# EECS 487, Fall 2006
# Exam 2

December 21, 2006

This is a closed-book exam. Notes are not permitted.
Basic calculators are permitted, but not needed.
Explain or show your work for each question.

Name: ______________________________

uniqname: ______________________________

**Honor code pledge:** I have neither given nor received aid on this exam.

Signature: ________________________________

| ***problem*** | ***points*** |
|---:|---:|
| 1. | /4 |
| 2. | /12 |
| 3. | /12 |
| 4. | /4 |
| 5. | /16 |
| 6. | /8 |
| 7. | /8 |
| 8. | /4 |
| 9. | /4 |
| 10. | /12 |
| 11. | /8 |
| 12. | /8 |
| bonus | /2 |
| Total | /100 |

## *Splines*

**1. (4 points)** Suppose we specify a quadratic spline curve $\mathbf{f}(t) = \mathbf{a}_0 + \mathbf{a}_1 t + \mathbf{a}_2 t^2$ as follows. The curve starts at $\mathbf{p}_0$, ends at $\mathbf{p}_1$, and its tangent vector at the halfway mark is $\mathbf{v}$. In other words: $\mathbf{f}(0) = \mathbf{p}_0$, $\mathbf{f}(1) = \mathbf{p}_1$, and $\mathbf{f}'(½) = \mathbf{v}$. Recall that the constraint matrix $\mathbf{C}$ relates $\mathbf{p}_0$, $\mathbf{p}_1$, and $\mathbf{v}$ to the polynomial coefficients $\mathbf{a}_0$, $\mathbf{a}_1$, $\mathbf{a}_2$ as follows:

$$\begin{pmatrix} \mathbf{p}_0 \\ \mathbf{p}_1 \\ \mathbf{v} \end{pmatrix} = \mathbf{C} \begin{pmatrix} \mathbf{a}_0 \\ \mathbf{a}_1 \\ \mathbf{a}_2 \end{pmatrix}$$

What is $\mathbf{C}$ in this case?

**2. (12 points)** If we change the definition of the spline in problem 1, replacing "$\mathbf{f}'(½) = \mathbf{v}$" with "$\mathbf{f}(½) = \mathbf{p}_{½}$" (i.e., specifying the *location*, not derivative, at $t = ½$), the resulting basis matrix is:

$$\mathbf{B} = \begin{pmatrix} 1 & 0 & 0 \\ -3 & 4 & -1 \\ 2 & -4 & 2 \end{pmatrix}$$

and we have: $\mathbf{f}(t) = (1 \quad t \quad t^2)\ \mathbf{B}\ \mathbf{p}$, where $\mathbf{p}$ is the column "vector" containing $\mathbf{p}_0$, $\mathbf{p}_{½}$, and $\mathbf{p}_1$.

(a) What are the basis functions for this spline?

(b) What is $\mathbf{f}'(½)$? Express your answer as simply as possible in terms of $\mathbf{p}_0$, $\mathbf{p}_{½}$, and $\mathbf{p}_1$.

(c) It turns out that the constraint matrix in problem 1 is singular. (You should be able to verify this: note that the third row is a linear combination of the first two.) What is "wrong" with the spline specification in problem 1 that results in this singular constraint matrix?

## *Animation*

**3. (12 points)** For each of the following animation techniques, give one example of a type of motion for which the technique would be suitable, and another type of motion for which it would not be suitable.

(a) Keyframe animation

(b) Physics-based simulation

(c) Motion capture

**4. (4 points)** What is “secondary motion”? Why is it important in animation?

## *Ray tracing*

**5. (16 points)** Suppose a surface is defined implicitly as the set of 3D locations ($x$,$y$,$z$) that satisfy:

$$\left(\frac{x}{3}\right)^2+\left(\frac{y}{2}\right)^2+z^2=1$$

Let $\mathbf{r}(t)$ = (-3,-3,-3) + $t$ (1,1,1) define a ray. It can be checked that the ray does intersect the surface in two places.

(a) What procedure can we use to verify that the ray intersects the surface?

(b) What are the $t$-values at the two intersection points?

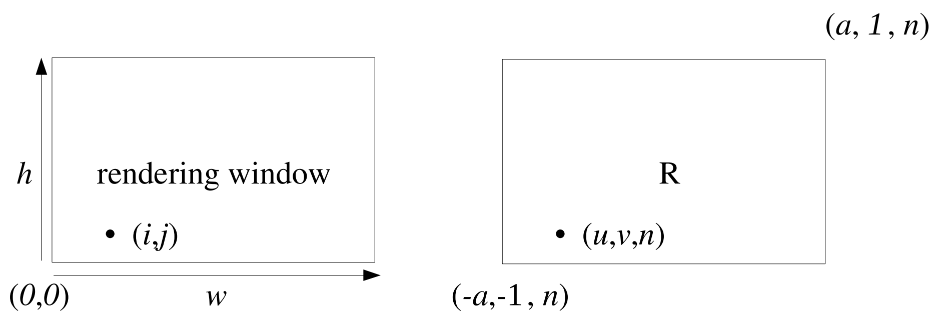
(c) What are the intersection points in 3D?

(d) What procedure can we use to find the surface normal at a given intersection point? What are the surface normals at the two intersection points in this case?

**6. (8 points)** Suppose the rendering window is $w$ pixels wide and $h$ pixels high, and let $a$ denote the aspect ratio ($a = w/h$). Suppose we define pixel coordinates so that integer coordinates refer to the *lower-left corner* (not the center) of a given pixel. For example, coordinate (0,0) refers to the lower left corner of the lower left pixel (see figure below).

