Министерство образования и науки Российской Федерации

**Федеральное государственное бюджетное образовательное учреждение науки и высшего образования**

«Владимирский государственный университет имени Александра Григорьевича и Николая Григорьевича Столетовых»

(ВлГУ)

Кафедра информационных систем и программной инженерии

**Лабораторная работа № 15**

**по дисциплине**

**«Программирование компьютерной графики»**

**Тема: «Системы частиц»**

Выполнил:

ст. гр. ПРИ-117

Хлызова В.Г.

Принял:

Жигалов И.Е.

Владимир, 2021

ЦЕЛЬ РАБОТЫ

Изучение методов использования в компьютерной графике 3D-объектов, не имеющих чётких геометрических границ и описываемых стохастическими алгоритмами.

ЗАДАНИЕ

1. Ознакомиться по методическим указаниям и литературе с теоретическим материалом.

2. Выполнить действия, приведенные в разделе 15.2, 15.3. При разработке программы имя проекта, создаваемого в MS Visual Studio, должно содержать фамилию студента и группу (например, Ivanov\_Ivan\_ISG\_105\_lab\_1).

3. При выполнении 15.2, 15.3 изменить программу таким образом, чтобы частицы вращались в процессе полета. Взрыв должен быть применен в сцене, разработанной при выполнении лабораторной работы №14 согласно варианту задания.

4. Создать направленный взрыв. Направление взрыва выбрать произвольным. Увеличить сетку поверхности.

5. Добавить два новых положения камеры, которые бы максимально удобно и зрелищно демонстрировали направленность взрыва.

ХОД РАБОТЫ

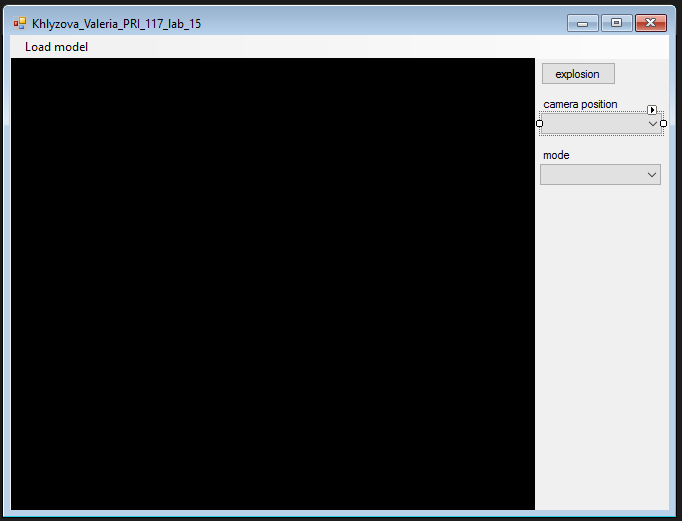


Рисунок 1 - Конструктор формы.

Обработчик формы:

using System;

using System.Windows.Forms;

using Tao.DevIl;

using Tao.FreeGlut;

using Tao.OpenGl;

namespace Khlyzova\_Valeria\_PRI\_117\_lab\_15

{

public partial class Form1 : Form

{

float global\_time = 0;

private float[,] camera\_date = new float[5, 7];

private Explosion BOOOOM\_1 = new Explosion(1, 10, 1, 300, 500);

Random rnd = new Random();

double a = 0, b = 0, c = -5, d = 0, zoom = 1;

int os\_x = 1, os\_y = 0, os\_z = 0;

anModelLoader Model = null;

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

{

if (openFileDialog1.ShowDialog() == DialogResult.OK)

{

Model = new anModelLoader();

Model.LoadModel(openFileDialog1.FileName);

RenderTimer.Start();

}

}

public Form1()

{

InitializeComponent();

AnT.InitializeContexts();

}

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

{

Glut.glutInit();

Glut.glutInitDisplayMode(Glut.GLUT\_RGB | Glut.GLUT\_DOUBLE | Glut.GLUT\_DEPTH);

Il.ilInit();

