Computer Graphics **Instructor: Steve Seitz**

CSE 457, Spring 2002

Homework #2

Hidden Surfaces, Shading, Ray Tracing, Texture Mapping, Parametric Curves, Particle Systems

Prepared by: Eugene Hsu, Khoi Nguyen, and Steve Seitz

Assigned: Friday, May 10th
Due: Wednesday, May 29th, **at the beginning of class**

Directions: Please provide short written answers to the questions in the space provided. If you require extra space, you may staple additional pages to the back of your assignment. Feel free to discuss the problems with classmates, but please ***answer the questions on your own*.**

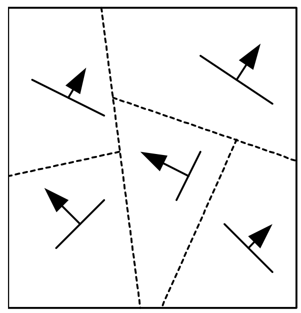
Name:____

Student ID:____

Problem 1. BSP Trees (20 Points)

BSP trees are widely used in computer graphics. Many variations can be used to increase performance. The following questions deal with some of these variations.

For the version of BSP trees that we learned about in class, polygons in the scene (or more precisely, their supporting planes) were used to do the scene splitting. However, it is not necessary to use existing polygons – one can choose arbitrary planes to split the scene:



a. What is one advantage of being able to pick the plane used to divide the scene at each step? What is one disadvantage of not just using existing polygons? Draw an example to illustrate your point for each case.

BSP Trees (cont'd)

Recall that when using a BSP tree as described in class, we must draw *all* the polygons in the tree. This is inefficient, since many of these polygons will be completely outside of the view frustum. However, it is possible to store information at the internal nodes in a BSP tree that will allow us to easily determine if any of the polygons below that node will be visible. If none of the polygons in that sub-tree will be visible, we can completely ignore that branch of the tree.

b. Explain what extra information should be stored at the internal nodes to allow this, and how it would be used to do this "pruning" of the BSP tree.

c. In class, we talked about doing a "back to front" traversal of a BSP tree. But it is sometimes preferable to do a "front to back" traversal of the tree, in which we draw polygons closer to the viewer before we draw the polygons farther away. (See part (d) for one reason why this is useful) How should the tree traversal order be changed in order to do a front to back traversal?

BSP Trees (cont'd)

When we traverse a BSP tree in back to front order, we may draw over the same pixel location many times, which is inefficient since we would do a lot of "useless" shading computations. Assume we instead traverse the tree in front to back order. As we scan convert each polygon, we would like to be able to know whether or not each pixel of it will be visible in the final scene (and thus whether we need to compute shading information for that point).

d. What simple information about the screen do we need to maintain in order to know if each pixel in the next polygon we draw will be visible or not? About how many bits/bytes are needed to encode this information?

Problem 2. Shading (21 Points)

Respond TRUE or FALSE to each of these statements and ***explain your reasoning.***

a. The Phong model provides an accurate physical model for how light interacts with real surfaces.

b. A rough surface with many tiny microfacets is likely to have a large diffuse reflection coefficient. Assume that the microfacets are oriented in random directions.

c. In the Phong model, specular reflection does not depend on viewing angle.

Phong and Gouraud Shading (cont'd)

In the following, suppose you are approximating a curved surface with a triangular mesh (i.e., a set of adjacent triangles).

d. Gouraud interpolation cannot produce specular highlights

e.The brightest point on an object shaded with Gouraud interpolation can be brighter than the brightest point shaded with Phong interpolation. (hint: can points within a polygon be brigher than the vertices?).

f. By increasing the number of polygons, you can make the difference between Gouraud interpolation and Phong interpolation to be arbitrarily small—you can make the polygons small enough that there is no perceivable difference.

g. Gouraud interpolation requires fewer shading operations (evaluations of the Phong illumination equation) than Phong shading.

Problem 3. Ray Tracing Implicit Surfaces (20 Points)

There are many ways to represent a surface. One such way is to define a function of the form $f(x, y, z) = 0$. Such a function is called an *implicit surface* representation. For example, the equation $f(x, y, z) = x^2 + y^2 + z^2 - r^2 = 0$ defines a sphere of radius r. Suppose that you want to ray trace the paraboloid surface defined by the function $z = x^2 + y^2$.

a. Find the implicit function *f* such that the equation $f(x, y, z) = 0$ defines this paraboloid.

b. Now suppose that you are given a ray $\mathbf{p} + t\mathbf{d}$, where **p** is the origin of the ray and **d** is the direction of the ray. Solve for *t* such that the ray intersects the paraboloid. Show your work, and have your answer be in terms of $\mathbf{p} = (p_x, p_y, p_z)$ and $\mathbf{d} = (d_x, d_y, d_z)$.

