МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний аерокосмічний університет ім. М. Є. Жуковського

«Харківський авіаційний інститут»

Факультет програмної інженерії та бізнесу

Кафедра інженерії програмного забезпечення

Практичні роботи

Minor *«Розробник ігрових додатків»*

дисципліна *«Програмування віртуальної реальності»*

(назва дисципліни)

Виконав: студент 4 курсу групи 632ПСТ

напряму підготовки (спеціальності):

121 інженерія програмного забезпечення

(шифр і назва напряму підготовки / спеціальності)

Бреславець М.О.

(прізвище й ініціали студента)

Прийняв: *доц. каф 603, к.т.н, Лучшев П.О.*

(посада, науковий ступінь, прізвище й ініціали)

Національна шкала:

Кількість балів:

Оцінка ECTS:

Зміст

[1. Формування простору віртуальної реальності 3](#_Toc178945056)

[1.1. Моделювання та генерація ландшафту 3](#_Toc178945057)

[1.2. Моделювання поверхні води 5](#_Toc178945058)

[2. Фотореалістичність об'єктів віртуальної реальності 6](#_Toc178945059)

[2.1. Перспективна проекція та управління точкою зору 6](#_Toc178945060)

[2.2. Використання текстур 6](#_Toc178945061)

[3. Створення та моделювання об'єктів віртуальної реальності 7](#_Toc178945062)

[3.1. Назва роботи 7](#_Toc178945063)

[4. Сприйняття віртуальної реальності людиною 8](#_Toc178945064)

[4.1. Назва роботи 8](#_Toc178945065)

[5. Зв'язок віртуального та реального світу 9](#_Toc178945066)

[5.1. Назва роботи 9](#_Toc178945067)

[Загальний перелік посилань 10](#_Toc178945068)

[Додаток А. Лістинг програми до практичної роботи № 1.1 11](#_Toc178945069)

[Додаток Б. Лістинг програми до практичної роботи № 2.1 12](#_Toc178945070)

[Додаток ?. Лістинг програми до практичної роботи №? 13](#_Toc178945071)

# 1. Формування простору віртуальної реальності

## 1.1. Моделювання та генерація ландшафту

### Завдання

Обговорити з викладачем та вибрати один із існуючих підходів до формування віртуальної моделі ландшафту. Деякі з методів розглянуті у теоретичному введенні.

Погодити інструментальні засоби та розробити програму для формування та відображення на екрані тривимірної поверхні, яка за допомогою обраних методів моделює ландшафт. Генерована поверхня повинна виводитися на екран у двох режимах: у вигляді каркасної моделі (рис. 1.1) та з градієнтним

### Хід виконання

Роботу виконав як за допомогою вже стандартного об’єкта ігрового простору Unity 3D. «terrain», так і за допомогою пустого об’єкта, як кажуть з нуля. Отже почнемо із використання стандартного об’єкта «terrain». На рисунку 1-3 – зображений вигляд сцени.

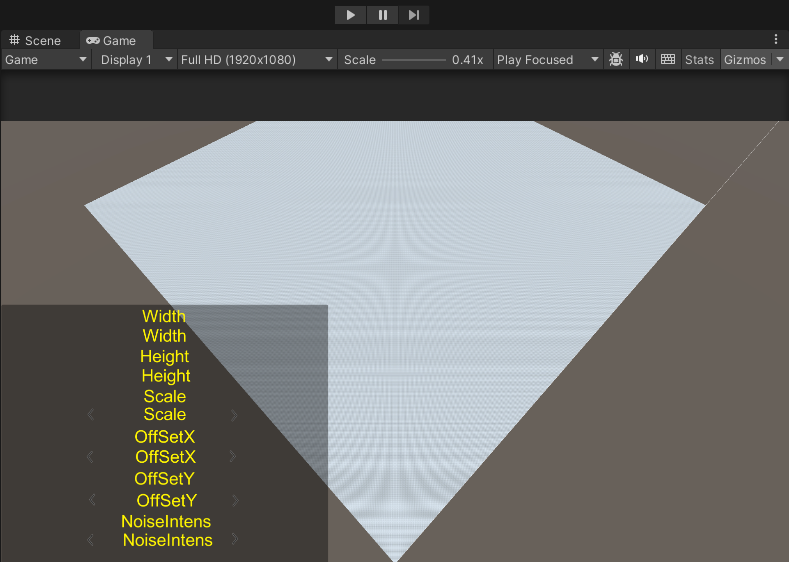


Рисунок 1 – Вигляд сцени до запуску програми.

