

КАФЕДРА Системы автоматизированного проектирования (РК-6)

---

по дисциплине: «Компьютерная графика»

## Вариант лабораторной работы 2

## Оценка

*Москва, 2022 г.*

## ОГЛАВЛЕНИЕ

ЦЕЛЬ РАБОТЫ .....	3
Задание .....	3
Вводная часть .....	3
Разбор кода.....	5
Результаты работы программы.....	7
ВЫВОДЫ.....	7
СПИСО ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ .....	8

## ЦЕЛЬ РАБОТЫ

Цель лабораторной работы - ознакомиться с синтаксисом базовых функций OpenGL.

### Задание

Лабораторная работа состоит из двух частей:

1. Смоделировать фигуру (см. рис. 1) в OpenGL с помощью базовых функций и примитивов, изученных по методическим указаниям.
2. На одну из граней (закрашенную зеленым цветом – см. рис. 1) полученной фигуры наложить текстуру, которая хранится в директории Data.

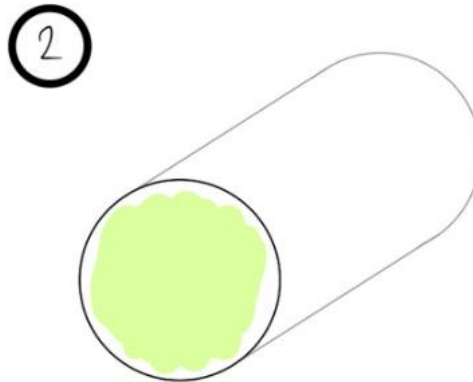


Рисунок 1. Вариант лабораторной работы – «Цилиндр»

### Вводная часть

Для построения модели потребовалось разделить отрисовку на отдельные блоки: отрисовка боковой грани цилиндра, дна и крышки.

Для создания тела цилиндра использовался примитив *GL\_QUAD\_STRIP*. Работает он следующим образом: рисуются связанные четырехугольники. Первая, вторая, третья и четвертая вершина определяют первый четырехугольник. Третья, четвертая, пятая и шестая вершина - второй четырехугольник и т.д.  $(2n-1)$ ,  $2n$ ,  $(2n+1)$  и  $(2n+2)$  вершины задают  $n$ -ый четырехугольник.

В нашем случае, для построения боковой грани, задача состояла в том, чтобы пройти по всей окружности ( $2\pi$ ) с заданным шагом (чем он меньше, тем более гладкая будет грань) и передать в функцию *glVertex3f* координаты текущей точки на окружности и координаты на оси Z.

Для построения крышки и дна цилиндра, использовался примитив *GL\_POLYGON*, у которого все вершины определяют один многоугольник.

Здесь также было необходимо пройти по всей окружности ( $2\pi$ ) с заданным шагом и передать в функцию *glVertex3f* координаты текущей точки на окружности и координаты на оси Z.

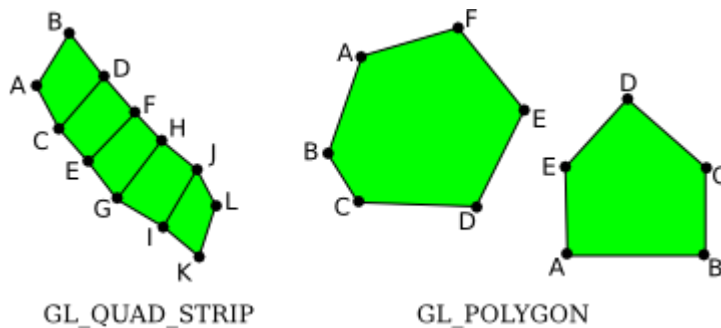


Рисунок 2. Построение примитивов *GL\_QUAD\_STRIP* и *GL\_POLYGON*

Если в процессе отрисовки примитивов не учесть, что центр фигуры должен находиться в центре глобальной системы координат, то при вращении модель будет смещена от центра рабочего окна. Чтобы этого избежать, необходимо в качестве верхней и нижней точек цилиндра указывать значения  $height / 2$  и  $-height / 2$  соответственно; *height* – высота цилиндра.

