**Московский авиационный институт**

**(Национальный исследовательский университет)**

Факультет: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

**Лабораторные работы №4, 5**

по курсу «Компьютерная графика»

Тема: «Ознакомление с технологией OpenGL»

Студент: Мариничев И. А.

Группа: М8О-308Б-19

Преподаватель: Филиппов Г. С.

Оценка:

Москва

2021

**1. Постановка задачи.**

Создать графическое приложение с использованием OpenGL. Используя результаты Л.Р.№3, изобразить заданное тело (то же, что и в л.р. №3) с использованием средств OpenGL 2.1. Использовать буфер вершин. Точность аппроксимации тела задается пользователем. Обеспечить возможность вращения и масштабирования многогранника и удаление невидимых линий и поверхностей. Реализовать простую модель освещения на GLSL. Параметры освещения и отражающие свойства материала задаются пользователем в диалоговом режиме.

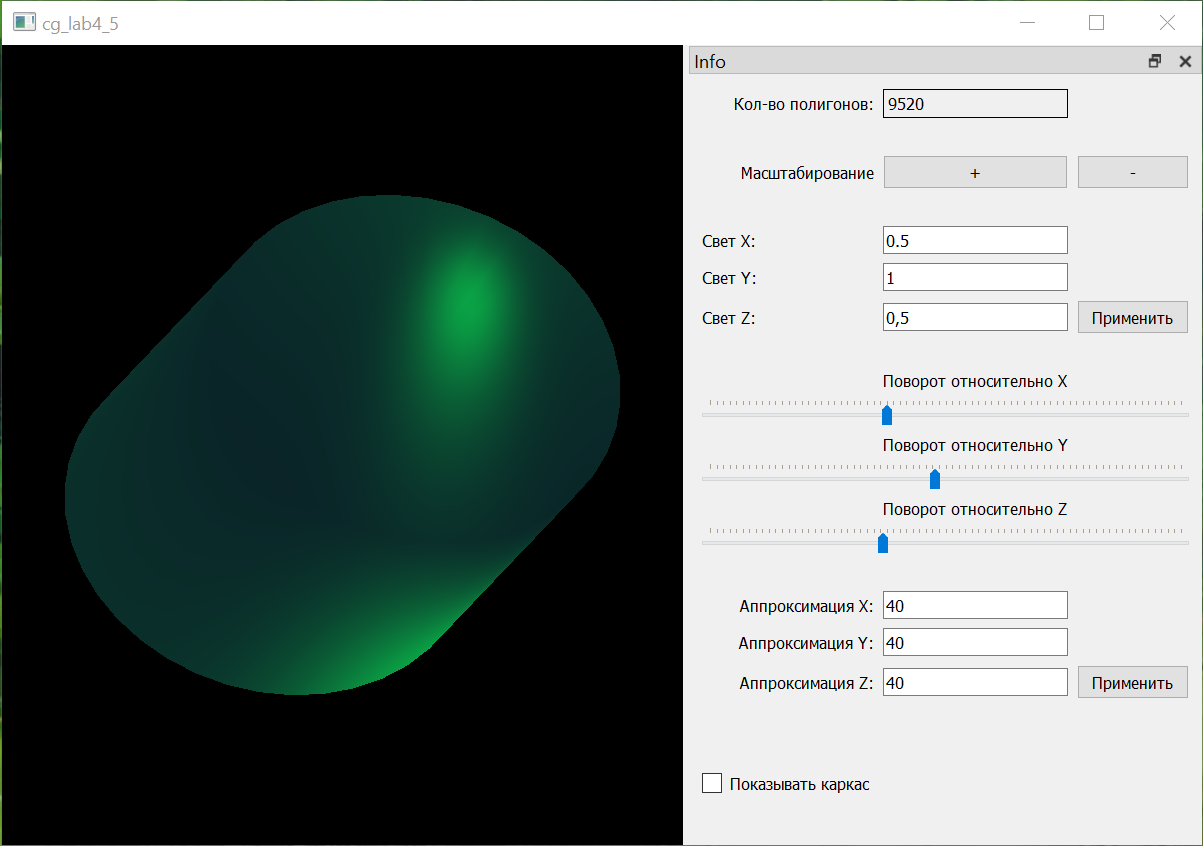
**Вариант №8**: Наклонный круговой цилиндр.