Il.ilEnable(Il.IL\_CONV\_PAL);

Gl.glClearColor(255, 255, 255, 1);

Gl.glViewport(0, 0, AnT.Width, AnT.Height);

Gl.glMatrixMode(Gl.GL\_PROJECTION);

Gl.glLoadIdentity();

Glu.gluPerspective(45, (float)AnT.Width / (float)AnT.Height, 0.1, 200);

Gl.glMatrixMode(Gl.GL\_MODELVIEW);

Gl.glLoadIdentity();

Gl.glEnable(Gl.GL\_DEPTH\_TEST);

Gl.glEnable(Gl.GL\_LIGHTING);

Gl.glEnable(Gl.GL\_LIGHT0);

Gl.glEnable(Gl.GL\_COLOR\_MATERIAL);

comboBox1.SelectedIndex = 1;

comboBox2.SelectedIndex = 0;

camera\_date[0, 0] = -3;

camera\_date[0, 1] = 0;

camera\_date[0, 2] = -20;

camera\_date[0, 3] = 0;

camera\_date[0, 4] = 1;

camera\_date[0, 5] = 0;

camera\_date[0, 6] = 0;

camera\_date[1, 0] = -3;

camera\_date[2, 1] = 2;

camera\_date[1, 2] = -20;

camera\_date[1, 3] = 30;

camera\_date[1, 4] = 1;

camera\_date[1, 5] = 0;

camera\_date[1, 6] = 0;

camera\_date[2, 0] = -3;

camera\_date[2, 1] = 2;

camera\_date[2, 2] = -20;

camera\_date[2, 3] = 30;

camera\_date[2, 4] = 1;

camera\_date[2, 5] = 1;

camera\_date[2, 6] = 0;

camera\_date[3, 0] = -3;

camera\_date[3, 1] = -3;

camera\_date[3, 2] = -20;

camera\_date[3, 3] = 70;

camera\_date[3, 4] = 0;

camera\_date[3, 5] = 1;

camera\_date[3, 6] = 0;

camera\_date[4, 0] = 3;

camera\_date[4, 1] = 2;

camera\_date[4, 2] = -30;

camera\_date[4, 3] = 1;

camera\_date[4, 4] = 1;

camera\_date[4, 5] = 1;

camera\_date[4, 6] = 1;

RenderTimer.Start();

}

private void RenderTimer\_Tick(object sender, EventArgs e)

{

global\_time += (float)RenderTimer.Interval / 1000;

Draw();

}

private void Draw()

{

if (comboBox2.SelectedIndex == 0)

{

Gl.glClearColor(255, 255, 255, 1);

}

else

{

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

}

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

Gl.glLoadIdentity();

if (comboBox2.SelectedIndex == 0)

{

Gl.glColor3d(0, 0, 0);

}

else

{

Gl.glColor3d(255, 255, 255);

}

Gl.glPushMatrix();

int camera = comboBox1.SelectedIndex;

Gl.glTranslated(camera\_date[camera, 0], camera\_date[camera, 1], camera\_date[camera, 2]);

Gl.glRotated(camera\_date[camera, 3], camera\_date[camera, 4], camera\_date[camera, 5], camera\_date[camera, 6]);

Gl.glPushMatrix();

Gl.glTranslated(a, b, c);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

DrawMatrix(30);

BOOOOM\_1.Calculate(global\_time);

if (Model != null)

Model.DrawModel();

Gl.glPopMatrix();

Gl.glPopMatrix();

Gl.glFlush();

AnT.Invalidate();

}

private void DrawMatrix(int x)

{

float quad\_size = 1;

Gl.glBegin(Gl.GL\_LINES);

for (int ax = -10; ax < x + 1; ax++)

{

Gl.glVertex3d(quad\_size \* ax, -5, 0);

Gl.glVertex3d(quad\_size \* ax, -5, quad\_size \* x);

}

for (int bx = 0; bx < x + 1; bx++)

{

Gl.glVertex3d(-10, -5, quad\_size \* bx);

