UCS 1712 – GRAPHICS AND MULTIMEDIA LAB

ASSIGNMENT – 10

VISHAL N

185001198

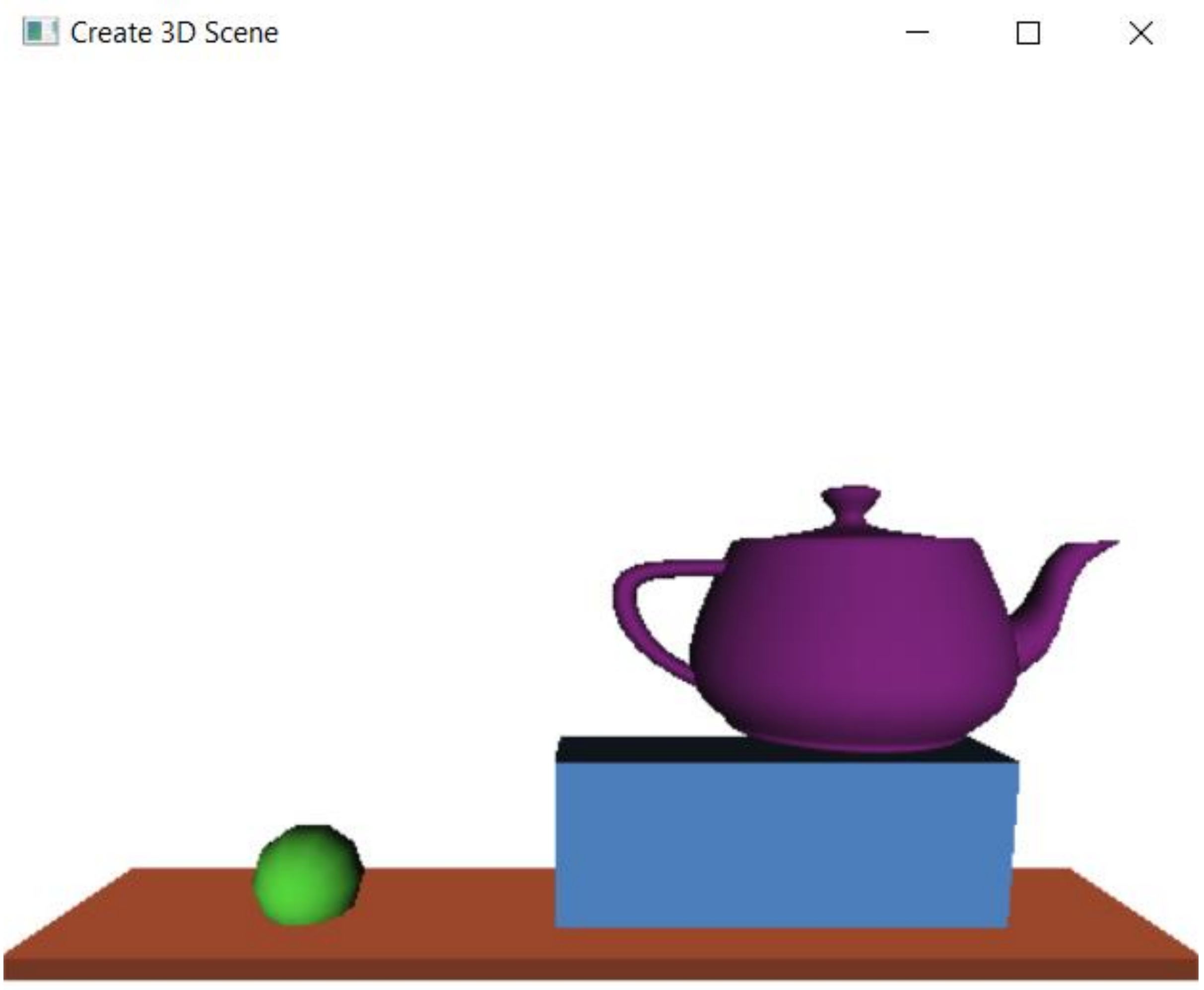
28.10.2021 **CSEC**

1. CREATE 3D SCENE:

```
#pragma warning(disable : 4996)

#include <GL/glut.h>
#include <GL/glu.h>
#include <stdlib.h>
#include <stdio.h>

int INC = 1;

void initialize(void) {
   glClearColor(1.0, 1.0, 1.0, 0.0);
   glShadeModel(GL_SMOOTH);
   GLfloat light_diffuse[] = { 1.0, 1.0, 1.0, 1.0 };
   GLfloat light_position[] = { 0, 0, 1, 0 };
   glLightfv(GL_LIGHT0, GL_DIFFUSE, light_diffuse);
   glLightfv(GL_LIGHT0, GL_POSITION, light_position);
   glEnable(GL_LIGHTING);
   glEnable(GL_LIGHT0);
   glEnable(GL_DEPTH_TEST);
}

GLuint LoadTexture(const char* filename) {
   GLuint texture;
   int width, height;
   unsigned char* data;
   FILE* file;
   file = fopen(filename, "rb");
   if (file == NULL) return 0;
   width = 474;
   height = 395;
   data = (unsigned char*)malloc(width * height * 3);
   fread(data, width * height * 3, 1, file);
   fclose(file);
   for (int i = 0; i < width * height; ++i) {
      int index = i * 3;
      unsigned char B, R;
      B = data[index];
```

```
        R = data[index + 2];
        data[index] = R;
        data[index + 2] = B;
    }

    glGenTextures(1, &texture);
    glBindTexture(GL_TEXTURE_2D, texture);
    glTexEnvf(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_MODULATE);
    glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER,
        GL_LINEAR_MIPMAP_NEAREST);
    glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER,
        GL_LINEAR);
    glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
    glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);
    gluBuild2DMipmaps(GL_TEXTURE_2D, 3, width, height, GL_RGB,
        GL_UNSIGNED_BYTE, data);
    free(data);
    return texture;
}

void drawScene(int state) {
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glLoadIdentity();
    gluLookAt(0.0, 1.0, 7.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);