Также важно отметить, что в зависимости от шага, грань цилиндра может отрисоваться не до конца, из-за чего в цилиндре появятся «щели». Для предотвращения этого эффекта, имеет смысл итерироваться не до  $2\pi$ , а до  $2\pi + angle\_stepsize$ , где *angle\_stepsize* – шаг.

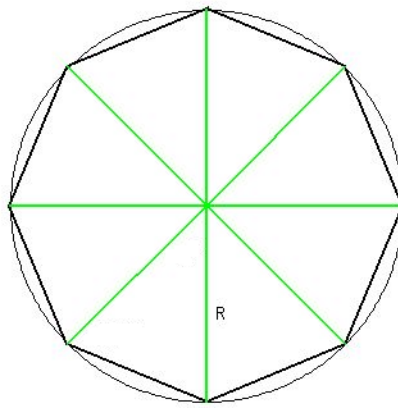


Рисунок 3. Построение окружности с помощью GL\_POLYGON

### Разбор кода

Для выполнения лабораторной работы был изменен код в функциях *DrawGLScene* и *LoadGLTextures*. Разберём его:

1. *GLvoid LoadGLTextures()* - функция для загрузки картинки и конвертирования её в текстуру.

```

37  // Загрузка картинки и конвертирование в текстуру
38  GLvoid LoadGLTextures()
39  {
40      // Загрузка картинки
41      AUX_RGBImageRec* texture1;
42      texture1 = auxDIBImageLoad("Data/Mask1.bmp");
43
44      // Создание текстуры
45      glGenTextures(1, &texture[0]);
46      glBindTexture(GL_TEXTURE_2D, texture[0]);
47      glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
48      glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
49      glTexImage2D(GL_TEXTURE_2D, 0, 3, texture1->sizeX, texture1->sizeY, 0,
50                  GL_RGB, GL_UNSIGNED_BYTE, texture1->data);
51  }
```

2. *int DrawGLScene(GLvoid)* – функция отрисовки сцены. Очистим экран, выполним сброс просмотра, выполним сдвиг по оси Z «от экрана», зададим вращение по осям X, Y, Z, указываем OpenGL на область памяти с текстурой.

```

147  glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT); // Clear Screen And Depth Buffer
148  glLoadIdentity(); // Reset The Current Modelview Matrix
149  glTranslatef(0.0f,0.0f,-3.5f); // Сдвиг по оси Z "от экрана"
150
151  glRotatef(xrot,1.0f,0.0f,0.0f); // Вращение по оси X
152  glRotatef(yrot,0.0f,1.0f,0.0f); // Вращение по оси Y
153  glRotatef(zrot,0.0f,0.0f,1.0f); // Вращение по оси Z
154  glBindTexture(GL_TEXTURE_2D, texture[0]);
```

3. Вызовем функцию *GLvoid DrawCylinder(GLfloat radius, GLfloat height)* для отрисовки цилиндра с аргументами *0.4f, 2.0f* – радиус и высота цилиндра. Далее инициализируем переменные.

```
94     GLfloat z_center = height / 2;
95     GLfloat x = 0.0f;
96     GLfloat y = 0.0f;
97     GLfloat angle_cyl = 0.0f;
98     GLfloat angle_stepsize = 0.1f;
99     GLfloat angle_end = 2 * PI + angle_stepsize;
```

4. Построим боковую грань цилиндра с помощью примитива *GL\_QUAD\_STRIP*.

```
101     // Отрисовка боковой грани цилиндра
102     glBegin(GL_QUAD_STRIP);
103     angle_cyl = 0.0;
104
105     while (angle_cyl < angle_end) {
106         x = radius * cos(angle_cyl);
107         y = radius * sin(angle_cyl);
108         glVertex3f(x, y, -z_center);
109         glVertex3f(x, y, z_center);
110         angle_cyl += angle_stepsize;
111     }
112
113     glEnd();
```

5. Построим верхнюю грань цилиндра с помощью примитива *GL\_POLYGON*.

```
115     // Отрисовка верхушки цилиндра
116     glBegin(GL_POLYGON);
117     angle_cyl = 0.0;
118
119     while (angle_cyl < angle_end) {
120         x = radius * cos(angle_cyl);
121         y = radius * sin(angle_cyl);
122         glTexCoord2f(x, y);
123         glVertex3f(x, y, z_center);
124         angle_cyl += angle_stepsize;
125     }
126
127     glEnd();
```