We'll use the same conventions as in our text to define "eye space" via camera location **e**, direction **u** that points to the right of the camera, direction **v** that points up, and direction **w** that points behind the camera (so that **-w** points forward along the line of sight). Each of **e**, **u**, **v**, and **w** are defined in world space.

For rendering, we will define a rectangle R in eye space that is mapped to the rendering window. R lies in the plane $w = n$ and is centered at $(0, 0, n)$. It has the same aspect ratio as the rendering window, and its lower left corner is $(-a, -1, n)$, and its upper right corner is $(a, 1, n)$. Note: That is "1" (numeral one) not "$l$" (letter l).




(a) What is the transform that maps floating point pixel coordinates $(i,j)$ to corresponding eye-space coordinates $(u,v,-n)$ in R? Specifically, tell how to compute $u$ in terms of $i$, and $v$ in terms of $j$.

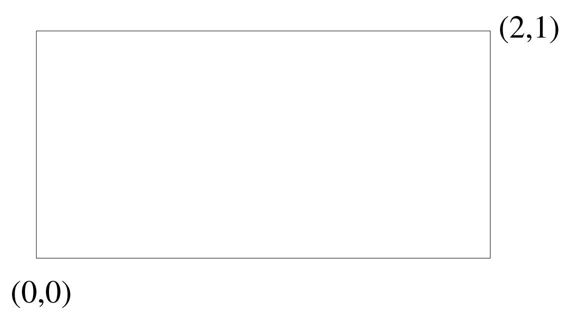
(b) Given an eye space coordinate $(u,v,n)$, what is the corresponding world space location?

**7. (8 points)** Compared to Monte-Carlo ray tracing, classical ray tracing makes simplifying assumptions that greatly reduce the number of computations necessary to render an image.

(a) Describe a simplifying assumption made in classical ray tracing that is not made in Monte-Carlo ray tracing, and explain what realistic effect is thereby supported by Monte-Carlo ray tracing but not by classical ray tracing.

(b) Describe a realistic effect that is not supported in Monte-Carlo ray tracing. What simplifying assumption made in Monte-Carlo ray tracing is responsible for the failure?

**8. (4 points)** Describe a procedure to generate random samples *uniformly* within the following rectangle that is 2 units wide and 1 unit tall:



## *Radiosity*

**9. (4 points)** Which of the following realistic rendering effects is supported by radiosity?

- depth of field
- motion blur
- soft shadows
- glossy reflection
- subsurface scattering
- color bleeding
- caustics

**10. (12 points)** Suppose a simple scene is represented with two patches (numbered 0 and 1). The radiosity of the scene is modeled by the radiosity equation:

$$\mathbf{B} = \mathbf{E} + \mathbf{MB}$$

where **B** is the 2-vector of radiosity values (one per patch), **E** is the 2-vector of light emission values (telling how much light is emitted by each patch), and **M** is a 2x2 matrix that encodes how light leaving the patches is subsequently received and reflected by them. *i.e.*, **MB** is a 2-vector that tells how much light is reflected by each patch. It does this by summing the incoming light at each patch and multiplying by the material color of the patch.

(a) What is the formal solution of the radiosity equation (above)?
Express **B** symbolically in terms of **E** and **M**.

Suppose we are given the vector of emission values $\mathbf{E} = \begin{pmatrix} \frac{1}{2} \\ 0 \end{pmatrix}$, and matrix $\mathbf{M} = \begin{pmatrix} \frac{1}{5} & \frac{2}{5} \\ \frac{1}{2} & \frac{1}{3} \end{pmatrix}$.

(b) What is the radiosity **B** in this case? (Recall: $\begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1} = \frac{1}{(ad - bc)} \begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$ ).

(c) Radiosity can be solved iteratively, starting with an initial estimate **B** = **E**, then iteratively updating **B** using the radiosity equation. Using this method, what is the radiosity estimate after one iteration?

## *Precomputed radiance transfer*

**11. (8 points)** Precomputed radiance transfer is a technique for computing global illumination effects like soft shadows and color bleeding at interactive frame rates.

(a) What information is "precomputed"?

(b) What assumption is made that lets us re-use the precomputed information over multiple frames at run-time?

(c) Why does the method only support low-frequency lighting?

(d) What would go wrong if a high-frequency lighting environment was used?

## *Non-photorealistic rendering*

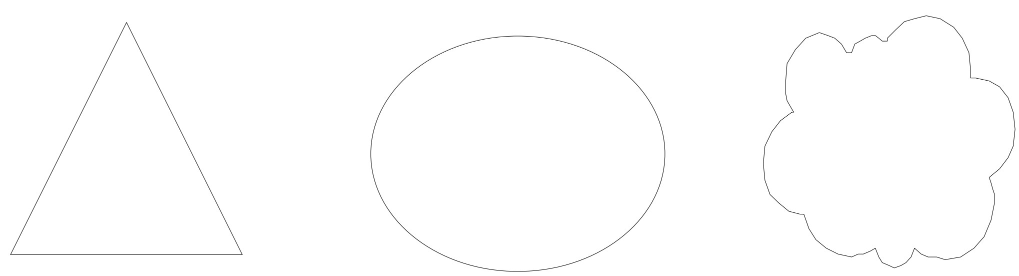
**12. (8 points)**

(a) Describe an application for which photorealistic images are usually better than non-photorealistic images, and another application for which the situation is reversed.
What is the reason for the difference?

(a) Describe the basic idea of "G-buffers" in Saito and Takahashi's work on "comprehensible rendering." Give an example of a G-buffer that differs from an ordinary shaded rendering of a scene, and explain how image processing can be applied to the image to produce a useful effect.

## *Bonus question (2 points)*

For each of the following shapes, tell whether it is $C^0$, $C^1$, or $C^2$.



(a) (b) (c)