МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНОЛОГИЧЕСКИЙ
УНИВЕРСИТЕТ «МИСИС»

Институт информационных технологий и компьютерных наук

Кафедра инженерной кибернетики

Курсовая работа

по дисциплине «Технологии программирования»
на тему
«Визуализация ландшафта с помощью карты высот»

Выполнила:
студентка 2-го курса,
гр. БПМ-21-3 Хлобустова С.М.

Проверил:
доцент, к.т.н. Полевой Д.В.

Москва, 2023

Содержание

1 Описание проекта

Рендеринг карты местности в 3д по серой png картинке с возможностью просмотра пользователем деталей отрисовки.

Визуализация местности в труднодоступных местах. Проект, который предоставляет более точные данные для наглядного демонстрирования ландшафта и дальнейшего проектирования, строительства или изучения на местности.

Карта высот представляет собой растровое изображение в оттенках серого со значением текселя (рисунок 1), соответствующим расстоянию, на которое вершина должна быть перемещена по нормали. Далее программа обрабатывает предыдущее изображение и представляет 3д местность (рисунок 2) с возможностью перемещения по ней пользователя.

Тексель (текселерация) — это величина, которая является отношением размера текстуры (в пикселях) к габаритам 3d модели в сцене, пиксель текстуры.

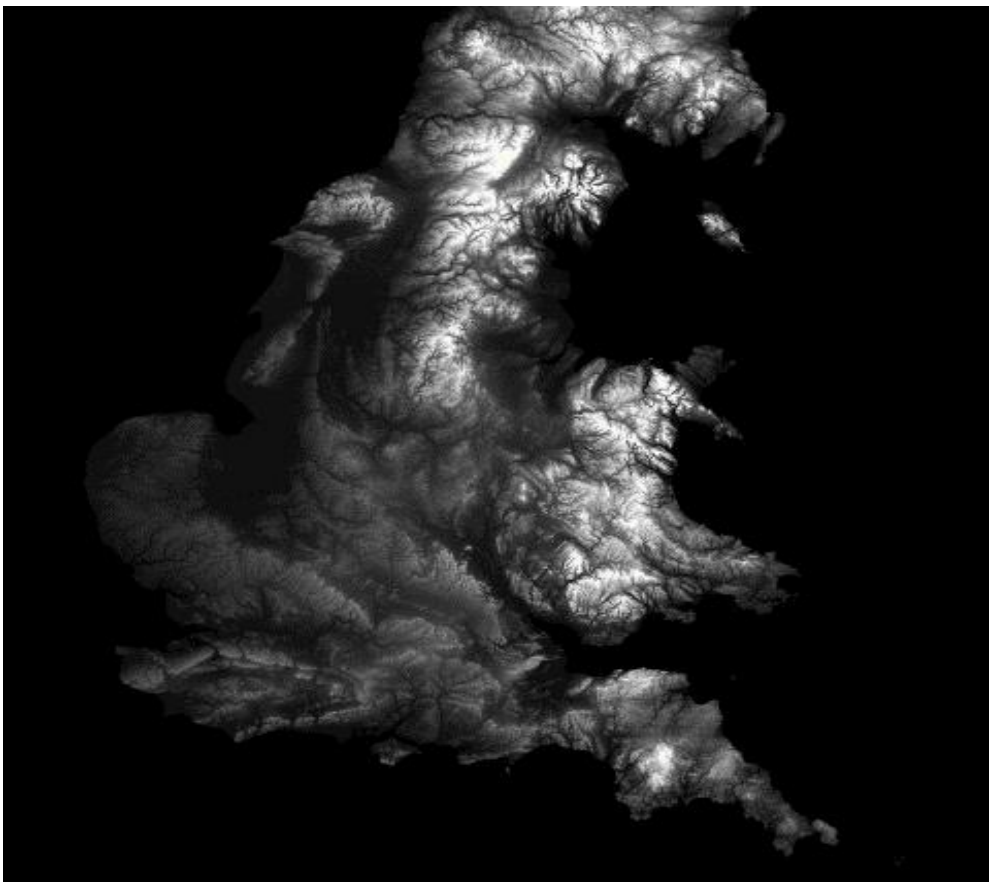


Рисунок *1 - 2д картинка на входе*

