

## Computer Graphics 50:198:456/56:198:556 (Spring 2009)

<b>Homework:</b> 6	<b>Professor:</b> Suneeta Ramaswami
<b>Due Date:</b> 4/29/08	<b>E-mail:</b> rsuneeta@camden.rutgers.edu
<b>Office:</b> 321 BSB	<b>URL:</b> <a href="http://crab.rutgers.edu/~rsuneeta">http://crab.rutgers.edu/~rsuneeta</a>
	<b>Phone:</b> (856)-225-6439

---

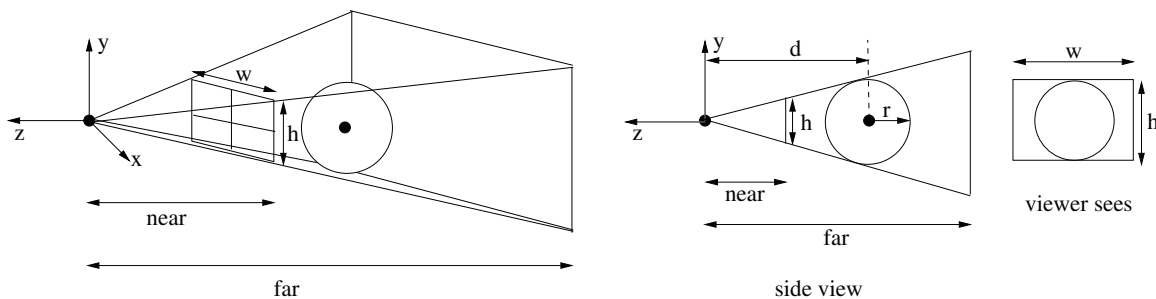
### Written Assignment #2

There are 6 questions in all. Undergraduate students are required to answer questions 1-4 ( $2 \times 10 + 2 \times 15$  points). Graduate students are required to answer all 6 questions ( $2 \times 5 + 4 \times 10$  points). Please show all your work; answers without justification will not receive any credit. Turn in a legible copy of your answers **on or before April 29, 2009**.

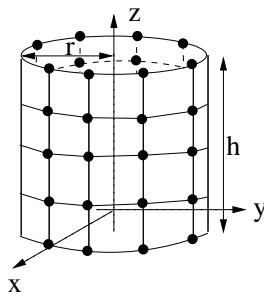
1. In class, we derived the perspective transformation matrix used by `OpenGL` for the command `glFrustum(1,r,b,t,n,f)`. Using that, derive the transformation matrix for `gluPerspective(v,a,n,f)`. In other words, specify the perspective transformation matrix in terms of  $v, a, n$  and  $f$ .
2. Suppose you want to render a luminous blue sphere, which is tinged with red on the east (without any highlights) due to an appropriate light source on the positive  $x$ -axis. The red fades or gets brighter as the light source is moved rightwards or leftwards, respectively. Write the `myinit` routine giving the necessary lighting parameters to achieve this. For full credit, submit a complete working program.
3. Suppose you want to render a scene consisting of a unit sphere centered at the origin and one moving light source, `LIGHT0`. The sphere has a diffuse reflectance of  $(1.0, 1.0, 1.0)$ . `LIGHT0` is initially at  $(2.0, 0.0, 0.0)$  and has a red diffuse component. Assume the default values for all other lighting parameters. The viewer is at  $(5.0, 5.0, 5.0)$  and looking at the origin, with the “up” direction being the positive  $y$  axis. When the left mouse button is clicked, `LIGHT0` is rotated counterclockwise about the  $y$ -axis by one degree. Write the relevant `init`, `display` and `mouse` routines to render the scene. For full credit, submit a complete working program.
4. Consider a line (segment) in 3D going from the world-coordinate points  $P_1(6, 10, -3)$  to  $P_2(-3, -5, -2)$  and a semi-infinite viewing pyramid in the region  $-z \leq x \leq z, -z \leq y \leq z$ , which is bounded by the planes  $z = +x, z = -x, z = +y, z = -y$ . The projection plane is at  $z = -1$ .
  - (a) Clip the line segment in 3D (using parametric line equations). What are the endpoints of the clipped segment?
  - (b) What are the endpoints of the clipped segment when it is projected onto the projection plane?
5. On the distant planet of Omicron Persei 8, they prefer a different way of specifying the perspective transformation. As with `gluPerspective`, they give the distances to the near and far clipping planes and the window’s aspect ratio,  $a = w/h$ . However, rather than giving the  $y$ -field of view, they instead give the distance  $d$  to a sphere of a given radius  $r$  that is centered along the viewing direction. The  $y$ -field of view is set so that the sphere exactly fills

the window's height. (See the figure below.) Write a procedure which, given these parameters, produces an equivalent call to `gluPerspective`. You may assume that the window is wider than tall, that is,  $a \geq 1$ . Here are the function prototypes.

```
void op8Perspective(double d, double r, double aspect, double near, double far)
void gluPerspective(double fovy, double aspect, double near, double far)
```



6. Recall that on the first written assignment, you were asked to write a procedure to generate a rendering of a cylinder in OpenGL. The cylinder is centered along the  $z$ -axis, has a height of  $h$  units, and has a radius of  $r$  units. Because OpenGL can only display polygons, you are to split the cylinder into  $v_s$  vertical stacks (along the  $z$ -axis), and  $r_s$  radial slices (around the  $z$ -axis). For example, in the figure below,  $v_s = 4$  and  $r_s = 8$ . Each face is drawn as a `GL_POLYGON`.



We have already seen how to draw the cylinder in OpenGL in the solutions to the first written assignment. You are now asked to wrap a texture around the cylinder. The texture image is  $256 \times 256$  pixels. The texture should be mapped using `GL_REPEAT`, so that exactly four copies of the image go all the way around the cylinder. Explain how to modify the above procedure to provide the proper texture coordinates for each vertex. (You do not need to give any of the other OpenGL texture commands.)