    glMatrixMode(GL_MODELVIEW);
    glPushMatrix();
    GLfloat cube_color[] = { 0.26, 0.46, 0.7, 1.0 };
    glMaterialfv(GL_FRONT, GL_DIFFUSE, cube_color);
    glScalef(4, 1.5, 1.0);
    glTranslatef(0.4, -1.0, 0.0);
    glutSolidCube(1.0);
    glPopMatrix();
    glPushMatrix();
    glEnable(GL_TEXTURE_2D);
    GLfloat teapot_color[] = { 0.9, 0.2, 0.9, 0.0 };
    GLfloat mat_shininess[] = { 10 };
    glMaterialfv(GL_FRONT, GL_DIFFUSE, teapot_color);
    glMaterialfv(GL_FRONT, GL_SHININESS, mat_shininess);
    glScalef(2, 2, 2);
    glTranslatef(1.1, 0.25, 0.0);
    glutSolidTeapot(0.7);
    glDisable(GL_TEXTURE_2D);
    glPopMatrix();
    glPushMatrix();
    GLfloat ramp_color[] = { 0.8, 0.34, 0.19, 1.0 };
    mat_shininess[0] = 100;
    glMaterialfv(GL_FRONT, GL_DIFFUSE, ramp_color);
    glMaterialfv(GL_FRONT, GL_SHININESS, mat_shininess);
    glRotatef(0, 0, 1, 1);
    glTranslatef(0.0, -2.4, 0);
    glScalef(10.0, 0.2, 1.9);
    glutSolidCube(1.0);
    glPopMatrix();
    glPushMatrix();
    GLfloat ball_color[] = { 0.3, 0.8, 0.2, 0.1 };
    glMaterialfv(GL_FRONT, GL_DIFFUSE, ball_color);
    glRotatef(-0.1, 0, 0, 1);
    glTranslatef(-2.5 - 0.25, -2, 0);
    glutSolidSphere(0.5, 10, 10);
    glPopMatrix();
    glutSwapBuffers();
    glutTimerFunc(1000 / 60, drawScene, state + INC);
}

void reshape(int w, int h) {
    glViewport(0, 0, (GLsizei)w, (GLsizei)h);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluPerspective(75, 1, 1, 20);
    glMatrixMode(GL_MODELVIEW);
}

void sceneDemo() {
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glutTimerFunc(1000 / 60, drawScene, 0);
}

int main(int argc, char** argv) {
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB);
    glutInitWindowSize(500, 500);
    glutCreateWindow("Create 3D Scene");
    initialize();
    glutDisplayFunc(sceneDemo);
    glutReshapeFunc(reshape);
    glutMainLoop();
    return 0;
}
```

OUTPUTS:


Create 3D Scene