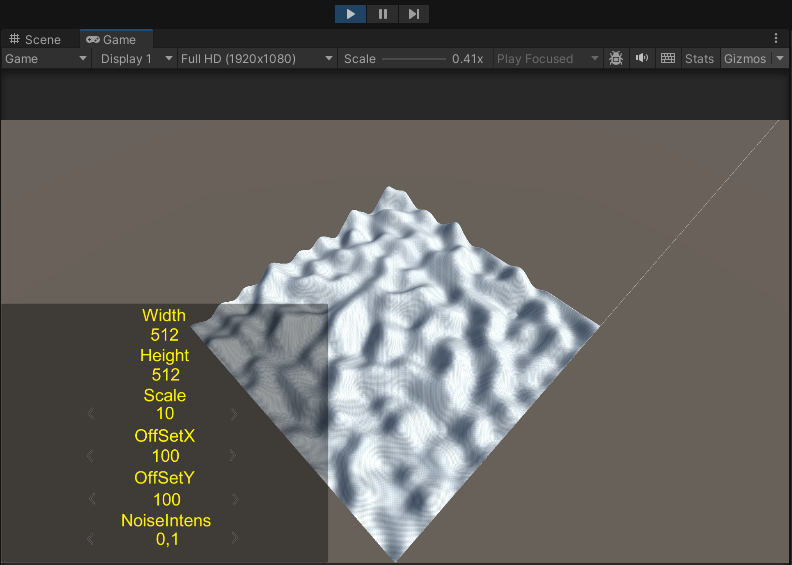


Рисунок 2 – Вигляд сцени після запуску програми.

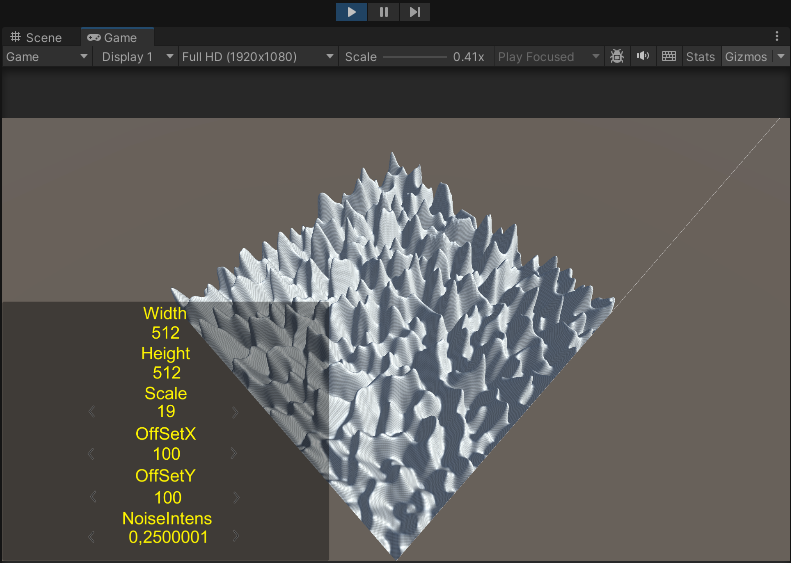


Рисунок 3 – Вигляд сцени при зміні налаштування.

Генерація ландшафту відбувається тому що ігровий об’єкт «terrain» має компонент, тобто скрипт, який і виконує генерацію початкового рельєфу, а також містить декілька методів для взаємодії зміни налаштувань генерації із кнопками на «Canvas», текст скрипту надаю в додатку А.

Тепер перейдемо до більш цікавішого, а саме побудова ландшафту з нуля, для цього був створений пустий об’єкт на сцені, до нього були додані такі компоненти як «Mesh Filter», «Mesh Renderer» та головне це скрипти:

* «Mesh Generator» - будова сітки з трикутних мешів, далі за допомогою шума зміна їх в просторі (та градієнта розкраска мешів в залежності від їх розміщення(вгорі чи внизу )).
* «Mesh Control Penal» - для взаємодії з елементами керування розміру карти (в залежності від цього параметру скрипт керує висотою камери – автоматично), та інтенсивності шуму від користувача, під час роботи програми.

