**Московский авиационный институт**

**(Национальный исследовательский университет)**

Факультет: «Информационные технологии и прикладная математика»

Кафедра: 804 «Теория вероятностей и компьютерное моделирование»

Дисциплина: «Компьютерная графика»

**Лабораторная работа № 3**

Тема: Основы построения фотореалистичных изображений.

Студент: Шевчук П.В.

Группа: 80-304

Преподаватель: Чернышов Л.Н.

Дата: 30.10.18

Оценка:

Москва, 2018

**1. Постановка задачи**

Используя результаты Л.Р. № 2, аппроксимировать заданное тело выпуклым многогранником. Точность аппроксимации задается пользователем. Обеспечить возможность вращения и масштабирования многогранника и удаление невидимых линий и поверхностей. Реализовать простую модель закраски для случая одного источника света.

Параметры освещения и отражающие свойства материала задаются пользователем в диалоговом режиме.

Вариант тела: Сектор эллипсоида

**2. Решения задачи**

Язык программирования: C#.

Выбранный язык удобен для решения задачи, так как он обладает удобными библиотеками для построения фигур и вывода их на экран.

Пространство имен System.Drawing обеспечивает доступ к базовым функциональным возможностям графического интерфейса.

Пространство имен System.Collections.Generic содержит интерфейсы и классы, определяющие универсальные коллекции, которые позволяют обеспечить повышенную производительность и безопасность типов по сравнению с строго типизированными коллекциями.

С помощью парабол и окружностей строится сектор эллипсоида по вычисленным координатам. Затем добавляем источник света и создаются массивы распределения света.

В рассеянном освещении интенсивность считается как произведение коэффициента диффузного отражения рассеянного света на интенсивность рассеянного. Интенсивность зеркального освещения выражается эмпирической формулой по закону Фонга. При диффузном отражении и отражении рассеянного света, цветовые координаты отраженного луча определяются произведением соответствующих цветовых координат падающего света и цвета поверхности. При зеркальном отражении цвет отраженного света определяется только цветом источника.

**3. Руководство по использованию программы**

В окне вывода пользователь может менять масштаб полученной фигуры. Степень аппроксимации задается с помощью полей «Количество окружностей» и «Количество парабол». При запуске строится фигура без освещения(рис.1).

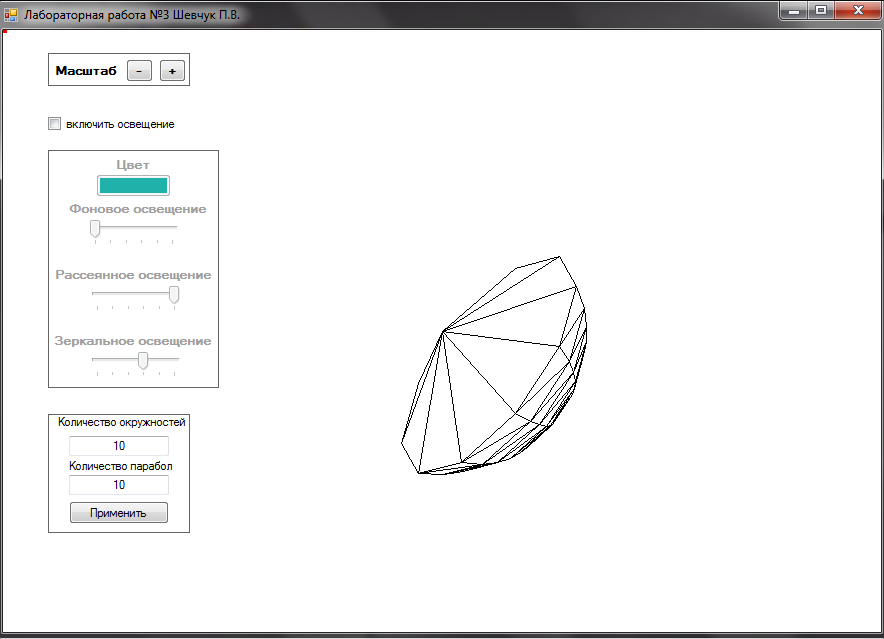


Рис 1. Незакрашенная фигура

Пользователь может включить источник света и регулировать степень освещения с помощью фонового и рассеянного освещения(рис.2). Фоновое освещение можно считать как более однотонное, а рассеянное позволяет увидеть разную степень освещения той или иной грани.

