Институт приборостроения, автоматизации и информационных технологий

Кафедра информационных систем и цифровых технологий

Дисциплина «Выпуск и сопровождение программных продуктов»

Отчет к лабораторной работе № 3

«Генерация сетки лабиринта – или же меша»

Выполнил:

Василения Иван Валерьевич

Приняла:

Булгакова С.А.

Орёл, 2025г

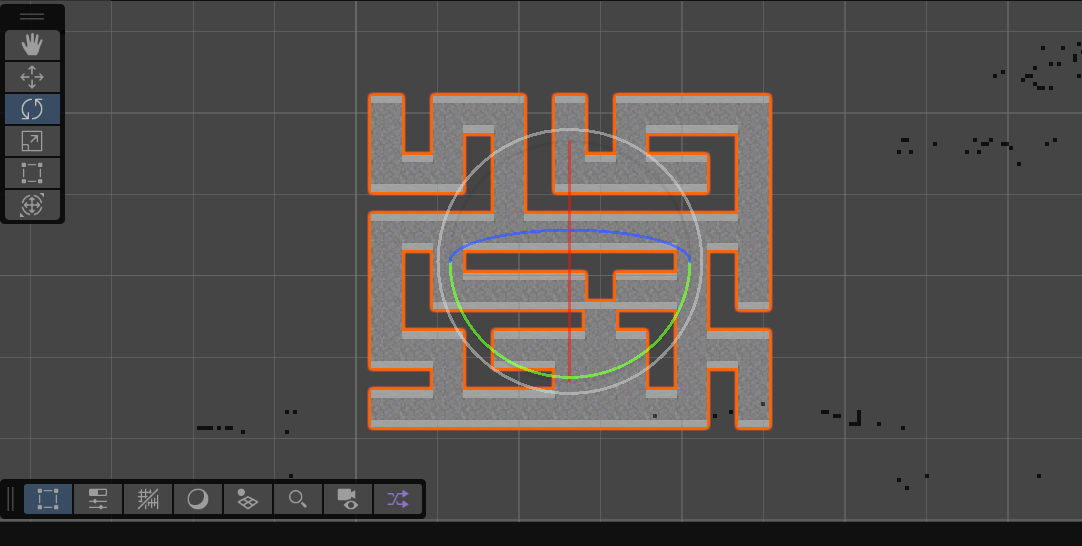


Рисунок 1 – Пример генерации лабиринта

Листинг файла MazeMeshGenerator.cs:

using System.Collections.Generic;

using UnityEngine;

public class MazeMeshGenerator

{

    // generator params

    public float width;     // how wide are hallways

    public float height;    // how tall are hallways

    public MazeMeshGenerator()

    {

        width = 3.75f;

        height = 3.5f;

    }

    public Mesh FromData(int[,] data)

    {

        Mesh maze = new Mesh();

        List<Vector3> newVertices = new List<Vector3>();

        List<Vector2> newUVs = new List<Vector2>();

        maze.subMeshCount = 2;

        List<int> floorTriangles = new List<int>();

        List<int> wallTriangles = new List<int>();

        int rMax = data.GetUpperBound(0);

        int cMax = data.GetUpperBound(1);

        float halfH = height \* .5f;

        //4

        for (int i = 0; i <= rMax; i++)

        {

            for (int j = 0; j <= cMax; j++)

            {

                if (data[i, j] != 1)

                {

                    // floor

                    AddQuad(Matrix4x4.TRS(

                        new Vector3(j \* width, 0, i \* width),

                        Quaternion.LookRotation(Vector3.up),

                        new Vector3(width, width, 1)

                    ), ref newVertices, ref newUVs, ref floorTriangles);

                    // ceiling

                    AddQuad(Matrix4x4.TRS(

                        new Vector3(j \* width, height, i \* width),

                        Quaternion.LookRotation(Vector3.down),

                        new Vector3(width, width, 1)

                    ), ref newVertices, ref newUVs, ref floorTriangles);

                    // walls on sides next to blocked grid cells

                    if (i - 1 < 0 || data[i-1, j] == 1)

                    {

                        AddQuad(Matrix4x4.TRS(

                            new Vector3(j \* width, halfH, (i-.5f) \* width),

                            Quaternion.LookRotation(Vector3.forward),

                            new Vector3(width, height, 1)

                        ), ref newVertices, ref newUVs, ref wallTriangles);

                    }

                    if (j + 1 > cMax || data[i, j+1] == 1)

                    {

                        AddQuad(Matrix4x4.TRS(

                            new Vector3((j+.5f) \* width, halfH, i \* width),

                            Quaternion.LookRotation(Vector3.left),

                            new Vector3(width, height, 1)

                        ), ref newVertices, ref newUVs, ref wallTriangles);

                    }

                    if (j - 1 < 0 || data[i, j-1] == 1)

                    {

                        AddQuad(Matrix4x4.TRS(

                            new Vector3((j-.5f) \* width, halfH, i \* width),

                            Quaternion.LookRotation(Vector3.right),

                            new Vector3(width, height, 1)

                        ), ref newVertices, ref newUVs, ref wallTriangles);

                    }

                    if (i + 1 > rMax || data[i+1, j] == 1)

                    {

                        AddQuad(Matrix4x4.TRS(

                            new Vector3(j \* width, halfH, (i+.5f) \* width),

                            Quaternion.LookRotation(Vector3.back),

                            new Vector3(width, height, 1)

                        ), ref newVertices, ref newUVs, ref wallTriangles);

                    }

                }

            }

        }

        maze.vertices = newVertices.ToArray();

        maze.uv = newUVs.ToArray();

        maze.SetTriangles(floorTriangles.ToArray(), 0);

        maze.SetTriangles(wallTriangles.ToArray(), 1);

        //5

        maze.RecalculateNormals();

        return maze;

    }

    //1, 2

    private void AddQuad(Matrix4x4 matrix, ref List<Vector3> newVertices,

        ref List<Vector2> newUVs, ref List<int> newTriangles)

    {

        int index = newVertices.Count;

        // corners before transforming

        Vector3 vert1 = new Vector3(-.5f, -.5f, 0);

        Vector3 vert2 = new Vector3(-.5f, .5f, 0);

        Vector3 vert3 = new Vector3(.5f, .5f, 0);

        Vector3 vert4 = new Vector3(.5f, -.5f, 0);

        newVertices.Add(matrix.MultiplyPoint3x4(vert1));

        newVertices.Add(matrix.MultiplyPoint3x4(vert2));

        newVertices.Add(matrix.MultiplyPoint3x4(vert3));

        newVertices.Add(matrix.MultiplyPoint3x4(vert4));

        newUVs.Add(new Vector2(1, 0));

        newUVs.Add(new Vector2(1, 1));

        newUVs.Add(new Vector2(0, 1));

        newUVs.Add(new Vector2(0, 0));

        newTriangles.Add(index+2);

        newTriangles.Add(index+1);

        newTriangles.Add(index);

        newTriangles.Add(index+3);

        newTriangles.Add(index+2);

        newTriangles.Add(index);

    }

}