Gl.glVertex3d(quad\_size \* x, -5, quad\_size \* bx);

}

Gl.glEnd();

}

private void button1\_Click(object sender, EventArgs e)

{

Random rnd = new Random();

BOOOOM\_1.SetNewPosition(5, 5, 5);

BOOOOM\_1.SetNewPower(rnd.Next(20));

BOOOOM\_1.Boooom(global\_time);

}

}

}

Класс Partilce:

namespace Khlyzova\_Valeria\_PRI\_117\_lab\_15

{

class Partilce

{

private float[] position = new float[3];

private float \_size;

private float \_lifeTime;

private float[] Grav = new float[3];

private float[] power = new float[3];

private float attenuation;

private float[] speed = new float[3];

private float LastTime = 0;

public Partilce(float x, float y, float z, float size, float lifeTime, float start\_time)

{

\_size = size;

\_lifeTime = lifeTime;

position[0] = x;

position[1] = y;

position[1] = z;

speed[0] = 0;

speed[1] = 0;

speed[2] = 0;

Grav[0] = 0;

Grav[1] = -9.8f;

Grav[2] = 0;

attenuation = 3.33f;

power[0] = 0;

power[0] = 0;

power[0] = 0;

LastTime = start\_time;

}

public void SetPower(float x, float y, float z)

{

power[0] = x;

power[1] = y;

power[2] = z;

}

public void InvertSpeed(int os, float attenuation)

{

speed[os] \*= -1 \* attenuation;

}

public float GetSize()

{

return \_size;

}

public void setAttenuation(float new\_value)

{

attenuation = new\_value;

}

public void UpdatePosition(float timeNow)

{

float dTime = timeNow - LastTime;

\_lifeTime -= dTime;

LastTime = timeNow;

for (int a = 0; a < 3; a++)

{

if (power[a] > 5)

{

power[a] -= attenuation \* dTime;

if (power[a] <= 0)

power[a] = 0;

}

position[a] += (speed[a] \* dTime + (Grav[a] + power[a]) \* dTime \* dTime);

speed[a] += (Grav[a] + power[a]) \* dTime;

}

}

public bool isLife()

{

if (\_lifeTime > 0)

{

return true;

}

else

{

return false;

}

}

public float GetPositionX()

{

return position[0];

}

public float GetPositionY()

{

return position[1];

}

public float GetPositionZ()

{

return position[2];

}

}

}

Класс Explosion:

using System;

using Tao.OpenGl;

namespace Khlyzova\_Valeria\_PRI\_117\_lab\_15

{

class Explosion

{

private float[] position = new float[3];

private float \_power;

private int MAX\_PARTICLES = 1000;

private int \_particles\_now;

private bool isStart = false;

private Partilce[] PartilceArray;

private bool isDisplayList = false;

private int DisplayListNom = 0;

public Explosion(float x, float y, float z, float power, int particle\_count)

{

position[0] = x;

position[1] = y;

position[2] = z;

\_particles\_now = particle\_count;

\_power = power;

if (particle\_count > MAX\_PARTICLES)

{

particle\_count = MAX\_PARTICLES;

}

PartilceArray = new Partilce[particle\_count];

}

public void SetNewPosition(float x, float y, float z)

{

position[0] = x;

position[1] = y;

position[2] = z;

}

public void SetNewPower(float new\_power)

{

\_power = new\_power;

}

private void CreateDisplayList()

{

DisplayListNom = Gl.glGenLists(1);

Gl.glNewList(DisplayListNom, Gl.GL\_COMPILE);

Gl.glPointSize(3);

Gl.glBegin(Gl.GL\_POINTS);

Gl.glVertex3d(0, 0, 0);

Gl.glEnd();

Gl.glEndList();

isDisplayList = true;

}

public void Boooom(float time\_start)

{

Random rnd = new Random();

if (!isDisplayList)

{

CreateDisplayList();

}

for (int ax = 0; ax < \_particles\_now; ax++)