При увеличении степени аппроксимации фигура становится более гладкой(рис.3).

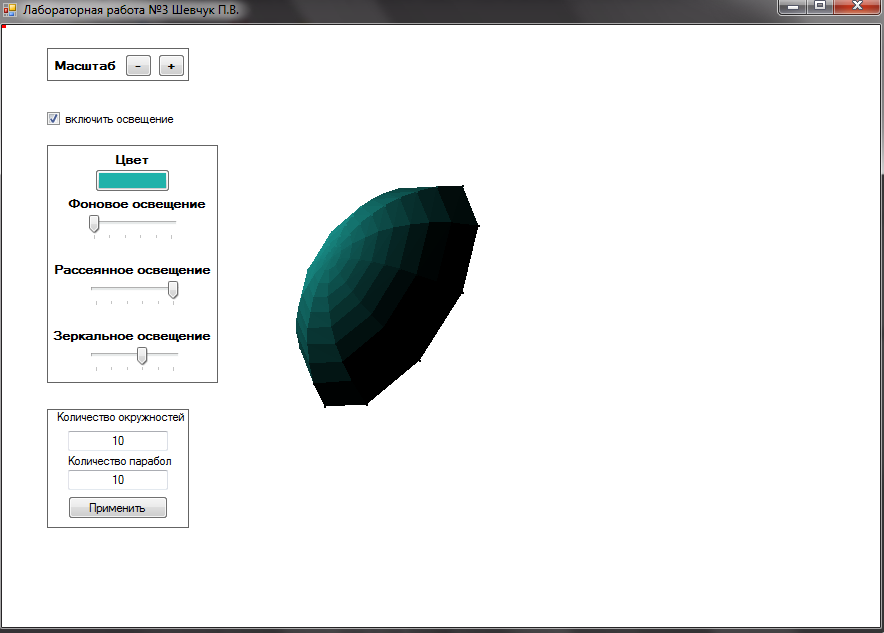


Рис 2. Фигура с источником света

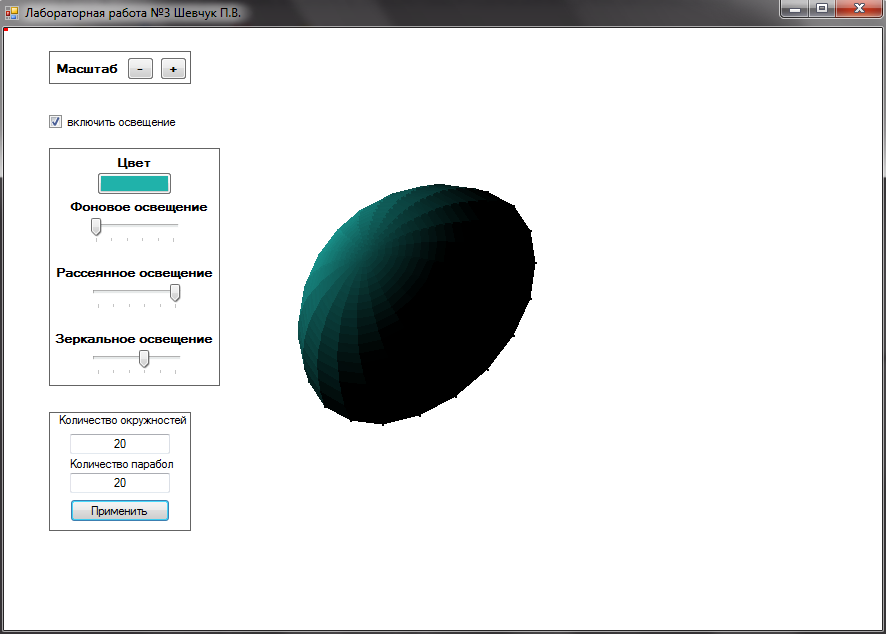


Рис 3 . Увеличиваем степень аппроксимации

**4. Листинг программы**

using System;

using System.Drawing;

using System.Windows.Forms;

namespace lab3CG

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

// точка света

light = new MyPoint(1, 0, 0, 1);

// двумерные матрицы для освещения

iNofLight = new double[3, 3];

kMat = new double[3, 3];

// создание объекта класса Ellipsoid

ellipsoid = new Ellipsoid(int.Parse(textBoxOfNumberOfCircles.Text), int.Parse(textBoxOfNumberOfParabols.Text));

// передаём в конструктор число окружностей и число парабол

// текущие координаты курсора и координаты его предыдущего положения

mx = 0;

my = 0;

cx = 0;

cy = 0;

// масштаб

scale = 30;

mashtabK = 0;

isMouseDown = false;

}

// объявление переменны

int mx, my, cx, cy;

float scale;

double mashtabK;

MyPoint light;

double[,] iNofLight;

double[,] kMat;

bool isMouseDown;

Ellipsoid ellipsoid;

// задание координат освещения, диапозона цветов и построение графика

private void Form1\_Paint(object sender, PaintEventArgs e)

{

// рассеянное

kMat[0, 0] = 1;

kMat[0, 1] = 1;

kMat[0, 2] = 1;

// фоновое

kMat[1, 0] = 1;

kMat[1, 1] = 1;

kMat[1, 2] = 1;

