Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ  
ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

|  |
| --- |
| Институт информационных технологий и анализа данных |

наименование института

|  |
| --- |
| Допускаю к защите  Руководитель:  З.А. Бахвалова |
| И.О. Фамилия |

Разработка системы управления частицами на языке C#

наименование темы

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе по дисциплине

|  |
| --- |
| Технологии программирования |

|  |
| --- |
| 1.030.00.00 ПЗ |
| обозначение документа |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Выполнил студент группы | ИСТб 20-3 |  |  |  | Ярков М.Д. |
|  | шифр группы |  | подпись |  | Фамилия И.О |
| Нормоконтроль |  |  |  |  | З.А. Бахвалова |
|  |  |  | подпись |  | Фамилия И.О |

|  |  |
| --- | --- |
| Курсовая работа защищена с оценкой |  |

Иркутск 2021 г.

**Министерство науки и высшего образования Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**ЗАДАНИЕ**

**НА КУРСОВУЮ РАБОТУ**

|  |  |  |
| --- | --- | --- |
| По курсу | Технологии программирования | |
| Студенту | Яркову Максиму Денисовичу | |
|  | (фамилия, инициалы) | |
| Тема работы: | Разработка системы управления частицами на языке C# | | |
| Исходные данные: | | Вариант 6, 9, дополнительное задание | |
| Рекомендуемая литература: | | | |
| 1. Троелсен, Эндрю Язык программирования C# 5.0 и платформа .NET 4.5 / Эндрю Троелсен. - М.: Вильямс, 2015. - 486 c  2. Система частиц, часть 1. [Электронный ресурс] // aqua.tealeaf.su : чаинка, 2021. URL: http://aqua.tealeaf.su/particle-system.html (дата обращения: 01.12.2021). | | | |
|  | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Графическая часть на | - | | листах. | |
| Дата выдачи задания |  | 23 / 11 / 2021 г. | | |
| Задание получил студент | | | |  | |  | Ярков М.Д. |
|  | | | | подпись | |  | Фамилия И.О. |

|  |  |
| --- | --- |
| Дата представления работы руководителю | 13 / 12 / 2021 г. |
| Руководитель курсовой работы |  |  | З.А. Бахвалова |
|  | подпись | Фамилия И.О. |

Содержание

[Введение 4](#_Toc91158781)

[1 Индивидуальный вариант 5](#_Toc91158782)

[2 Внешний вид главного окна 6](#_Toc91158783)

[3 Код входа 7](#_Toc91158784)

[4 Основной код программы 7](#_Toc91158785)

[5 Код частиц 9](#_Toc91158786)

[6 Код чёрной дыры 11](#_Toc91158787)

[7 Код анализатора аудиопотока 11](#_Toc91158788)

[8 Описание работы интерфейса 14](#_Toc91158789)

[Заключение 18](#_Toc91158790)

[Список использованных источников 19](#_Toc91158791)

# Введение

В данной работе необходимо разработать графический интерфейс для управления системой частиц.

Система частиц — используемый в компьютерной графике способ представления объектов, не имеющих чётких геометрических границ (различные облака, туманности, взрывы, струи пара, шлейфы от ракет, дым, снег, дождь и т. п.).

Система частиц состоит из определённого (фиксированного или произвольного) количества частиц. Математически каждая частица представляется как материальная точка с дополнительными атрибутами, такими как внешний вид реализуемого с помощью метода render, скоростью, запасом жизни и т. п.

В ходе работы программы каждая частица изменяет своё состояние по определённому, общему для всех частиц системы, закону. Например, частица может подвергаться воздействию гравитации, менять размер, цвет, скорость и так далее, и, после проведения всех расчётов, частица визуализируется.

Новые частицы испускаются так называемым «эмиттером». Эмиттером может быть точка, тогда новые частицы будут возникать в одном месте. Так можно смоделировать, например, взрыв: эмиттером будет его центр. Эмиттером может быть отрезок прямой или плоскость: например, частицы дождя или снега должны возникать на высоко расположенной горизонтальной плоскости. Эмиттером может быть и произвольный геометрический объект: в этом случае новые частицы будут возникать на всей его поверхности.

