Homework #2

Shading, Ray Tracing, Texture Mapping, and Parametric Curves

Due: Wednesday, November 23rd
at the beginning of class

Directions: Please provide short written answers to the following questions, using this page as a cover sheet. Be sure to justify your answers when requested. Feel free to discuss the problems with classmates, but please ***answer the questions on your own***.

Be sure to write your name on your homework solution.
You may (optionally) use this page as a cover sheet.

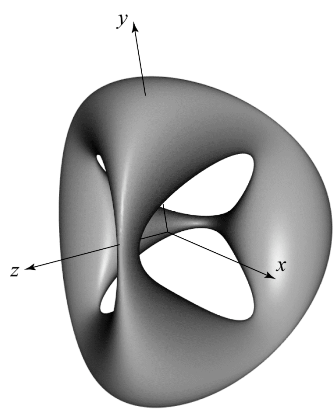
Name:____

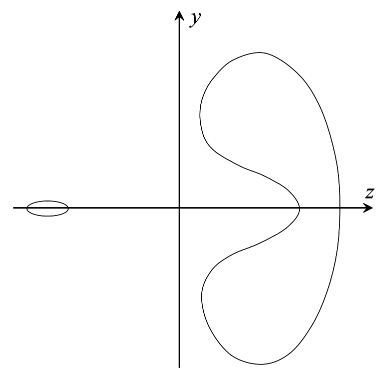
Problem 1. Ray intersection with implicit surfaces (25 points)

There are many ways to represent a surface. One 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 we wanted to ray trace a "quartic chair," described by the equation:

$$(x^2+y^2+z^2-ak^2)^2-b\left[(z-k)^2-2x^2\right]\left[(z+k)^2-2y^2\right]=0$$

On the left is a picture of a quartic chair, and on the right is a slice through the *y-z* plane.





For this problem, we will assume a = 0.95, b = 0.8, and k = 5.

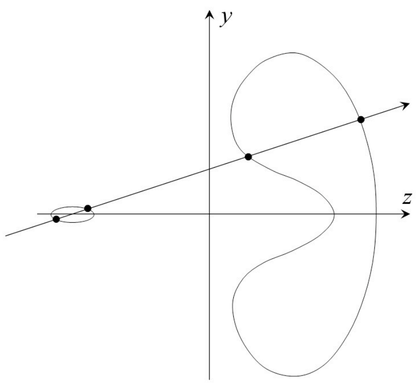
In the next problem steps, you will be asked to solve for and/or discuss ray intersections with this primitive. Performing the ray intersections will amount to solving for the roots of a polynomial, much as it did for sphere intersection. For your answers, you need to keep a few things in mind:

- You will find as many roots as the order (largest exponent) of the polynomial.
- You may find a mixture of real and complex roots. When we say complex here, we mean a number that has a non-zero imaginary component.
- All complex roots occur in complex conjugate pairs. If $A + iB$ is a root, then so is $A - iB$.
- Sometimes a real root will appear more than once, i.e., has multiplicity > 1. Consider the case of sphere intersection, which we solve by computing the roots of a quadratic equation. A ray that intersects the sphere will usually have two distinct roots (each has multiplicity = 1) where the ray enters and leaves the sphere. If we were to take such a ray and translate it away from the center of the sphere, those roots get closer and closer together, until they merge into one root. They merge when the ray is tangent to the sphere. The result is one distinct real root with multiplicity = 2.

a) (10 points) Consider the ray $P+t\mathbf{d}$, where $P=\begin{pmatrix}0 & 0 & 0\end{pmatrix}$ and $\mathbf{d}=\begin{pmatrix}0 & 0 & 1\end{pmatrix}$. Solve for all values of *t* where the ray intersects the quartic chair (including negative values of *t*). Which value of *t* represents the intersection we care about for ray tracing? In the process of solving for t, you will be computing the roots of a polynomial. How many distinct real roots do you find? How many of them have multiplicity > 1? How many complex roots do you find?

