

Department of Computer Science **Fog in OpenGL**

```
glEnable( GL_FOG )
```

- If enabled, blend a fog color into the post texturing color.

```
glFog( ... )
```

- Specify fog parameters

```
glHint( GL_FOG_HINT, mode )
```

- Specify implementation-specific hints
- *mode*: `GL_FASTEST`, `GL_NICEST`, and `GL_DONT_CARE`

5/20/2007 CG1 - Surfaces (v1.00) 1

Department of Computer Science **Fog**

```
glFog[if]( property, value )
```

- Depth Cueing – specify a range for a linear fog ramp
 - `GL_FOG_LINEAR`
- Environmental effects – simulate more realistic fog
 - `GL_FOG_EXP`
 - `GL_FOG_EXP2`
- Properties and default values:

<code>GL_FOG_MODE</code>	equation to use to compute fog blend factor	<code>GL_EXP</code>
<code>GL_FOG_DENSITY</code>	density, >= 0	1
<code>GL_FOG_START</code>	near distance	0.0
<code>GL_FOG_END</code>	far distance	1.0
<code>GL_FOG_INDEX</code>	color index	0
<code>GL_FOG_COLOR</code>	RGBA fog color	(0,0,0,0)

5/20/2007 CG1 - Surfaces (v1.00) 2

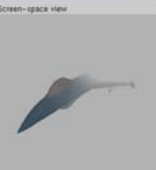
Department of Computer Science **Fog Tutorial**

Fog equation

$$f = \frac{\text{end} - z}{\text{end} - \text{start}}$$

z is the distance in eye coordinates from origin to fragment being fogged.

Screen-space view



Command manipulation window

```
GLfloat color[4] = { 0.70 , 0.70 , 0.70 , 1.00 };
glFogf(GL_FOG_COLOR, color);
glFogf(GL_FOG_START, 0.50 );
glFogf(GL_FOG_END, 2.00 );
glFogi(GL_FOG_MODE, GL_LINEAR);
```

Click on the arguments and move the mouse to modify values.

5/20/2007 3

Department of Computer Science **Smooth Curves and Surfaces**

- Many real-world objects are inherently smooth
- Much of CG involves modeling the real world
- Two common cases:
 - Modeling existing objects
 - Modeling objects that do not exist
- For existing objects, we commonly approximate the shape of the object with spheres, planes, etc.
- For new objects, we're not approximating – we're creating

5/20/2007 CG1 - Surfaces (v1.00) 4

Department of Computer Science **Surface Modeling Techniques**

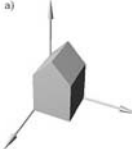
- Many different techniques
- We'll talk about these:
 - *Polygon meshes*
 - Approximate the surface with flat polygons
 - *Parametric models* – bicubic patches
 - Use a mathematical formula to construct a network of intersecting curves
 - *Blobs*
 - Sculpt rounded, organic surfaces based on spheres and gravity formulas
 - *Subdivision surfaces*
 - Form surfaces from the fractal subdivision of polygon meshes; a relatively new technique from Pixar

5/20/2007 CG1 - Surfaces (v1.00) 5

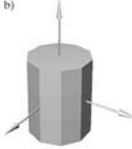
Department of Computer Science **Polygonal Meshes**

- Surface given by a set of connected polygons
- Collection of edges, vertices, and polygons
 - Each edge is shared by at most two polygons
 - Each edge is part of at least one polygon
- Advantages:
 - Compact representation
 - Polygonal mathematics is fairly straight forward
 - Rendering hardware optimized for rendering polygons


a)



b)



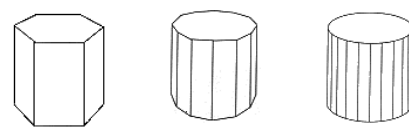
c)



5/20/2007 CG1 - Surfaces (v1.00) 6

Polygonal Meshes

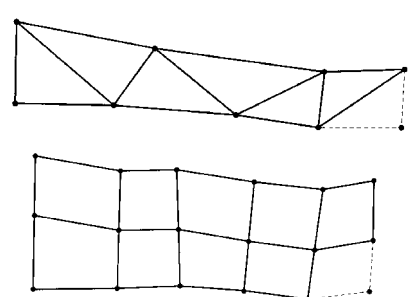
- Best at representing planar surfaces
- Can approximate rounded surfaces by increasing polygon count:



5/20/2007 CG1 - Surfaces (v1.00) 7

Polygonal Meshes

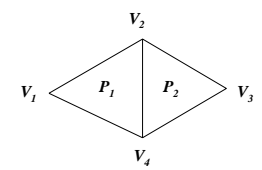
- Triangle and quad meshes



5/20/2007 CG1 - Surfaces (v1.00) 8

Representing Meshes

- Vertex list representation:



$$V = (V_1, V_2, V_3, V_4)$$

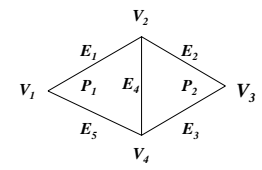
$$P_1 = (1, 2, 4)$$

$$P_2 = (4, 2, 3)$$

5/20/2007 CG1 - Surfaces (v1.00) 9

Representing Meshes

- Edge list representation:



$$V = (V_1, V_2, V_3, V_4)$$

$$E_1 = (V_1, V_2, P_1)$$

$$E_2 = (V_2, V_3, P_2)$$

$$E_3 = (V_3, V_4, P_2)$$

$$E_4 = (V_4, V_2, P_1, P_2)$$

$$E_5 = (V_4, V_1, P_1)$$

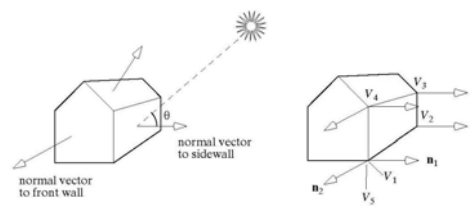
$$P_1 = (E_1, E_4, E_5)$$

$$P_2 = (E_2, E_3, E_4)$$

5/20/2007 CG1 - Surfaces (v1.00) 10

Mesh Normals

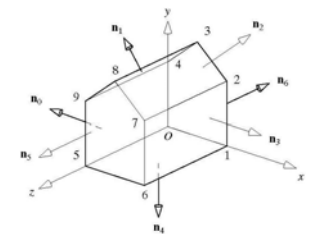
- Along with vertex and edge information, must specify normals
- Most shading algorithms make use of the normal
 - Normals determine brightness, etc.
- Can specify normals for faces or vertices



5/20/2007 CG1 - Surfaces (v1.00) 11

Mesh Normals

- Specifying normals at vertices can be advantageous
 - Helps with clipping and shading
- Example: the “basic barn” with seven faces
- For simplicity, base will be square – can always scale



5/20/2007 CG1 - Surfaces (v1.00) 12

Department of Computer Science **Basic Barn – Vertices**

Vert.	x	y	z
0	0	0	0
1	1	0	0
2	1	1	0
3	0.5	1.5	0
4	0	1	0
5	0	0	1
6	1	0	1
7	1	1	1
8	0.5	1.5	1
9	0	1	1

5/20/2007 CG1 - Surfaces (v1.00) 13

Department of Computer Science **Basic Barn – Normals**

Norm.	n_x	n_y	n_z
0	-1	0	0
1	-0.707	0.707	0
2	0.707	0.707	0
3	1	0	0
4	0	-1	0
5	0	0	1
6	0	0	-1

5/20/2007 CG1 - Surfaces (v1.00) 14

Department of Computer Science **Basic Barn – Faces**

Face	Vertices	Associated Normals
0 (left)	0, 5, 9, 4	0, 0, 0, 0
1 (roof L)	3, 4, 9, 8	1, 1, 1, 1
2 (roof R)	2, 3, 8, 7	2, 2, 2, 2
3 (right)	1, 2, 7, 6	3, 3, 3, 3
4 (bottom)	0, 1, 6, 5	4, 4, 4, 4
5 (front)	5, 6, 7, 8, 9	5, 5, 5, 5, 5
6 (back)	0, 4, 3, 2, 1	6, 6, 6, 6, 6

- Could enumerate vertices for faces in several ways
- Handy convention:
Traverse the polygon counterclockwise, as seen from outside the object.

