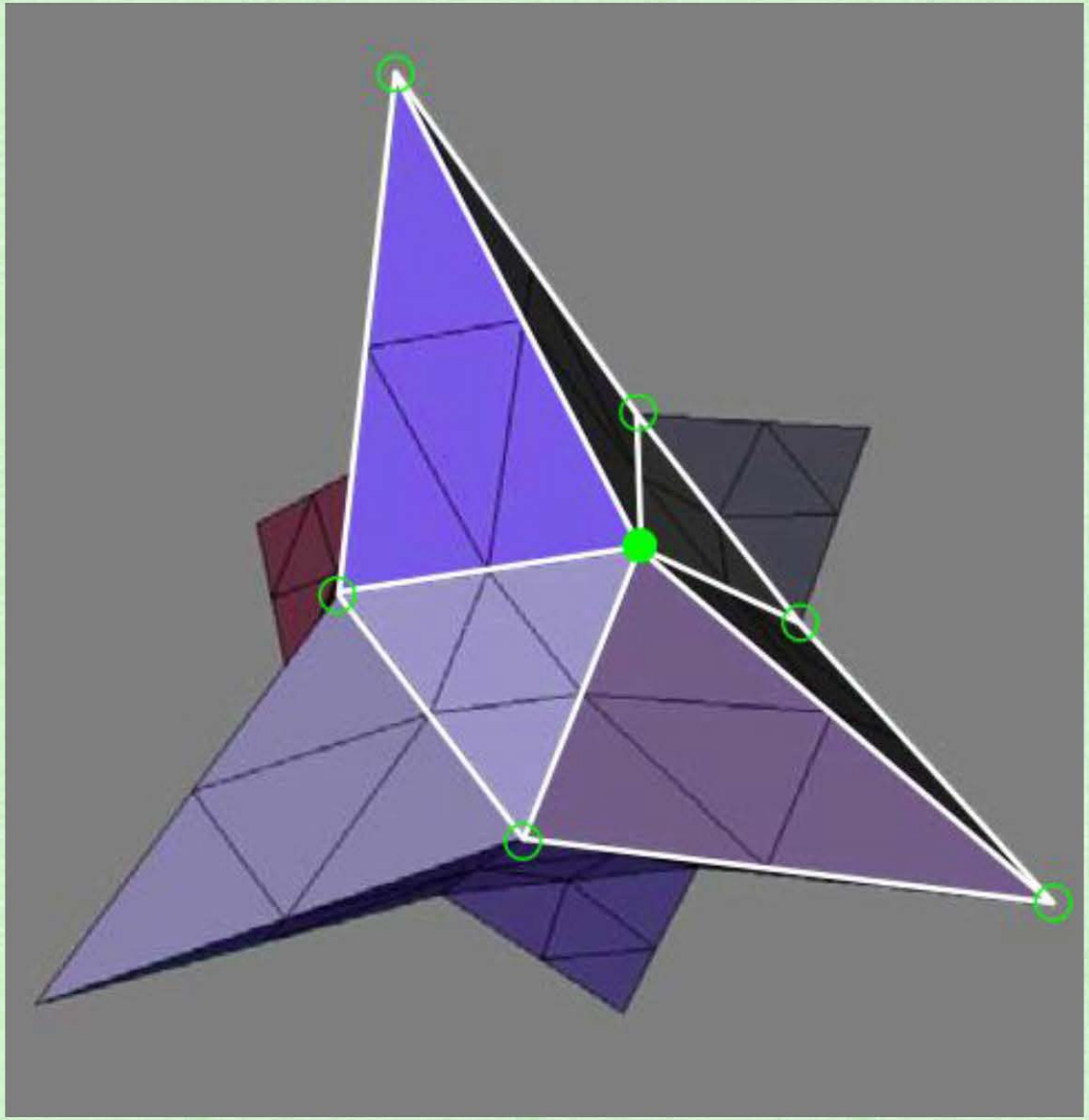
New Vertex and Stencil



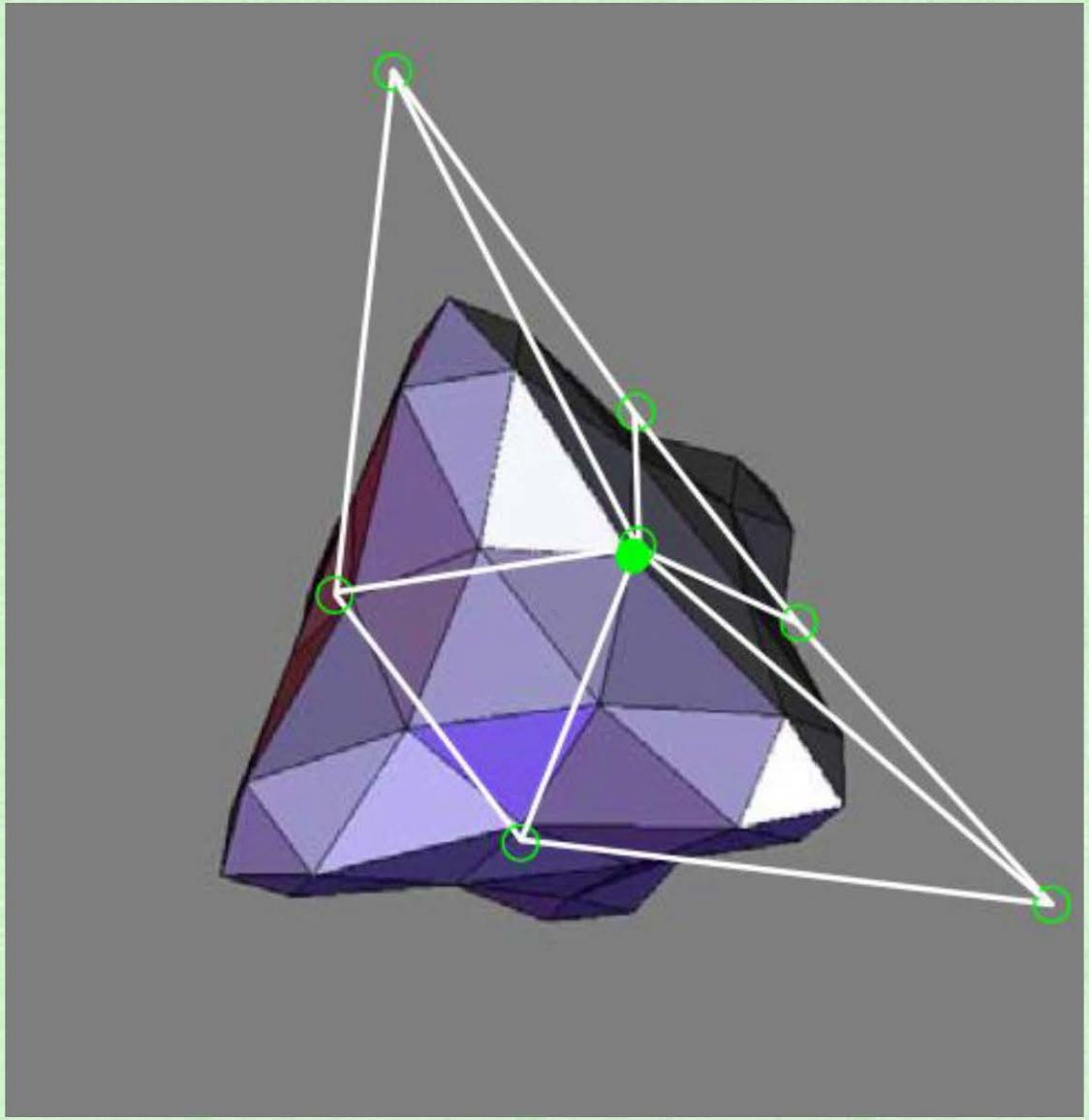
Move New Vertex



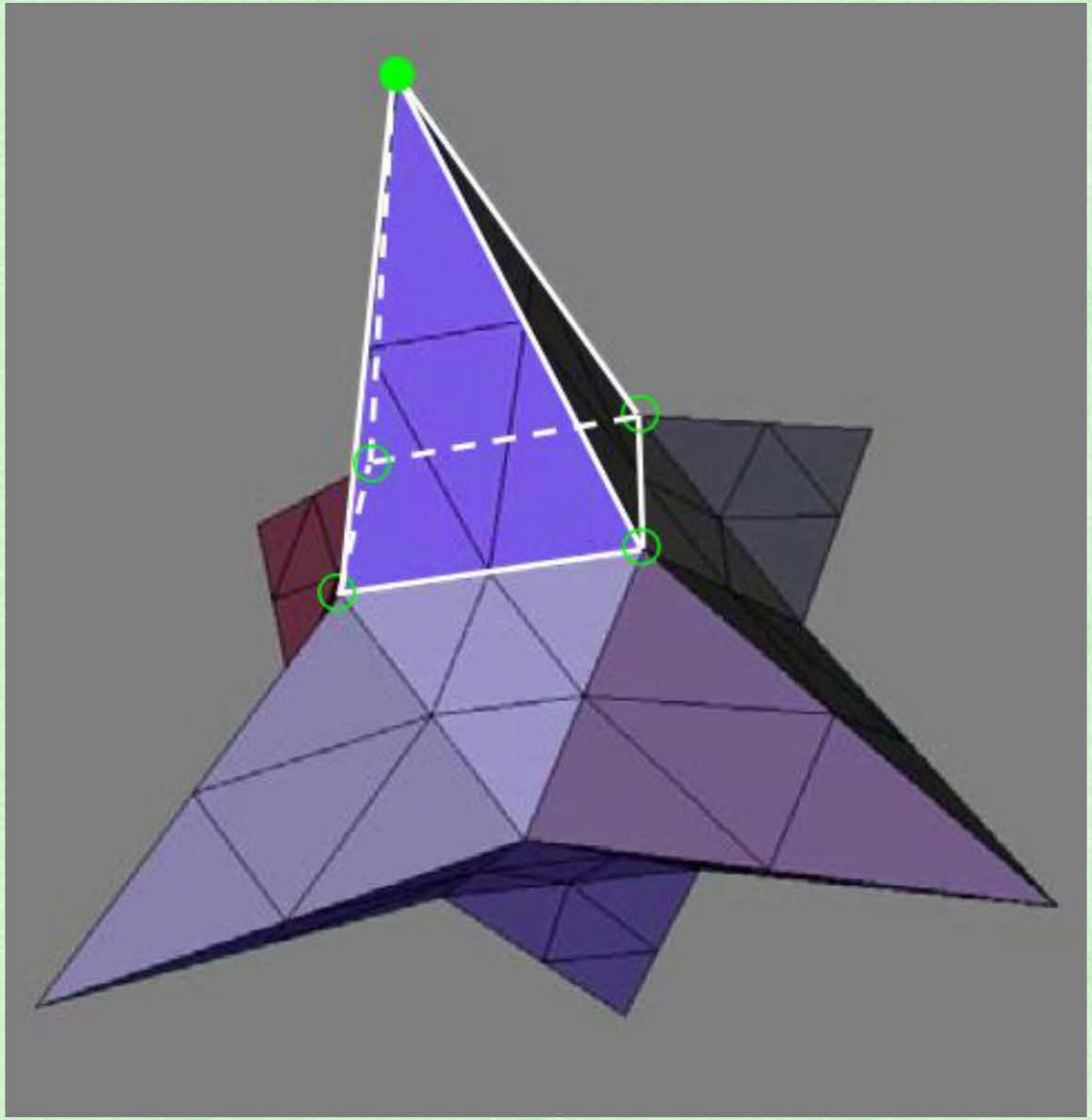
Original (Regular) Vertex and Stencil



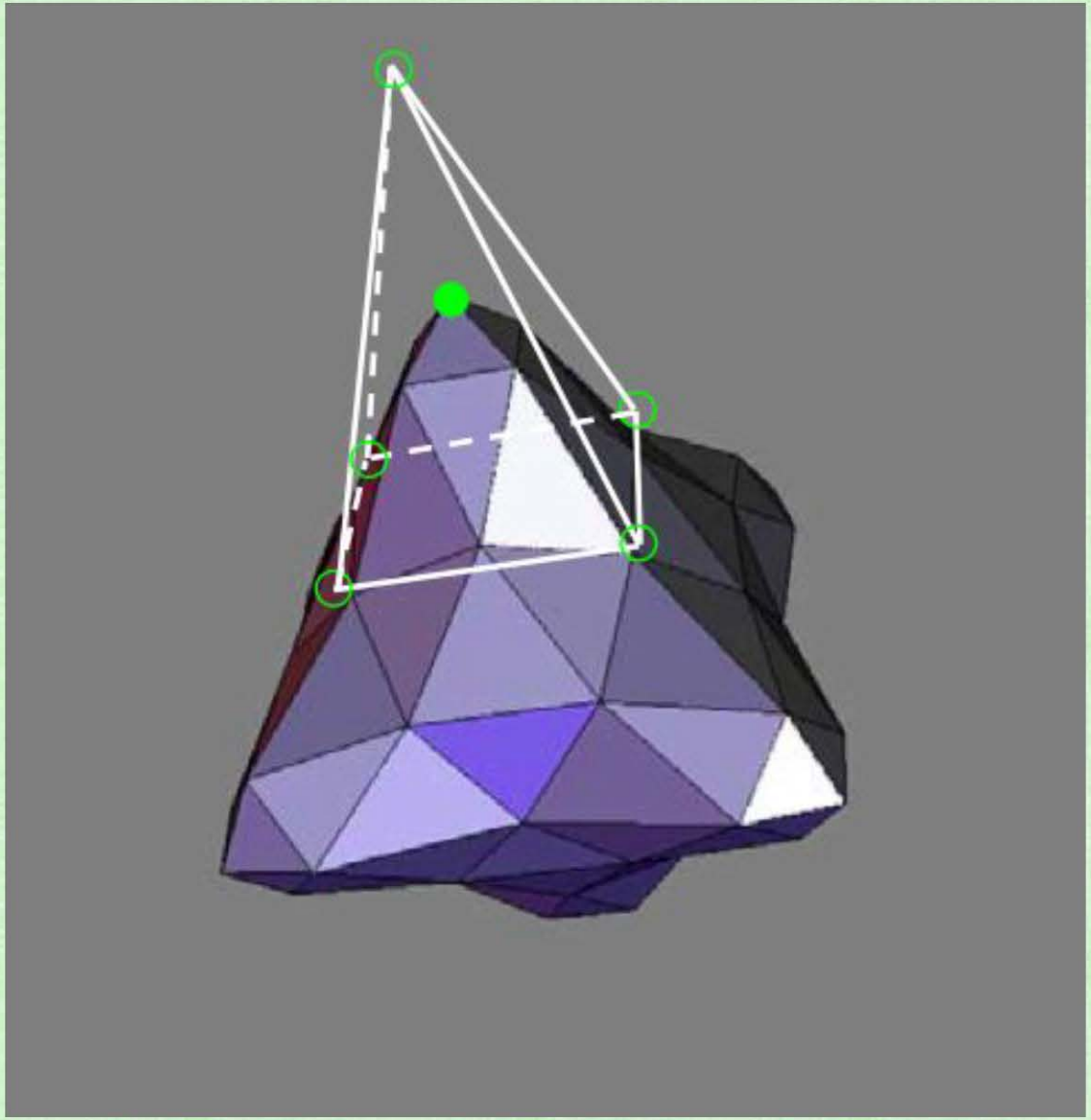
Move Original (Regular) Vertex