Problem 1 (cont'd)

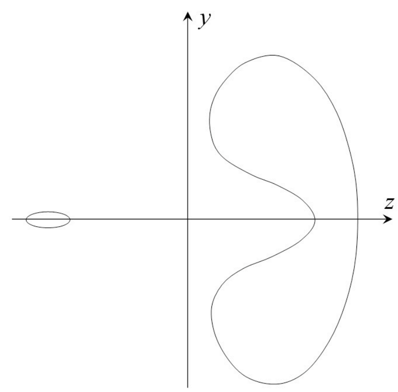
b) (15 points) What are all the possible combinations of roots, not counting the one in part (a)? For each combination, describe the 4 roots as in part (a), draw a ray in the *y-z* plane that gives rise to that combination, and place a dot at each intersection point. There are five diagrams below that have not been filled in. You may not need all five; on the other hand, if you can actually think of more distinct cases than spaces provided, then we might just give extra credit. The first one has already been filled in. (Note: not all conceivable combinations can be achieved on this particular implicit surface. For example, there is no ray that will give a root with multiplicity 4.) ***Please write on this page and include it with your homework solution. You do not need to justify your answers.***



\# of distinct real roots: **4**

\# of real roots w/ multiplicity > 1: **0**

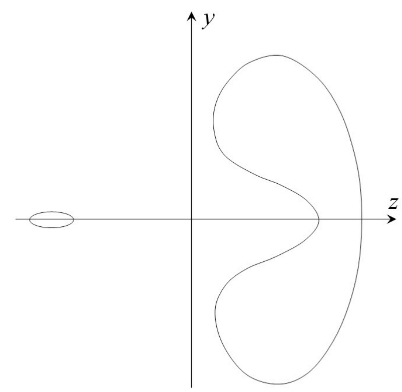
\# of complex roots: **0**



\# of distinct real roots:

\# of real roots w/ multiplicity > 1:

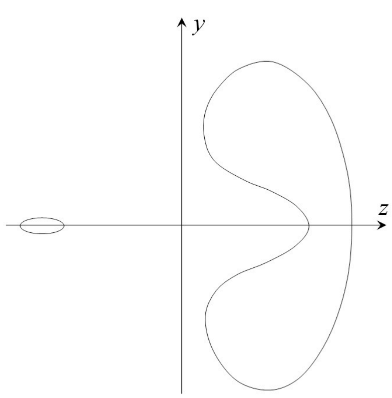
\# of complex roots:



\# of distinct real roots:

\# of real roots w/ multiplicity > 1:

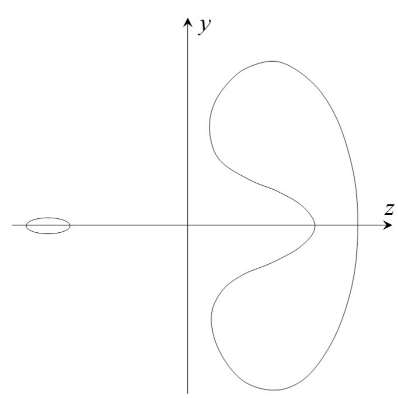
\# of complex roots:



\# of distinct real roots:

\# of real roots w/ multiplicity > 1:

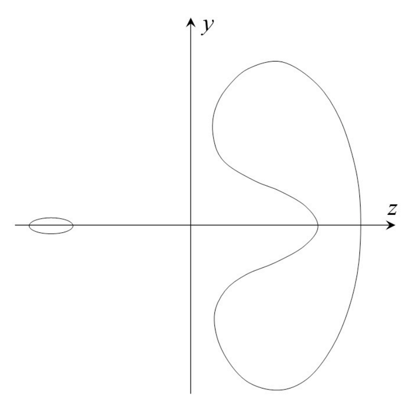
\# of complex roots:



\# of distinct real roots:

\# of real roots w/ multiplicity > 1:

\# of complex roots:



\# of distinct real roots:

\# of real roots w/ multiplicity > 1:

\# of complex roots:

Problem 2. Counting rays (25 points)

In this problem, we study the number of rays traced for using different ray tracing algorithms. Consider the following setup:

m x m pixels
k x k supersampling
n geometric primitives
ℓ light sources
d bounces (reflections and/or refractions)