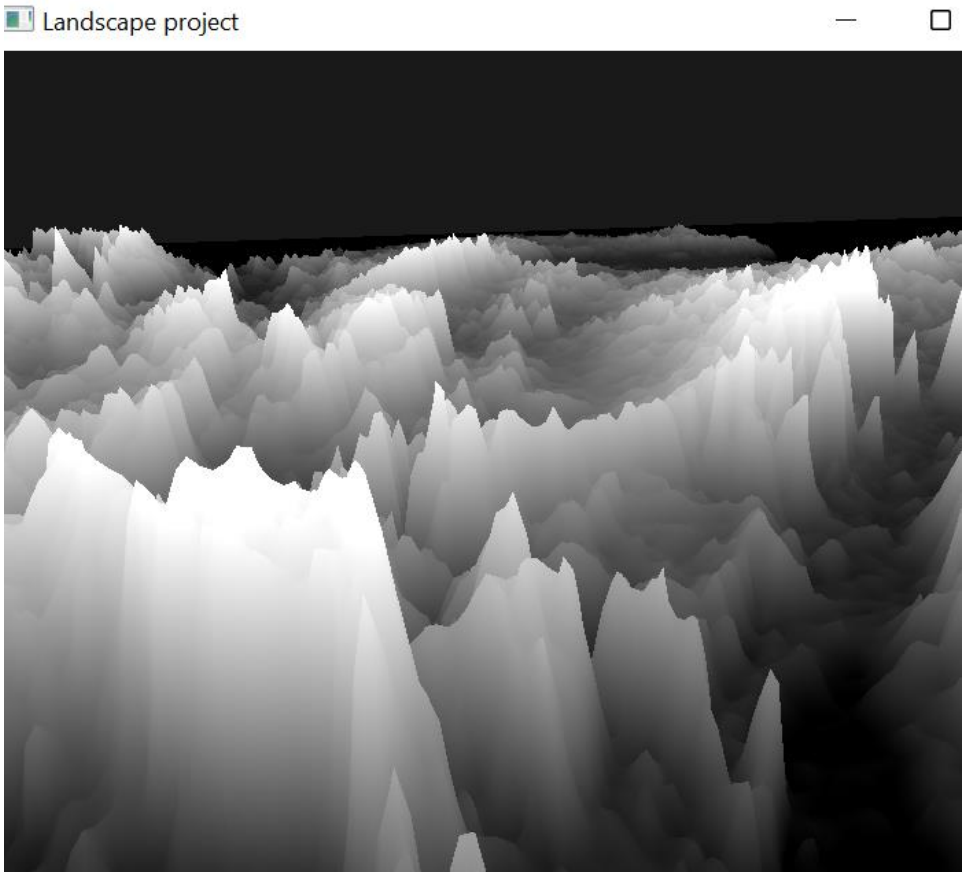


Рисунок 2 - 3д местность на выходе

2 Техническое задание

- Создание окна приложения отрисовки GLFW

- Обработка изображения, инициализация массива вершин значениями оттенков серого – составление карты высот

- Создание трехмерной местности на основе карты высот
- Изменение положения камеры внутри окна приложения с помощью обработки нажатий на клавиатуру и мышь

3 Пользовательский интерфейс

Нажатия клавиш на клавиатуре

Клавиши	*Действия*
W	Увеличить масштаб / движение вперед
S	Уменьшить масштаб / движение назад
A	Изменение положения камеры в левую сторону / движение влево
D	Изменение положения камеры в правую сторону / движение вправо
ESC	Выход из окна отрисовки
mouse move	Захват мышки приложением и изменение угла обзора камеры
Scroll mouse	Посредствам вращения колёсика мышки происходит изменение положения камеры воль оптической оси – приближение/отдаление изображения
Right mouse button	При нажатии правой кнопки мыши появляется курсор мыши, которым можно нажать полноэкранный режим При нажатии ещё раз, курсор мыши захватывается приложением (mouse move)

- Пользователь может выходить за пределы местности

- Пользователь может менять входное изображение, вводя в командную строку путь до него

4 Сборка

4.1 Требования к сборке

- Версия C++ не ниже 11

- Версия CMake не ниже 3.

