SESSION 2018/2019 SEMESTER 2

LAB 4

NAME: TAN SEE JOU (A17CS0218)

LECTURER NAME: DR NORHAIDA BTE MOHD SUAIB

SUBJECT NAME: FUNDAMENTAL OF COMPUTER GRAPHICS

SUBJECT CODE: SCSV2213

SECTION: 01

Introduction

Lab 4 is focus on hierarchical modelling. Hierarchical modelling is a way of composing complex objects out of multiple simpler primitives. That is a complex object can be made up of several simpler objects, which turn out be made up of even simpler objects, and so on until the simple geometric primitive that can be draw directly and easily.

Lab Requirement

At first, we need to run and study the codes provided for hierarchical modelling in OpenGL (planet and sun AND robot arm). Edit and produce our own hierarchical model.

Lab Findings

From Lab4, we found out that the transformations are specified in the opposite order from the order in which they are applied to the object and the transformation are specified before the drawing the object. Besides, the transform are usually in the order of **scaling → rotate → translate**, as the scaling and rotation based on the rotation point (fixed point). Once the transformation of scaling and rotation done, it is easy to apply translation to move the object to the desired place.

Besides, the modelling transformation that that are applied to certain object in the scene will affect the other objects that on the same scene as well. Thus, in order to prevent and limit the transformation to that particular objects**, glPushMatrix(void)** and **glPopMatrix(void)** is applied. The concept is LIFO (Last in First Out) stack concept with the glPushMatrix() pushes the current matrix stack down by one, duplication the current matrix. That is, after a glPushMatrix() call, the matrix on top of the stack is identical to the one below. It does not change the current transformation; it just merely saves a copy. Later when glPopMatrix() is called, it will retrieve that copy and will replace the current modelling transform with the retrieved transform.

Through Lab4, we also learn the how to set and apply some of the OpenGL transformation functions.

i. **glRotatef** (GL float angle, GL float x, GL float y, GL float z);

- ✓ produces a rotation of angle degrees around the vector x, y and z.
- ✓ angle of rotation is in degrees.
- ✓ x, y, z are coordinates of a vector

ii. **glTranslatef** (GLfloat x, GLfloat y, GL float z);

- ✓ produces a translation by x, y and z.
- ✓ x, y, z are coordinates of a vector

iii. **glScalef** (GLfloat x, GLfloat y, GLfloat z);

- ✓ produces a nonuniform scaling along the x, y and z axes. The three parameters indicate the desired scale factor along each of the three axes.
- ✓ x, y, and z specify the scale factor along the x, y and z axes respectively.

iv. **glutWireSphere** (GLdouble radius, Glint slices, Glint stacks);

- ✓ Radius - the radius of the sphere;
  Slices - the number of subdivisions around the Z axis (which similar to the lines of longitude);
  Stacks - the number of subdivisions along the Z-axis (which similar to the lines of latitude).

v. **glutWireCube** (GLdouble size);

- ✓ render a wireframe cube, the cube is centered at the momdelling coordinates origin with sides of length size.

Lab Output

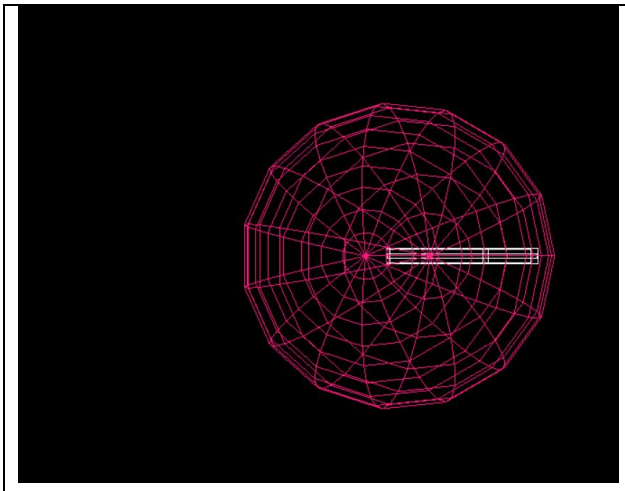

Figure 1: Without any key applied

Our lab output is a 3D clock which composing of a sphere and 3 cube.