Для реализации используется язык C#. C# – это язык программирования, предназначенный для разработки самых разнообразных приложений. Язык C# прост, строго типизирован и объектно-ориентирован. Благодаря множеству нововведений C# обеспечивает возможность быстрой разработки приложений, но при этом сохраняет выразительность и элегантность, присущую си подобным языкам.

# 1 Индивидуальный вариант

6. Реализовать точку-счётчик частиц «чёрная дыра»:

* при клике левой кнопкой мыши добавлять новый счётчик
* при клике правой кнопкой мыши удалять счётчик
* при попадании частицы в счётчик увеличивать размер в зависимости от здоровья частицы, увеличивать счётчик на единицу и удалять частицу

9. Реализовать debug режим:

* в котором рисуются вектора скорости каждой частицы
* добавить ползунок для замедления симуляции и полной остановки
* добавить кнопочку для пошагового выполнения симуляции
* при наведении мышки на частицу, выводить её координаты и значение Life

Дополнительное задание. Реализовать различные типы генерации:

* Из центра
* Везде
* Под спектр звуковых частот

# 2 Внешний вид главного окна

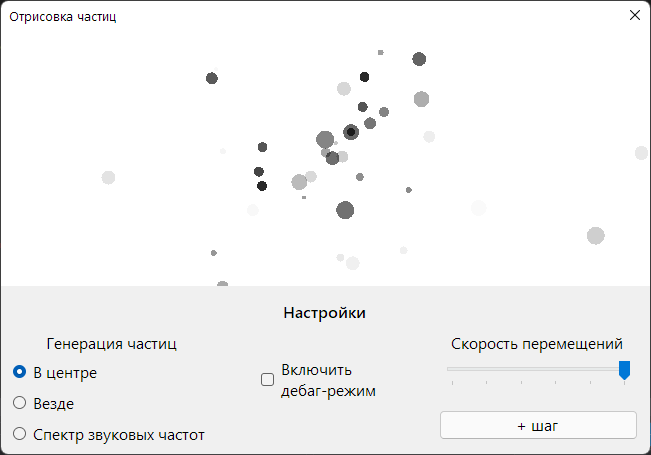


Рисунок 2.1 – Скриншот вешнего вида главного окна

# 3 Код входа

using System;

using System.Windows.Forms;

namespace CourseWork {

static class Program {

/// <summary>

/// Главная точка входа для приложения.

/// </summary>

[STAThread]

static void Main() {

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new DrawerForm());

}

}

}

# 4 Основной код программы

using System;

using System.Collections.Generic;

using System.Drawing;

using System.Text;

using System.Windows.Forms;

namespace CourseWork {

public partial class DrawerForm : Form {

List<Particle> particles = new List<Particle>(); // список частиц

List<BlackHole> blackHoles = new List<BlackHole>(); // список частиц

Analyzer analyzer = new Analyzer();

bool \_debug = false;

int \_ticks = 0, \_updateSpeed, \_generationType;

public DrawerForm() {

InitializeComponent();

drawing.Image = new Bitmap(drawing.Width, drawing.Height);

radioButton1.Checked = true;

// подтягивание инфы из интерфейса

OnSpeedChange(null, null);

OnDebugChange(null, null);

}

private void OnTick(object sender = null, EventArgs e = null) { // каждый тик

\_ticks++;

if ((\_ticks / \_updateSpeed > 0) || (sender.Equals(tickButton))) {

\_ticks %= \_updateSpeed;

UpdateParticles(); // обновляем частицы

UpdateBlackHoles(); // обновляем чёрные дыры

using (var g = Graphics.FromImage(drawing.Image)) {

g.Clear(Color.White);

foreach (var particle in particles) { // сначала рисуем частицы

particle.Draw(g, \_debug);