4.2 Инструкция по сборке

- git clone https://github.com/SvetlanaKhlobustova50809/courseworkk.git

(копировать в любое место, только чтобы в пути к репозиторию не было папок с названиями, отличными от английского языка)

- В переменную CMAKE_INSTALL_PREFIX в CMake указать путь, куда нужно выгрузить exe файл
- Собрать проект с помощью CMake
- Собрать INSTALL, собрать проект landscape
- После сборки в указанном пути CMAKE_INSTALL_PREFIX в папке exefile должен находится landscape.exe, его также можно запускать для тестирования программы, должно выводится окно отрисовки как на рисунке 3
- После запуска проекта должно вывестись окно отрисовки как на рисунке 3

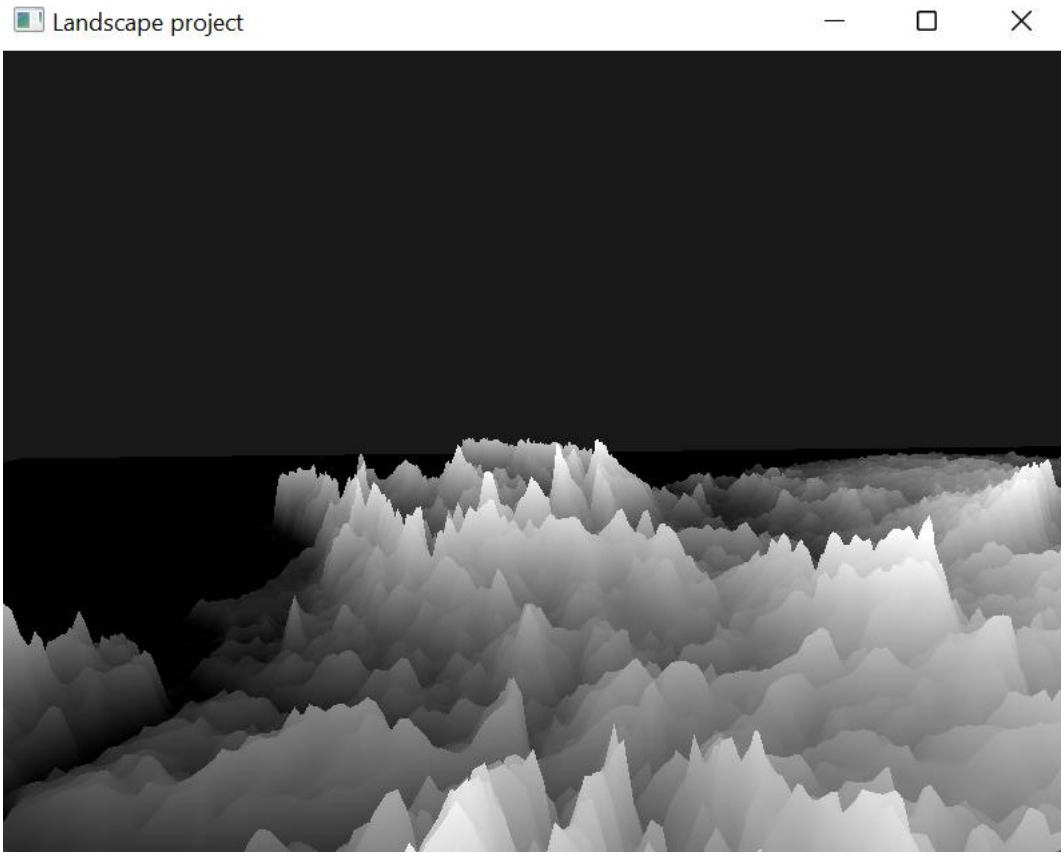


Рисунок 3 – окно отрисовки 3д местност

4.3 Работа с командной строкой

После вышеуказанных действий в командной строке cmd нужно выполнить следующее:

- с помощью команды cd указать прямой путь до папки, которая указана в переменной CMAKE_INSTALL_PREFIX и добавить \exefile (там должен лежать landscape.exe)

```
cd C:\Users\Svt\Desktop\exs\exefile
```

- ввести в качестве второго параметра путь (прямой или как указано относительный) до входной картинки в формате .png

```
C:\Users\Svt\Desktop\exs\exefile>landscape.exe ../textures/5.png
```