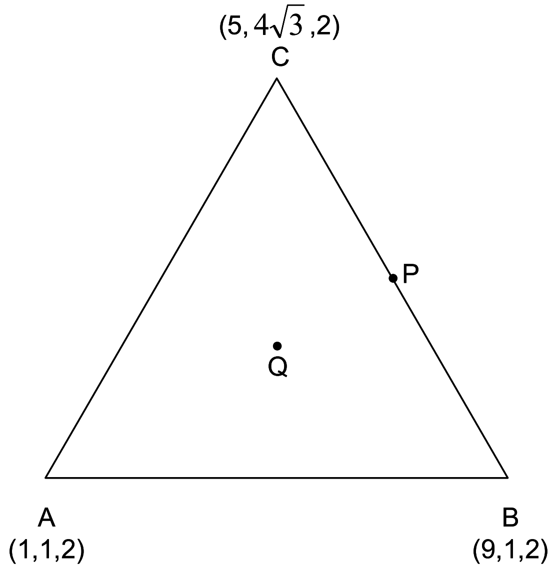
Ray Tracing Implicitly Defined Surfaces (cont'd)

c. Suppose that $\mathbf{p} = (2,4,8)$ and $\mathbf{d} = (-1,-1,-1)$. Where does this ray intersect the surface? If there are multiple intersections, be sure to give all points.

d. The normal N of an implicitly defined surface is given by the gradient of the implicit function. In other words, $\mathrm{N}(x,y,z) = \left(\frac{\partial f}{\partial x}(x,y,z), \frac{\partial f}{\partial y}(x,y,z), \frac{\partial f}{\partial z}(x,y,z) \right)$. Give the function N for the paraboloid. What is a *unit* normal at $(1,2,5)$? ***Show your work.***

Problem 4. Barycentric Coordinates (15 Points)

Given a set of points in three-dimensional space, we can define a local coordinate system with respect to these points using something called *barycentric coordinates*. Barycentric coordinates are quite useful, and are used extensively in working with triangles.



If Q is a point inside the triangle formed by A, B, and C, then the coordinates of Q can be expressed as a weighted average of the coordinates of A, B, and C, as follows:

$$Q = \alpha A + \beta B + \gamma C$$
$$\alpha + \beta + \gamma = 1$$

We call α, β, and γ, the barycentric coordinates of Q with respect to A, B, and C.
To solve for Q's barycentric coordinates, use the following equations:

$$\alpha = \frac{Area(QBC)}{Area(ABC)} \qquad \beta = \frac{Area(AQC)}{Area(ABC)} \qquad \gamma = \frac{Area(ABQ)}{Area(ABC)}$$

Barycentric Coordinates (cont'd)

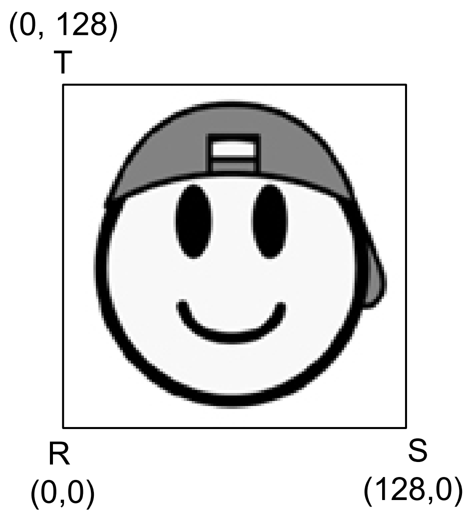
a. Calculate the barycentric coordinates for points Q, P, and A for the equilateral triangle shown on the previous page (hint: using the fact that it is equilateral will make the computation easier).

- Point Q is the center of the triangle.
- Point P is the midpoint of the edge BC.

b. Barycentric coordinates can also be used to interpolate vertex properties such as shading coefficients, normals, and texture coordinates. Suppose vertex A, B, and C were shaded with colors S_A, S_B, and S_C, respectively. Given a point Q with barycentric coordinates α, β, and γ, give the equation for the interpolated color of Q using α, β, and γ. Check to make sure your formula gives the right answer for the vertices.

Barycentric Coordinates (cont'd)

Suppose you have a texture image shown below, and you know the vertices A, B, C of the triangle map to R, S, and T respectively. The size of the texture image is 128x128 pixels.



c. Calculate the texture image coordinate that corresponds to points P using the barycentric coordinates found in part a. Draw the approximate location on the texture image above.

