Tp OpenGL Louis Thevenet & Guillaume Sablayrolles

3.2.2 Display

1. A solid teapot instead of a wire one. It seems to be missing lighting and texture.
2. As we add slices and stacks, the resolution of the sphere improves.

3.2.3 Viewport

1.

```
// Function called every time the main window is resized
void reshape(int width, int height)
{
    // define the viewport transformation;
    int minwh = std::min(width, height);
    int maxwh = std::max(width, height);

    int offset = maxwh - minwh;
    int offset_x = (width > height) ? offset : 0;
    int offset_y = (width < height) ? offset : 0;

    glViewport(offset_x/2, offset_y/2, minwh , minwh );
}
```

3.2.4 Keyboard

```
bool wire = false;

// function called everytime the windows is refreshed
void display()
{
    // clear window
    glClear(GL_COLOR_BUFFER_BIT);

    // draw scene
    if (wire) {
        glutSolidTeapot(0.7);

    }
    else {
        glutWireTeapot(0.7);
    }

    // flush drawing routines to the window
    glFlush();
}



// Function called everytime a key is pressed
void key(unsigned char key, int, int)
{
    switch(key)
    {
        case 'w':
            wire=!wire;
            break;
        // the 'esc' key
        case 27:
        // the 'q' key
        case 'q': exit(EXIT_SUCCESS); break;
        default: break;
    }
    glutPostRedisplay();
}
```

4.2

1. When we decrease `fovy`, the object takes more space on the screen since the field of view is narrower.
2. When `zNear` is increased, if parts of the objet escape the clipping volume, it will not be rendered anymore.

4.2.2

1. Setting the up vector to (0, -1, 0) makes the camera look at the teapot upside down.
2.

(a)

```
gluLookAt(
        0.0, 1.2,0.0,
        0.0,0.0,0.0,
        0.0001,-1.0,0.0
```

```
    );
```

(b)

```
gluLookAt(
        0.0, 0.0,-1.7,
        0.0,0.0,0.0,
        0.5,0.5,1.0
    );
```

4.3.2

1. When the second `glPopMatrix` is removed, as soon as the window is updated and a new frame is rendered, we see that the objects are not rendered in the same place as before. This is due to the stack not being restored at the return of the `display` function. The current matrix is the previous one so the operations are being accumulated.
2. Since we are making operations on state variables (matrix on the stack), if we set the camera after the second `glPushMatrix`, it will only be affected by the operations perfomed at its scope.
3. In this case, the current matrix gets set to the indentity matrix, and the transformations performed on the teapots don't start from the camera's position anymore, the scene we see is different.

4.4

```
#define SPEED 0.1
#define ANG_SPEED 0.5

void move_camera(double x, double y, double z) {
    glMatrixMode(GL_PROJECTION);
    glTranslatef(x, y, z);

}
void rotate_camera(double x, double y, double z) {
    glMatrixMode(GL_PROJECTION);
    glRotatef(ANG_SPEED, x, y, z);
}

void arrow_keys(int key, int, int) {
    switch(key) {
        case GLUT_KEY_LEFT:
            rotate_camera(0.0, -1.0, 0.0);
            break;
        case GLUT_KEY_RIGHT:
            rotate_camera(0.0, 1.0, 0.0);
            break;

        case GLUT_KEY_UP:
            rotate_camera(-1.0, 0.0, 0.0);
            break;

        case GLUT_KEY_DOWN:
            rotate_camera(1.0, 0.0, 0.0);
            break;

        default:
            break;
    }
    glutPostRedisplay();

}
// Function called everytime a key is pressed
void key(unsigned char key, int, int)
{
    switch(key)
    {
        case 27:
            exit(EXIT_SUCCESS);
            break;

        case 'z':
            move_camera(.0, .0, SPEED);
            break;

        case 's':
            move_camera(0.,0., -SPEED);
            break;

        case 'q':
            move_camera(SPEED,0., 0.);
            break;

        case 'd':
            move_camera(-SPEED,0., 0.);
            break;
        case 'a':
            move_camera(0.,-SPEED, 0.);
            break;
        case 'w':
            move_camera(0.,SPEED, 0.);
            break;

        default:
            break;
    }
    glutPostRedisplay();
}

int main(int argc, char* argv[])
{
    // [...]
    // for the arrows
    glutSpecialFunc(arrow_keys);
    // [...]
}
```