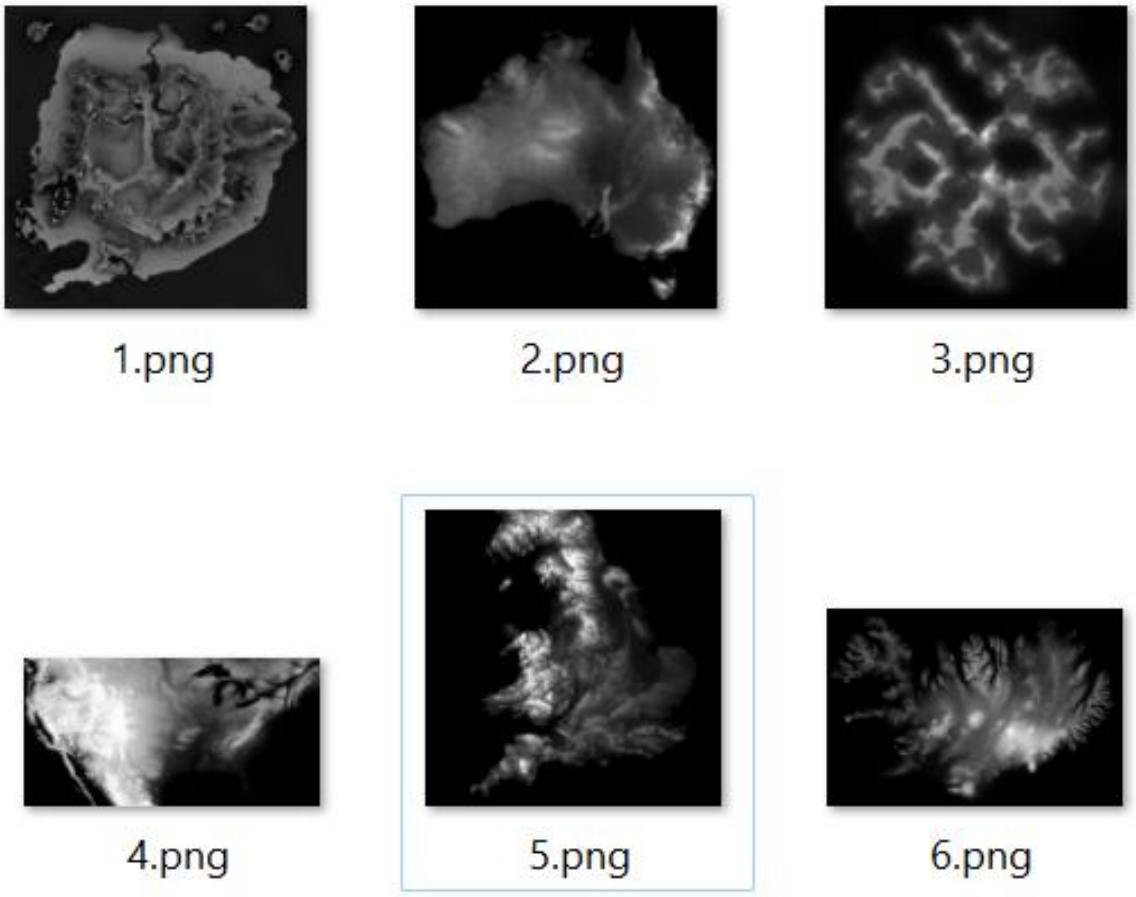
Если пользователь всё сделал правильно, то должно выйти окно отрисовки как на рисунке 3 и текст в консоли:

```
Loaded heightmap of size 1200 x 1200
Loaded 1440000 vertices
Loaded 2877600 indices
Created lattice of 1199 strips with 2398 triangles each
Created 2875202 triangles total
```

Если пользователь неправильно указал путь или указанная картинка не существует, то выведется текст

```
Введите путь до картинки в формате ../textures/2.png
```

Картинку можно выбрать из папки textures (путь ../textures/{номер от 1 до 6}.png):



Или можно скачать готовую карту высот или сгенерировать ее с помощью [сайта]. Браузер данных о высотах, который "автоматически выставляет" дисплей таким образом, чтобы самые высокие и самые низкие видимые высоты были белыми и черными.

URL: https://tangrams.github.io/heightmapper/

Картинка должна быть

- в формате .png

- черно-белая

- без прозрачного фона (квадратная/прямоугольная)

5 Описание реализации

Данные карты высот считываются в массив, в котором хранятся пиксельные данные.