Original (Extraordinary) Vertex and Stencil

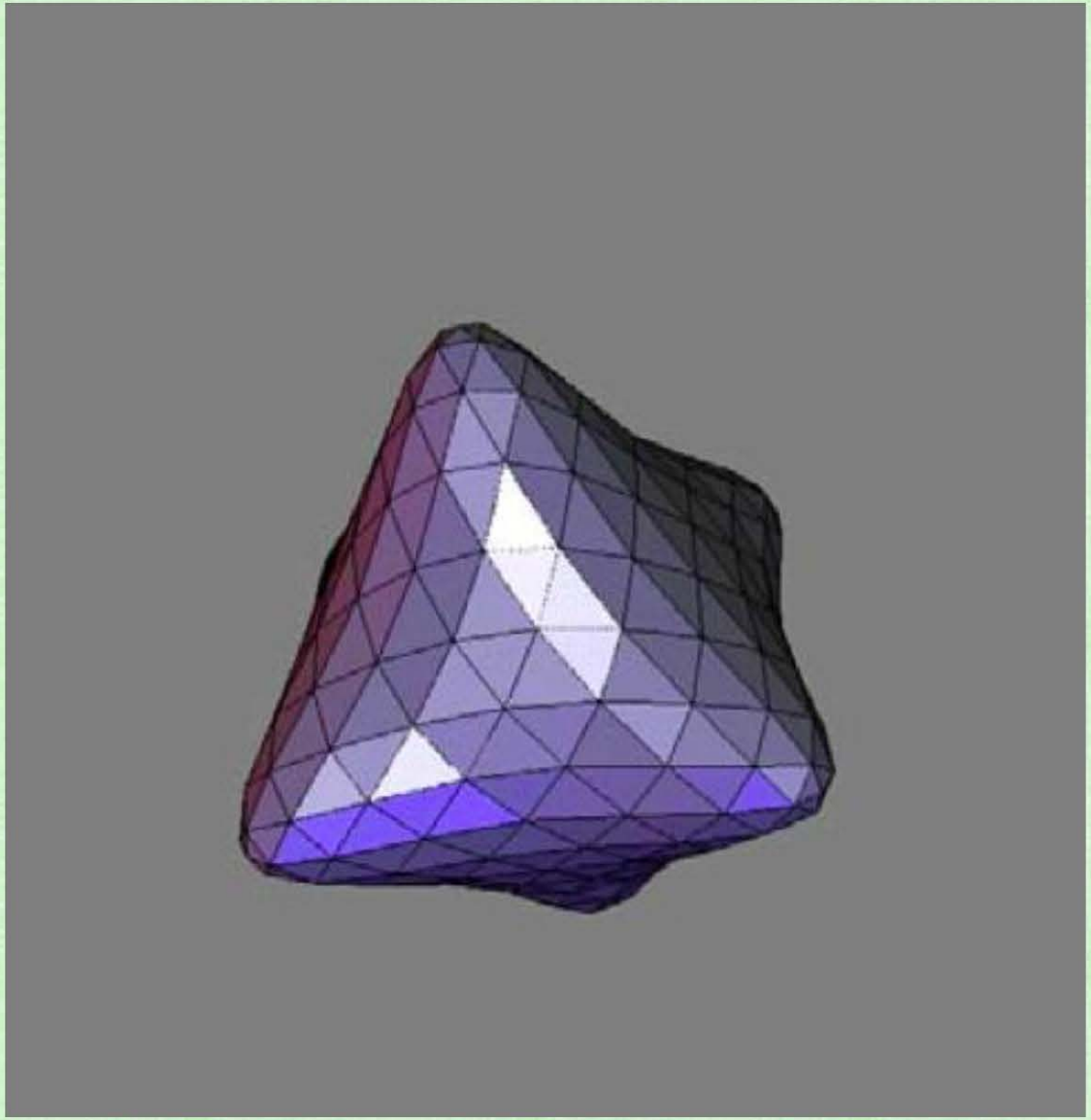


Move Original (Extraordinary) Vertex

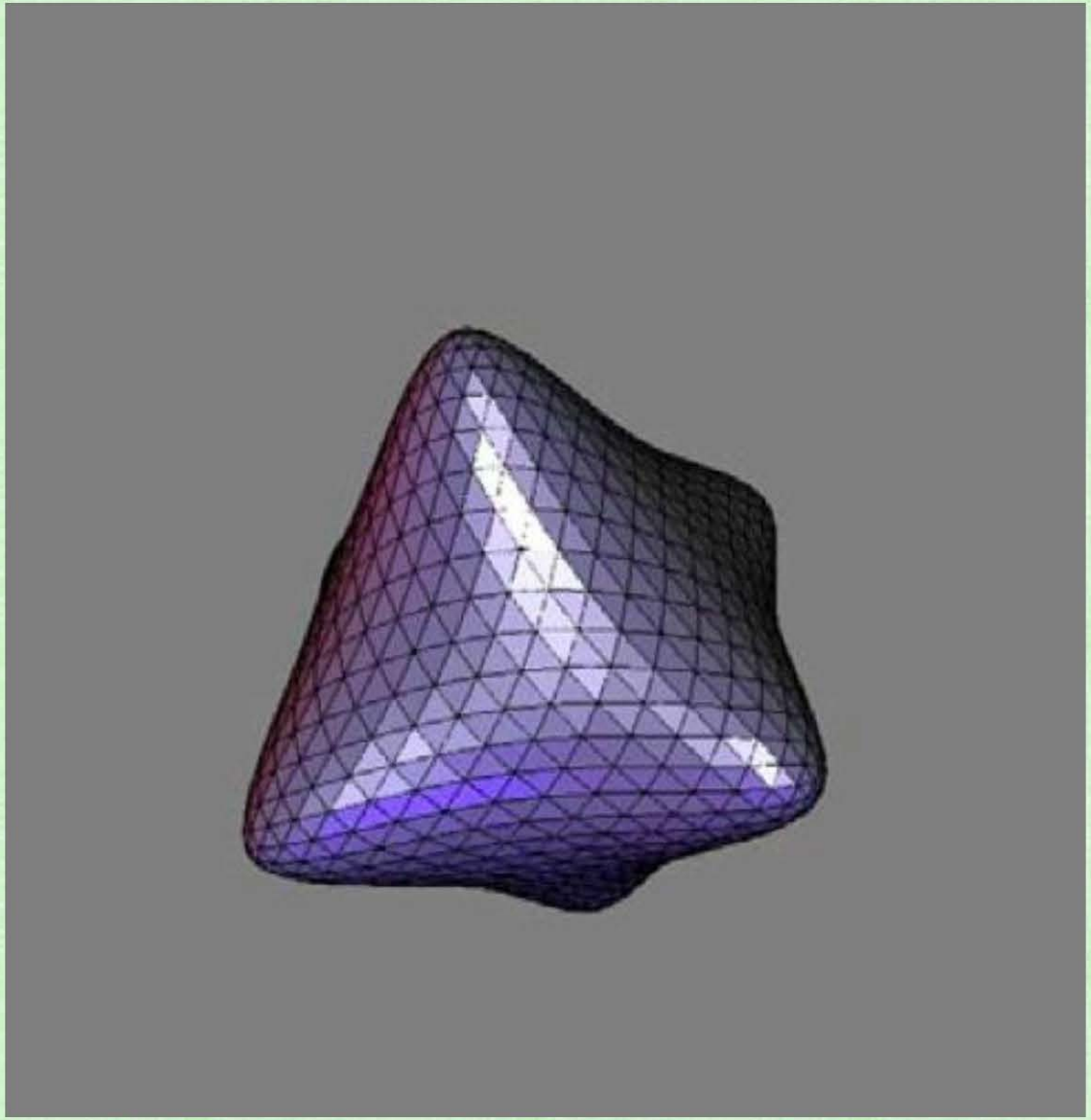


Subdivided Surface

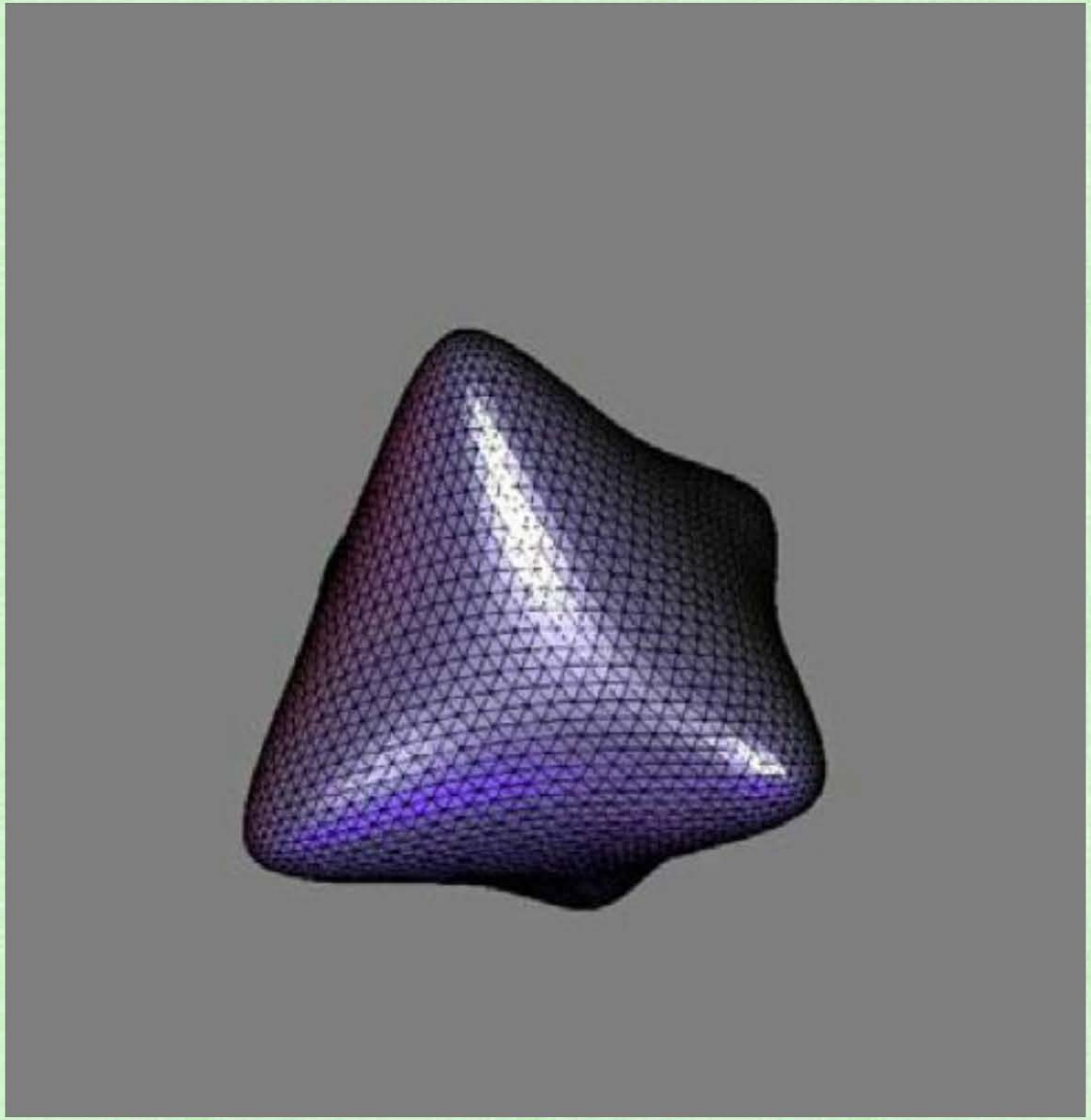
Subdivide Again



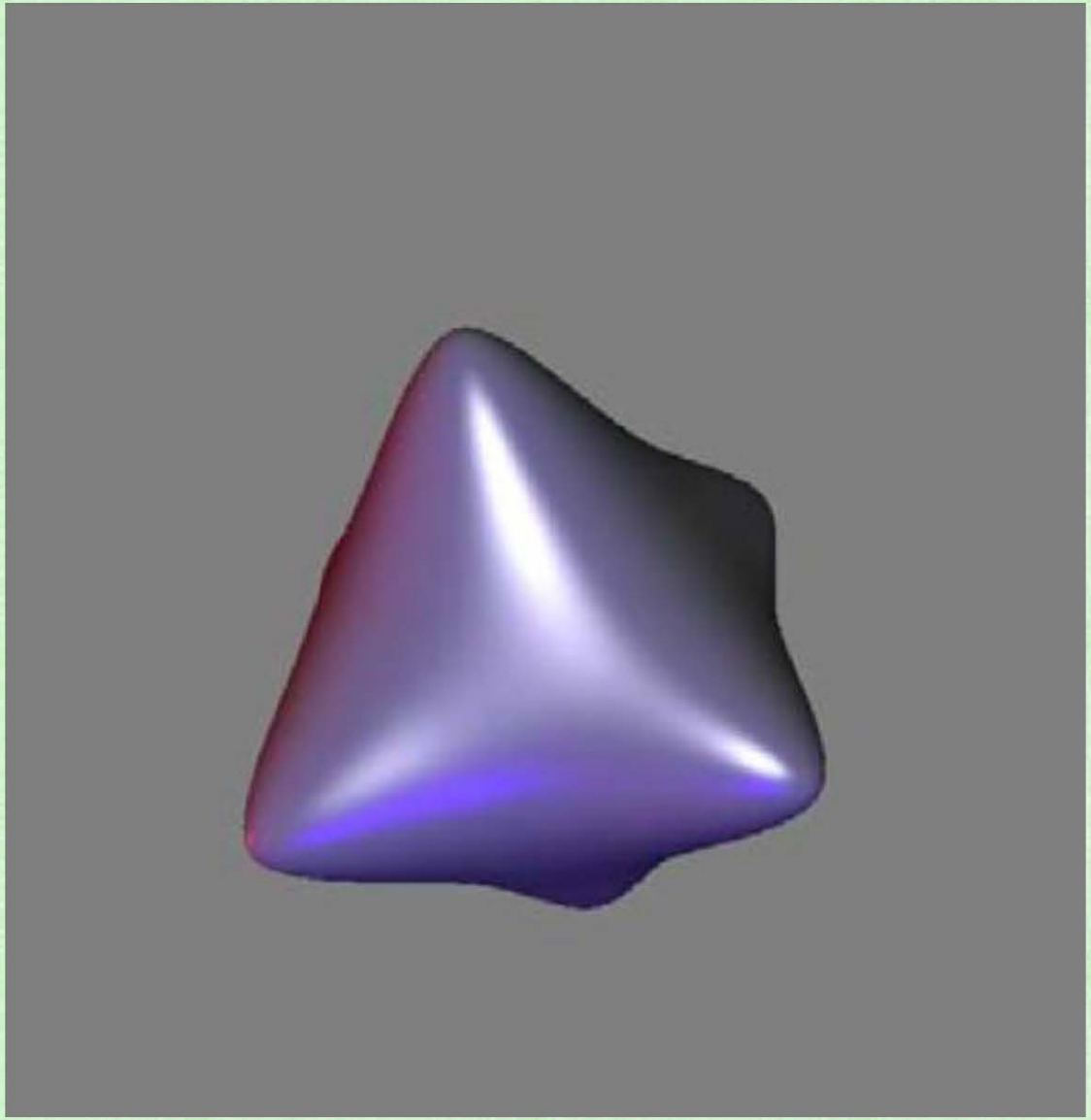
