Университет ИТМО

Факультет программной инженерии
и компьютерной техники

Кафедра вычислительной техники

Компьютерная графика
Лабораторная работа

Выполнили:
Милосердов А. О.
Калугин Ф. И.
Группа P3410

Преподаватель:
Бессмертный И. А.

Санкт-Петербург
2017 г.

Описание

Был реализован кроссплатформенный движок для FPS игр.
Особенности:

- Сборка и установка игровых файлов в игровой каталог (шейдеры, текстуры, карты, бинарь) на Windows и Linux.
- Высокоуровневый объектно-ориентированный Modern C++ API для работы с графикой.
- FPS камера.
- Использование различных текстур, модель освещения Phong, динамические тени (Shadow mapping).
- Обработка коллизий игрока и объектов мира.
- Физика гравитации.
- Загрузка карт мира в формате BMP.

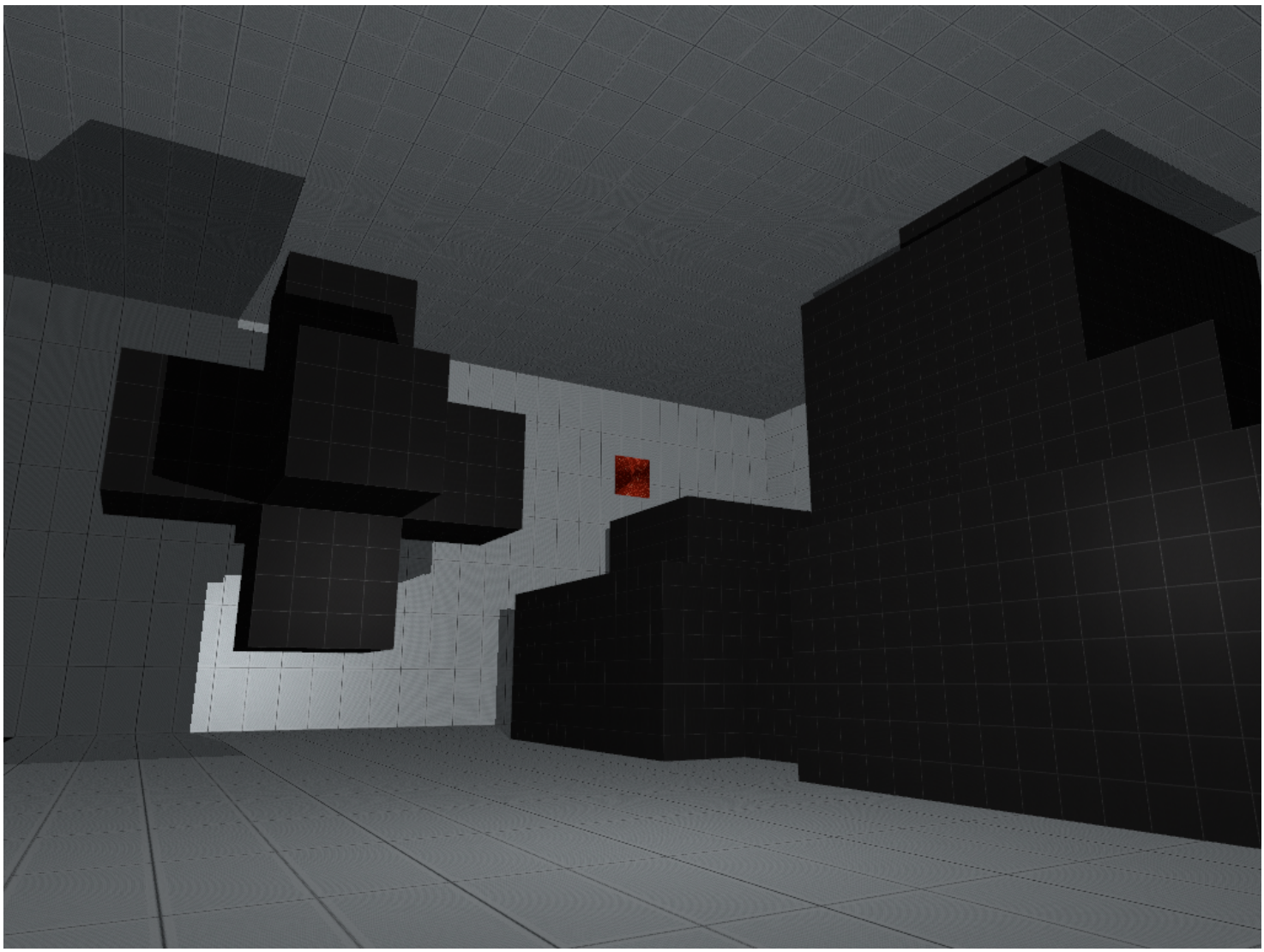

Рис. 1: Скриншот демонстрационного проекта (игрок выстрелил красным октаэдром который в полете испускает свет)

Карта мира представляет собой продвинутую карту высот/текстур:

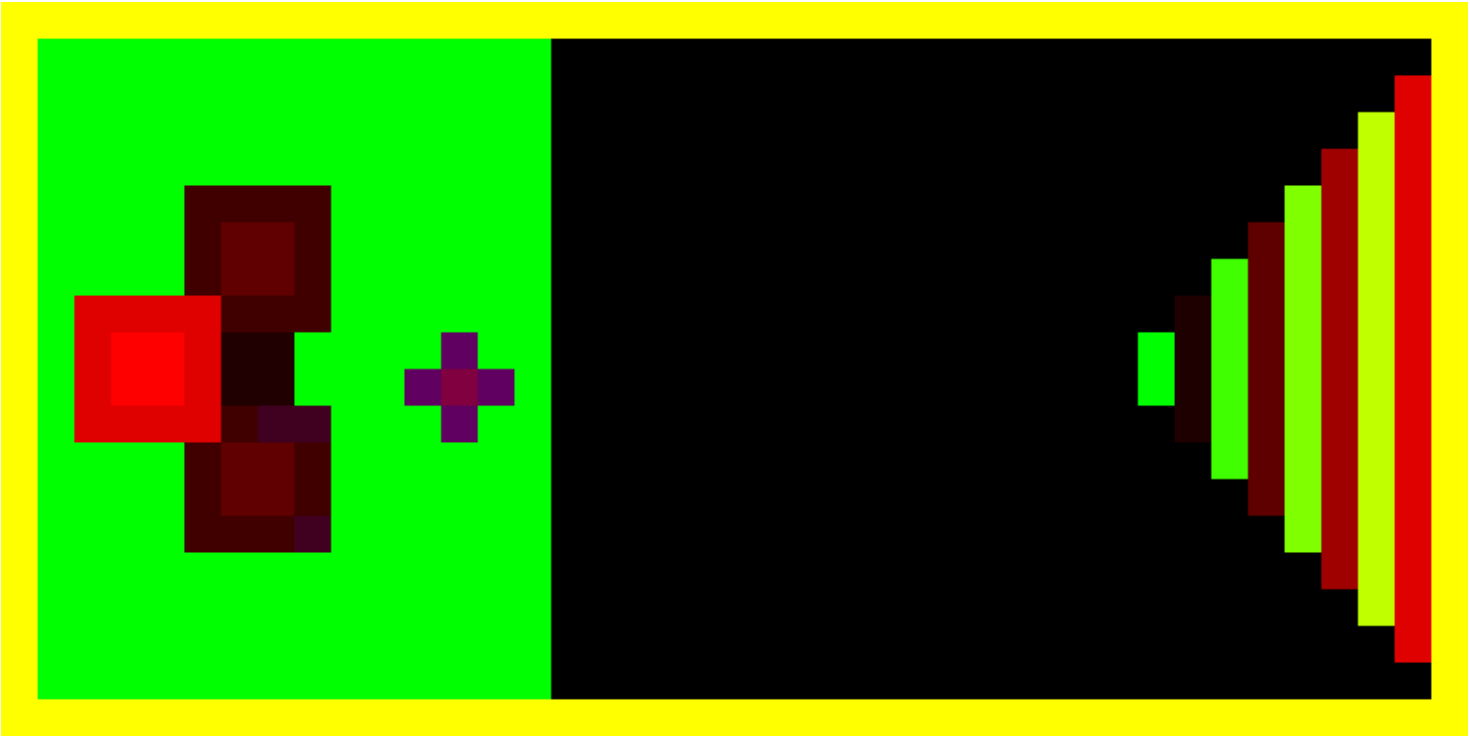

Рис. 2: Карта мира в формате bmp

- Синий канал - стартовый уровень высоты данного участка.
- Красный канал - конечный уровень высоты данного участка.
- Зеленый канал - номер используемой текстуры участка.

Каждый пиксель занимает примерно 2 на 2 метра реального мира.

Modern C++ API

```
std::vector<std::shared_ptr<Texture>> textures;
textures.push_back(std::make_shared<Texture>("textures/portal_black.png"));
textures.push_back(std::make_shared<Texture>("textures/portal_white.jpg"));
...
std::list<std::unique_ptr<OpenGLModel>> objects = Map::load("maps/map01.bmp", textures);
...
Shader screenShader;
screenShader.attach("shaders/shadows.frag");
screenShader.attach("shaders/shadows.vert");
screenShader.link();
...
for (auto &obj : objects) {
    obj->draw(screenShader.getUniformLocation("model"));
}
```

Листинг 1: Пример использования

Листинги

```cpp
#include "model.hpp"

#include <glad/glad.h>
#include <glm/gtc/matrix_transform.hpp>
#include <vector>
#include <iostream>

OpenGLModel::OpenGLModel(std::shared_ptr<Texture> texture) {
    mTexture = texture;
    glGenVertexArrays(1, &mID);
}

void OpenGLModel::initVBO(std::vector<float> &coordinates) {
    glBindVertexArray(mID);

    glGenBuffers(1, &mVBO);
    glBindBuffer(GL_ARRAY_BUFFER, mVBO);