Создаётся сетка, соответствующая разрешению пользовательского изображения. **stbi_load()** метод установит переменные width и height к размеру изображения и data будет содержать width * height * nChannels элементов. Тогда сетка будет представлять собой массив, состоящий из width * height вершин. Для предоставленного примера карты высот (../textures/5.png) это приведет к созданию сетки из 1200 x 1200 вершин, в общей сложности 1 440 000 вершины.

Сам ландшафт разделен на сетку значений высоты, которые расположены на равном расстоянии друг от друга, но имеют различную высоту, на основе данных поля высоты (рисунок 4). Результатом является сетка, представляющая ландшафт в сцене.

В проекте используется Объект буфера вершин (VBO) для указания данных вершин сетки. Полученная сетка будет центрирована в начале координат, лежать в плоскости XZ и иметь размер width * height. Каждая вершина будет находиться на расстоянии одной единицы в пространстве модели. Затем вершина будет отображаться по нормали к поверхности (ось Y) на основе соответствующего местоположения на карте высот.

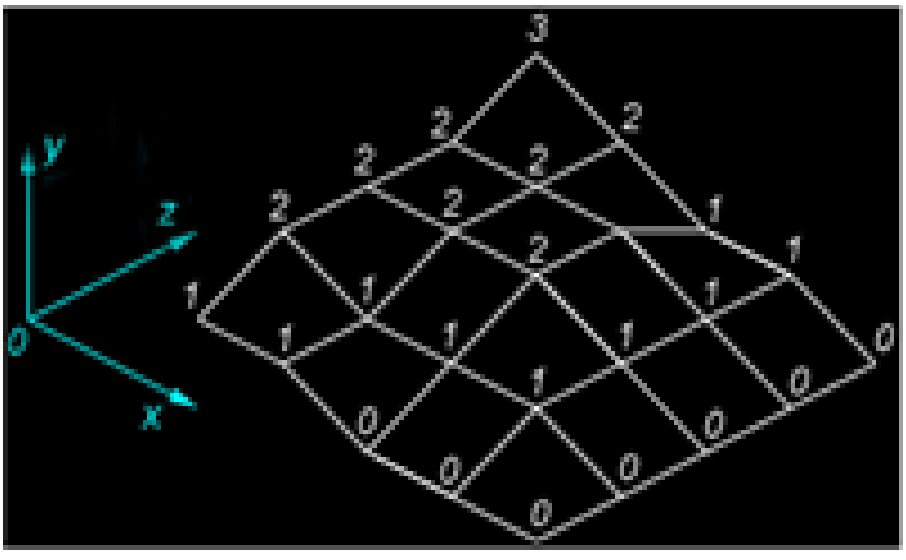


Рисунок 4 – графическое представление массива

Карта высот представлена в оттенках серого, поэтому нужно сохранить первый канал (независимо от того, состоит ли фактический texel из трех каналов RGB или одного канала) в качестве значения, которое будет использовать для координаты y, или высоты, вершины на сетке.

Сетка представляется в виде последовательности треугольных полос, поэтому следует использовать Объект буфера элемента (EBO) для соединения вершин в треугольники. Сетка будет разбита на треугольные полоски поперек каждого ряда.

На рисунке 5 каждая цветная треугольная полоса соответствует одной треугольной полосе для строки i по столбцам j.