And Again



And Again



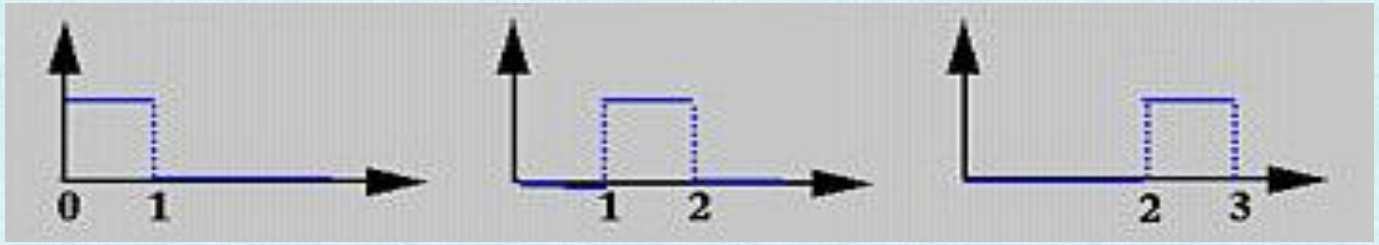
Final (Smooth) Limit Surface



B-Spline Basis Functions

- Knots: node locations $\{u_0, u_1, \ldots u_m\}$ in parameter space
- Lowest level building blocks are piecewise constant (m of them):