}

foreach (var blackHole in blackHoles) { // потом дыры

blackHole.Draw(g);

}

foreach (var blackHole in blackHoles) { // потом текст, чтоб не было наслоения черноты на информацию

blackHole.DrawInfo(g);

}

}

}

drawing.Invalidate(); // отображаем рисунок

}

private void UpdateParticles() {

if (\_generationType != 2) {

if (particles.Count < 500) {

Particle particle = new Particle(drawing.Image.Width, drawing.Image.Height, \_generationType);

particles.Add(particle); // добавляю в список

}

}

else {

byte[] rawData = analyzer.getRawData();

for (int i = 0; i < rawData.Length; i++) {

Particle particle = new Particle(5+i\*10, 260-rawData[i], \_generationType);

particles.Add(particle); // добавляю в список

}

}

for (int i = particles.Count-1; i > -1; i--) {

var particle = particles[i];

particle.Life -= \_generationType == 2 ? 4 : 1; // уменьшаю здоровье

if (particle.Life < 0) // если здоровье кончилось

particles.Remove(particle);

else

particle.Move();

}

}

private void UpdateBlackHoles() {

foreach (var blackHole in blackHoles) {

foreach (var particle in particles) {

if (blackHole.IsTouching(particle))

blackHole.Eat(particle);

}

}

}

private void MouseDebug(object sender, MouseEventArgs e) {

if (\_debug) {

int x = e.X, y = e.Y;

using (var g = Graphics.FromImage(drawing.Image)) {

foreach (var particle in particles)

particle.DrawInfo(g, x, y);

}

drawing.Invalidate(); // отображаем рисунок

}

}

private void MouseInteraction(object sender, MouseEventArgs e) {

if (e.Button == MouseButtons.Left)

blackHoles.Add(new BlackHole(e.X, e.Y));

if (e.Button == MouseButtons.Right) {

for (int i = 0; i < blackHoles.Count; i++) {

var blackHole = blackHoles[i];

var x = Math.Abs(e.X - blackHole.X);

var y = Math.Abs(e.Y - blackHole.Y);

var lenght = Math.Sqrt(x \* x + y \* y);

if (lenght < blackHole.Radius) {

blackHoles.Remove(blackHole);

i--;

}}}}

private void OnDebugChange(object sender, EventArgs e) {

\_debug = debugCheckBox.Checked;

}

private void OnSpeedChange(object sender, EventArgs e) {

\_updateSpeed = speedBar.Maximum - speedBar.Value;

if (\_updateSpeed == speedBar.Maximum)

\_updateSpeed = -1; // полная остановка

else

\_updateSpeed = (int)Math.Pow(2, \_updateSpeed);

}

private void GenerateInCenter(object sender, EventArgs e) {

\_generationType = 0;

}

private void GenerateEverywhere(object sender, EventArgs e) {

\_generationType = 1;

}

private void GenerateBySound(object sender, EventArgs e) {

\_generationType = 2;

}

}

}

# 5 Код частиц

using System;

using System.Drawing;

namespace CourseWork {

class Particle {

public int Radius;

public float X, Y, Direction, Speed, Life;

private static readonly Random rand = new Random();

public Particle(int x, int y, int generationType) {

if (generationType == 0) {

X = x/2; Y = y/2;

}

else if (generationType == 1) {

X = rand.Next(x); Y = rand.Next(y);

}

else { // (generationType == 2)

X = x; Y = y;

}

if (generationType == 2) {

Direction = 270;

Speed = 2;

Radius = 4;

Life = 100;

}

else { // создание рандомных параметров

Direction = rand.Next(360);

Speed = rand.Next(1, 10);

Radius = rand.Next(2, 10);

Life = rand.Next(20, 100);

}

}

public void Move() {

var directionInRadians = Direction / 180 \* Math.PI;

X += (float)(Speed \* Math.Cos(directionInRadians));

Y -= (float)(Speed \* Math.Sin(directionInRadians));