Вміст скриптів додам відповідно в додаток А.

На рисунку 4-6 – зображений вигляд сцени.

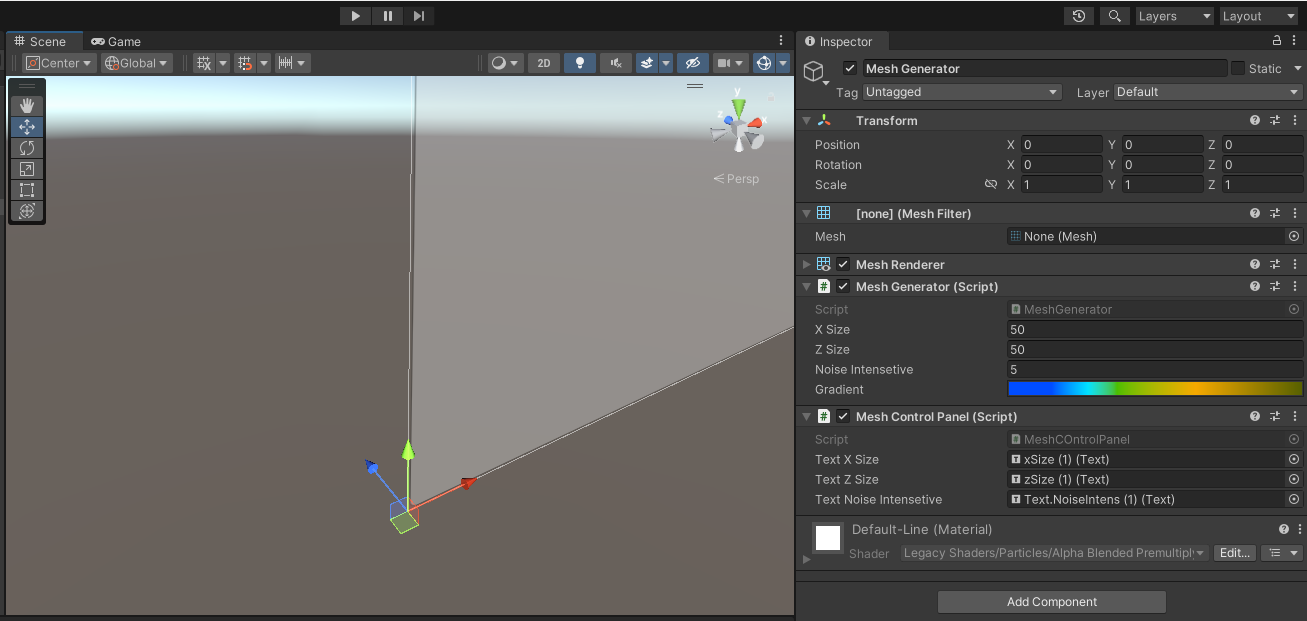


Рисунок 4 – Вигляд сцени до запуску програми.

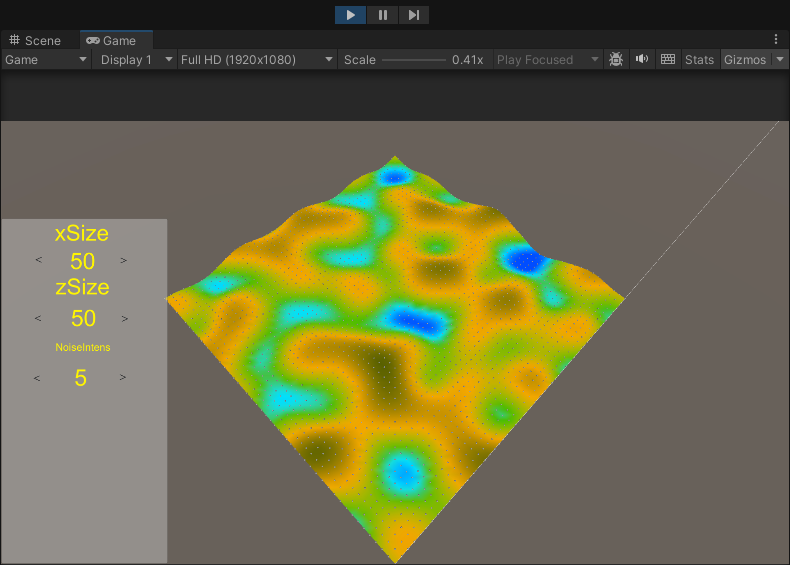


Рисунок 5 – Вигляд сцени після запуску програми.