{

PartilceArray[ax] = new Partilce(position[0], position[1], position[2], 5.0f, 10, time\_start);

int direction\_x = rnd.Next(1, 3);

int direction\_y = rnd.Next(1, 3);

int direction\_z = rnd.Next(1, 3);

if (direction\_x == 2)

direction\_x = -1;

if (direction\_y == 2)

direction\_y = -1;

if (direction\_z == 2)

direction\_z = -1;

float \_power\_rnd = rnd.Next((int)\_power / 20, (int)\_power);

PartilceArray[ax].setAttenuation(\_power / 2.0f);

PartilceArray[ax].SetPower(\_power\_rnd \* ((float)rnd.Next(100, 1000) / 1000.0f) \* direction\_x, \_power\_rnd \* ((float)rnd.Next(100, 1000) / 1000.0f) \* direction\_y, \_power\_rnd \* ((float)rnd.Next(100, 1000) / 1000.0f) \* direction\_z);

}

isStart = true;

}

public void Calculate(float time)

{

if (isStart)

{

for (int ax = 0; ax < \_particles\_now; ax++)

{

if (PartilceArray[ax].isLife())

{

PartilceArray[ax].UpdatePosition(time);

Gl.glPushMatrix();

float size = PartilceArray[ax].GetSize();

Gl.glTranslated(PartilceArray[ax].GetPositionX(), PartilceArray[ax].GetPositionY(), PartilceArray[ax].GetPositionZ());

Gl.glScalef(size, size, size);

Gl.glCallList(DisplayListNom);

Gl.glPopMatrix();

if (PartilceArray[ax].GetPositionY() < 0)

{

PartilceArray[ax].InvertSpeed(1, 0.6f);

}

}

}

}

}

}

}

Загрузка моделей и наложение текстур были рассмотрены в лабораторной работе №14.

Результат выполнения

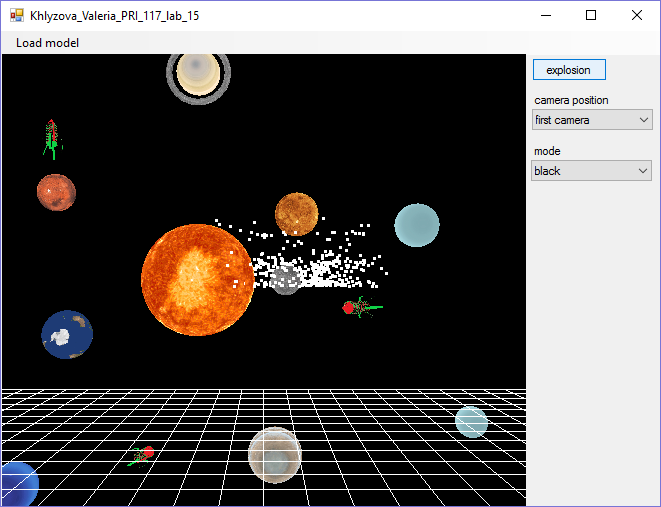


Рисунок 2 - Результат выполнения программы.

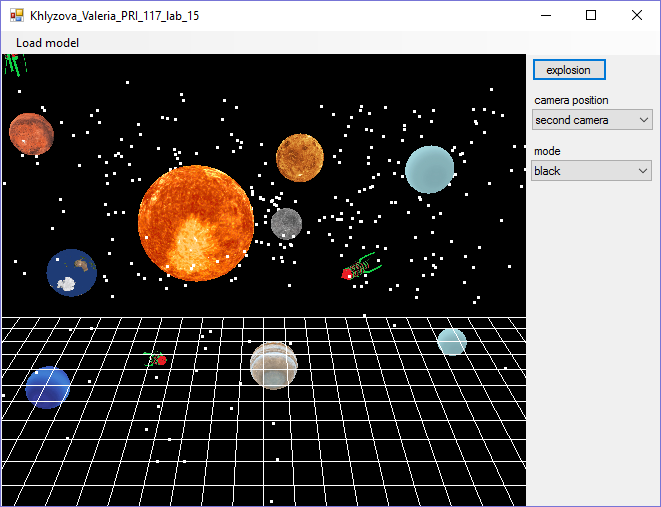


Рисунок 3 - Результат выполнения программы.