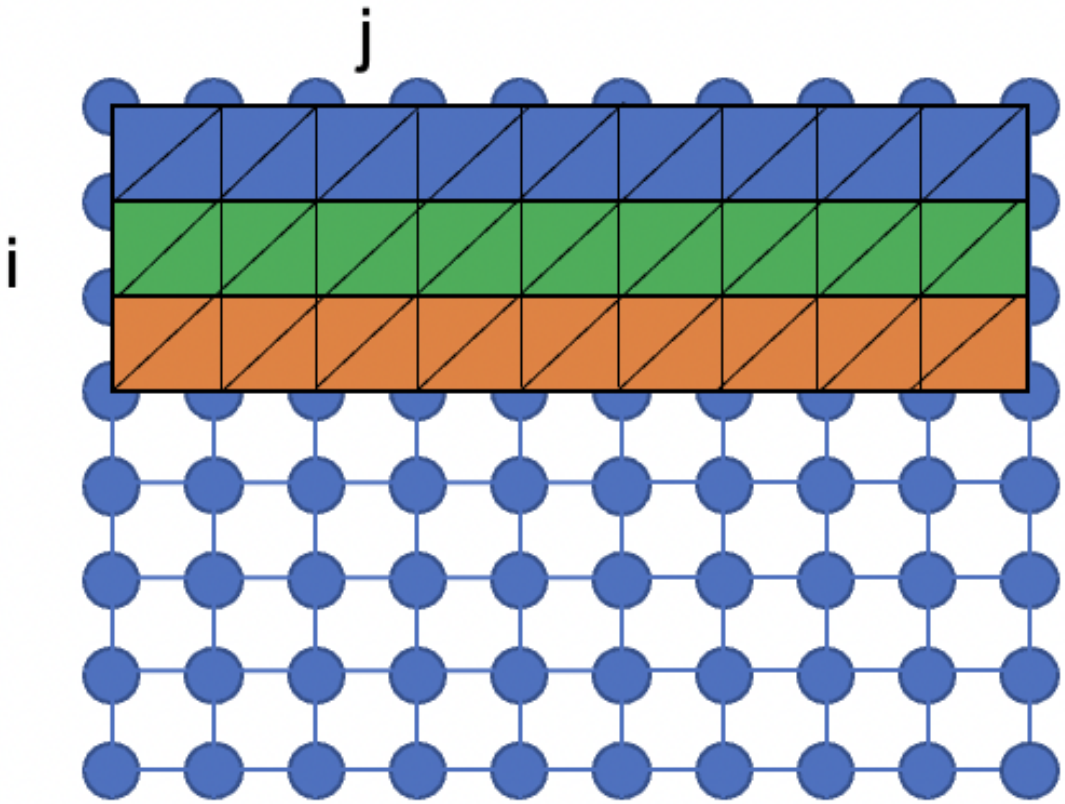


*Рисунок 5 – сетка размером i*j цветных полос*

Далее настраивается Объект массива вершин (VAO) на графическом процессоре и визуализируется сетка полосу за полосой.

Загрузка фрагментного и вершинного шейдеров производится в форматах .fs и .vs для задания массивов и отрисовки местности.

При рендеринге происходит передача координаты у нашей вершины из вершинного шейдера в фрагментный шейдер. Затем в шейдере фрагмента нормализуется это значение (используя обратный сдвиг и масштаб сверху), чтобы преобразовать его в значение в оттенках серого. Графический вывод осуществляется через OpenGL API. Скриншоты различных поверхностей - рисунки 6 и 7.