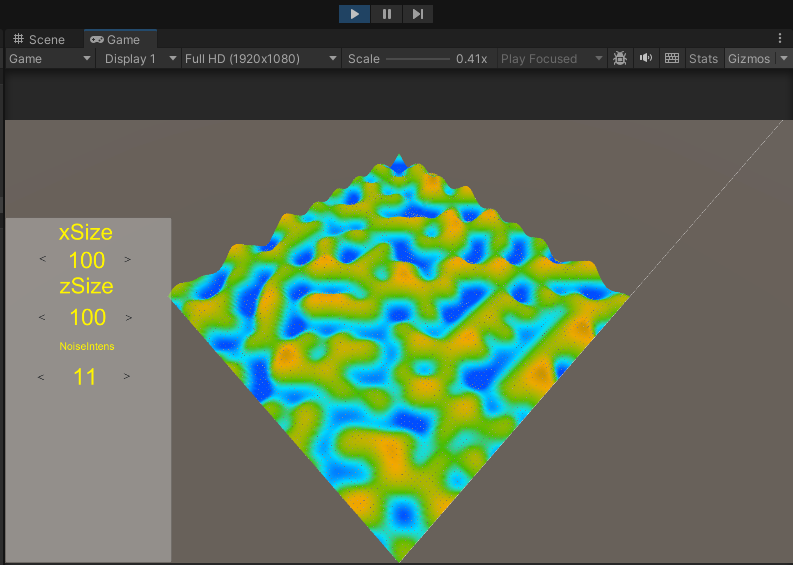


Рисунок 6 – Вигляд сцени при зміні налаштування генерації.

### Контроль виконання вимог та елементів завдання

В результаті виконання практичної роботи були повністю виконані елементи базового рівня та підвищеного рівня складності, що відображено в таблиці 1.1.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Таблиця 1.1 | | | | |
| № | Складність | Вимоги до роботи | Бали | Зроблено |
|  | Базовий рівень | Формування ландшафту у вигляді каркасної моделі | 2 | 2 |
|  | Градієнтне забарвлення поверхні залежно від висоти ландшафту | 2 | 2 |
|  | ООП. Реалізація об'єкта, що моделюється, у вигляді одного або декількох класів власної розробки | 1 | 1 |
|  | Підвищений рівень | Інтерактивне керування параметрами модельованого ландшафту (порожнина, рівень деталізації, розмальовка тощо) | 3 | 3 |

## 1.2. Моделювання поверхні води

### Завдання

Загальне завдання та варіант

### Теоретичні відомості

Самостійно вивчені засоби, методи, алгоритми та інші елементи віртуальної реальності, які були використані в роботі з посиланнями на джерело).

### Результати виконання практичної роботи

Розв'язання завдання

Перерахування вимог базового та підвищеного рівня складності з таблиці 2.1 з копіями екрана, які демонструють виконання поставленого завдання та/або з посиланнями на додатки та рядки коду, в яких ці вимоги реалізовані

### Контроль виконання вимог та елементів завдання

В результаті виконання практичної роботи були повністю виконані елементи базового рівня та частково підвищеного рівня складності, що відображено в таблиці 1.2.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Таблиця 1.2 | | | | |
| № з/п | Складність | Вимоги | Бали | Зроблено |
|  | Базовий рівень |  |  |  |
|  |  |  |  |
|  | Підвищений рівень |  |  |  |
|  |  |  |  |

# 2. Фотореалістичність об'єктів віртуальної реальності

## 2.1. Перспективна проекція та управління точкою зору

### Завдання

### Теоретичні відомості

### Результати виконання практичної роботи

Розв'язання завдання

### Контроль виконання вимог та елементів завдання

В результаті виконання практичної роботи були повністю виконані елементи базового рівня та частково підвищеного рівня складності, що відображено в таблиці 2.1.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Таблиця 2.1 | | | | |
| № з/п | Складність | Вимоги | Бали | Зроблено |
|  | Базовий рівень |  |  |  |
|  |  |  |  |
|  | Підвищений рівень |  |  |  |
|  |  |  |  |