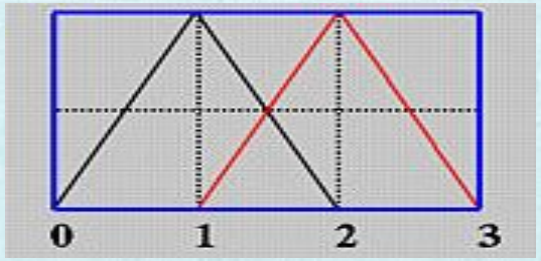
$$N_{i,0}(u) = \begin{cases} 1 & if\ u_i \le u < u_{i+1} \\ 0 & otherwise \end{cases}$$



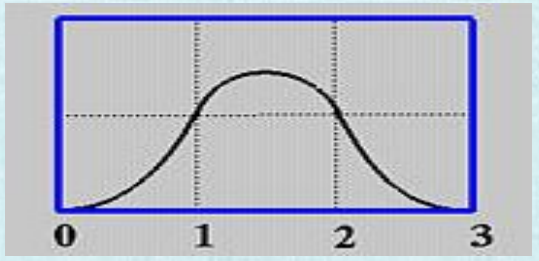
0-th order: $N_{0,0}$ $N_{1,0}$ $N_{2,0}$

- Higher order building blocks are defined recursively (i-th basis function of order j is…):

$$N_{i,j}(u) = \frac{u - u_i}{u_{i+j} - u_i} N_{i,j-1}(u) + \frac{u_{i+j+1} - u}{u_{i+j+1} - u_{i+1}} N_{i+1,j-1}(u)$$



1st order: $N_{0,1}$ $N_{1,1}$

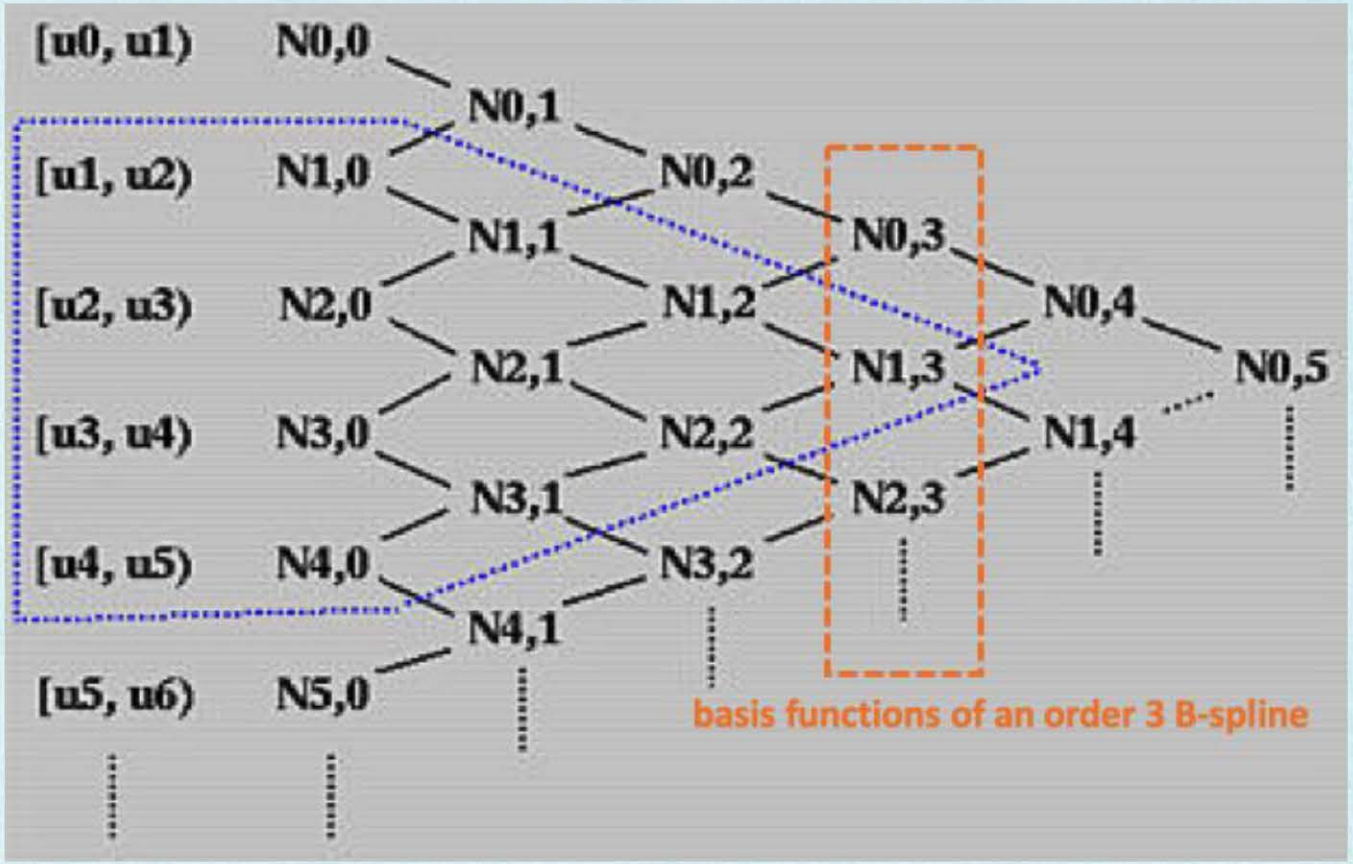


2nd order: $N_{0,2}$

B-Spline Basis Functions

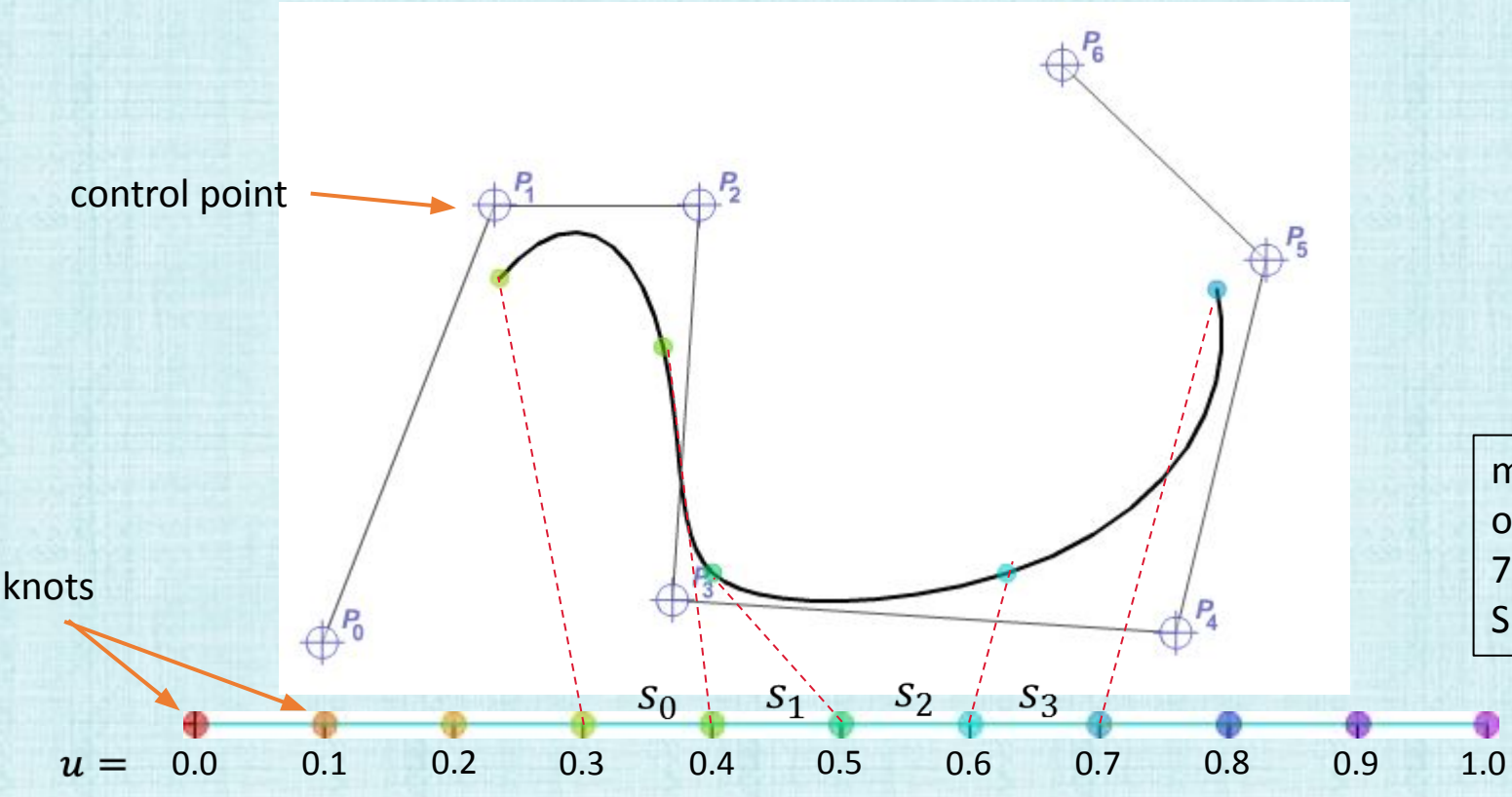
- The basis functions for a B-spline of order p are $N_{i,p}(u)$
- Let $\boldsymbol{P}_0, \boldsymbol{P}_1, \ldots, \boldsymbol{P}_n$ be control points (in 2D or 3D)
- The B-spline curve is defined as: $C(u) = \sum_{i=0}^{n} N_{i,p}(u)\boldsymbol{P}_i$