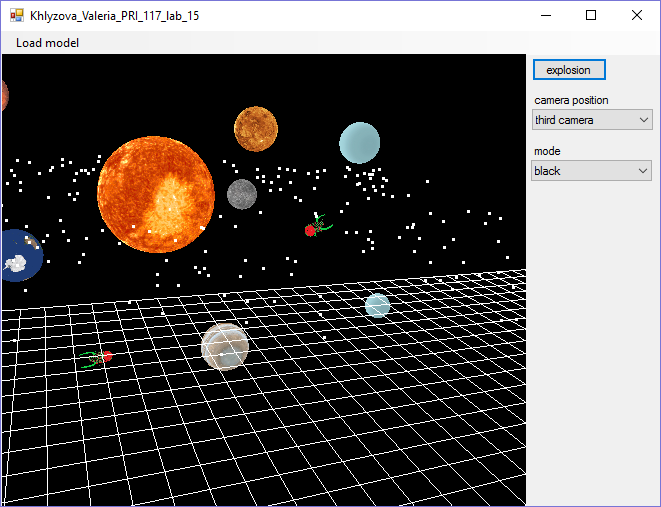


Рисунок 4 - Результат выполнения программы.

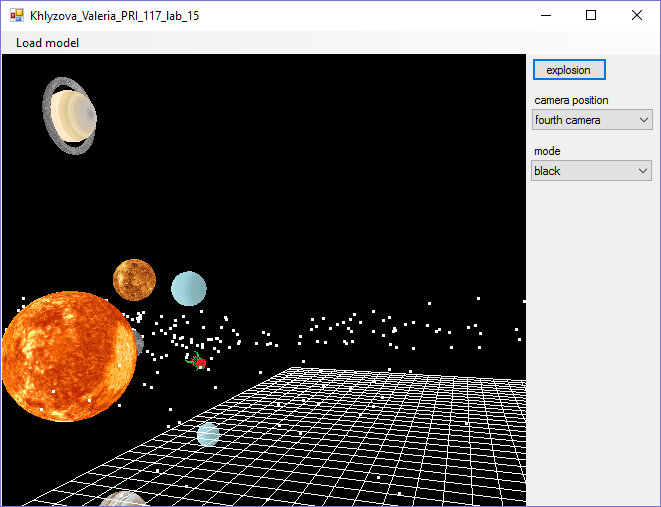


Рисунок 5 - Результат выполнения программы.

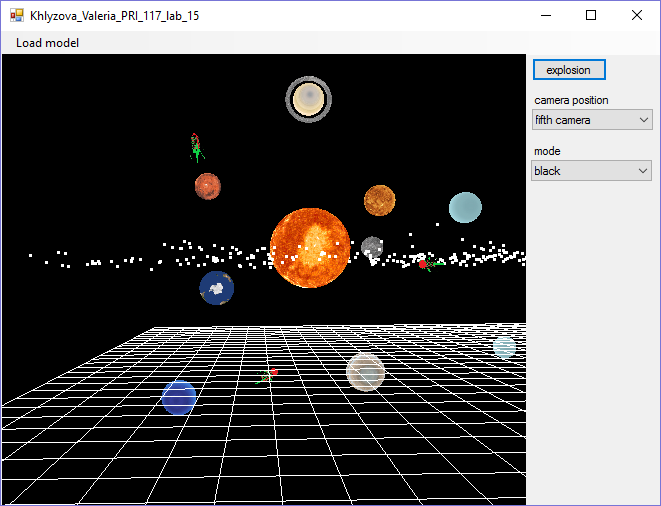


Рисунок 6 - Результат выполнения программы.

ВЫВОД

В ходе выполнения лабораторной работы были изучены методы использования в компьютерной графике 3D-объектов, не имеющих чётких геометрических границ и описываемых стохастическими алгоритмами.