CS 559 Final Exam

December 17, 2008

Please write your name on every page (we may unstaple the exams for grading)

Write numerical answers in fractional form or use radicals (square root symbols) – we would prefer to see $\frac{\sqrt{3}}{2}$ than .866. You should not need a calculator for this exam.

Unless otherwise noted, assume that everything is a right-handed coordinate system and that angles are measured counter clockwise. E.g. to find the direction of rotation, point your thumb along the axis and curl your fingers.

If you need extra space, use the back of a page, but clearly mark what everything is. We may look at your work to determine partial credit.

The exam has 100 points

You have the entire exam period to complete the exam.

Question 1: What did that word mean and why did we use it? (5pts each = 15)

For each part (A,B,C) there are two subparts. Please be clear where your answers are for each part. A sentence or two is sufficient for each.

1.A.1) The XYZ color system uses "**imaginary** colors" as its primaries. Why are these colors "**imaginary**"?

1.A.2) Why would you need a color system with **imaginary** primaries (rather than one with non-imaginary primaries)?

Imaginary refers to the fact that the colors are not physically realizable

An imaginary color system is used to create a 3-primary color system that can describe the entire range of possibilities for the human visual system. While you can't implement such a color system, it is useful in analyzing color systems that you can.

1.B.1) What are the three things referred to by the "tri" in **tri-linear** interpolation?

1.B.2) Where is **tri-linear** interpolation used, and why would you prefer it to "less than tri-" linear interpolation (like bi-linear or uni-linear)?

Tri-linear interpolation interpolates in position (x,y) and level of the mip map.

Tri-linear interpolation (and mip-mapping) provides a fast approximation to texture filtering. Interpolating in one level of the mip map restricts you to filtering that will be either too blurry or too sharp (lots of aliasing).

1.C.1) What are the two parts refered to in the "bi" in **bilateral** filtering?

1.C.2) What is **bilateral** filtering useful for that plain (uni-lateral) filtering is not?

Bi-Lateral filtering uses both the distance in space and value difference.

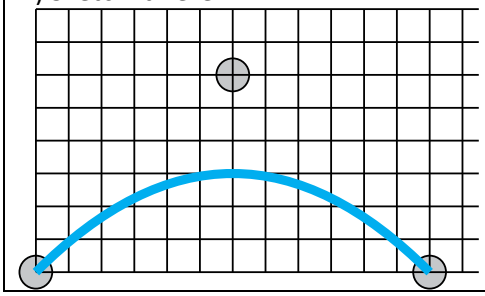
Bi-Lateral filtering preserves edges better than traditional low-pass filtering (it is sometimes referred to as edge-preserving smoothing), and is more "robust" (outliers don't throw things off). This gets used in tone mapping.

Note: just saying tone mapping is only ½ the story.

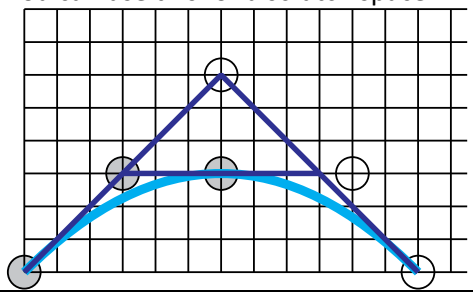
Question 2: Bézier Curves (2+4+5+2+3 = 16pts)

A quadratic Bézier Curve in 2D has its control points at (0,0), (6,6) and (12,0).

A) Sketch it here:



You can use this for a scratch space



B) Divide the curve in Part A in half – that is create a new that is the same as the first half of the given curve. Give the positions of the control points of the new curve.

(0,0), (3,3), (6,3) – work is shown in the diagram (DeCastlejau alg)

C) A Cubic Bézier is the same curve as the curve in Part A (Hint: it is sufficient for the endpoints and the first derivatives at the endpoints to match). What are the positions of its control points?