## 2.2. Використання текстур

### Завдання

### Теоретичні відомості

### Результати виконання практичної роботи

### Контроль виконання вимог та елементів завдання

# 3. Створення та моделювання об'єктів віртуальної реальності

## 3.1. Назва роботи

### Завдання

### Теоретичні відомості

### Результати виконання практичної роботи

### Контроль виконання вимог та елементів завдання

# 4. Сприйняття віртуальної реальності людиною

## 4.1. Назва роботи

### Завдання

### Теоретичні відомості

### Результати виконання практичної роботи

### Контроль виконання вимог та елементів завдання

# 5. Зв'язок віртуального та реального світу

## 5.1. Назва роботи

### Завдання

### Теоретичні відомості

### Результати виконання практичної роботи

### Контроль виконання вимог та елементів завдання

# Загальний перелік посилань

1. Microsoft. glDrawArrays function [Електронний ресурс] / Microsoft – Режим доступу до ресурсу: <https://learn.microsoft.com/en-us/windows/win32/opengl/gldrawarrays>.
2. Diamond-square algorithm [Електронний ресурс] – Режим доступу до ресурсу: <https://en.wikipedia.org/wiki/Diamond-square_algorithm>.

# Додаток А. Лістинг програми до практичної роботи № 1.1

### Код файлу (TerreinGen.cs)

using System.Collections;

using System.Collections.Generic;

using UnityEngine.UI;

using UnityEngine;

public class TerreinGen : MonoBehaviour

{

public Text widthCan, heightCan, scaleCan, offsetXCan, offsetYCan, noiseIntensityCan;

private int width = 512; // Width of the terrain

private int height = 512; // Height of the terrain

private float scale = 10f; // Scale of the terrain

private float offsetX = 100f; // X offset for noise

private float offsetY = 100f; // Y offset for noise

private float noiseIntensity = 0.1f; //Intensity of the noise

Terrain terrain;

//Renderer render;

// public Material material;

private void Start()

{

terrain = GetComponent<Terrain>();

//render = GetComponent<Renderer>();

//render.material = material;

NewTerrain();

OnRenameInfo();

}

private void NewTerrain() {

// Створення нового екземпляра TerrainData

TerrainData terrainData = new TerrainData();

// Встановити роздільну здатність карти висот і розмір TerrainData

terrainData.heightmapResolution = width;

terrainData.size = new Vector3(width, 600, height);

// Створення висот рельєфу

float[,] heights = GenerateHeights();

terrainData.SetHeights(0, 0, heights);

terrain.terrainData = terrainData;

}

private void OnRenameInfo()

{

widthCan.text = width.ToString();

heightCan.text = height.ToString();

scaleCan.text = scale.ToString();

offsetXCan.text = offsetX.ToString();

offsetYCan.text = offsetY.ToString();

noiseIntensityCan.text = noiseIntensity.ToString();

}

private float[,] GenerateHeights()

{

float[,] heights = new float[width, height];

for (int x = 0; x < width; x++)

{

for (int y = 0; y < height; y++)

{

// Створення значення шуму Perlin для поточної позиції

float xCoord = (float)x / width \* scale + offsetX;

float yCoord = (float)y / height \* scale + offsetY;

float noiseValue = Mathf.PerlinNoise(xCoord, yCoord);

//Встановіть висоту рельєфу на основі рівня шуму

heights[x, y] = noiseValue \* noiseIntensity;

}

}

return heights;

}

public void ButtonClick(bool triger, int inf)

{

switch (inf)

{

case 3: if (triger) scale++; else scale--; break;

case 4: if (triger) offsetX++; else offsetX--; break;

case 5: if (triger) offsetY++; else offsetY--; break;

case 6: if (triger) noiseIntensity += 0.01f; else noiseIntensity -= 0.01f; break;

}

NewTerrain();

OnRenameInfo();

}

public void ButtonNext(int inf)

{

ButtonClick(true, inf);

}

public void ButtonPrev(int inf)

{

ButtonClick(false, inf);

}

}

### Код файлу (MeshGenerator.cs)

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UIElements;

