|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство образования и науки Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ Робототехники и комплексной автоматизации

КАФЕДРА Системы автоматизированного проектирования (РК-6)

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ**

по дисциплине: «Компьютерная графика»

Студент Гусаров Аркадий Андреевич

Группа РК6-63Б

Тип задания Лабораторная работа №1-2

Название «Знакомство с OpenGL»

Вариант лабораторной работы 2

Студент **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_Гусаров А.A.**

*подпись, дата фамилия, и.о.*

Преподаватель **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_Витюков Ф.A.**

*подпись, дата фамилия, и.о.*

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*La*

*Москва, 2022 г.*

**Оглавление**

[Цель работы 3](#_Toc99363763)

[Задание 3](#_Toc99363764)

[Вводная часть 3](#_Toc99363765)

[Разбор кода 5](#_Toc99363766)

[Результаты работы программы 6](#_Toc99363767)

[Выводы 7](#_Toc99363768)

[Список используемых источников 7](#_Toc99363769)

# **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.

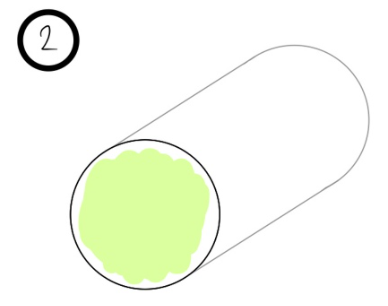


Рис. 1. Вариант лабораторной работы – «Цилиндр»

# **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 (2π) 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 (2π) с заданным шагом и передать в функцию *glVertex3f* координаты текущей точки на окружности и координаты на оси Z.

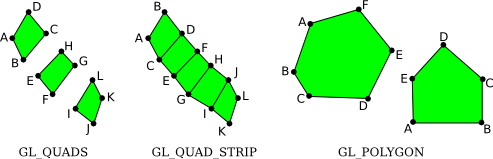


Рис. 2. Построение примитивов GL\_QUAD\_STRIP и GL\_POLYGON

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 2π, а до 2π + *angle\_stepsize*, где *angle\_stepsize* – шаг.

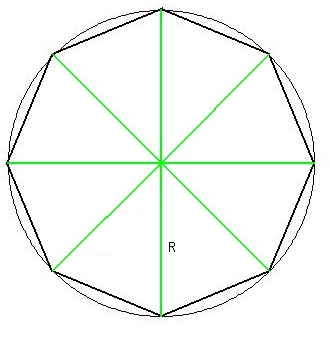


Рис. 3. Построение окружности с помощью GL\_POLYGON

# **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.

// Загрузка картинки и конвертирование в текстуру

GLvoid LoadGLTextures()

{

// Загрузка картинки

AUX\_RGBImageRec\* texture1;

texture1 = auxDIBImageLoad("Data/Mask1.bmp");

// Создание текстуры

glGenTextures(1, &texture[0]);

glBindTexture(GL\_TEXTURE\_2D, texture[0]);

glTexParameteri(GL\_TEXTURE\_2D, GL\_TEXTURE\_MAG\_FILTER, GL\_LINEAR);

glTexParameteri(GL\_TEXTURE\_2D, GL\_TEXTURE\_MIN\_FILTER, GL\_LINEAR);

glTexImage2D(GL\_TEXTURE\_2D, 0, 3, texture1->sizeX, texture1->sizeY, 0,

GL\_RGB, GL\_UNSIGNED\_BYTE, texture1->data);

}

1. *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.

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

glLoadIdentity(); // Reset The Current Modelview Matrix

glTranslatef(0.0f,0.0f,-3.5f);

glRotatef(xrot,1.0f,0.0f,0.0f); // Вращение по оси X

glRotatef(yrot,0.0f,1.0f,0.0f); // Вращение по оси Y

glRotatef(zrot,0.0f,0.0f,1.0f); // Вращение по оси Z

glBindTexture(GL\_TEXTURE\_2D, texture[0]);

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

GLfloat z\_center = height / 2;

GLfloat x = 0.0f;

GLfloat y = 0.0f;

GLfloat angle\_cyl = 0.0f;

GLfloat angle\_stepsize = 0.1f;

GLfloat angle\_end = 2 \* PI + angle\_stepsize;

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

// Отрисовка боковой грани цилиндра

glBegin(GL\_QUAD\_STRIP);

angle\_cyl = 0.0;

while (angle\_cyl < angle\_end) {

x = radius \* cos(angle\_cyl);

y = radius \* sin(angle\_cyl);

glVertex3f(x, y, -z\_center);

glVertex3f(x, y, z\_center);

angle\_cyl += angle\_stepsize;

}

glEnd();

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

// Отрисовка верхушки цилиндра

glBegin(GL\_POLYGON);

angle\_cyl = 0.0;

while (angle\_cyl < angle\_end) {

x = radius \* cos(angle\_cyl);

y = radius \* sin(angle\_cyl);

glTexCoord2f(x, y);

glVertex3f(x, y, z\_center);

angle\_cyl += angle\_stepsize;

}

glEnd();

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

// Отрисовка дна цилиндра

glBegin(GL\_POLYGON);

angle\_cyl = 0.0;

while (angle\_cyl < angle\_end) {

x = radius \* cos(angle\_cyl);

y = radius \* sin(angle\_cyl);

glTexCoord2f(x, y);

glVertex3f(x, y, -z\_center);

angle\_cyl += angle\_stepsize;

}

glEnd();

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

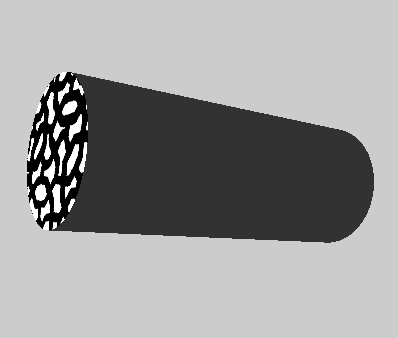
Sleep(8);

return TRUE; // Keep Going

Весь код хранится на [GitHub](https://github.com/Arcady1/University_labs/tree/master/Computer_Graphics/lab_1_2/lab1_2).

# **Результаты работы программы**

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

# **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>