As part of figuring this out, you'll need to first figure out what the derivatives of the curve in Part A are. Write them here so we can give you partial credit if you get the answer wrong:

(12,12), (12,-12) (e.g. 2(P2-P1) and 2(P3-P2))

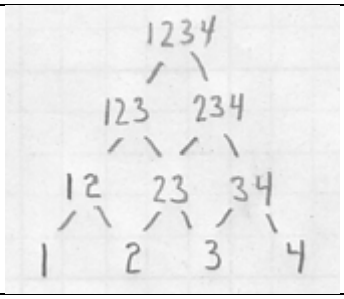
The Positions for the cubic control points are:

(0,0), (4,4), (8,4), (12,0)

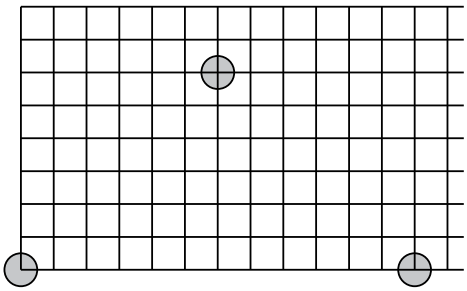
There are several ways to have "guessed" this (or checked that its right). For one, the symmetry of it.

Question 2: Bézier Curves (continued)

D) You could compute the value of the curve that is your answer to part C for a particular U value by doing a number of linear interpolations. How many?

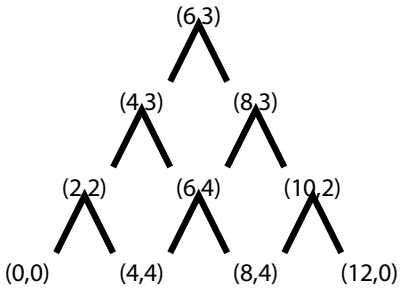
6. The picture at right is a helpful reminder of what the different required pairings are (123 is an interpolation of 12 and 23).	

E) Compute the value of your curve in Part C at parameter (U) value .5 by doing the sequence of linear interpolations. Give the positions of all of the intermediate points. If Part C is correct, this answer should be related to Part A/B – but we're checking to see that you get the right answer for the numbers you have in part C.



You can use this grid for scratch space.

From the other parts, you should know that the answer is 6,3. But to actually do it:



Question 3: Déjà vu (4pts + 4 pts = 8pts)

If these questions seem familiar, it's because they are very similar (but not exactly the same) as questions on the midterm.

A) For the town project, a student said they wrote a Phong shader (e.g. that implements lighting using the Phong lighting model and Phong shading) that is used for all objects in the world. We're not sure if they aren't just doing the standard OpenGL lighting (using Phong lighting and Gouraud shading).

Describe an object that we could add to the world that would make it easy to tell if the student really did Phong shading. Describe how it would look different if it were really Phong shading (versus just being Gouraud shading).

The easiest things are big, specular polygons. Especially when the normals face different directions. In this case, just computing lighting at the vertices will be really insufficient. You can get a specular highlight just in the middle with Phong.

B) A vertex (point) is drawn at the origin. It is viewed through a camera that is positioned with the viewing matrix:

½	-1/2	0	-2
½	½	0	-2
0	0	0	2
0	0	0	1

The object that the vertex is drawn with transformation matrix:

0	-1	0	2
1	0	0	4
0	0	1	6
0	0	0	1

This simple projective transform matrix is used:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

Where does the point appear in screen coordinates? (give the x,y position)

The point (0,0,0) is put through the object transform matrix giving (2,4,6,1). Then the camera matrix is applied giving (-3, 1,8,1), and the projection gives (-3, 1,1,-8). Dividing by W gives:

(3/8, -1/8)

Question 4: Subdivision (4pts + 9 pts = 13 pts)

The control mesh (the initial polygon for subdivision) in all of these examples is a tetrahedron.
A Tetrahedron is a 4-sided polyhedron where every side is a triangle.
Describe the polygons (number and number of sides) you would have if you did:

0) No subdivision (example)
   4 polygons, all with 3 sides
A) One round of Loop subdivision

16 triangles

B) One round of Catmull-Clark subdivision

12 quads

C) One round of Butterfly (or modified Butterfly) subdivision

16 triangles

Describe the vertices (with the number of edges) that you would have if you did:

0) No subdivision (example)
   a. Four vertices, all with 3 edges connected to them
A) One round of Loop subdivision

10 vertices total:
4 vertices with 3 edges (the original ones)
6 vertices with 6 edges (each edge has a new vertex, and each new vertex is regular)

B) One round of Catmull-Clark subdivision

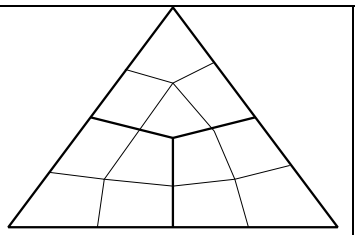
14 vertices total:
8 vertices with 3 edges - 4 original vertices and 4 face points
6 vertices with 4 edges (each of the edge points)

C) Two rounds of Catmull-Clark subdivision

8 vertices with 3 edges each
N vertices with 4 edges each
This was the important part (remembering that we only introduce extraordinary points in round 1, so we copy the 8 from there)

To get N – notice that each face has 6 regular points inside it (see the picture are right), and each edge has 3 regular points on it. So N is 4*6+6*3 = 42

So, if you're keeping score:
50 vertices, 8 with 3 edges, 42 with 4 edges



Question 5: Programmable Shading (5 pts each = 15)

While these questions use GLSL terminology (since we used it in class, and it was in the readings), the concepts are more general and apply to other current systems.

A) Explain what a varying variable is. Is this something you specify in your host (e.g. C++) program?

A varying variable is something output by the vertex program that is interpolated, and then given to the fragment shader.

Since its interpolated, you cannot specify it directly. It always is determined by 3 vertices.

B) Can a vertex program pass a value directly to a fragment program? Explain why or why not. Your explanation should be about how the hardware works, not just the syntax of the language.

No, since each fragment depends on 3 vertices, the fragment shader gets something that is interpolated between these – it can't get a value from a particular vertex.

Possible alternate answers: point primitives only have a single vertex, so maybe. Or you could imagine sending all 3 values to the fragment shader and having it choose.

C) A type of "mapping" was done that used a texture to make a flat surface appear bumpy. The effect is very convincing: even the silhouette (e.g. when you look edge on) is correct. (so when you look at the flat surface from the side, you can see the bumps). Was this bump mapping? Could it have been done in the pixel shader? Explain how you know.

No – its displacement mapping.

Since the pixel shader cannot change the positions of the fragment. In order to make the edge not look like a line, something had to change positions. So this could not have been done in the pixel shader.

Question 6: Some Definitions (4 pts each = 20pts)

Explain the following terms in a sentence or two:

A) Limit Surface

The surface obtained after an infinite number of applications of the subdivision rules.

B) Metamer

A color (distribution of light frequency) that is perceptually equivalent.

C) Caustic (in the way it was used in the rendering lecture)

Where a specular reflection serves as a light source on a diffuse surface.

Technically, its where a curved surface focuses light onto another surface. But the simpler definition is OK.

D) Tone Mapping

Mapping a range of intensities in an image to the range available on a display.

E) Euler Angles

A representation of a 3D rotation as three numbers, each one is the amount of rotation about a fixed axis.

Question 7: Lighting (4+4+5= 13 points)

A) What term (or lighting type) in the standard OpenGL lighting model is used to fake global illumination. Explain how it approximates it.

Ambient.

It assumes that even if there is no direct lighting, some light gets to everything indirectly. It just makes the amount of indirect light constant.

B) Can global illumination be created easily with a (traditional from the eye) Ray-Tracer? Give a brief explanation why or why not.

No. It would require a lot of rays. And potentially handling an infinite recurrence (since objects effect one another).

C) Distribution Ray Tracing uses a number of rays (usually randomly distributed) where a traditional ray tracer might just send one ray. Describe 3 effects that are easy to create using distribution (that are hard without it). (Note: 2 points for the first one, 2 points for the second, 1 point for the third)

C1) Soft Shadows

C2) Anti-Aliasing

C3) Motion Blur

Depth in field

Glossy reflections (imperfect surfaces)