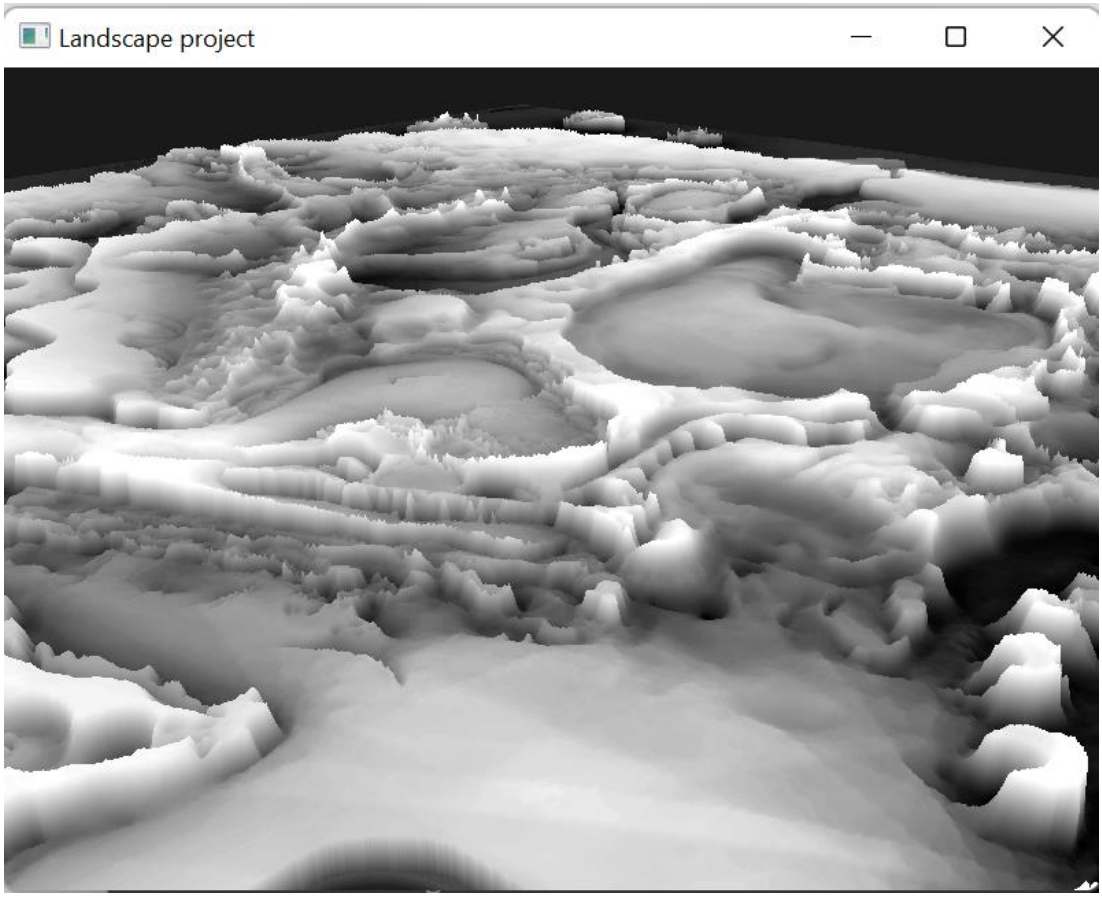


Рисунок 6

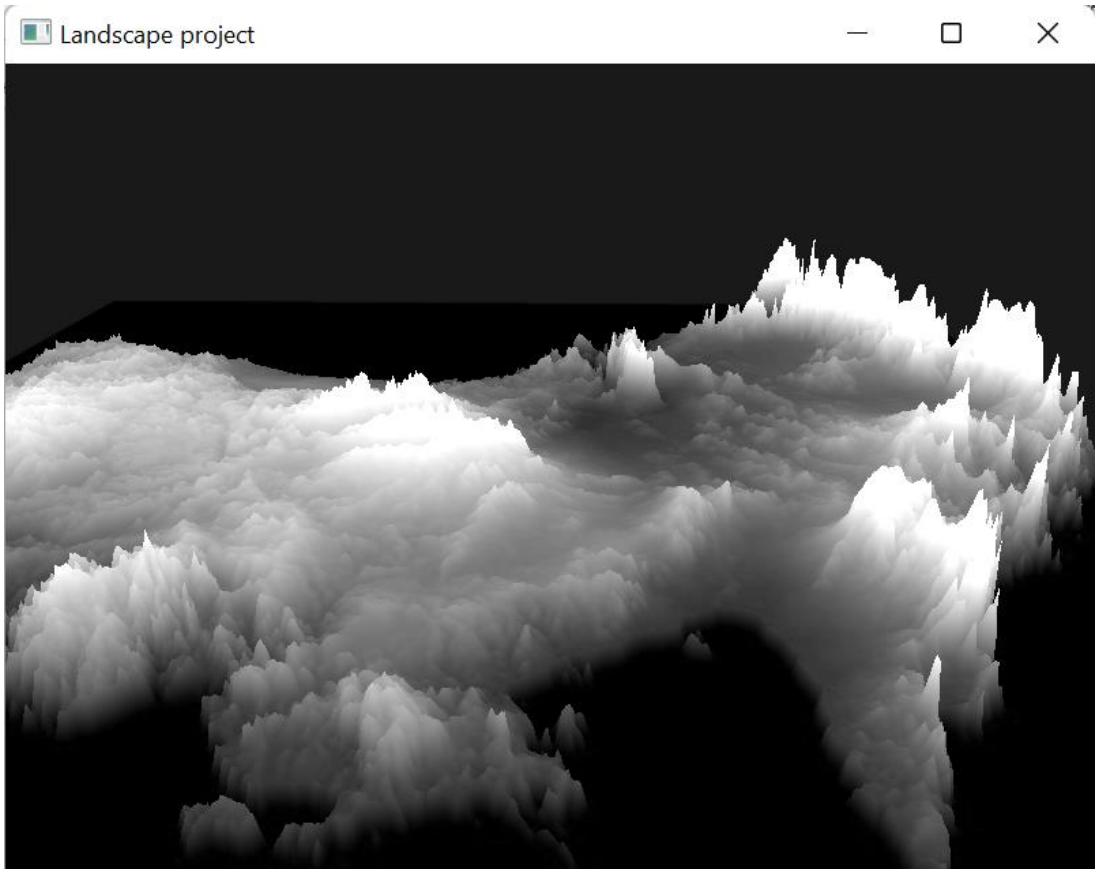


Рисунок 7

6 Библиотеки

1. OpenGL 3.3 Спецификация OpenGL описывает каким будет результат выполнения каждой конкретной функции и что она должна делать. Большая часть работы OpenGL — это преобразование 3D координат в 2D пространство для отрисовки на экране. OpenGL является только стандартом / спецификацией, конкретная реализация реализована разработчиком драйвера для конкретной видеокарты.

