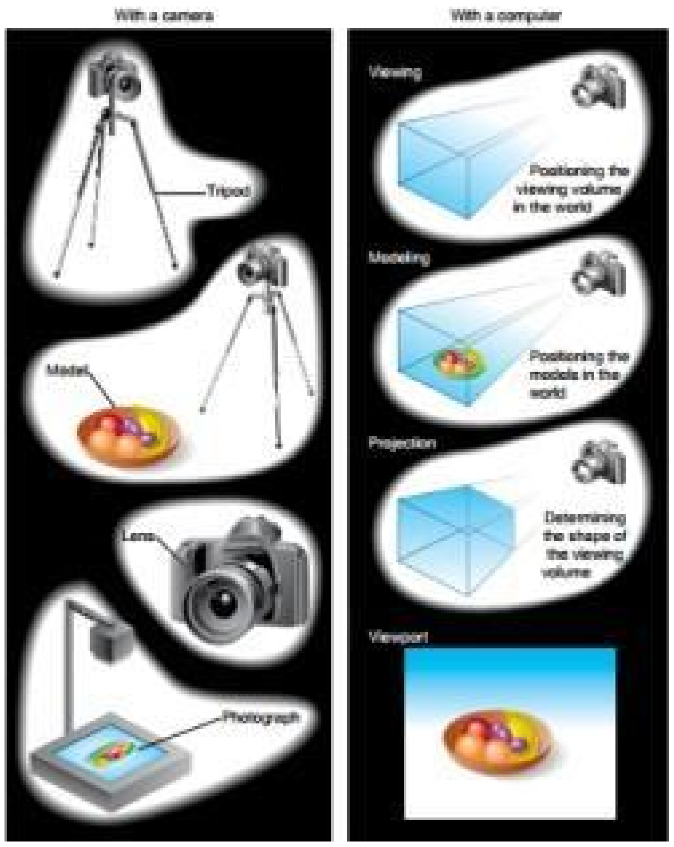
IS2107 - Graphics and Visualization

2024.07.04

Practical 05

Viewing and Transformations in OpenGL

Overview


With a camera
With a computer
Tripod
Model
Lens
Photograph
Viewing
Positioning the viewing volume in the world
Modeling
Positioning the models in the world
Projection
Determining the shape of the viewing volume
Viewport

Modeling Transformations

1. Translate :

```
void glTranslate{fd}(TYPE x, TYPE y, TYPE z);
```

Multiplies the current matrix by a matrix that moves (translates) an object by the given x-, y-, and z-values (or moves the local coordinate system by the same amounts).

2. Rotate :

```
void glRotate{fd}(TYPE angle, TYPE x, TYPE y, TYPE z);
```

Multiplies the current matrix by a matrix that rotates an object (or the local coordinate system) in a counterclockwise direction about the ray from the origin through the point (x, y, z). The angle parameter specifies the angle of rotation in degrees.

3. Scale :

```
void glScale{fd}(TYPE x, TYPE y, TYPE z);
```

Multiplies the current matrix by a matrix that stretches, shrinks, or reflects an object along the axes. Each x-, y-, and z-coordinate of every point in the object is multiplied by the corresponding argument x, y, or z. With the local coordinate system approach, the local coordinate axes are stretched, shrunk, or reflected by the x-, y-, and z-factors, and the associated object is transformed with them.

V iewing Transformations

```
void gluLookAt(GLdouble eyex, GLdouble eyey, GLdouble eyez,
GLdouble centerx, GLdouble centery, GLdouble centerz,
GLdouble upx, GLdouble upy, GLdouble upz);
```

Defines a viewing matrix and multiplies it to the right of the current matrix. The desired viewpoint is specified by eyex, eyey, and eyez. The centerx, centery, and centerz arguments specify any point along the desired line of sight, but typically they specify some point in the centre of the scene being looked at. The upx, upy, and upz arguments indicate which direction is up (that is, the direction from the bottom to the top of the viewing volume).

Projection Transformations

1. Perspective Projection:

```
void glFrustum(GLdouble left, GLdouble right, GLdouble
bottom, GLdouble top,GLdouble near, GLdouble far);
```

Creates a matrix for a perspective-view frustum and multiplies the current matrix by it. The frustum's viewing volume is defined by the parameters: (left, bottom, –near) and (right, top, –near) specify the (x, y, z) coordinates of the lower left and upper right corners, respectively, of the near clipping plane; near and far give the distances from the viewpoint to the near and far clipping planes. They should always be positive.

2. Orthographic Projection :

```
void glOrtho(GLdouble left, GLdouble right, GLdouble bottom,
GLdouble top, GLdouble near, GLdouble far);
```

Creates a matrix for an orthographic parallel viewing volume and multiplies the current matrix by it. (left, bottom, –near) and (right, top, –near) are points on the near clipping plane that are mapped to the lower left and upper right corners of the viewport window, respectively. (left, bottom, –far) and (right, top, –far) are points on the far clipping plane that are mapped to the same respective corners of the viewport. Both near and far may be positive, negative, or even set to zero. However, near and far should not be the same value.

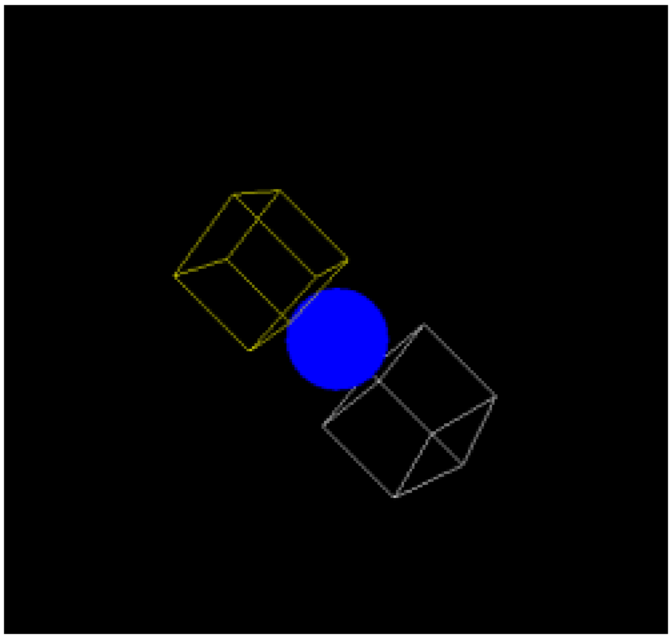
V iewport Transformation

```
void glViewport(GLint x, GLint y, GLsizei width, GLsizei
height);
```

Defines a pixel rectangle in the window into which the final image is mapped. The (x, y) parameter specifies the lower left corner of the viewport, and width and height are the size of the viewport rectangle. By default, the initial viewport values are (0, 0, winWidth, winHeight), where winWidth and winHeight specify the size of the window

Activity

1. Create two WireCubes and a SolidSphere (You can use any colour and size as you prefer).
2. Apply necessary transformations to render a final scene as given below.



3. Create a zip folder containing all of your executable code and final output image.
4. Rename it as <your_index>.zip and upload it to the VLE.