Problem 5. Parametric Curves (16 Points)

Fill out the following table describing properties of the listed curves. Put an "X" in each cell where the specified curve has the given property.

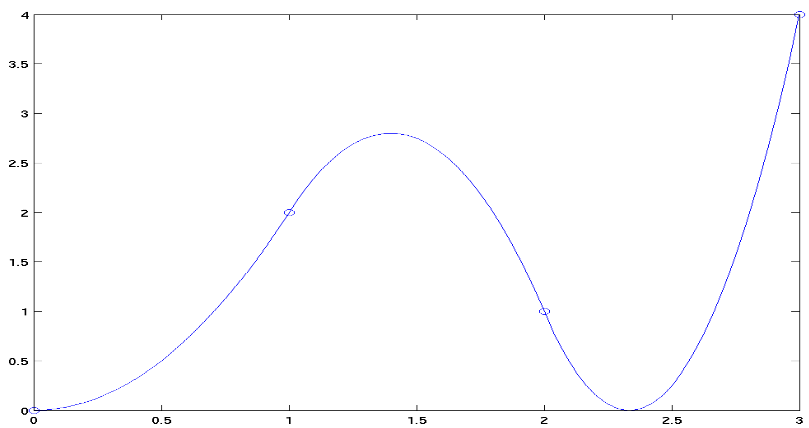
	C2 continuous?	Local control?	Interpolates *all* control points?	Within convex hull of control points?
Bezier				
B-spline				
C2 interpolating				
Catmull-Rom				

Problem 6. Quadratic Splines (20 points)

Suppose that you wish to construct a C1 continuous piecewise quadratic function $f(x)$ that satisfies $f(0)=0,\ f(1)=2,\ f(2)=1,\ f(3)=4$. More specifically, $f(x)$ is defined as follows.

$$f(x)=\begin{cases} f_0(x)= & a_0+b_0x+c_0x^2 & 0\le x<1 \\ f_1(x)= & a_1+b_1(x-1)+c_1(x-1)^2 & 1\le x<2 \\ f_2(x)= & a_2+b_2(x-2)+c_2(x-2)^2 & 2\le x\le 3 \end{cases}$$

A picture of the solution is shown below. Your goal is to solve for the coefficients.



a. What does it mean for a curve to be C0 continuous? Given that $f(x)$ is C0 continuous and that it interpolates the given points, write out the constraint equations that arise from the definition of the curve and solve for as many coefficients as possible. For example, we know that $f_1(0)=0$, so we can solve for a_0 easily. Because of C0 continuity, we can relate a_0, b_0, and c_0 to a_1, b_1, and c_1 by looking at the values of $f_1(1)$ and $f_2(1)$, etc. Show your work!

Piecewise Quadratic Curves (cont'd)

b. What does it mean for a curve to be C1 continuous? Given that $f(x)$ is C1 continuous, write out constraint equations and solve for as many coefficients as possible. Show your work.

c. Suppose you are given that $\frac{df}{dt}(0) = 0$. Now there is enough information to uniquely determine $f(x)$. Solve for the coefficients. Show your work. You may wish to check your solution with the graph.

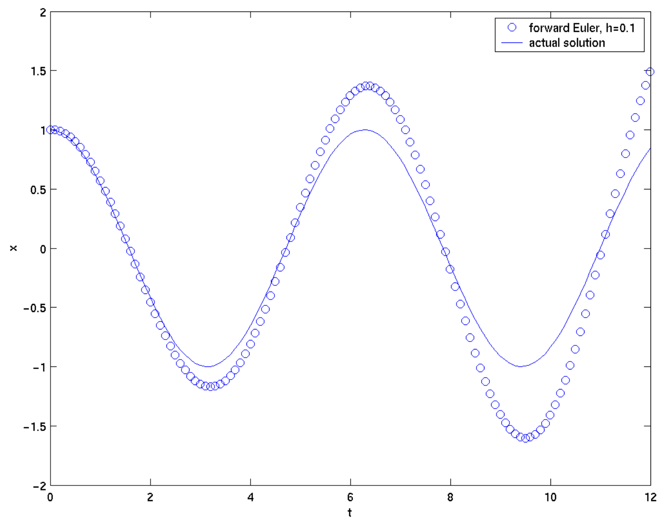
d. Use your results to compute $f(1.9)$ and $f(2.7)$.