6. Построим нижнюю грань цилиндра с помощью примитива *GL\_POLYGON*.

```
129     // Отрисовка дна цилиндра
130     glBegin(GL_POLYGON);
131     angle_cyl = 0.0;
132
133     while (angle_cyl < angle_end) {
134         x = radius * cos(angle_cyl);
135         y = radius * sin(angle_cyl);
136         glTexCoord2f(x, y);
137         glVertex3f(x, y, -z_center);
138         angle_cyl += angle_stepsize;
139     }
140
141     glEnd();
```

7. Далее в функции *DrawGLScene* зададим угол вращения фигуры и задержку в итерациях 8 мс.

```
162 | Sleep(8);  
163 | return TRUE;  
// Keep Going
```

Весь код хранится на [GitHub](#).

### Результаты работы программы

В результате выполнения программы происходит построение цилиндра, на верхнюю грань которого наложена текстура.

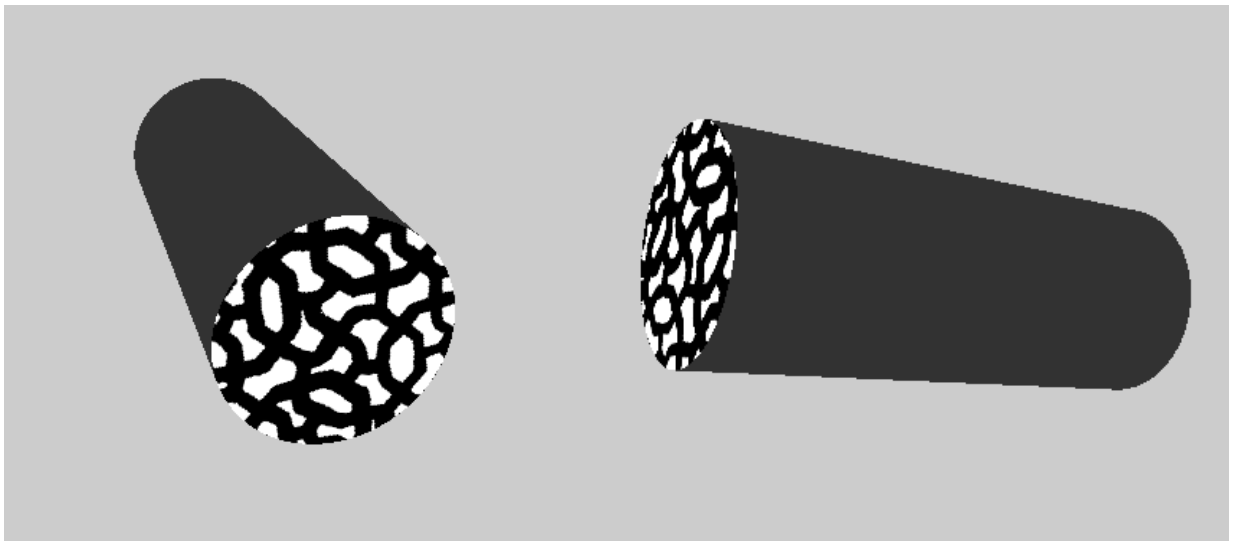


Рисунок 4. Итоговая модель

### ВЫВОДЫ

В результате выполнения лабораторной работы были изучены базовые функции OpenGL, их синтаксис и принципы построения 3D-моделей. В ходе выполнения работы была получена 3D-модель цилиндра.

Также были изучены функции создания и настройки параметров окна, функции построения геометрии объекта, способ UV-маппирования и отрисовки с помощью разных примитивов.

## СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Работа с OpenGL [Электронный ресурс] – Режим доступа:  
<http://pmg.org.ru/nehe/index.html>
2. Витюков Ф. А. Лекции по дисциплине «Компьютерная графика» – Москва:  
МГТУ им. Н. Э. Баумана, 2022.
3. Github Arcady1.  
[https://github.com/Arcady1/University\\_labs/tree/master/Computer\\_Graphics/lab\\_1\\_2/lab1\\_2](https://github.com/Arcady1/University_labs/tree/master/Computer_Graphics/lab_1_2/lab1_2)