// зеркальное

kMat[2, 0] = 1;

kMat[2, 1] = 1;

kMat[2, 2] = 1;

// Освещение, диапозон цветов от начальных значений

iNofLight[0, 0] = (double)buttonOfColor.BackColor.R \* ((double)trackBarRas.Value / 5) / 255;

iNofLight[0, 1] = (double)buttonOfColor.BackColor.G \* ((double)trackBarRas.Value / 5) / 255;

iNofLight[0, 2] = (double)buttonOfColor.BackColor.B \* ((double)trackBarRas.Value / 5) / 255;

iNofLight[1, 0] = (double)buttonOfColor.BackColor.R \* ((double)trackBarFon.Value / 5) / 255;

iNofLight[1, 1] = (double)buttonOfColor.BackColor.G \* ((double)trackBarFon.Value / 5) / 255;

iNofLight[1, 2] = (double)buttonOfColor.BackColor.B \* ((double)trackBarFon.Value / 5) / 255;

iNofLight[2, 0] = (double)buttonOfColor.BackColor.R \* ((double)trackBarMir.Value / 5) / 255;

iNofLight[2, 1] = (double)buttonOfColor.BackColor.G \* ((double)trackBarMir.Value / 5) / 255;

iNofLight[2, 2] = (double)buttonOfColor.BackColor.B \* ((double)trackBarMir.Value / 5) / 255;

Pen pen = new Pen(Color.Black, 1.0f);

e.Graphics.DrawRectangle(new Pen(Color.Red, 3.0f), (int)light.x, -(int)light.y, 1, 1);

// задание масштаба

double zoom\_level = (scale + mashtabK) / 1000;

double coeff = Math.Max(e.ClipRectangle.Width, e.ClipRectangle.Height) \* zoom\_level;

// матрица сдвига

ShiftMatrix sh = new ShiftMatrix(e.ClipRectangle.Width / 2, e.ClipRectangle.Height / 2, 0);