5/20/2007 CG1 - Surfaces (v1.00) 15

Department of Computer Science **Drawing Meshes in OpenGL**

- Mesh is a collection of faces
- Each face is a polygon

```

for( each face f in object ) {
    glBegin( GL_POLYGON );
    for( each vertex v in face f ) {
        glNormal3f( normal for this vertex );
        glVertex3f( position of vertex v );
    }
    glEnd();
}
    
```

5/20/2007 CG1 - Surfaces (v1.00) 16

Department of Computer Science **Polygon Meshes**

- Pros:
 - Fast rendering
 - Scanned data is easily acquired
- Cons:
 - High polygon density necessary to approximate areas in detail

5/20/2007 CG1 - Surfaces (v1.00) 17

Department of Computer Science **Parametric Models**

- Polylines and polygons are first-degree approximations
 - Polylines: approximate curves
 - Polygons: approximate surfaces
- Work well for piecewise linear objects
 - Other objects require lots of endpoint coordinates to achieve reasonable accuracy
- For higher accuracy and more efficiency, we want to use a higher-degree approximation
- Three methods:
 - Explicit functions
 - Implicit functions
 - Parametric representation

5/20/2007 CG1 - Surfaces (v1.00) 18

Explicit Functions

- Represent y and z as functions of x
 $y = f(x), z = g(x)$
- Problems:
 - Can only get one y or z for each x , so circles and ellipses will require more than one segment
 - Not rotationally invariant
 - Describing curves with vertical tangents will require representing slopes of infinity

5/20/2007 CG1 - Surfaces (v1.00) 19

Implicit Functions

- Model curves as solutions to implicit equations, e.g.
 $f(x,y,z) = 0$
- Problems:
 - Circles can be modeled with, e.g., $x^2 + y^2 = 1$ – how do you model *half* a circle?
 - If two implicitly defined curves are joined, must ensure that tangent directions at the join point agree

5/20/2007 CG1 - Surfaces (v1.00) 20

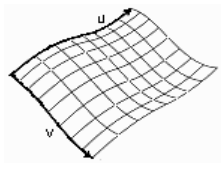
Parametric Representation

- Parametric forms overcome the problems with explicit and implicit forms
- Parametric representation:
 $x = x(t), y = y(t), z = z(t)$
- Tangent slopes can be infinite; parametric tangent vectors cannot
- Curves are approximated by piecewise polynomial curves, not piecewise linear curves
- Each segment Q of the overall curve is represented by equations for $x, y,$ and z which are cubic polynomials in the variable t

5/20/2007 CG1 - Surfaces (v1.00) 21

Bicubic Parametric Patches

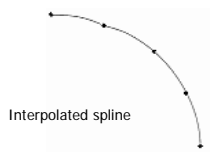
- Defines surfaces using mathematical formulas
- Patch: two connected curves
 - One in s direction, the other in t direction
 - Surface between curves is determined by interpolation.
 - Surfaces are curved
- Each patch is defined by two *splines*, u & v



5/20/2007 CG1 - Surfaces (v1.00) 22

Splines

- Curves drawn to conform to a series of control points
- Splines can be *interpolated* or *approximated*
- Interpolated splines: curve is drawn through the points
- Approximated splines: curve is drawn near the points, but not through them

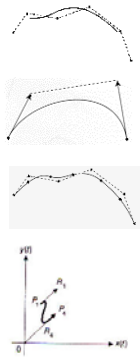


Interpolated spline

5/20/2007 CG1 - Surfaces (v1.00) 23

Approximated Splines

- Many different forms
- B-spline: curve passes near, but not through, points
- Bezier curve: curve is controlled by tangent vectors
- Non-uniform, rational B-splines (NURBS): curves pass through first and last points, near all others
- Hermite curve: specified by endpoints and tangent vectors from each endpoint



5/20/2007 CG1 - Surfaces (v1.00) 24

Parametric Models

- x, y, z given as function of t
- Approximated by cubic polynomials

$$x(t) = a_x t^3 + b_x t^2 + c_x t + d_x$$

$$y(t) = a_y t^3 + b_y t^2 + c_y t + d_y$$

$$z(t) = a_z t^3 + b_z t^2 + c_z t + d_z$$

5/20/2007 CG1 - Surfaces (v1.00) 25

Parametric Models

- To deal with finite curve segments, we restrict t to the interval $[0,1]$
- Define T as

$$T = [t^3 \ t^2 \ t \ 1]^T$$

- Define C as the matrix of coefficients of the polynomials

$$C = \begin{bmatrix} a_x & b_x & c_x & d_x \\ a_y & b_y & c_y & d_y \\ a_z & b_z & c_z & d_z \end{bmatrix}$$

