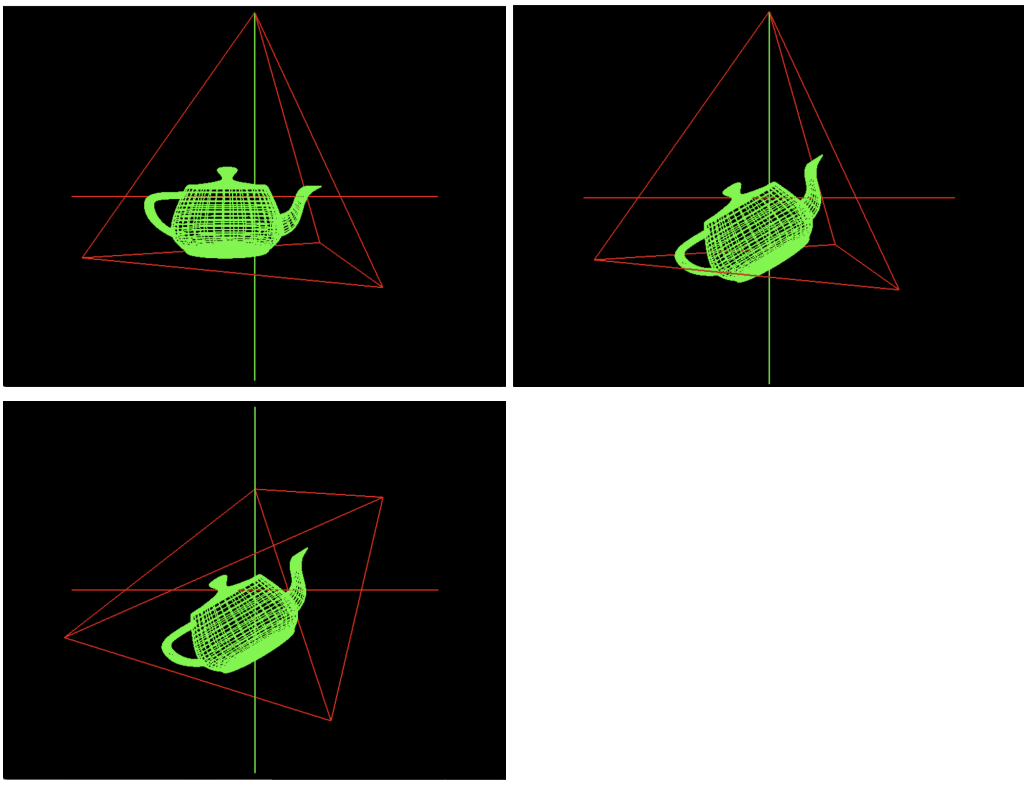
Барщевский Никита,
группа 3530904/10105

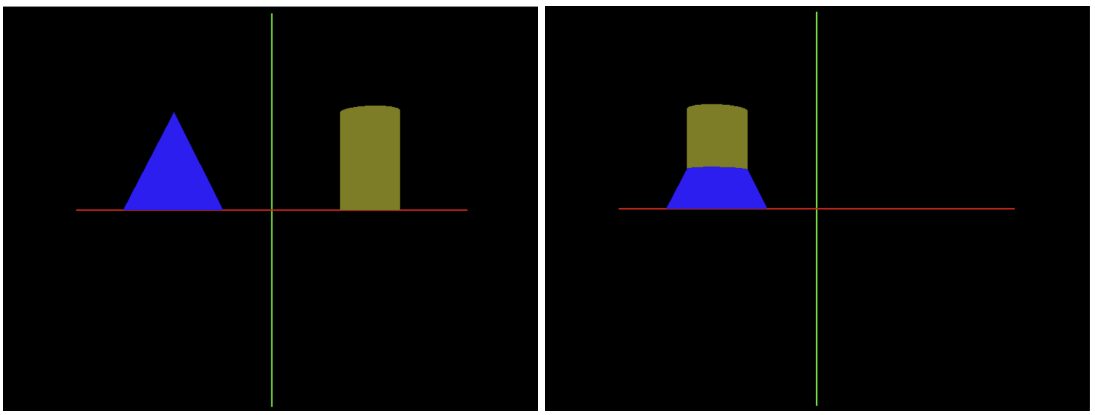
Отчет по дисциплине
“Компьютерная графика”
Работа №1

Целью работы является ознакомление с основами векторной графики и получение навыков работы с базовыми функциями графического API и трехмерными графическими примитивами. Требуется при помощи стандартных функций бибилиотеки (OpenGL/Vulkan или DirectX) изобразить указанные объекты и произвести необходимые преобразования.

Задание 45.
1. Изобразить каркасный чайник и каркасный тетраэдр, описанный вокруг чайника. Размеры примитивов задать самостоятельно.
2. Выполнить поворот чайника на =30 вокруг оси Z, тетраэдра - на =-45 вокруг оси Y.
3. Изобразить конус и цилиндр, одинаковой высоты
4. Переместить один из них так, чтобы центры основания конуса и цилиндра совпали.