- Given *m* intervals, a B-spline of order *p* requires *m-p* control points $\boldsymbol{P}_i$
- For example, the *m=6* intervals (to the right) with order *p=3* requires *m-p=3* (orange box) control points $\boldsymbol{P}_i$
- The summation (above) would go from 0 to 2 to include 3 control points



B-Spline

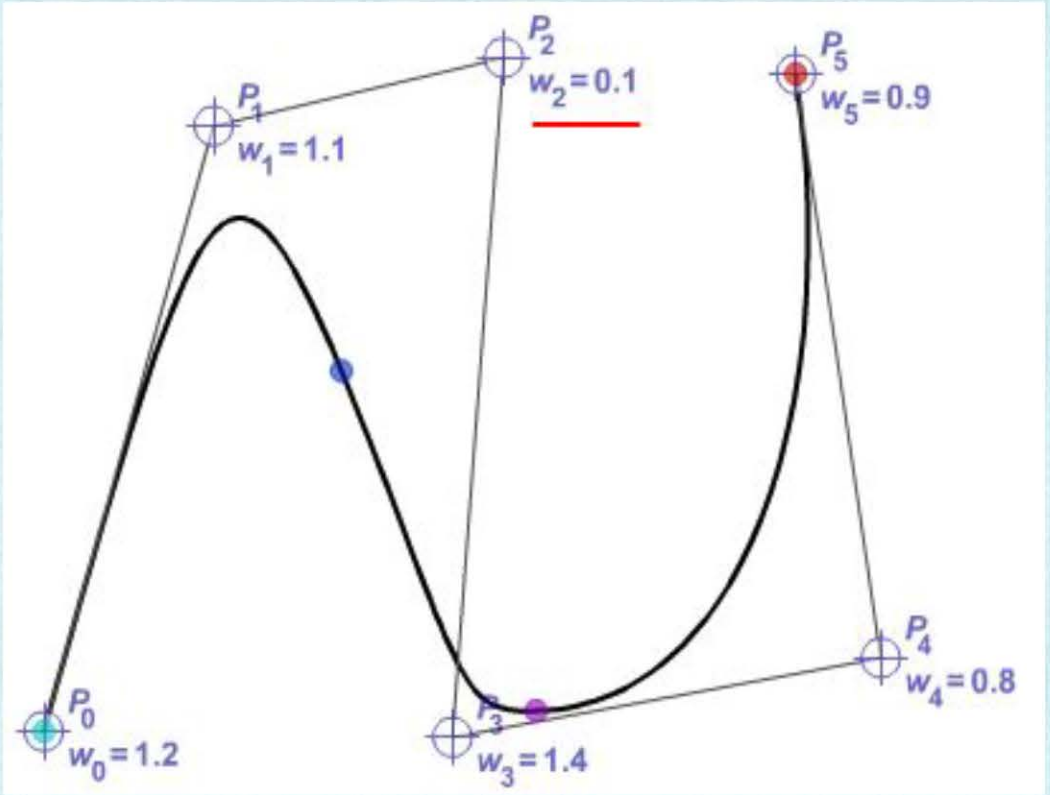
- Ignoring boundaries (where there are only 1, 2, 3 cubic basis functions $N_{i,3}$), 4 control points define the shape of each curve segment
  - e.g., s_0 = (.3,.4) is controlled P_0, P_1, P_2, P_3
  - [u3,u4) is the first interval to have a full representation in the orange box (of prior slide): $N_{0,3}, N_{1,3}, N_{2,3}, N_{3,3}$
  - 3 parametric domain intervals are lost off of each side (of the domain)

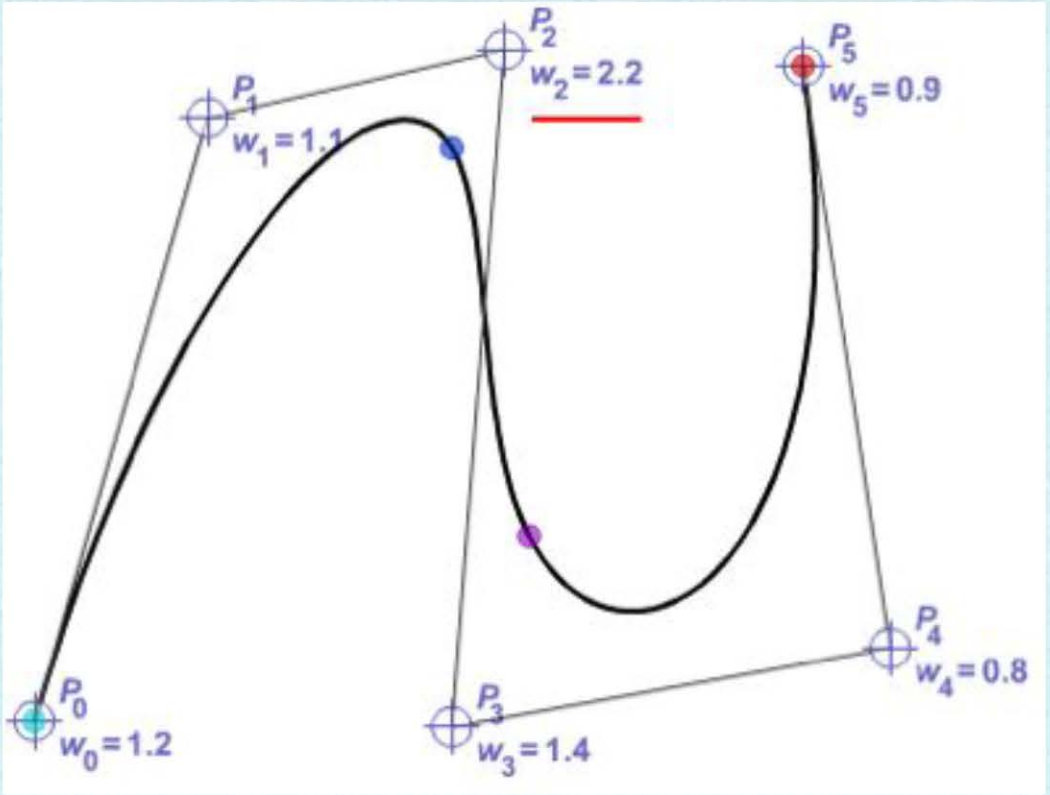


m=10 intervals
order p=3
7 control points
Sum: i = 0 to 6 (n=6)

NURBS Curve

- Non-Uniform Rational B-Spline (NURBS): $C(u) = \frac{\sum_{i=0}^{n} N_{i,p}(u) w_i \boldsymbol{P}_i}{\sum_{i=0}^{n} N_{i,p}(u) w_i}$
- Increasing the weight w_i on point $\boldsymbol{P}_i$ pulls the curve closer to $\boldsymbol{P}_i$, while decreasing w_i releases the curve to move farther away from $\boldsymbol{P}_i$

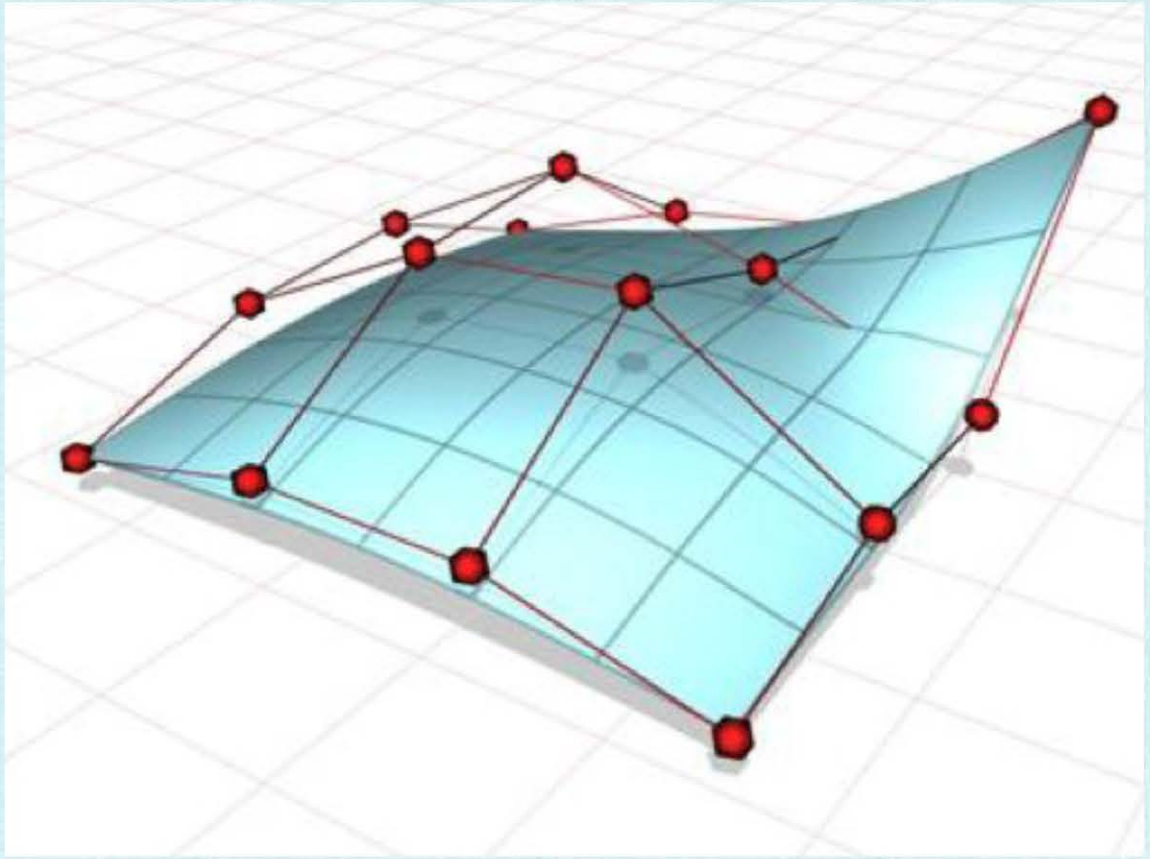




NURBS Surface

- Extending the ideas from curves to surfaces, a NURBS surface is defined as:

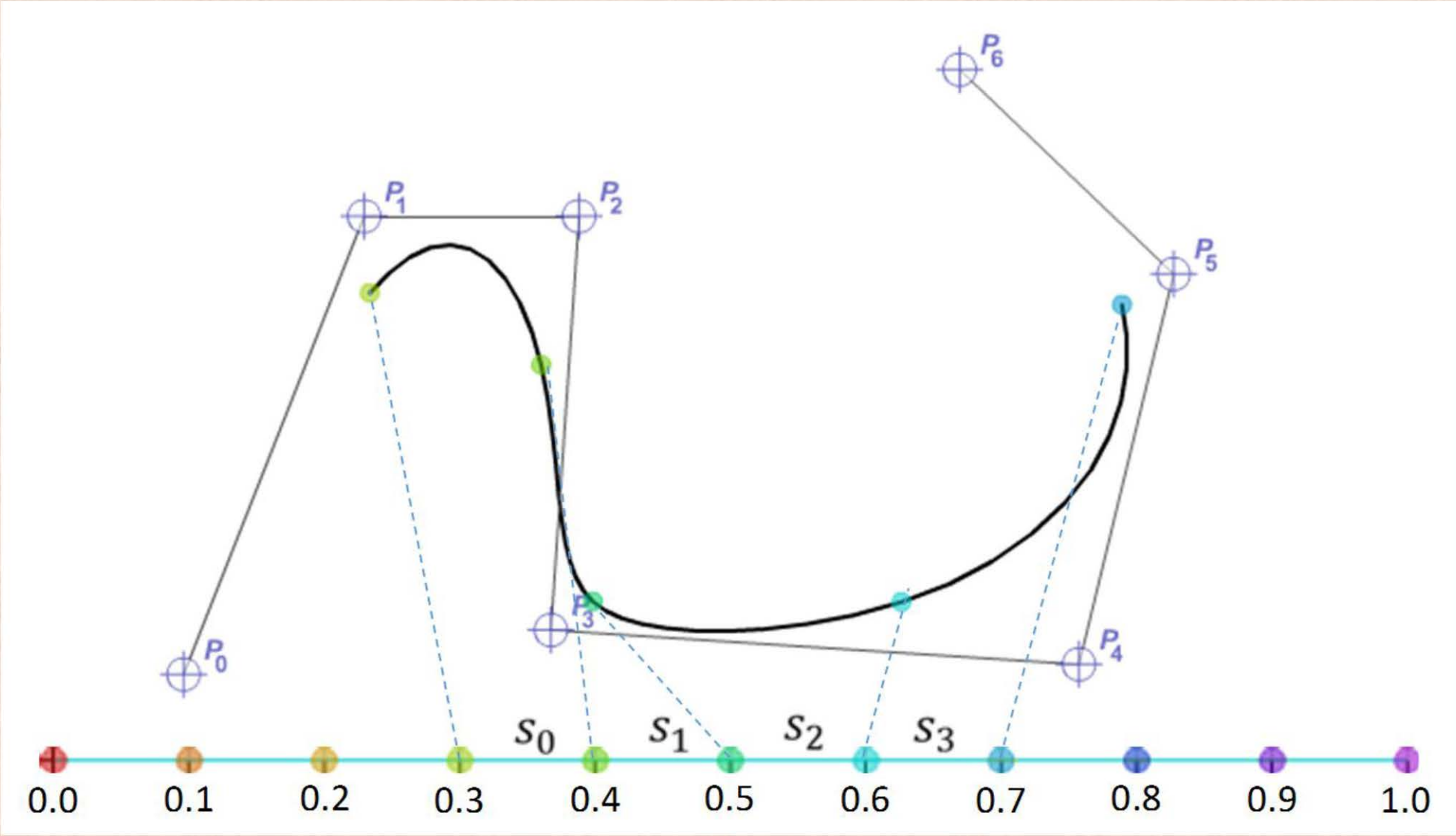
$$S(u,v) = \frac{\sum_{i=0}^{m}\sum_{j=0}^{n} N_{i,p}(u)N_{j,q}(v)w_{i,j}\boldsymbol{P}_{i,j}}{\sum_{i=0}^{m}\sum_{j=0}^{n} N_{i,p}(u)N_{j,q}(v)w_{i,j}}$$



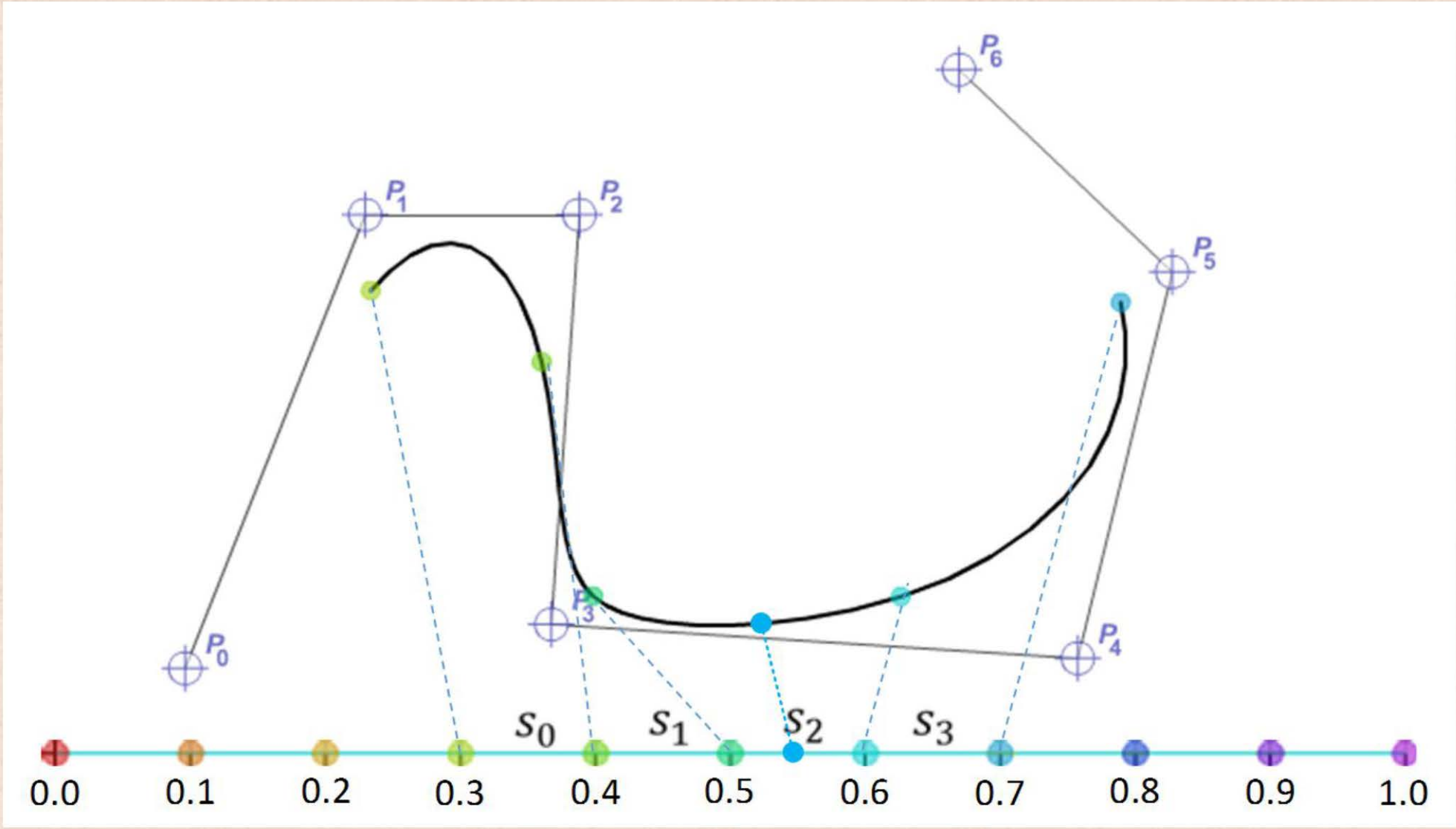
Subdividing Spline Curves

- First, insert new knots (and control points) without changing the shape of the curve
- Then, can move around control points to modify the shape of the curve