- Then,

$$Q(t) = [x(t) \ y(t) \ z(t)]^T = C \cdot T$$

5/20/2007 CG1 - Surfaces (v1.00) 26

General Bicubic Patch

- Q specifies the location of shape of the patch
- P_{ij} are the control points
- B_i and B_j represent the defining splines

$$Q(u, v) = \sum_{i=0}^3 \sum_{j=0}^3 P_{ij} B_i(u) B_j(v)$$

5/20/2007 CG1 - Surfaces (v1.00) 27

Control Points

- Control the shape of the curve
- Moving control points changes the appearance

5/20/2007 CG1 - Surfaces (v1.00) 28

Basis and Geometry Matrices

- Separate C matrix into basis + geometry matrices

$$C = G \cdot M$$

- M : basis matrix

$$M = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ m_{41} & m_{42} & m_{43} & m_{44} \end{bmatrix}$$

- G : geometry matrix

$$G = \begin{bmatrix} g_{1x} & g_{2x} & g_{3x} & g_{4x} \\ g_{1y} & g_{2y} & g_{3y} & g_{4y} \\ g_{1z} & g_{2z} & g_{3z} & g_{4z} \end{bmatrix}$$

$$G = [G_1 \ G_2 \ G_3 \ G_4]$$

5/20/2007 CG1 - Surfaces (v1.00) 29

Basis and Geometry Matrices

- Basis and control points distinguish types of curves
- Control points are in geometry matrix

$$Q(t) = G \cdot M \cdot T$$

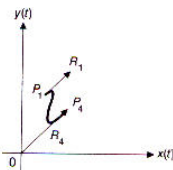
$$= \begin{bmatrix} g_{1x} & g_{2x} & g_{3x} & g_{4x} \\ g_{1y} & g_{2y} & g_{3y} & g_{4y} \\ g_{1z} & g_{2z} & g_{3z} & g_{4z} \end{bmatrix} \cdot \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ m_{41} & m_{42} & m_{43} & m_{44} \end{bmatrix} \cdot \begin{bmatrix} t^3 \\ t^2 \\ t^1 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} x(t) \\ y(t) \\ z(t) \end{bmatrix}$$

5/20/2007 CG1 - Surfaces (v1.00) 30

Hermite Curves

- Hermite curves
 - Specified by endpoints and tangent vectors from each endpoint



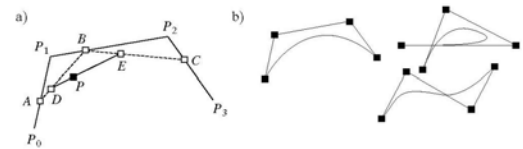
$$M_H = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

$$G_H = [P_1 \quad P_4 \quad R_1 \quad R_4]$$

5/20/2007 CG1 - Surfaces (v1.00) 31

Bezier Curves

- Bezier curves are specified by endpoints and intermediate points, not on the curve, that approximate the tangent vectors



$$M_B = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & -3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

$$G_H = [P_0 \quad P_1 \quad P_2 \quad P_3]$$

5/20/2007 CG1 - Surfaces (v1.00) 32

Parametric Continuity

- Parametric continuity of joined curves
 - C^0 continuity - curve segments are joined at endpoints
 - C^1 continuity - direction and magnitude of tangent vectors at join points are equal (first parametric derivatives equal)
 - C^2 continuity - direction and magnitude of n th derivative are equal (first and second parametric derivatives equal)
- Hermite and Bezier curves have C^0 and C^1 continuity
- Splines have C^0 , C^1 , C^2 continuity

5/20/2007 CG1 - Surfaces (v1.00) 33

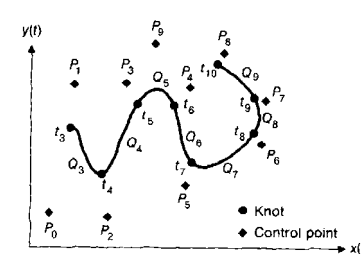
Splines

- Natural splines
 - Control points lie on the curves themselves
 - Global control
 - Moving one control point changes the entire curve
 - Moving one control point forces re-evaluation of entire curve.
- B-splines
 - Control points are outside of the curve
 - Local control
 - Moving one control point only affects a portion of the entire curve.
 - Individual curve segments share control points.
 - Knots** - points connecting individual curve segments.