Результат работы программы:





Код программы:

```
#include <Windows.h>
#include <GL/glew.h>
#include <GL/freeglut.h>
#include <iostream>

float angleX = 0.0f, angleY = 0.0f, angleZ = 0.0f;
bool isRotatingTeapot = false, isRotatingTetrahedron = false;
bool isMovingCylinder = false;
float targetAngleTeapot = 0.0f, targetAngleTetrahedron = 0.0f;
int counter = 0;
float cylinderPosition = 1.0f;

void setupProjection(int width, int height) {
	glMatrixMode(GL_PROJECTION);
	glLoadIdentity();
	gluPerspective(45.0, (double)width / (double)height, 0.1, 100.0);
	glMatrixMode(GL_MODELVIEW);
}

void drawAxes() {
	glLineWidth(2.0);
	glColor3f(1.0, 0.0, 0.0);
	glBegin(GL_LINES);
	glVertex3f(-2.0, 0.0, 0.0);
	glVertex3f(2.0, 0.0, 0.0);
	glEnd();
	glColor3f(0.0, 1.0, 0.0);
	glBegin(GL_LINES);
	glVertex3f(0.0, -2.0, 0.0);
	glVertex3f(0.0, 2.0, 0.0);
	glEnd();
	glColor3f(0.0, 0.0, 1.0);
	glBegin(GL_LINES);
	glVertex3f(0.0, 0.0, -2.0);
```

```
        glVertex3f(0.0, 0.0, 2.0);
        glEnd();
}

void display() {
        glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
        glLoadIdentity();
        gluLookAt(0.0, 0.0, 5.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);
        drawAxes();

        if (counter < 3) {
        glPushMatrix();
        glColor3f(0.0, 1.0, 0.0);
        glScalef(0.6, 0.6, 0.6);
        glRotatef(angleZ, 0.0, 0.0, 1.0);
        glTranslatef(-0.5, -0.3, 0);
        glutWireTeapot(1.0);
        glPopMatrix();
        glPushMatrix();
        glColor3f(1.0, 0.0, 0.0);
        glScalef(2.0, 2.0, 2.0);
        glRotatef(90, 0.0, 0.0, 1.0);
        glRotatef(angleY, 0.0, 1.0, 0.0);
        glutWireTetrahedron();
        glPopMatrix();
        }
        else if (counter >= 3) {
        glPushMatrix();
        glColor3f(0.0, 0.0, 1.0);
        glTranslatef(-1.0, 0.0, 0.0);
        glRotatef(-90, 1.0, 0.0, 0.0);
        glutSolidCone(0.5, 1.0, 50, 50);
        glPopMatrix();
        glPushMatrix();
        glColor3f(0.5, 0.5, 0.0);
        glTranslatef(cylinderPosition, 0.0, 0.0);
        glRotatef(-90, 1.0, 0.0, 0.0);
        glutSolidCylinder(0.3, 1.0, 50, 50);
        glPopMatrix();
        }
        glFlush();
}

void reshape(int width, int height) {
        if (height == 0) height = 1;
        glViewport(0, 0, width, height);
        setupProjection(width, height);
}
```

```
void idle() {
	if (isRotatingTeapot) {
	if (angleZ < targetAngleTeapot) {
	angleZ += 0.1f;
	if (angleZ >= targetAngleTeapot) {
		angleZ = targetAngleTeapot;
		isRotatingTeapot = false;
	}
	}
	}
	if (isRotatingTetrahedron) {
	if (angleY < targetAngleTetrahedron) {
	angleY += 0.1f;
	if (angleY >= targetAngleTetrahedron) {
		angleY = targetAngleTetrahedron;
		isRotatingTetrahedron = false;
	}
	}
	}
	if (isMovingCylinder) {
	if (cylinderPosition > -1.0f) {
	cylinderPosition -= 0.001f;
	}
	else {
	cylinderPosition = -1.0f;
	isMovingCylinder = false;
	}
	}
	glutPostRedisplay();
}

void mouse(int button, int state, int x, int y) {
	if (button == GLUT_LEFT_BUTTON && state == GLUT_DOWN) {
	if (counter == 0) {
	targetAngleTeapot = 30.0f;
	isRotatingTeapot = true;
	}
	else if (counter == 1) {
	targetAngleTetrahedron = 45.0f;
	isRotatingTetrahedron = true;
	}
	else if (counter == 3) {
	isMovingCylinder = true;
	}
	counter++;
	}
}
```

```
void keyboard(unsigned char key, int x, int y) {
	if (key == 'r') {
	angleX = 0.0f;
	angleY = 0.0f;
	angleZ = 0.0f;
	counter = 0;
	cylinderPosition = 1.0f;
	isMovingCylinder = false;
	}
	glutPostRedisplay();
}

int main(int argc, char* argv[]) {
	glutInit(&argc, argv);
	glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB | GLUT_DEPTH);
	glutInitWindowSize(800, 600);
	glutCreateWindow("Lab 1");
	glEnable(GL_DEPTH_TEST);
	glutDisplayFunc(display);
	glutReshapeFunc(reshape);
	glutIdleFunc(idle);
	glutMouseFunc(mouse);
	glutKeyboardFunc(keyboard);
	glutMainLoop();
	return 0;
}
```