}

public void DrawInfo(Graphics g, int \_x, int \_y) {

var x = Math.Abs(X - \_x);

var y = Math.Abs(Y - \_y);

var lenght = Math.Sqrt(x \* x + y \* y);

if (lenght < Radius) {

var b = new SolidBrush(Color.SkyBlue);

g.FillRectangle(b, X, Y, 50, 50);

StringFormat stringFormat = new StringFormat();

stringFormat.Alignment = StringAlignment.Center;

stringFormat.LineAlignment = StringAlignment.Center;

g.DrawString("X: " + (int)X, new Font("Arial", 10), Brushes.Black, X + 25, Y + 10, stringFormat);

g.DrawString("Y: " + (int)Y, new Font("Arial", 10), Brushes.Black, X + 25, Y + 25, stringFormat);

g.DrawString("Life: " + (int)Life, new Font("Arial", 10), Brushes.Black, X + 25, Y + 40, stringFormat);

}

}

public void Draw(Graphics g, bool debug) {

int alpha = (int)(Math.Min(1f, Life / 100) \* 255);

var color = Color.FromArgb(alpha, Color.Black);

var b = new SolidBrush(color); // создали кисть для рисования

g.FillEllipse(b, X - Radius, Y - Radius, Radius \* 2, Radius \* 2); // отрисовали

if (debug) {

var directionInRadians = -Direction / 180 \* Math.PI;

g.DrawLine(new Pen(Color.OrangeRed, 2), X, Y, (int)(X + Math.Cos(directionInRadians) \* 5), (int)(Y + Math.Sin(directionInRadians) \* 5));

}

b.Dispose(); // почистили объект из памяти, чтобы сборщику мусора было попроще

}

public float Feed() {

float life = Life;

Life = 0;

return life;

}

}

}

# 6 Код чёрной дыры

using System;

using System.Drawing;

namespace CourseWork {

class BlackHole {

public int X, Y, Eaten;

public float Radius = 20, Heaviness = 0;

public BlackHole(int x, int y) {

X = x; Y = y;

}

public void Draw(Graphics g) {

var b = new SolidBrush(Color.Black); // создали кисть для рисования

g.FillEllipse(b, X - Radius, Y - Radius, Radius \* 2, Radius \* 2); // отрисовали

b.Dispose(); // почистили объект из памяти, чтобы сборщику мусора было попроще

}

public void DrawInfo(Graphics g) {

StringFormat stringFormat = new StringFormat();

stringFormat.Alignment = StringAlignment.Center;

stringFormat.LineAlignment = StringAlignment.Center;

g.DrawString("" + Eaten, new Font("Arial", 10), Brushes.White, X, Y, stringFormat);

}

public void Eat(Particle p) {

Heaviness += p.Feed() / 100;

Radius = 20 + (float)Math.Sqrt(Heaviness);

Eaten++;

}

public bool IsTouching(Particle particle) {

var x = Math.Abs(particle.X - X);

var y = Math.Abs(particle.Y - Y);

var lenght = Math.Sqrt(x \* x + y \* y);

return lenght < Radius;

}

}

}

# 7 Код анализатора аудиопотока

using System;

using System.Collections.Generic;

using System.Windows.Forms;

using Un4seen.Bass;

using Un4seen.BassWasapi;

namespace CourseWork {

internal class Analyzer {

private WASAPIPROC \_process;

public Analyzer() {

\_process = new WASAPIPROC(Process);

int availableDevice = -1;

for (int i = 0; i < BassWasapi.BASS\_WASAPI\_GetDeviceCount(); i++) {

var device = BassWasapi.BASS\_WASAPI\_GetDeviceInfo(i);

if (device.IsEnabled && device.IsLoopback) {

availableDevice = i;

break;

}

}

if (availableDevice == -1)

throw new Exception("Аудиопоток недоступен");

