Modeling

A Volkswagan Beetle becomes the subject of a 1970 simulation project. Ivan Sutherland (left) and assistants plot coordinates for digitizing the car.

Modeling the Everyday World

Three broad areas:
- Modeling (Geometric) = Shape
- Animation = Motion/Behavior
- Rendering = Appearance
Geometric Modeling

1. How to represent 3d shapes
   - Polygonal meshes

Stanford Bunny
69451 triangles
David, Digital Michelangelo Project
28,184,526 vertices, 56,230,343 triangles

Geometric Modeling

1. How to represent 3d shapes
   - Smooth surfaces
     - Bicubic spline surfaces
     - Subdivision surfaces

Caltech Head
Utah Teapot
Geometric Modeling

1. How to represent 3D shapes
2. How to create 3D shapes
   1. CAD tools
   2. Scanners
   3. Procedurally
3. How to manipulate 3D shapes
   1. Deform/skin/morph/animate
   2. Smooth/compress
   3. Set operations, ...

OpenGL Primitives
Primitive API

```c
glBegin(GL_POLYGON);
    glVertex3f(-1.0,-1.0,0.0);
    glVertex3f(1.0,-1.0,0.0);
    glVertex3f(1.0,1.0,0.0);
    glVertex3f(-1.0,1.0,0.0);
    glVertex3f(-1.0,1.0,0.0);
    glVertex3f(-1.0,-1.0,0.0);
    glEnd();
```

Polygons

```c
float v1[3] = {-1.0,-1.0,0.0};
float v2[3] = { 1.0,-1.0,0.0};
float v3[3] = { 1.0, 1.0,0.0};
float v4[3] = {-1.0, 1.0,0.0};

glBegin(GL_POLYGON);
    glVertex3fv(v1);
    glVertex3fv(v2);
    glVertex3fv(v3);
    glVertex3fv(v4);
    glEnd();
```
#f - #e + #v = 2

```c
typedef float Point[3];

Point verts[8] = {
  {-1.,-1.,-1.},
  { 1.,-1.,-1.},
  { 1., 1.,-1.},
  {-1., 1.,-1.},
  {-1.,-1., 1.},
  { 1.,-1., 1.},
  { 1., 1., 1.},
  {-1., 1., 1.},
};

face(int a, int b, int c, int d) {
  glBegin(GL_POLYGON);
  glVertex3fv(verts[a]);
  glVertex3fv(verts[b]);
  glVertex3fv(verts[c]);
  glVertex3fv(verts[d]);
  glEnd();
}
```

// Note consistent ccw orientation!

cube() {
  face(0,3,2,1);
  face(2,3,7,6);
  face(0,4,7,3);
  face(1,2,6,5);
  face(4,5,6,7);
  face(0,1,5,4);
}
Points/Polygons

typedef float Point[3];
Point verts[8] = {
{-1.,-1.,-1.},
{ 1.,-1.,-1.},
{ 1., 1.,-1.},
{-1., 1.,-1.},
{-1.,-1., 1.},
{ 1.,-1., 1.},
{ 1., 1., 1.},
{-1., 1., 1.},
};
int polys[6][4] = {
{0,3,2,1},
{2,3,7,6},
{0,4,7,3},
{1,5,4,3},
{4,5,6,7},
{0,1,5,4},
};

face(int a, int b, int c, int d) {
  glBegin(GL_POLYGON);
  glVertex3fv(verts[a]);
  glVertex3fv(verts[b]);
  glVertex3fv(verts[c]);
  glVertex3fv(verts[d]);
  glEnd();
}
cube() {
  for(int i = 0; i < 6; i++)
    face(polys[i][0],
         polys[i][1],
         polys[i][2],
         polys[i][3]);
}

Representations

Polygons
  + Simple
  - Redundant information
Points/Polygons
  + Share vertices (compress/consistency)
Additional topological information
  + Constant time access to neighbors
    More advanced algorithms such as
      surface normal calculation, subdivision ...
  - Additional storage for topology
  - More complicated data structures
Triangle Adjacency

```c
struct Vert {
    Point pt;
    Face *f;
};

struct Face {
    Vert *v[3];
    Face *f[3];
};
```

Recursive Subdivision

Bezier curves
Subdivision surfaces
Fractals
Recursively divide into two curves

Left side

\[ Q_0 = P_0 \]
\[ Q_1 = P_0^1 \]
\[ Q_2 = P_0^2 \]
\[ Q_3 = P_0^3 \]
Beziers Curves – Midpoint Subdivision

Recursively divide into two curves

Right side

\[ R_0 = P_0^3 = \frac{1}{8} P_0 + \frac{1}{4} P_1 + \frac{1}{4} P_2 + \frac{1}{8} P_3 \]
\[ R_1 = P_1^2 = \frac{1}{4} P_1 + \frac{1}{2} P_2 + \frac{1}{4} P_3 \]
\[ R_2 = P_2^1 = \frac{1}{2} P_2 + \frac{1}{2} P_3 \]
\[ R_3 = P_3 \]
Subdivision Surfaces

Triangle Mesh
Triangle Mesh – Subdivide

Loop Algorithm - Edge
Loop Algorithm - Vert

Semi-Regular Meshes

Extraordinary Points
Loop Subdivision – Extraordinary Vertex

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

\[ 1 - k \beta \]

k neighbors ...

Fractal Subdivision

\[ \Delta x \]
Fractal Subdivision

\[ \Delta x \]

Fractal Subdivision

\[ \Delta y = \text{random()} \cdot \Delta x \]

\[ \Delta x \]
Fractal Subdivision: Height Field
Things to Remember

Common representations
- Dense polygon mesh data structures
  - Polygon
  - Points/Polygon (MeshArray)

Subdivision algorithms
- Subdivision surfaces and the Loop subdivision algorithm
- Fractal subdivision algorithm

Operations
- Computing normals from polygonal mesh