2. GLFW - это мультиплатформенная библиотека C с открытым исходным кодом для разработки OpenGL. Он предоставляет простой API для создания окон, контекстов и поверхностей, получения входных данных и событий. Он обеспечивает доступ к вводу с клавиатуры, мыши и джойстиков.

3. GLAD управляет указателями на функции для OpenGL. Это библиотека с открытым исходным кодом, которую можно настроить в предоставляемом ею онлайн-сервисе.

4. GLM библиотеки математических вычислений для OpenGL.

5. STB однофайловые общедоступные библиотеки для C / C ++. stb_image.h нужен для загрузки изображений.

7 Документация

Заполнение сетки

Данный модуль, предназначен для заполнения сетки.

Функции

void vertexGridPlaceholder (int width, int height, unsigned char *data, unsigned bytePerPixel) - Функция инициализации массива vertices.

Аргументы

width,height	количество клеток в высоту и ширину сетки
data	массив с пиксельными данными карты высот
bytePerPixel	байт пикселя в перспективе

void indicesGridPlaceholder (int width, int height, int rez) - Функция инициализации массива indices.

Аргументы

width,height	количество клеток в высоту и ширину сетки
rez	шаг, с которым нужно заполнять массив indices

void VAO_Setup () - Функция настройки Объекта массива вершин (VAO) на графическом процессоре.

void visualisationGrip (const int numStrips, const int numTrisPerStrip) - Функция визуализации сетки полосу за полосой.

Аргументы

numStrips	номер полосы
numTrisPerStrip	количество треугольников, из которых будет состоять полоса

Класс Camera

Абстрактный класс камеры, который обрабатывает входные данные и вычисляет соответствующие углы Эйлера, векторы и матрицы для использования в OpenGL

Конструкторы:

Camera::Camera (glm::vec3 position = glm::vec3(0.0f, 0.0f, 0.0f), glm::vec3 up = glm::vec3(0.0f, 1.0f, 0.0f), GLfloat yaw = YAW, GLfloat pitch = PITCH)[inline] - конструктор с параметрами векторов

Camera::Camera (GLfloat posX, GLfloat posY, GLfloat posZ, GLfloat upX, GLfloat upY, GLfloat upZ, GLfloat yaw, GLfloat pitch)[inline] - конструктор с параметрами скалярных значений

Методы:

glm::mat4 Camera::GetViewMatrix ()[inline] - возвращает матрицу вида, рассчитанную с использованием углов Эйлера и матрицы LookAt

void Camera::ProcessKeyboard (Movement_Direction direction, GLfloat deltaTime)[inline] - обрабатывает ввод, полученный от любой системы ввода, подобной клавиатуре. Принимает входной параметр в виде определяемого камерой перечисления (чтобы абстрагировать его от оконных систем)

void Camera::ProcessMouseMovement (GLfloat xoffset, GLfloat yoffset, GLboolean constrainPitch = true)[inline] - обрабатывает ввод, полученный от системы ввода с помощью мыши. Ожидает значение смещения как в направлении x, так и в направлении y.

void Camera::ProcessMouseScroll (GLfloat yoffset)[inline] - обрабатывает входные данные, полученные от события колеса прокрутки мыши. Требуется ввод только по вертикальной оси колеса]

8 Материалы и ссылки

- Изучение современного OpenGL через использование основных функций и использование некоторых понятий математики. URL: https://learnopengl.com/

- Туториал по технике создания ландшафтов. Рассматривается карта высот, простейший алгоритм визуализации и оптимизация на уровне OpenGL.

URL: https://gamedev.ru/articles/?id=30015

- Пример построения местности в 3д и алгоритм генерации ландшафта.

URL: https://www.codeproject.com/Articles/14154/OpenGL-Terrain-Generation-An-Introduction