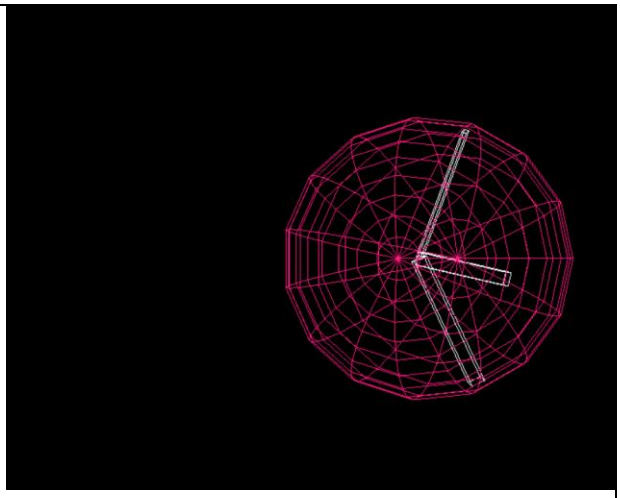

Figure 2: With s key applied

By pressed s key, all the second, minute and hour hand of the clock rotate. However, the sphere is not rotate.

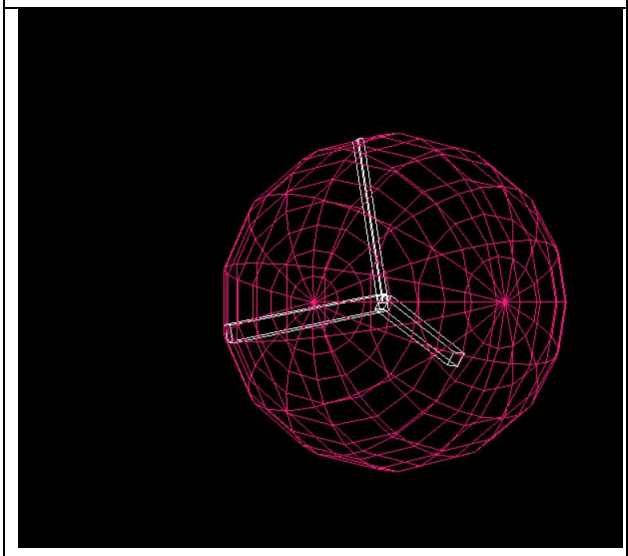

Figure 3: With d key applied

By pressed d key, all the hands of the clock rotate and the clock rotate itself as well.

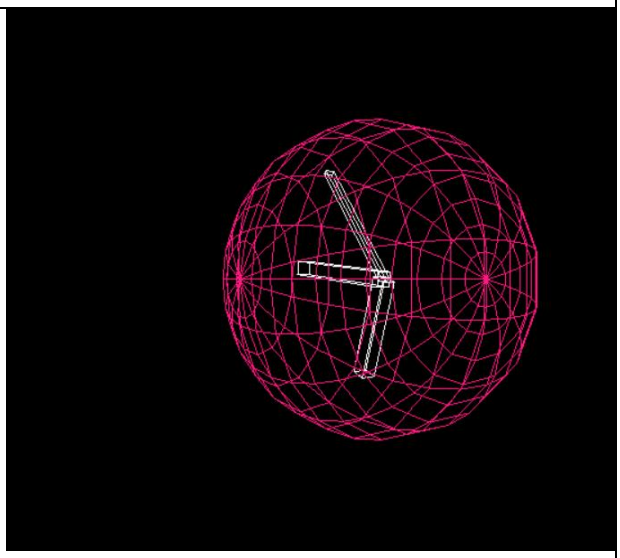

Figure 4: With y key applied

By pressed y key, all the hands of the clock rotate and clock itself rotate with a fixed point.