For each of the algorithms and scenarios discussed in parts (a)-(e) below, assume the following:

- You are counting rays cast, including primary rays, shadow (light) rays, reflected rays, and (when asked for in the problem) refracted rays.
- *No* acceleration techniques are used.
- *Every* recursively traced (reflected or refracted) ray hits an object, including the primary rays.
- You will always cast a ray to the light source after intersecting an object, and this does not count as a recursive "bounce" (but certainly counts as a cast ray).
- Each ray cast to a light source counts as a single ray-cast, even when accounting for transparent shadows. (The transparent shadow case can be handled by keeping track of all intersections encountered – not just the closest – when casting a ray to a light, so this is a reasonable assumption.)

You do not need to justify your answers, though doing so may help you to earn partial credit. For each sub-problem, write out a summation (with the Σ symbol for the summation) and then, if possible, convert it to closed form.

a) (5 points) For Whitted ray tracing, assuming reflection (but *no* refraction) at every surface, how many rays are cast?

b) (5 points) For Whitted ray tracing, assuming reflection *and* refraction at every surface, how many rays are cast?

c) (5 points) Suppose now, in order to get glossy reflections, you recursively cast k x k rays around the reflection direction at each bounce. Assuming glossy reflection (but *no* refraction) at every surface, how many rays are cast?

d) (5 points) In addition, in order to get translucent (blurry) refraction effects, you recursively cast k x k rays around the refraction direction at each bounce. Assuming glossy reflection and translucent refraction at every surface, how many rays are cast?

e) (5 points) Suppose now you switch to using distribution ray tracing. Assuming glossy reflection and translucent refraction at every surface, how many rays are cast?

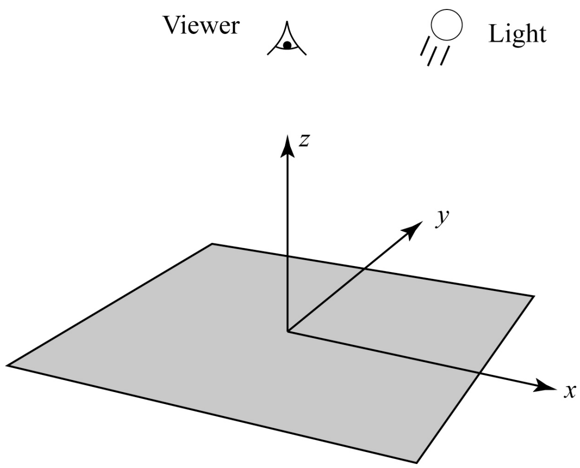
Problem 3. Shading, displacement mapping, and normal mapping (34 points)

In this problem, an opaque surface will be illuminated by one directional light source and will reflect light according to the following Phong shading equation:

$$I = A_{shadow} L \left(k_d \left(\mathbf{N} \cdot \mathbf{L} \right)_+ + k_s B \left(\mathbf{V} \cdot \mathbf{R} \right)_+^{n_s} \right)$$

Note the inclusion of a shadowing term, which takes on a value of 0 or 1. For simplicity, we will assume a monochrome world where I, L, k_d, and k_s are scalar values.

Suppose a viewer is looking down at an infinite plane (the x-y plane) as illustrated below. The scene is illuminated by a directional light source, also pointing straight down on the scene.



Answer the following questions below, giving brief justifications of each answer. Note that lighting and viewing directions are from the point of view of the light and viewer, respectively, and need to be negated when considering the surface-centric shading equation above. [In general, you don't need to solve equations and precisely plot functions. It is enough to describe the variables involved, how they relate to each other, and how this relationship will determine, e.g., the appearance of the surface. If you're more comfortable making the answers analytical with equations and plots, however, you are welcome to do so.]

a) (2 points) Assume: Perspective viewer at (0,0,1) looking in the (0,0,-1) direction, angular field of view of 90 degrees, lighting direction of (0,0,-1), $k_d = 0.5$, $k_s = 0$. Describe the brightness variation over the image seen by the viewer. Justify your answer.

b) (2 points) Assume: Perspective viewer at (0,0,1) looking in the (0,0,-1) direction, , angular field of view of 90 degrees, lighting direction of (0,0,-1), $k_d = 0.5$, $k_s = 0.5$, $n_s = 10$. Describe the brightness variation over the image seen by the viewer. Justify your answer.