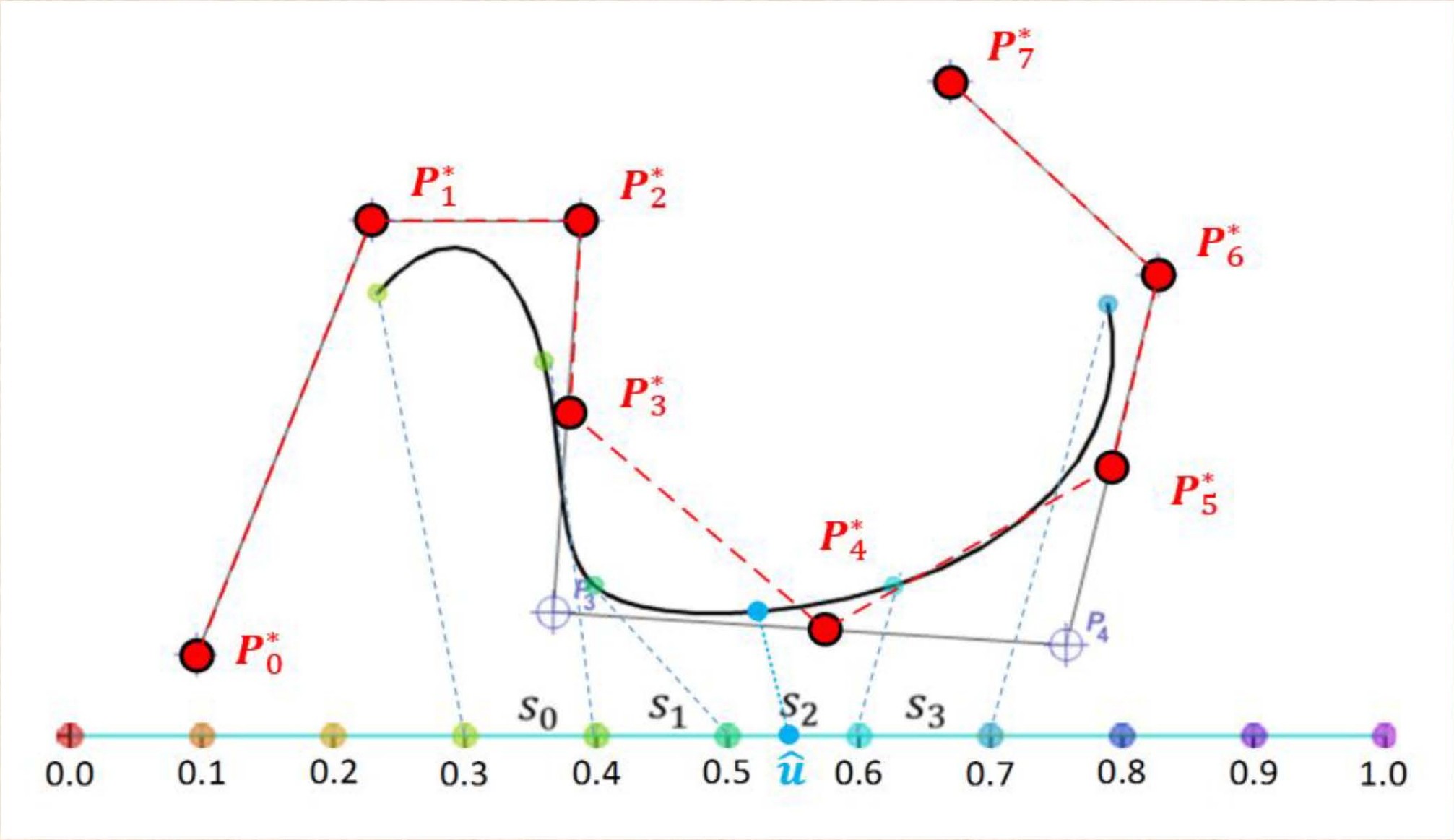
Original Curve



Insert a Knot

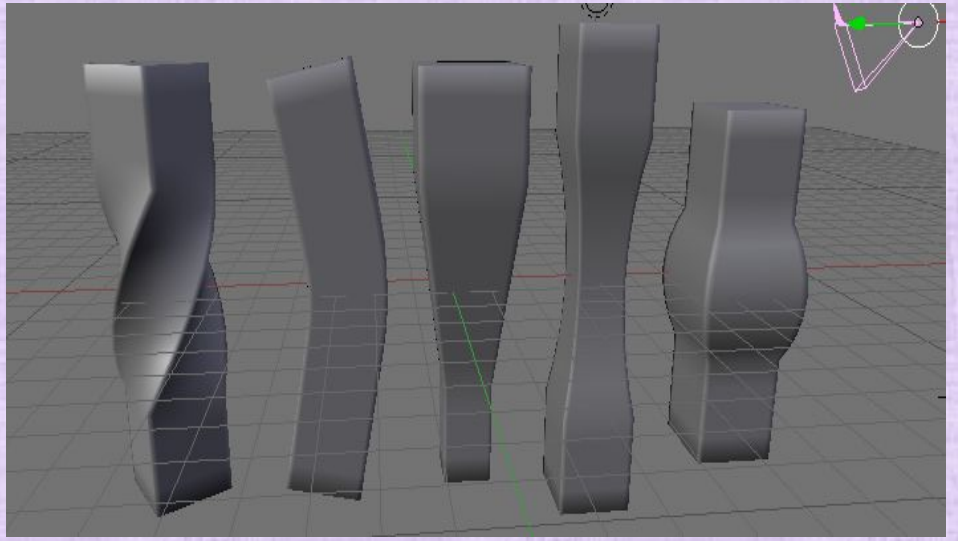


Add/Adjust Control Points



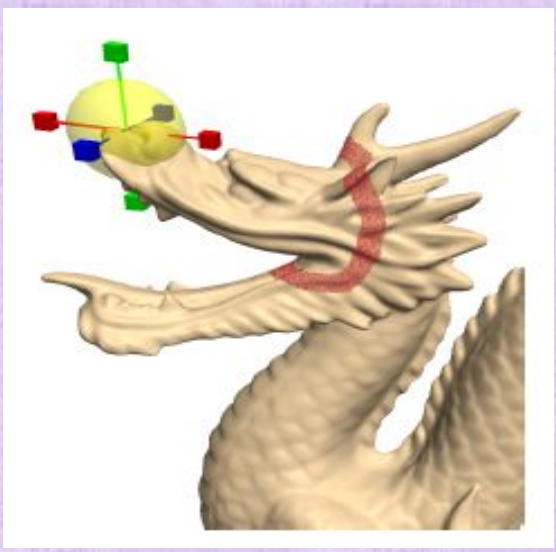
Mesh Editing

- Manipulate a few control/mesh points and use an algorithm to procedurally deform the mesh (e.g., twist, bend, stretch, etc.)
- Spline (e.g. B-spline, NURBS) mesh editing uses control points (as seen in previous slides)
- Laplacian mesh editing allows one to directly move mesh points



mesh editing in Blender

spline mesh editing



Laplacian mesh editing

Differential Coordinates

- The differential coordinate is the difference between a vertex's position x_i and the average position of its neighbors:

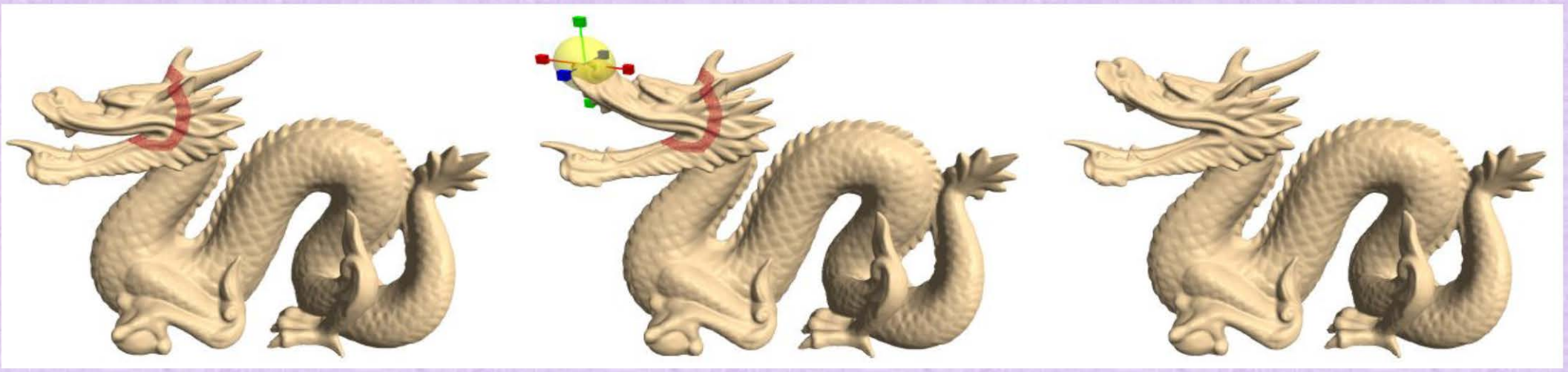
$$d_i = L(x_i) = x_i - \frac{1}{n_i}\sum_{j \in N_i} x_j$$

- Differential coordinates approximate the local shape:
  - The direction of d_i approximates the normal vector direction
  - The magnitude of d_i approximates the mean curvature

- Let X be the vector of mesh vertices (one entry for each vertex)
- Let D be the vector of differential coordinates (one entry for each vertex)
- Since each d_i is linear in x_i as well as all the x_j, can write $D = MX$ where M is a constant coefficient and sparse (vertices only interact with immediate neighbors) matrix

Laplacian Mesh Editing

- Select control points x_c and a region of interest (ROI) containing them
- Set a target position p_c for each control point
- Solve (sparse linear system) $M\hat{X} = D$ for all $\hat{x}_i$ in ROI, with soft constraints $\hat{x}_c = p_c$
- E.g., minimize $E(\hat{X}) = \sum_{i \in \text{ROI}} \|L(\hat{x}_i) - d_i\|_2^2 + \sum_c \|\hat{x}_c - p_c\|_2^2$
- Better results can be obtained by approximating part of the deformation via a transform $T_i(\hat{X})$, computed for each vertex via: $\min_{T_i} \left(\|T_i x_i - \hat{x}_i\|_2^2 + \sum_{j \in N_i} \|T_i x_j - \hat{x}_j\|_2^2 \right)$
- Then $E(\hat{X}) = \sum_{i \in \text{ROI}} \|L(\hat{x}_i) - T_i(\hat{X}) d_i\|_2^2 + \sum_c \|\hat{x}_c - p_c\|_2^2$



Crystal Explorer by Kent Vainio (2020)

METHODS:

- Cloth and rigid body simulations for the cape and treasure chest items
- Blender's hair particle system to create the fluffy fur, cape fur and antenna
- Multiple procedural textures (mostly Voronoi and Musgrave) along with various shaders (glass, emission, etc.) to create a vibrant scene
- Moogle (the fluffy creature - a character from the Final Fantasy video game series) was modeled and sculpted from scratch using reference images

Used loop cut, subdivision, solidify, bezel tool, bisect tool and displacement to create the whiskey, moon rock award, photo frames, etc
Used cloth simulation to model the newspaper. Took pictures of the book covers and created UV textures with a photo editing tool for books.
Combined LightPath node, Transparent, Glossy and Emission BSDF to create a reflective window and its imperfect surface
Downloaded the rocket ship, lunar lander and NASA medal in SolidWorks assembly format and converted with AutoDesk Inventor

Yanjia Li, Lingjie Kong, CS148 2020

Open Shading Language (OSL) Node; Displacement map; Dynamic Sky; Particle System; Motion Blur

- Geometrical low-poly modeling from scratch for all the geometries except for the grass (downloaded from TurboSquid)
- OSL node for procedural stone road texture (idea from Erindale YouTube Channel)
- Random displacement mapping to generate water wrinkles (from a subdivided plane) and tree crown (from an icosphere)
- Particle system with wind force for drifting cherry blossom petals; motion blur applied while rendering
- Blender official add-on “Dynamic Sky” for the basic light source and background; additional lighting for the reflection on the water wrinkles