[RequireComponent(typeof(MeshFilter))]

public class MeshGenerator : MonoBehaviour

{

Mesh mesh;

Vector3[] vertices;

int[] triangles;

Color[] colors;

public int xSize = 20, zSize = 20;

public float NoiseIntensetive = 2f;

public Gradient gradient;

float minTerrainHeiht, maxTerrainHeiht;

void Start()

{

mesh = new Mesh();

GetComponent<MeshFilter>().mesh = mesh;

OnUpdateMesh();

}

public void OnUpdateMesh()

{

CreateShape();

UpdateMesh();

}

void CreateShape()

{

vertices = new Vector3[(xSize + 1) \* (zSize + 1)];

for(int i=0, z = 0; z <= zSize; z++)

{

for(int x=0;x <= xSize; x++)

{

float y = Mathf.PerlinNoise(x \* .1f, z \* .1f) \* NoiseIntensetive;

vertices[i] = new Vector3(x, y, z);

if(y>maxTerrainHeiht)

maxTerrainHeiht = y;

if(y < minTerrainHeiht)

minTerrainHeiht = y;

i++;

}

}

triangles = new int[xSize \* zSize \* 6];

int vert = 0;

int tris = 0;

for (int z = 0; z < zSize; z++)

{

for (int x = 0; x < xSize; x++)

{

triangles[tris + 0] = vert + 0;

triangles[tris + 1] = vert + xSize + 1;

triangles[tris + 2] = vert + 1;

triangles[tris + 3] = vert + 1;

triangles[tris + 4] = vert + xSize + 1;

triangles[tris + 5] = vert + xSize + 2;

vert++;

tris += 6;

}

vert++;

}

colors = new Color[vertices.Length];

for (int i = 0, z = 0; z <= zSize; z++)

{

for (int x = 0; x <= xSize; x++)

{

float heidht =Mathf.InverseLerp(minTerrainHeiht, maxTerrainHeiht, vertices[i].y);

colors[i] = gradient.Evaluate(heidht);

i++;

}

}

}

void UpdateMesh()

{

mesh.Clear();

mesh.vertices = vertices;

mesh.triangles = triangles;

mesh.colors = colors;

mesh.RecalculateNormals();

}

private void OnDrawGizmos()

{

if(vertices == null)

return;

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

{

Gizmos.DrawSphere(vertices[i], .1f);

}

}

}

### Код файлу (MeshControlPanel.cs)

using UnityEngine;

using UnityEngine.UI;

public class MeshControlPanel : MonoBehaviour

{

public Text textXSize, textZSize, textNoiseIntensetive;

MeshGenerator meshGen;

void Start()

{

meshGen = this.GetComponent<MeshGenerator>();

UpdateTextConvas();

}

void UpdateTextConvas()

{

textXSize.text = meshGen.xSize.ToString();

textZSize.text = meshGen.zSize.ToString();

textNoiseIntensetive.text = meshGen.NoiseIntensetive.ToString();

int max = 10;

if(meshGen.xSize > meshGen.zSize)

max = meshGen.xSize;

else

max = meshGen.zSize;

Camera.main.transform.position = new Vector3(Camera.main.transform.position.x, max, Camera.main.transform.position.z);

}

public void ButtonClickNext(int inf){

switch (inf) {

case 1: if(meshGen.xSize < 100) meshGen.xSize += 5; break;

case 2: if (meshGen.zSize < 100) meshGen.zSize += 5;break;

case 3: if (meshGen.NoiseIntensetive < 15) meshGen.NoiseIntensetive+=1f; break;

}

meshGen.OnUpdateMesh();

UpdateTextConvas();

}

public void ButtonClickPrev(int inf){

switch (inf)

{

case 1: if (meshGen.xSize > 10) meshGen.xSize -= 5; break;

case 2: if (meshGen.zSize > 10) meshGen.zSize -= 5; break;

case 3: if (meshGen.NoiseIntensetive > 2) meshGen.NoiseIntensetive -= 1f; break;

}

meshGen.OnUpdateMesh();

UpdateTextConvas();

}

}

# Додаток Б. Лістинг програми до практичної роботи № 2.1

### Код файлу (???)

### Код файлу (???)

# Додаток ?. Лістинг програми до практичної роботи №?

### Код файлу (???)

### Код файлу (???)