Bass.BASS\_SetConfig(BASSConfig.BASS\_CONFIG\_UPDATETHREADS, false);

if (!Bass.BASS\_Init(0, 44100, BASSInit.BASS\_DEVICE\_DEFAULT, IntPtr.Zero))

throw new Exception(Bass.BASS\_ErrorGetCode().ToString());

if (!BassWasapi.BASS\_WASAPI\_Init(availableDevice, 0, 0, BASSWASAPIInit.BASS\_WASAPI\_BUFFER, 1f, 0.05f, \_process, IntPtr.Zero))

MessageBox.Show(Bass.BASS\_ErrorGetCode().ToString());

BassWasapi.BASS\_WASAPI\_Start();

}

public byte[] getRawData() {

int b0 = 0;

float[] fftBufferData = new float[8192];

List<byte> \_spectrumdata = new List<byte>();

if (BassWasapi.BASS\_WASAPI\_GetData(fftBufferData, (int)BASSData.BASS\_DATA\_FFT8192) < -1)

return \_spectrumdata.ToArray();

for (int x = 0; x < 64; x++) {

float peak = 0;

int b1 = (int)Math.Pow(2, x \* 10.0 / 63);

if (b1 > 1023) b1 = 1023;

if (b1 <= b0) b1 = b0 + 1;

for (; b0 < b1; b0++) {

if (peak < fftBufferData[1 + b0])

peak = fftBufferData[1 + b0];

}

int y = (int)(1.5 \* Math.Sqrt(peak) \* 255);

y = y > 255 ? 255 : (y < 0 ? 0 : y);

\_spectrumdata.Add((byte)y);

}

return \_spectrumdata.ToArray();

}

private int Process(IntPtr buffer, int length, IntPtr user) { // WASAPI callback, required for continuous recording

return length;

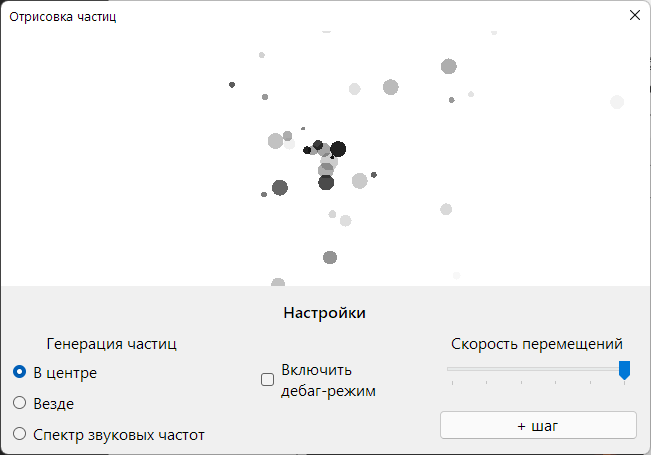
}

}

}

# 8 Описание работы интерфейса

Можно выбрать тип отрисовки частиц, выбрав параметр среди трёх типов **Генерации частиц**:

  
Рисунок 7.1 – Скриншот внешнего вида программы при генерации частиц в центре

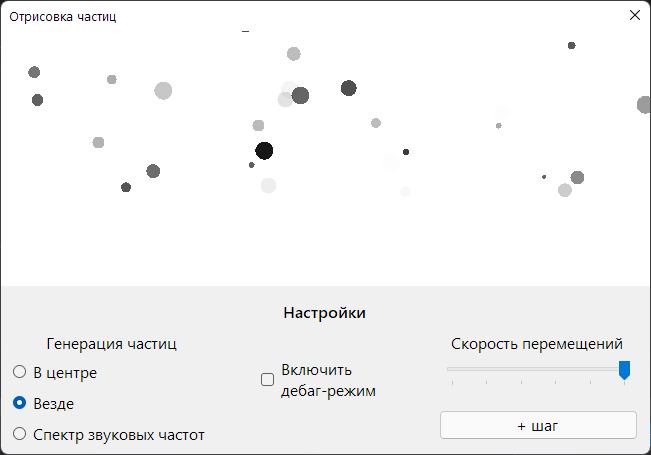


Рисунок 7.2 – Скриншот внешнего вида программы при генерации частиц везде

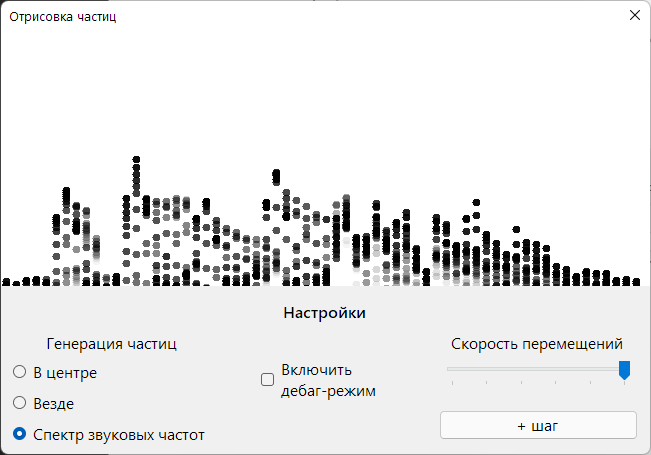


Рисунок 7.3 – Скриншот внешнего вида программы при генерации частиц в зависимости от спектра звуковых частот

Меняя ползунок **Скорость перемещений,** мы управляем скоростью частиц:

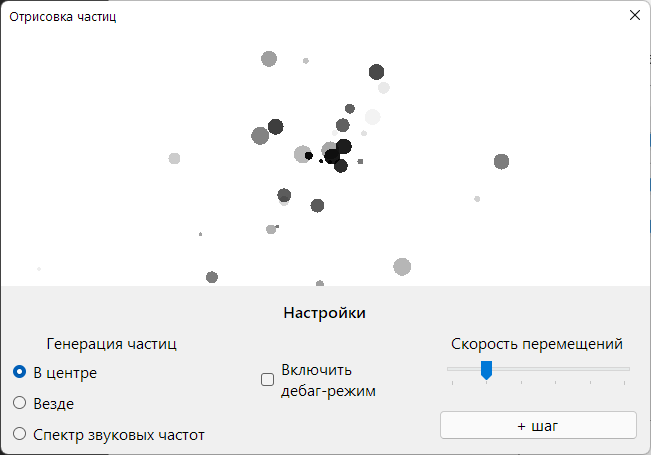


Рисунок 7.4 – Скриншот внешнего вида программы с изменённой скоростью перемещений

Нажимая на кнопку **+ шаг,** мы перемещаем частицы вручную. Это удобно, если скорость перемещений на нуле:

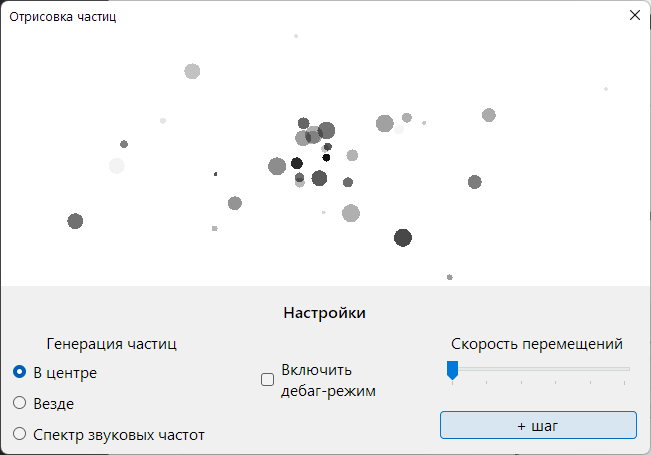


Рисунок 7.5 – Скриншот внешнего вида программы с выделенной кнопкой «+ шаг»

Включая чекбокс **Включить дебаг-режим,** можно видеть направления движения частиц и при наведении мыши узнавать значения частиц:

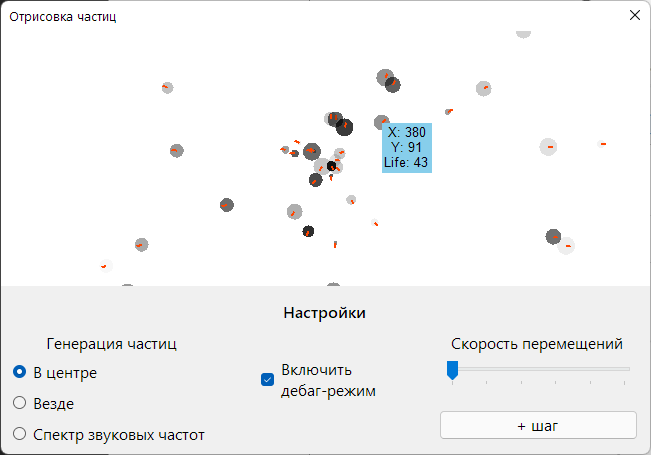


Рисунок 7.6 – Скриншот внешнего вида программы с включённым дебаг-режимом и наведении на одну из частиц

При клике на поле левой кнопкой мыши можно добавить **чёрную дыру**, а правой удалить. При попадании в неё частицы она увеличивается в зависимости от здоровья частицы и увеличивает счётчик:

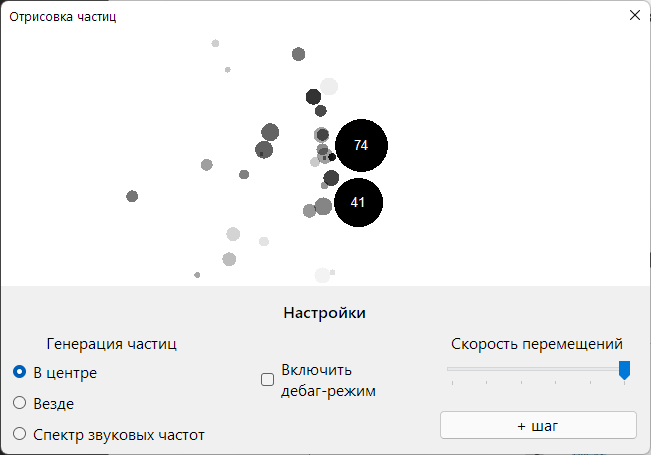


Рисунок 7.7 – Скриншот внешнего вида программы с добавленными чёрными дырами

# Заключение

В рамках курсовой работы было разработано приложение, позволяющее управлять системой частиц. В приложении реализованы все возможности, подразумевающиеся заданием.

Программа была успешно протестирована, ошибок выявлено не было. Все поставленные в рамках курсовой работы задачи были выполнены.

В рамках выполнения данной работы были получены знания и умения для работы с объектно-ориентированным языком программирования C#, разработкой графических приложений с отрисовкой графики и эмуляцией динамических систем.

# Список использованных источников

1. Троелсен, Эндрю Язык программирования C# 5.0 и платформа .NET 4.5 / Эндрю Троелсен. - М.: Вильямс, 2015. - 486 c.
2. Вагнер, Билл С# Эффективное программирование / Билл Вагнер. - М.: ЛОРИ, 2013. - 320 c.
3. Ишкова, Э. А. Самоучитель С#. Начала программирования / Э.А. Ишкова. - М.: Наука и техника, 2013. - 496 c.
4. Биллинг В.А. Основы программирования на C#. - Т.: Интернет-университет информационных технологий, Бином, 2012. - 488 с.
5. Евдокимов П.В. C# на примерах. - М.: Наука и техника,2016. - 304 с.
6. Нейгел К., Ивьен Б., Глинн Д., Уотсон К., Скиннер М. C# 4.0 и платформа .NET 4, 2011.
7. Система частиц, часть 1. [Электронный ресурс] // aqua.tealeaf.su: чаинка, 2021. URL: http://aqua.tealeaf.su/particle-system.html (дата обращения: 01.12.2021).