**2. Описание программы.**

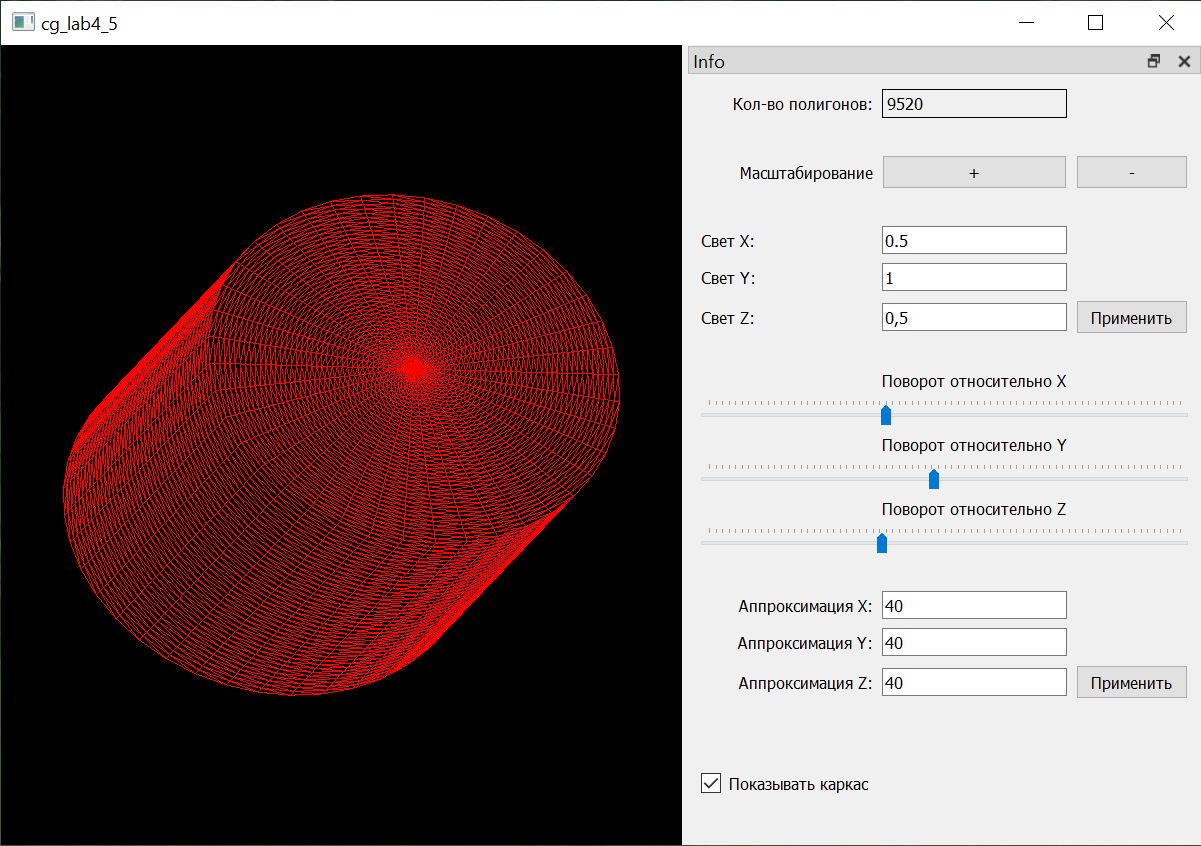
Фигура строится по аналогии с предыдущей лабораторной работой, только при помощи средств OpenGL. Здесь так же присутствует класс polygon для хранения полигонов, класс oblique\_circular\_cylinder, представляющий фигуру наклонный круговой цилиндр. Такая фигура состоит из множества полигонов. Пользователь может задавать аппроксимацию тела по разным осям. Все преобразования для фигуры выполняются средствами OpenGL.

**3. Демонстрация работы программы.**

1) Вид тела при средней аппроксимации по всем осям. На теле можно наблюдать блик.



2) Вид тела с прорисовкой каркаса.

****

**4. Основной код программы.**

void display::***initializeGL***() {

initializeOpenGLFunctions();

glClearColor(0.f, 0.f, 0.f, 1.f);

glEnable(GL\_DEPTH\_TEST);

}

void display::***resizeGL***(int width, int height) {

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

glViewport(0, 0, width, height);

}

void display::***paintGL***() {

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);

glEnable(GL\_DEPTH\_TEST);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

glRotatef(rotateX, 1.f, 0.f, 0.f);

glRotatef(rotateY, 0.f, 1.f, 0.f);

glRotatef(rotateZ, 0.f, 0.f, 1.f);

glScalef(scale, scale, scale);

*if* (displayCarcass) {

glPolygonMode(GL\_FRONT\_AND\_BACK, GL\_LINE);

glDisable(GL\_LIGHTING);

} *else* {

glPolygonMode(GL\_FRONT\_AND\_BACK, GL\_FILL);

glEnable(GL\_LIGHTING);

}

glPushMatrix();

glLoadIdentity();

glEnable(GL\_NORMALIZE);

glMaterialfv(GL\_FRONT\_AND\_BACK, GL\_AMBIENT, fig.get\_ambient\_color());

glMaterialfv(GL\_FRONT\_AND\_BACK, GL\_DIFFUSE, fig.get\_diffuse\_color());

glMaterialfv(GL\_FRONT\_AND\_BACK, GL\_SPECULAR, fig.get\_specular\_color());

glMaterialf(GL\_FRONT\_AND\_BACK, GL\_SHININESS, fig.get\_shininess());

float light\_ambient[] = {0.f, 0.22f, 0.51f, 1.f};

float light\_diffuse[] = {0.f, 0.55f, 0.128f, 1.f};

float light\_specular[] = {0.f, 0.44f, 0.102f, 1.f};

float light\_position[] = {lightPositionX,

lightPositionY,

lightPositionZ, 1.f};

glEnable(GL\_LIGHT0);

glLightfv(GL\_LIGHT0, GL\_DIFFUSE, light\_diffuse);

glLightfv(GL\_LIGHT0, GL\_SPECULAR, light\_specular);

glLightfv(GL\_LIGHT0, GL\_AMBIENT, light\_ambient);

glLightfv(GL\_LIGHT0, GL\_POSITION, light\_position);

glLightf(GL\_LIGHT0, GL\_SPOT\_EXPONENT, 128);

glLightf(GL\_LIGHT0, GL\_CONSTANT\_ATTENUATION, 1.f);

glPopMatrix();

glEnable(GL\_CULL\_FACE);

glCullFace(GL\_FRONT);

glColor3f(1.f, 0.f, 0.f);

*for* (*auto* polygon: fig.get\_polygons()) {

glBegin(GL\_POLYGON);

*for* (*auto* vertex: polygon.vertices) {

glVertex3f(vertex.x(), vertex.y(), vertex.z());

}

glEnd();

}

glDisable(GL\_CULL\_FACE);

glDisable(GL\_LIGHT0);

glDisable(GL\_LIGHTING);

glDisable(GL\_DEPTH\_TEST);

}

**5. Выводы.**

В ходе данной лабораторной работы я получил базовые навыки работы с OpenGL.