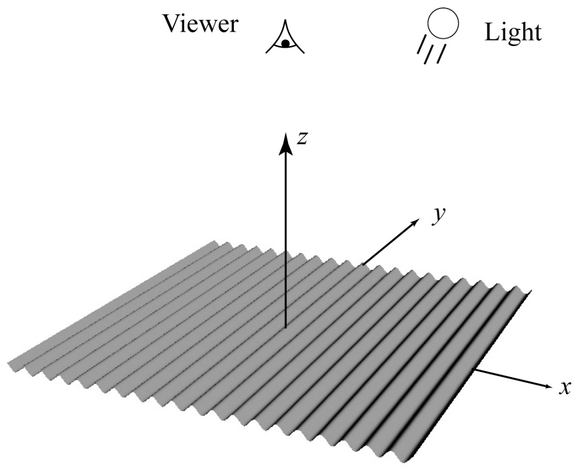
c) (2 points) Assume: Orthographic viewer looking in the (0,0,-1) direction, lighting direction of (0,0,-1), $k_d = 0.5$, $k_s = 0.5$, $n_s = 10$. Describe the brightness variation over the image seen by the viewer. Justify your answer.

d) (3 points) Assume: Orthographic viewer looking in the (0,0,-1) direction, $k_d = 0.5$, $k_s = 0$. The lighting direction starts at (-sqrt(2)/2 ,0, -sqrt(2)/2) and then rotates around the z-axis. Describe the brightness variation over time, as seen by the viewer. Justify your answer.

Problem 3. (cont'd)

e) (3 points) Assume: Orthographic viewer looking in the (0,0,-1) direction, k_d = 0.5, k_s = 0.5, n_s = 10. The lighting direction starts at (-sqrt(2)/2 ,0, -sqrt(2)/2) and then rotates around the z-axis. Describe the brightness variation over time, as seen by the viewer. Justify your answer.

Suppose now the infinite plane is replaced with a surface $z = \cos(x)$:



We can think of this as simply adding a displacement d=cos(x) in the normal direction to the x-y plane.

f) (6 points) Assume: Orthographic viewer looking in the (0,0,-1) direction, lighting direction of (0,0,-1), k_d = 0.5, k_s = 0. At what values of x is the surface brightest? At what values is it dimmest? Describe the appearance of the surface. Justify your answers.

g) (6 points) Assume: Orthographic viewer looking in the (0,0,-1) direction, lighting direction of (0,0,-1), k_d =0, k_s =0.5, n_s =10. At what values of x is the surface brightest? Describe the appearance of the surface. How does the appearance change as n_s increases to 100? Justify your answers.

Suppose now that we simply keep the normals used in (f)-(g) and map them over the plane from the first part of the problem. The geometry will be flat, but the shading will be based on the varying normals.

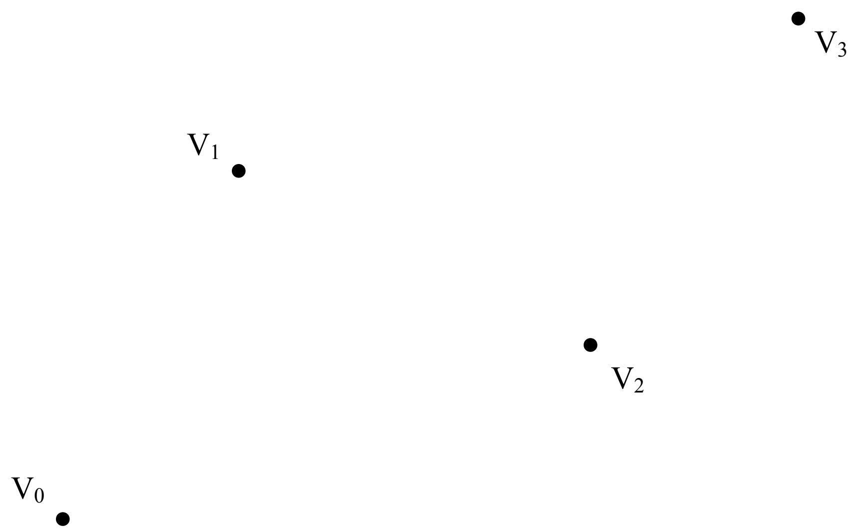
h) (6 points) Assume: Orthographic viewer looking in the (0,0,-1) direction, k_d =0.5, k_s =0. If we define the lighting to have direction (-sinθ, 0, -cosθ), will the normal mapped rendering look the same as the displacement mapped rendering for each of θ = 0, 10, and 80 degrees? Justify your answer.

