

DEPARTMENT OF COMPUTER SCIENCE  
CONCORDIA UNIVERSITY  
COMP 471/2 Fall 2008  
Brief Solution to Mid-term Examination

The following materials are allowed:  
One Computer Graphics textbook  
Notes up to one inch in thickness  
simple calculator

Time allowed: 1 hour  
Total Marks: 20 (20% of the final grade)

1. (6 marks) In this question, you have to calculate where a vertex will be displayed on the window. The answer should be given in window's coordinate, with the origin at the bottom left corner, and rounded to the nearest pixel.

Assume that the following calls have been made in the main program:

```
glutInitWindowSize(500,500);  
glutMatrixMode(GL_PROJECTION);  
glLoadIdentity();  
glFrustum(-2.5, 2.5, -2.5, 2.5, 4.0, 10.0);  
glMatrixMode(GL_MODELVIEW);  
glLoadIdentity();  
gluLookAt(0.0, 0.0, 5.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);
```

Where will the following two vertices be displayed?

- (a) (3 marks)

```
void display(){  
    glVertex3f(1.0, 0.0, 0.0);  
}
```

Answer: In eye coordinate, the vertex is at  $z = -5$ , and the near clipping plane is at  $z = -4$ . The  $x$  coordinate of the projection of the vertex on the near plane is given by  $x = 1.0 \times (4/5) = 0.8$ , and the  $y$  coordinate stays at 0. The near plane is of size  $5 \times 5$ . The window is of size  $500 \times 500$ . Thus, the point  $(x, y) = (0, 0)$  on the near plane is scaled to the point  $(250, 250)$  in window's coordinate, where the origin is the bottom left corner. The vertex, when projected on the near plane, has  $x = 0.8$ . The distance of 0.8 in the near plane scales to  $0.8 \times (500/5.0) = 80$  pixels in the window. Thus the vertex is drawn at  $(250, 250 + 80) = (250, 330)$ .

- (b) (3 marks)

```
void display(){  
    glMatrixMode(GL_MODELVIEW);
```

```

    glPushMatrix();
    glRotatef(90, 0.0, 0.0, 1.0);
    glVertex3f(1.0, 0.0, 0.0);
    glPopMatrix();
}

```

Answer: The extra rotation rotates 90 degrees anti-clockwise about the  $z$ -axis. It changes the vertex (1.0, 0.0, 0.0) to (0.0, 1.0, 0.0). Following a similar calculation as in part (a), this vertex projects to  $(x, y) = (0.0, 0.8)$  on the near plane. When scaled to the window, this vertex is at (330, 250).

2. (7 marks) This question concerns the `planet.c` program, a copy of which is attached at the end of this test.

Modify the program so that you also draw a moon revolving in a circular orbit around the small planet. The moon has a radius of 0.04, and its orbit has a radius of 1. The center of the moon's orbit is at the center of the small planet.

Your modification should be specified as a sequence of deletions and/or insertions in the program. Insertions should be specified as C/C++ codes.

Answer: The revised version of the `planet.c` program is attached at the end.

A new variable, `month`, keeps track of the position of the moon rotating around the planet. In order to better see the interaction of the moon rotation relative to those of the sun and the planet, I decide to let the moon rotate about the  $x$ -axis through the center of the planet.

Note that the moon has to follow the planet around the sun.

Note also that the rotation of the planet about its own  $y$ -axis (controlled by the `day` variable) should not affect the moon.

3. (7 marks) This question also concerns the `planet.c` program, a copy of which is attached at the end of this test.

Modify the original version of the program so that you can control the distance of the viewpoint by using the 'f' and 'b' keys. The 'f' key moves the viewpoint closer to the origin, and the 'b' key moves it away.

Your modification should be specified as a sequence of deletions and/or insertions in the program. Insertions should be specified as C/C++ codes.

Answer: The revised program is attached below.

A new variable `viewDist` is added to remember how far the camera has moved back along the  $z$ -axis, from the world coordinate's origin.

Note that after updating the `viewDist` in the keyboard function, a new `gluLookAt` has to be issued in order for the new camera location to take effect.

Note also that the `gluLookAt` call in the `reshape` function has to be changed to reflect the addition of the `viewDist` variable.

```

/*
 * planet.c
 * Modified to test answers for Q2 and Q3 of mid-term.
 * This program shows how to composite modeling transformations
 * to draw translated and rotated models.
 * Interaction: pressing the d and y keys (day and year)
 * alters the rotation of the planet around the sun.
 */
#include <GL/glut.h>
#include <stdlib.h>

static int year = 0, day = 0, month=0; /* month added */
      GLfloat viewDist = 5.0; /* viewDist added */

void init(void)
{
    glClearColor (0.0, 0.0, 0.0, 0.0);
    glShadeModel (GL_FLAT);
}

void display(void)
{
    glClear (GL_COLOR_BUFFER_BIT);
    glColor3f (1.0, 1.0, 1.0);

    glPushMatrix();
    glutWireSphere(1.0, 20, 16); /* draw sun */
    glRotatef ((GLfloat) year, 0.0, 1.0, 0.0);
    glTranslatef (2.0, 0.0, 0.0);
    glPushMatrix();
    glRotatef ((GLfloat) day, 0.0, 1.0, 0.0);
    glutWireSphere(0.2, 10, 8); /* draw smaller planet */
    glPopMatrix();
    /* next three lines added to draw a revolving moon*/
    glRotatef ((GLfloat) month, 1.0, 0.0, 0.0);
    glTranslatef(0.0, 1.0, 0.0);
    glutWireSphere(0.04, 8, 6);
    /* end of added part */
    glPopMatrix();
    glutSwapBuffers();
}

void reshape (int w, int h)
{

```

```

glViewport (0, 0, (GLsizei) w, (GLsizei) h);
glMatrixMode (GL_PROJECTION);
glLoadIdentity ();
gluPerspective(60.0, (GLfloat) w/(GLfloat) h, 1.0, 20.0);
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
gluLookAt (0.0, 0.0, viewDist, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0); /* modified */
}

```

```

void keyboard (unsigned char key, int x, int y)
{
    switch (key) {
        case 'd':
            day = (day + 10) % 360;
            glutPostRedisplay();
            break;
        case 'D':
            day = (day - 10) % 360;
            glutPostRedisplay();
            break;
        case 'y':
            year = (year + 5) % 360;
            glutPostRedisplay();
            break;
        case 'Y':
            year = (year - 5) % 360;
            glutPostRedisplay();
            break;
        /* keyboard control of moon added */
        case 'm':
            month = (month + 10) % 360;
            glutPostRedisplay();
            break;
        case 'M':
            month = (month - 10) % 360;
            glutPostRedisplay();
            break;
        /* 'f' and 'b' control added */
        case 'f':
            if (viewDist > 1.2) viewDist -= 0.1;
            glLoadIdentity();
            gluLookAt(0.0, 0.0, viewDist, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);
            glutPostRedisplay();
            break;
    }
}

```

```

    case 'b':
viewDist += 0.1;
glLoadIdentity();
gluLookAt(0.0, 0.0, viewDist, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);
glutPostRedisplay();
break;
    case 27:
        exit(0);
        break;
    default:
        break;
}
}

int main(int argc, char** argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode (GLUT_DOUBLE | GLUT_RGB);
    glutInitWindowSize (500, 500);
    glutInitWindowPosition (100, 100);
    glutCreateWindow (argv[0]);
    init ();
    glutDisplayFunc(display);
    glutReshapeFunc(reshape);
    glutKeyboardFunc(keyboard);
    glutMainLoop();
    return 0;
}

```