Yifan Wang, CS148 2020

Examples

- The following slides are all from 2013 or 2014 (so, worth updating, thus…)

- Extra Credit: 5 HW points (i.e. worth one full HW assignment) for anyone who can contribute:
  - a slide with a nice image
  - a few bullet points below the image on how the geometry was modeled
  - names of both partners

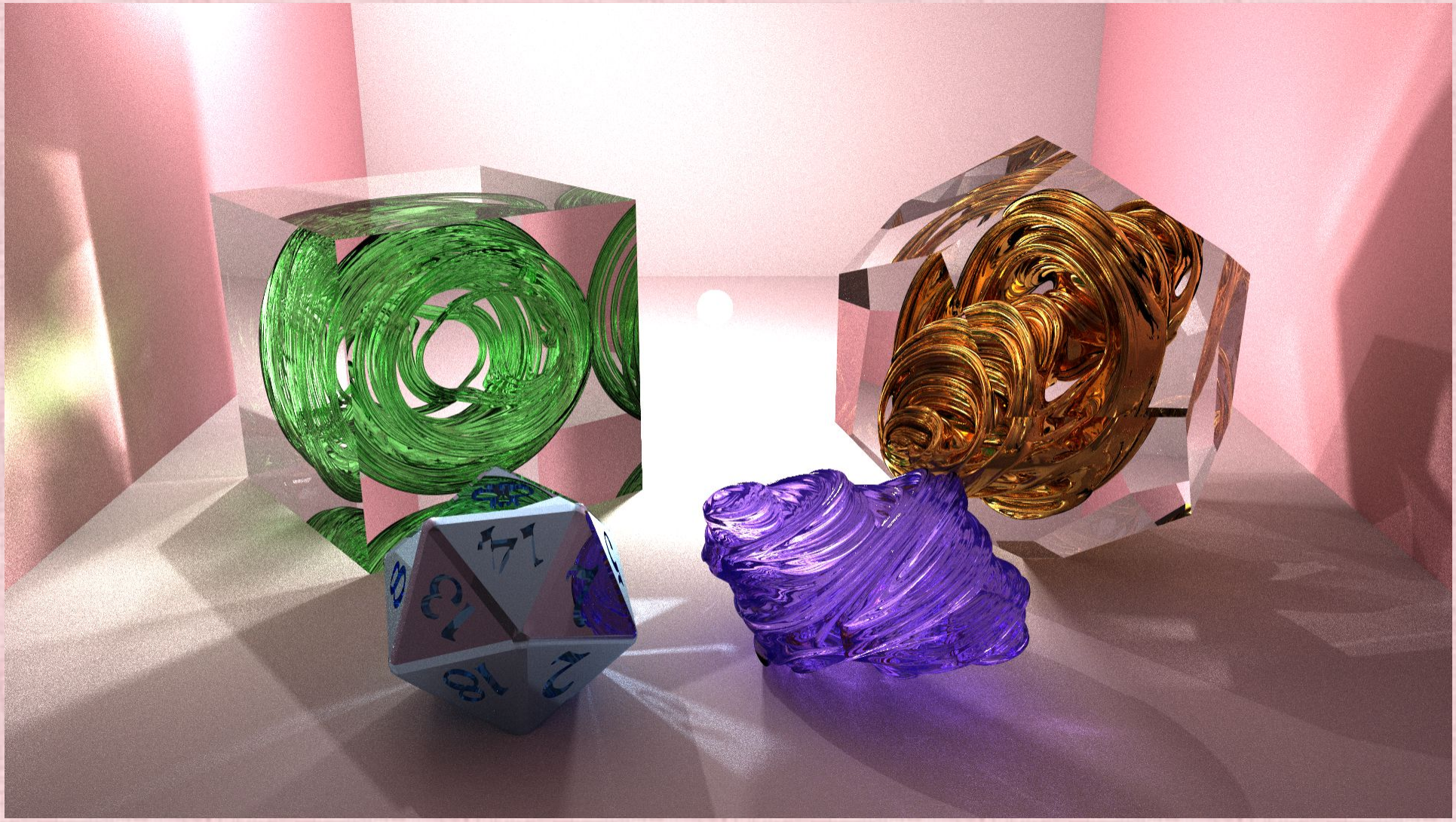
- We will assign anywhere from 0-5 HW points on any submission depending on how good, informative, and useful it is (guaranteed 5 points if we use it to update these slides)
- Recall: there is no overflow past the 50 percentile HW cap

- Deadline: end of the quarter

Maya, TF3DM, environment map, billboards

- The toy monster was modeled in Maya from geometric primitives
- The bridged and helicopter were downloaded from TF3DM
- The UVs and textures for the bridge were fixed in Maya
- The lights are billboards, the sky is an environment map, the water was a plane with a normal map

Marianna Neubauer, CS148 2014



Fractal Geometry: Raytracing Quaternion Julia Sets

- Calculate the ray intersection point (in quaternion space) by iterating over $z_{n+1} = z_n^2 + c$.
- z_n represents the origin of the ray and z_{n+1} represents the ray intersection point.
- c is a quaterion that defines the shape of the Julia Set.
- Caustic effects were accomplished with bidirectional path tracing.
- The D&D die was from TurboSquid.

Marianna Neubauer, CS148 2014

Blender, Microsoft Word and Paint

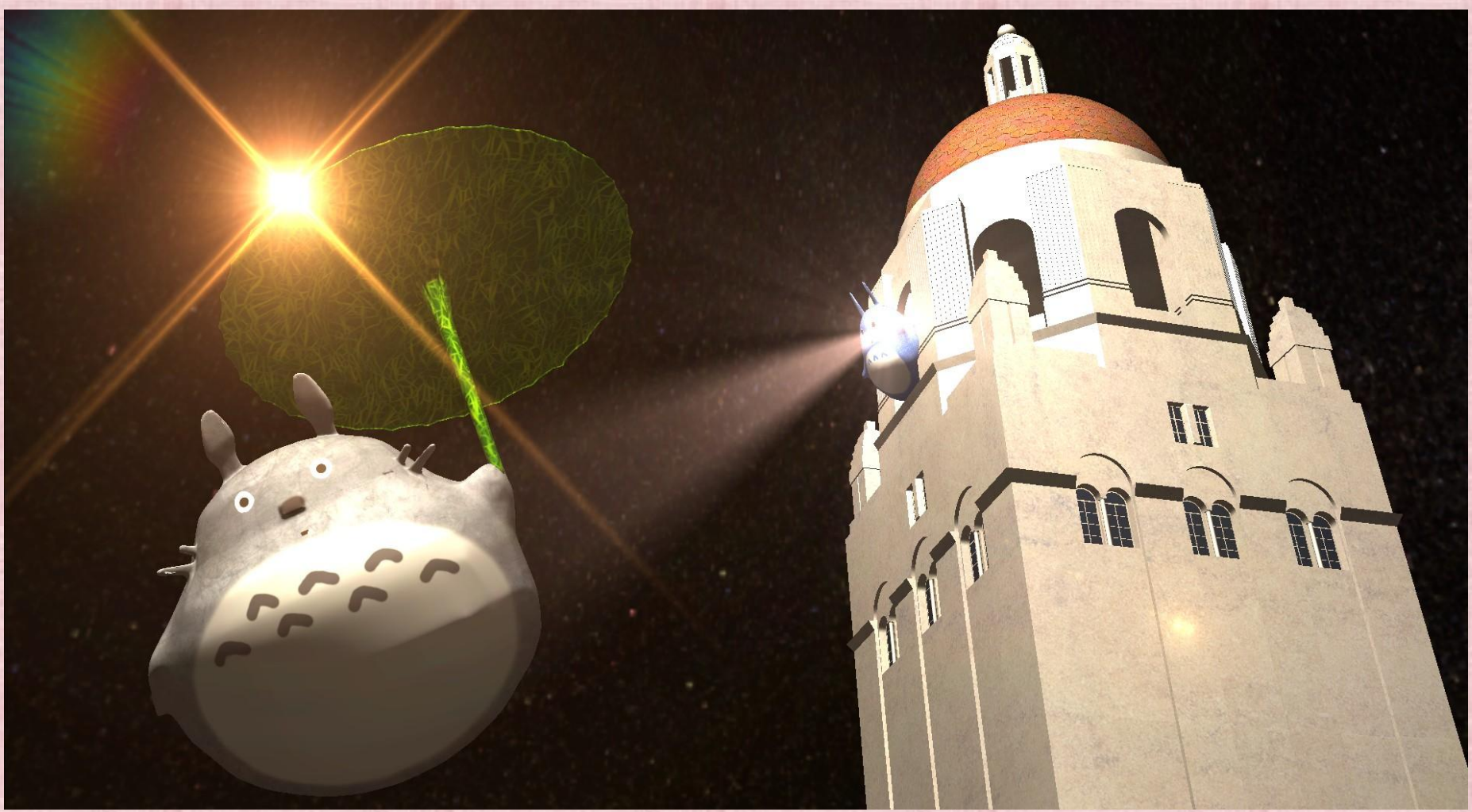
- Models are created, modified, and subdivided in Blender (wolf is downloaded online)
- Careful scaling and rotation of objects to make the scene realistic and seamless
- Textures are done using Blender, Word, and Paint

Angela Gong, CS148 2014

TurboSquid, Blender, Normal mapping, Multiple lights

- Canoe model is downloaded from TurboSquid, and candles and lily pads are created in Blender
- Normal mapping is applied on the water surface and image textures are used for scene objects
- 11 light sources are used to light up the scene, with alpha blending for transparency effect

Megan Faulk & Sarah Nader, CS148 2014



Blender, Maya, Meshlab

- Creates all the meshes with modeling software
- Extends the obj loader given in sample code to read vertex locations, normals, faces, and texture coordinates from files made in Maya and Blender

Adam Young and Winnie Lin, CS148 2013

Blender, Loop Subdivision, Fluid Simulator

- Modeled all the meshes with low resolution in Blender, and then loop-subdivide them for smoother looking
- Generate the icings of the donuts using a fluid simulator
- Generate nomal map from the texture using CrazyBump

Jean-Baptiste Boin, CS148 2013

Artistic Scene Layout with Mixed Downloaded and Manually Generated Meshes

- Find the cloth and other mesh objects available online
- Manually generate mesh for the diamond
- Put everything together artistically, create a scene that resembles traditional still life oil paintings

Yiju Hou and Erika Ji, CS148 2013