**Московский авиационный институт**

**(Национальный исследовательский университет)**

Факультет: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

Дисциплина: «Компьютерная графика»

**Лабораторная работа № 6**

Тема: Ознакомление с технологией OpenGL.

Студент: Чекушкин Д.И.

Группа: 80-304

Преподаватель: Чернышов Л.Н.

Дата:

Оценка:

Москва, 2018

1. Постановка задачи

Задание:Создание шейдерных анимационных эффектов в OpenGL 2.1

12. Анимация. Координата Y изменяется по закону Y = Y\*cos(t+Y)

1. Решения задачи

ЯП: Python

ОС: Ubuntu 16.04

Библиотеки: OpenGL.GL, math, sys

glClearColor - первоначальное закрашивание экрана

glEnable(GL\_LIGHTING)-glEnable(GL\_LIGHT0) - включить первый источник света

glLightfv(GL\_LIGHT0, GL\_POSITION, lightpos) - положение источника света

glTranslate - сдвиг матрицы

glRotate - вращения объекта вдоль оси

glBlendFunc - смешивания цветов

glVertex3fv - отрисовка векторов

glColor4f(x,y,z,u) - назначение цвета и прозрачности

tksd.askfloat - запуск диалогового окна

glEnable(GL\_LIGHTING) - включить освещение

glEnable(GL\_LIGHT0) - включить 1-ый источник света

glLightfv(GL\_LIGHT0, GL\_POSITION, lightpos) - положение источника света

Ход работы:

* С помощью функционала OpenGL строим цилиндр
* Добавляем источник света
* Добавляем анимацию
* Добавляем вращение с использованием стрелочек клавиатуры
* Отрисовываем

При тестировании были выявлены и исправлены незначительные ошибки, связанные с анимацией цилиндра

1. Руководство по использованию программы

Файл 6.py содержит код программы .

На вход подается положение источника света, на выход – график искомой поверхности, освещенной источником света.

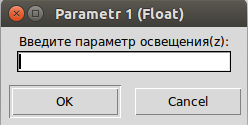
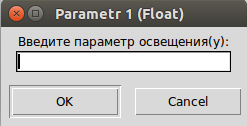
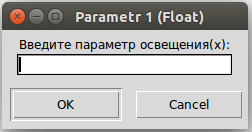


Рис 1 - Диалоговые окна

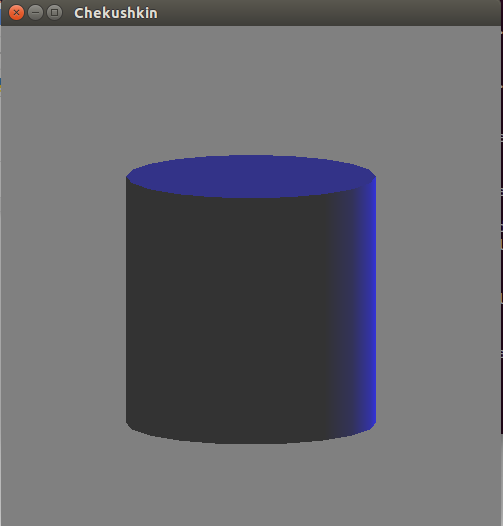


Рис 2 - Вывод программы

1. Листинг программы

#coding=utf-8

from OpenGL.GL import \*

from OpenGL.GLU import \*

from OpenGL.GLUT import \*

import math

import sys

#12. Анимация. Координата Y изменяется по закону Y = cos(t+Y)

try:

# for Python2

import Tkinter as tk

import tkSimpleDialog as tksd

except:

# for Python3

import tkinter as tk

import tkinter.simpledialog as tksd

root = tk.Tk()

q = tksd.askfloat("Parametr 1 (Float)", "Введите параметр освещения(x):",

parent=root, minvalue=0)

w = tksd.askfloat("Parametr 1 (Float)", "Введите параметр освещения(y):",

parent=root, minvalue=0)

e = tksd.askfloat("Parametr 1 (Float)", "Введите параметр освещения(z):",

parent=root, minvalue=0)

a=q

b=w

c=e

# Объявляем все глобальные переменные

global xrot # Величина вращения по оси x

global yrot # Величина вращения по оси y

global ambient # рассеянное освещение

global cylcolor # Цвет цилиндра

global lightpos # Положение источника освещения

global Y

Y = 0.1

# Процедура инициализации

def init():

global xrot # Величина вращения по оси x

global yrot # Величина вращения по оси y

global ambient # Рассеянное освещение

global cylcolor # Цвет цилиндра

global lightpos # Положение источника освещения

xrot = 10.0 # Величина вращения по оси x = 0

yrot = 0.0 # Величина вращения по оси y = 0

ambient = (1.0, 1.0, 1.0, 1) # Первые три числа цвет в формате RGB, а последнее - яркость

cylcolor = (0, 0, 1, 0.8) # Коричневый цвет для ствола

lightpos = (a, b, c) # Положение источника освещения по осям xyz

#lightpos = (2.0, -2.0, -1.0)

glClearColor(0.5, 0.5, 0.5, 1.0) # Серый цвет для первоначальной закраски

gluOrtho2D(-1.0, 1.0, -1.0, 1.0) # Определяем границы рисования по горизонтали и вертикали

glRotatef(-90, 1.0, 0.0, 0.0) # Сместимся по оси Х на 90 градусов

glLightModelfv(GL\_LIGHT\_MODEL\_AMBIENT, ambient) # Определяем текущую модель освещения

glEnable(GL\_LIGHTING) # Включаем освещение

glEnable(GL\_LIGHT0) # Включаем один источник света

glLightfv(GL\_LIGHT0, GL\_POSITION, lightpos) # Определяем положение источника света