// матрица масштабирования

ScalingMatrix sc = new ScalingMatrix(coeff, coeff, coeff);

// создаём две матрицы поворота

RotationMatrix rtx = new RotationMatrix('X', my \* Math.PI / 180.0);

RotationMatrix rty = new RotationMatrix('Y', -mx \* Math.PI / 180.0);

// матрица преобразования

Matrix preobr = sh \* rtx \* rty \* sc;

// построение сектора эллипсоида

ellipsoid.draw(preobr, pen, e.Graphics, light, iNofLight, kMat);

}

// передвижение мышкой

private void Form1\_MouseMove(object sender, MouseEventArgs e)

{

if (isMouseDown)

{

int delta\_x = e.X - cx;

int delta\_y = e.Y - cy;

mx += delta\_x;

my += delta\_y;

cx = e.X;

cy = e.Y;

Refresh();

}

}

// остальные функции формы

private void Form1\_MouseDown(object sender, MouseEventArgs e)

{

isMouseDown = true;

cx = e.X;

cy = e.Y;

}

private void checkBoxOfIlluminate\_CheckedChanged(object sender, EventArgs e)

{

ellipsoid.illuminate = checkBoxOfIlluminate.Checked;

panelOfLight.Enabled = checkBoxOfIlluminate.Checked;

//panelOfLightSource.Enabled = checkBoxOfIlluminate.Checked;

Refresh();

}

private void Form1\_MouseUp(object sender, MouseEventArgs e)

{

isMouseDown = false;

}

private void Form1\_SizeChanged(object sender, EventArgs e)

{

Refresh();

}

private void buttonOfApply1\_Click(object sender, EventArgs e)

{

ellipsoid.calcEllipsoid(int.Parse(textBoxOfNumberOfCircles.Text), int.Parse(textBoxOfNumberOfParabols.Text));

Refresh();

}

private void buttonRight\_Click(object sender, EventArgs e)

{

light.x += 10;

Refresh();

}

private void buttonLeft\_Click(object sender, EventArgs e)

{

light.x -= 10;

Refresh();

}

private void buttonUp\_Click(object sender, EventArgs e)

{

light.y += 10;

Refresh();

}

private void buttonDown\_Click(object sender, EventArgs e)

{

light.y -= 10;

Refresh();

}

private void buttonNaNas\_Click(object sender, EventArgs e)

{

light.z -= 10;

Refresh();

}

private void buttonOtNas\_Click(object sender, EventArgs e)

{

light.z += 10;

Refresh();

}

private void buttonOfColor\_Click(object sender, EventArgs e)

{

if (colorDialog1.ShowDialog() == DialogResult.OK)

{

buttonOfColor.BackColor = colorDialog1.Color; // выбор цвета диалога

Refresh();

}

}

private void trackBarFon\_ValueChanged(object sender, EventArgs e)

{

Refresh();

}

private void trackBarRas\_ValueChanged(object sender, EventArgs e)

{

Refresh();

}

private void trackBarMir\_ValueChanged(object sender, EventArgs e)

{

Refresh();

}

private void buttonOfMatApply\_Click(object sender, EventArgs e)

{

Refresh();

}

private void mashtabMinusButton\_Click(object sender, EventArgs e)

{

mashtabK += 5;

Refresh();

}

private void mashtabPlusButton\_Click(object sender, EventArgs e)

{

mashtabK += 5;

Refresh();}}}

**5. Вывод**

В ходе данной лабораторной работы получены основные навыки работы с трехмерной графикой C#, освоены методы реализации простой модели закраски для одного источника света и отрисовки плоскостей.

**6. Используемая литература**

1. Основные алгоритмы компьютерной графики [Электронный ресурс].

URL:<http://bourabai.ru/graphics/0211.htm>

2. Документация по .NET [Электронный ресурс].

URL:<https://docs.microsoft.com/ru-ru/dotnet/index?view=netframework-4.7.2>