Problem 7. Particle Systems (20 points, 5 points extra credit)

The motion of a particle on an undamped spring with spring constant 1 and rest length 0 can be physically modeled by the following second order differential equation.

$$\frac{d^2x}{dt^2} = -x$$

Suppose that we are given the initial conditions $x(0) = 1$ and $x'(0) = 0$; in other words, the particle is initially at position $x = 1$ and has velocity 0. Below is a plot of Euler's method for solving this differential equation versus the actual solution $x = \cos t$.



t vs *x* plot.

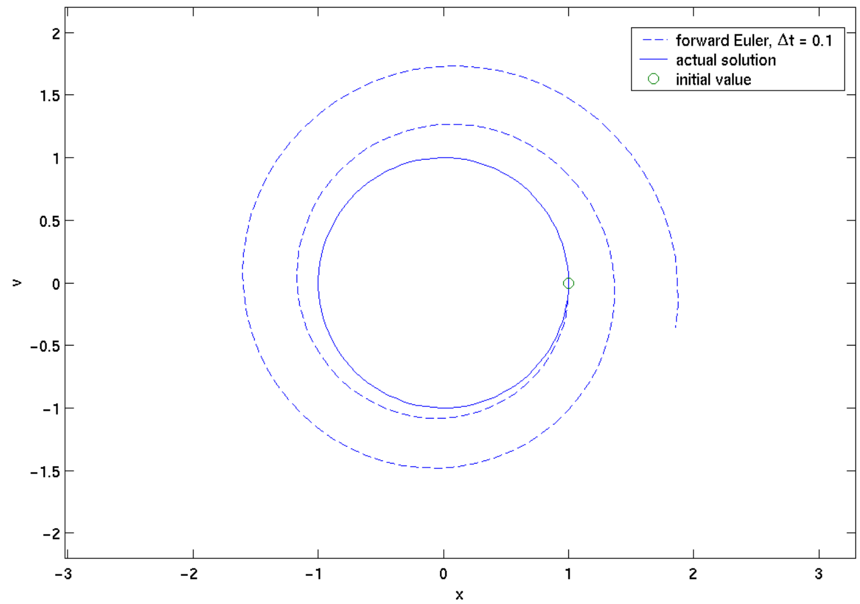
a. To solve this equation using Euler's method, we can write this system as two *first order* differential equations using phase space as described in lecture. Do this by making the substitution $\dot{x} = v$ (where *v* is the velocity of the particle) and fill in the following equation:

$$\begin{bmatrix} \dot{x}(t) \\ \dot{v}(t) \end{bmatrix} =$$

Give the initial conditions:

$$\begin{bmatrix} x(0) \\ v(0) \end{bmatrix} =$$

Particle Systems (cont'd)



x vs v plot.

b. Write down the formula for one iteration of Euler's method to solve the system that you gave in part a, given some Δt. If $\Delta t = 1/10$, then what are the results of the first *two* iterations of Euler's method? That is, what is the position and velocity of the particle at time $t = 0.1$ and $t = 0.2$? Show your work. If you wish, check your solution against the above graph, which plots the computed solution on the xv phase plane.

Particle Systems (cont'd)

c. With the given step size 0.1, you can see from both of the plots that the computed solution diverges from the actual solution. An interesting fact is that, for *any* positive step size, Euler's method will give an increasingly inaccurate solution as t goes to infinity. We will now show why this is true. Euler's method can also be written as follows.

$$\begin{bmatrix} x_{i+1} \\ v_{i+1} \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ -\Delta t & 1 \end{bmatrix} \begin{bmatrix} x_i \\ v_i \end{bmatrix}$$

This matrix can be written as **RS**, where **R** is a rotation matrix and **S** is a uniform scale:

$$\begin{bmatrix} 1 & \Delta t \\ -\Delta t & 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} s & 0 \\ 0 & s \end{bmatrix}$$

Find s and θ in terms of Δt (Hint: solve for s first by equating the length of the first column of **RS** with the length of the first column of the matrix on left. Then solve for θ by equating one of the entries of the matrices.). ***Show your work.***

What are s and θ when the step size is 0.1?

Particle Systems (cont'd)

d. Use your results from part c to explain why Euler's method eventually explodes for any $\Delta t > 0$. Why does the particle move in an increasingly large spiral in the *xv* phase plane for any $\Delta t > 0$ as shown in the plot? Specifically, why does the linear transformation **RS** create such a spiral? What does this tell us about the size of the spring oscillations?

e. Extra Credit: You've just shown that Euler's method is not stable for solving the undamped spring equation. Congratulations! Unfortunately, that won't help you at all if you actually want to implement a simulation of a particle on a spring. What could you do to make your simulation stable?