11/24/2020

Only one topic today (E1) another quiz next Tuesday (Shading).
These two quizzes will take most of the points of the quiz component.

Extra Credit deadline on 12/03/2020 at 5:00 pm

1) **Instructor evaluation proof of submission**
    a. **Email from IT**
    b. **Take a screenshot of the confirmation**
    c. **You can still offer qualitative (written) answers**
2) **Class evaluation**

Two alternatives, choose the one you prefer. Can do both but get extra credit only for 1.
Please comply with the requirements for 2 (in case you choose 2)

<u>Curves and Surfaces</u>

Last time concentrated on the general principals and OGL implementation.
Today, complete the OGL part.

Next, we will study how to use the conditions (control points etc.) to develop
The parametric equations.

The parametric equations sought

Based on the control points and restrictions (conditions) we obtain:

$x(u) = c_{0x} * u^0 + c_{1x} * u^1 + c_{2x} * u^{2\,+} c_{3x} * u^3$
$y(u) = c_{0y} * u^0 + c_{1y} * u^1 + c_{2y} * u^{2\,+} c_{3y} * u^3$
$z(u) = c_{0z} * u^0 + c_{1z} * u^1 + c_{2z} * u^{2\,+} c_{3z} * u^3$

Concentrating on
x(u)

$x(u) = c_0 * u^0 + c_1 * u^1 + c_2 * u^{2\,+} c_3 * u^3$

Find (solve) the C's
OGL is solving these C's for the Bezier Curve

How do we figure 4 unknowns (C's)?

Consider an interpolating curve.
We supply 4 points P_0, P_1, P_2, P_3.

We want to get a curve that "goes through" these 4 points.

The curve is expected to be represented by a cubic polynomial.
Assume that it starts at P_0 and ends at P_3.

OGL has to figure the 4 C's for X (12 for X, Y, and Z in 3-D)

What can we do to get these 4 unknowns

We have to figure out 4 equations , upon solving the equations we obtain the C's

In OGL the first part is:

Given 4 points P_0, P_1, P_2, P_3
OGL figures the 4 equation in 4 unknowns
Solves and get the 3 parametric functions.

$x(u) = c_{0x} * u^0 + c_{1x} * u^1 + c_{2x} * u^{2+} c_{3x} * u^3$
$y(u) = c_{0y} * u^0 + c_{1y} * u^1 + c_{2y} * u^{2+} c_{3y} * u^3$
$z(u) = c_{0z} * u^0 + c_{1z} * u^1 + c_{2z} * u^{2+} c_{3z} * u^3$

For the Bezier curve.

Done using the function glMap1f()

Next OpenGL plots the 3 functions

Internal loop
    u goes from 0 to 1 in increments of du (du is based on your specifications)
        glBegin(shape)
            Produce a vertex at $[x(u), y(u), z(u)),1]^T$ (using glEvalCoord1f())
        glEnd()

The shape can be any GL 2.x shape (point, line, line strip, quad strip).

To Produce a vertex at $[x(u), y(u), z(u)),1]^T$ (e.g. @ u=0.25)
Using glEvalCoord1f((float) 0.25);
Instead of glVertex() for the actual x, y, and z for u=0.25

Each call to glEvalCoord1f((float) d); with a given value d is instantiating a vertex

```
glBegin(shape)
glEvalCoord1f( (float) 0);
glEvalCoord1f( (float) 0.25);
glEvalCoord1f( (float) 0.4);
glEvalCoord1f( (float) 0.6);
glEvalCoord1f( (float) 1.0);
glEnd()
```

Not necessarily uniformly spaced between 0 and 1

Instead of the 2 OGL instructions
You can generate a plot where the point are uniformly spaced using

```
glMapGrid(100, 0.0, 1.0);
glEvalMesh1(GL_LINE, 0, 99);
```

OGL can only produce the 3 functions

$x(u) = c_{0x} * u^0 + c_{1x} * u^1 + c_{2x} * u^{2\,+} c_{3x} * u^3$
$y(u) = c_{0y} * u^0 + c_{1y} * u^1 + c_{2y} * u^{2\,+} c_{3y} * u^3$
$z(u) = c_{0z} * u^0 + c_{1z} * u^1 + c_{2z} * u^{2\,+} c_{3z} * u^3$

For a Bezier curve

How to interpolate or B-Spline

38:15 left most: what you “get” from the 4 points in GL
How to get interpolation ho to get a B-spline
|➔ use model view transformations.

38:13 how to get the Transformation
38:14 provides the transformation

For S5 you have to load the matrices to the Model view t get the right effect.