i) (4 points) Assume: Orthographic viewer, lighting direction of (0,0,-1), k_d =0.5, k_s =0. As we generally move the viewer around – rotating it to various viewing direcitons – will the normal mapped rendering look the same as the displacement mapped rendering? Justify your answer.

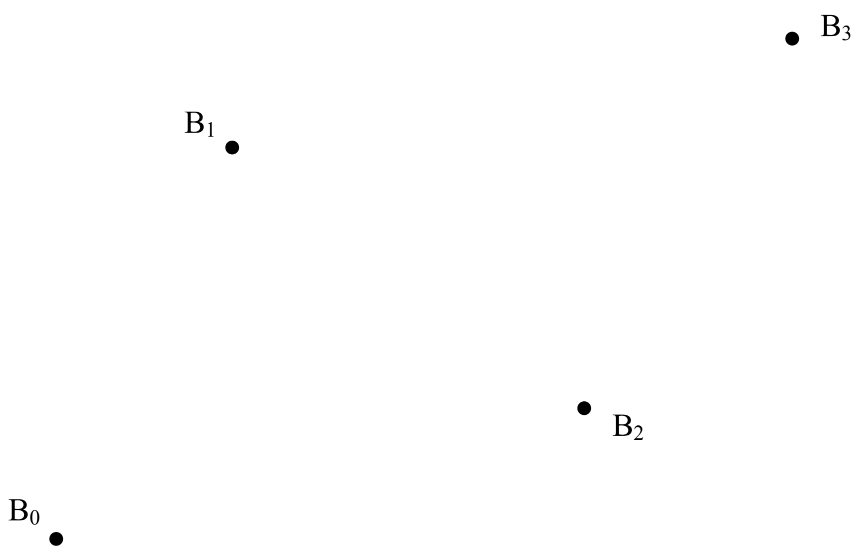
Problem 4. Parametric curves (16 points)

In this problem, we will explore the construction of parametric. ***Please write on the pages for this problem and include them with your homework solution.***

a) (4 points) Given the following Bezier control points, construct all of the de Casteljau lines and points needed to evaluate the curve at $u=1/3$. Mark this point on your diagram and then sketch the path the Bezier curve will take. The curve does not need to be exact, but it should conform to some of the geometric properties of Bezier curves (convex hull condition, tangency at endpoints).



b) (6 points) Given the following de Boor points, construct all of the lines and points needed to generate the Bezier control points for the B-spline. Assume that the first and last points are each repeated three times, so that the spline is endpoint interpolating. You must mark each Bezier point (*including any that coincide with a de Boor point*) with an X, but you do not need to label it (i.e., no need to give each Bezier point a name). Sketch the resulting spline curve, respecting the properties of Bezier curves noted above.



Problem 4 (cont'd)

c) (6 points) Given the following Catmull-Rom control points, construct all of the lines and points needed to generate the Bezier control points for the Catmull-Rom curve. Use a tension value of $\tau=1$. Assume that the first and last points are each repeated two times. You must mark each Bezier point (*including any that coincide with a Catmull-Rom control point*) with an X, but you do not need to label it (i.e., no need to give each Bezier point a name). Sketch the resulting spline curve, respecting the properties of Bezier curves noted above.