# Процедура обработки специальных клавиш

def specialkeys(key, x, y):

global xrot

global yrot

# Обработчики для клавиш со стрелками

if key == GLUT\_KEY\_UP: # Клавиша вверх

xrot -= 2.0 # Уменьшаем угол вращения по оси Х

if key == GLUT\_KEY\_DOWN: # Клавиша вниз

xrot += 2.0 # Увеличиваем угол вращения по оси Х

if key == GLUT\_KEY\_LEFT: # Клавиша влево

yrot -= 2.0 # Уменьшаем угол вращения по оси Y

if key == GLUT\_KEY\_RIGHT: # Клавиша вправо

yrot += 2.0 # Увеличиваем угол вращения по оси Y

glutPostRedisplay() # Вызываем процедуру перерисовки

# Процедура перерисовки

#def draw():

# global xrot

# global yrot

# global lightpos

# global greencolor

# global cylcolor

# glClear(GL\_COLOR\_BUFFER\_BIT) # Очищаем экран и заливаем серым цветом

# glPushMatrix() # Сохраняем текущее положение "камеры"

# glRotatef(xrot, 1.0, 0.0, 0.0) # Вращаем по оси X на величину xrot

# glRotatef(yrot, 0.0, 1.0, 0.0) # Вращаем по оси Y на величину yrot

# glLightfv(GL\_LIGHT0, GL\_POSITION, lightpos) # Источник света вращаем вместе с елкой

# Рисуем цилиндр

# Устанавливаем материал: рисовать с 2 сторон, рассеянное освещение, синий цвет

# glMaterialfv(GL\_FRONT, GL\_DIFFUSE, cylcolor)

# glTranslatef(0.0, 0.0, -0.7) # Сдвинемся по оси Z на -0.7

# Рисуем цилиндр с радиусом 0.1, высотой 0.2

# Последние два числа определяют количество полигонов

# glutSolidCylinder(0.5, 1, 20, 20)

# glPopMatrix() # Возвращаем сохраненное положение "камеры"

# glutSwapBuffers() # Выводим все нарисованное в памяти на экран

def draw():

global xrot, yrot, lightpos, greencolor, cylcolor, Y

Y=1

timeSinceStart = glutGet(GLUT\_ELAPSED\_TIME)

glEnable(GL\_DEPTH\_TEST)

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

glPushMatrix()

Y=Y\*math.cos(timeSinceStart\*0.002+Y)

glTranslatef(0.0, 0.5, Y)

glRotatef(xrot, 1.0, 0.0, 0.0)

glRotatef(yrot, 0.0, 1.0, 0.0)

glLightfv(GL\_LIGHT0, GL\_POSITION, lightpos)

glMaterialfv(GL\_FRONT, GL\_DIFFUSE, cylcolor)

glutSolidCylinder(0.5, 1, 20, 20)

glPopMatrix()

glutSwapBuffers()

glutPostRedisplay()

# Здесь начинается выполнение программы

# Использовать двойную буферизацию и цвета в формате RGB (Красный, Зеленый, Синий)

glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB | GLUT\_DEPTH)

glEnable(GL\_DEPTH\_TEST)

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

# Указываем начальный размер окна (ширина, высота)

glutInitWindowSize(500, 500)

# Указываем начальное положение окна относительно левого верхнего угла экрана

glutInitWindowPosition(50, 50)

# Инициализация OpenGl

glutInit(sys.argv)

glutCreateWindow(b"Chekushkin")

# Определяем процедуру, отвечающую за перерисовку

glutDisplayFunc(draw)

# Определяем процедуру, отвечающую за обработку клавиш

glutSpecialFunc(specialkeys)

# Вызываем нашу функцию инициализации

init()

# Запускаем основной цикл

glutMainLoop()

Выводы: благодаря проделанной работе мне удалось освоить создание анимации в Python, методы закраски и отрисовки плоскостей с помощью технологий OpenGL

Список литературы:

1. Самоучитель по графике [Электронный ресурс] Url: https://pythonworld.ru/novosti-mira-python/scientific-graphics-in-python.html
2. OpenGL manual [Электронный ресурс] Url: http://pyopengl.sourceforge.net/documentation/manual-3.0/
3. Методическое пособие по графике в OpenGL [Электронный ресурс] Url: https://rsdn.org/article/opengl/ogltut2.xml