5/20/2007 CG1 - Surfaces (v1.00) 34

B-Splines

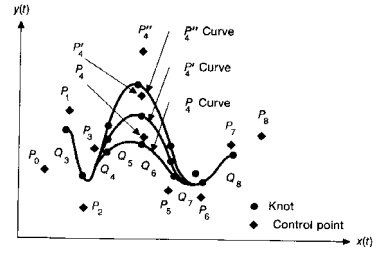
- B-splines



5/20/2007 CG1 - Surfaces (v1.00) 35

B-Splines

- B-splines - local control



5/20/2007 CG1 - Surfaces (v1.00) 36

B-Splines

- Segment Q_i uses control points $P_{i-3}, P_{i-2}, P_{i-1}$ and P_i

$$M_B = \frac{1}{6} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & -3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix}$$

$$G_B = [P_{i-3} \quad P_{i-2} \quad P_{i-1} \quad P_i]$$

5/20/2007 CG1 - Surfaces (v1.00) 37

Parametric Models

- Curve applet
 - <http://www.cc.gatech.edu/gvu/multimedia/nsfmmedia/graphics/edulib/Solomon/CurveDraw.html>

5/20/2007 CG1 - Surfaces (v1.00) 38

Types of Splines

- Uniform/non-uniform
 - Knots are/aren't equally spaced
- Rational/non-rational
 - Does/doesn't use homogeneous coordinates
- B (basis)
 - Curves are weighted sums of a basis function

5/20/2007 CG1 - Surfaces (v1.00) 39

Rational Splines

- Expressed using homogeneous coordinates

$$Q(t) = [X(t) \quad Y(t) \quad Z(t) \quad W(t)]$$

$$x(t) = \frac{X(t)}{W(t)}, \quad y(t) = \frac{Y(t)}{W(t)}, \quad z(t) = \frac{Z(t)}{W(t)}$$

5/20/2007 CG1 - Surfaces (v1.00) 40

Rational Splines

- Rational splines -- why bother?
 - Invariant under rotation, scaling, translation, and perspective transforms
 - Can define precisely any of the conic sections. (spheres, ellipsoids, paraboloids, etc)
- NURBS
 - Non-uniform Rational B-Splines
 - Non-uniform - knots need not be equally spaced.
 - Most useful and used method for representing curves in CG

5/20/2007 CG1 - Surfaces (v1.00) 41

Parametric Surfaces

- Connecting patches

5/20/2007 CG1 - Surfaces (v1.00) 42

Rendering

- Turn surface patches into polygons
- Recursively subdivide until patch closely resembles a plane surface.
- If do patch to polygon conversion, can use rendering hardware directly

5/20/2007 CG1 - Surfaces (v1.00) 43

Polygonal Meshes

- Pros:
 - Simple concept
 - Simple data storage
 - Fast rendering
- Cons:
 - Cannot represent a true curve
 - Difficult to build complex objects
 - Large data storage

5/20/2007 CG1 - Surfaces (v1.00) 44

Parametric Surfaces

- Pros:
 - Curves allow greater realism
 - Interactive editing potential
 - Use patch to polygon conversion to use existing rendering hardware
- Cons:
 - Slow rendering time due to the complex mathematics
 - Acquiring data via scans is difficult




Image © Hash, Inc.

5/20/2007 CG1 - Surfaces (v1.00) 45

Blobs

- Sculpt rounded, organic surfaces based on spheres and gravity formulas
- Pros:
 - Blobs lend them-selves naturally to organic shapes
- Cons:
 - Slow rendering
 - Not working directly with surfaces is non-intuitive




Image © Lorenzo Quintana

5/20/2007 CG1 - Surfaces (v1.00) 46

Subdivision Surfaces

- Begin with a control mesh of the model
- Subdivide each polygon, adjusting edges and vertices
 - New vertices at center of each face
 - New edges created as an average of midpoints of original edge with average of 2 new face points sharing an edge
 - Additional new vertices formed
 - Mesh is regenerated

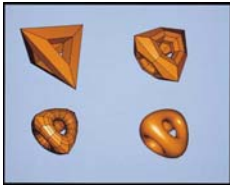


Image © Computer Graphics World

5/20/2007 CG1 - Surfaces (v1.00) 47

Subdivision Surfaces

- *Gerl's Game* uses this plus a set of infinitely sharp rules then smooth rules to produce limit surface
- Surfaces sharp at coarse scale, smooth at finer scale
- And vary sharpness of crease along its length





Image © Pixar

5/20/2007 CG1 - Surfaces (v1.00) 48

 Department of
Computer Science

Subdivision Surfaces

- Pros:
 - Easy to acquire scanned data
 - Represents curved surfaces well
- Cons:
 - Slow rendering




Image © Pixar

5/20/2007 CG1 - Surfaces (v1.00) 49