    glBufferData(GL_ARRAY_BUFFER, coordinates.size() * sizeof(float), coordinates.data(), GL_STATIC_DRAW);
    glVertexAttribPointer(0, 3, GL_FLOAT, GL_FALSE, 3 * sizeof(float), 0);
    glEnableVertexAttribArray(0);

    mVertexCount = coordinates.size() / 8 * 3;

    glEnableVertexAttribArray(0);
    glVertexAttribPointer(0, 3, GL_FLOAT, GL_FALSE, 8 * sizeof(float), (void*)0);
    glEnableVertexAttribArray(1);
    glVertexAttribPointer(1, 3, GL_FLOAT, GL_FALSE, 8 * sizeof(float), (void*)(3 * sizeof(float)));
    glEnableVertexAttribArray(2);
    glVertexAttribPointer(2, 2, GL_FLOAT, GL_FALSE, 8 * sizeof(float), (void*)(6 * sizeof(float)));
}

OpenGLModel::~OpenGLModel() {
    glDeleteVertexArrays(1, &mID);
    glDeleteBuffers(1, &mVBO);
}

void OpenGLModel::draw(GLuint modelLocation) {
    mTexture->activate();
    glBindVertexArray(mID);
    glm::mat4 model;
    model = glm::translate(model, mPosition);
    model = glm::rotate(model, glm::radians(mRotation.x), glm::vec3(1.0f, 0.0f, 0.0f));
    model = glm::rotate(model, glm::radians(mRotation.y), glm::vec3(0.0f, 1.0f, 0.0f));
    model = glm::rotate(model, glm::radians(mRotation.z), glm::vec3(0.0f, 0.0f, 1.0f));
    glUniformMatrix4fv(modelLocation, 1, GL_FALSE, &model[0][0]);
    glDrawArrays(GL_TRIANGLES, 0, mVertexCount);
}

void OpenGLModel::move(glm::vec3 position) {
    mPosition = position;
}

void OpenGLModel::rotate(glm::vec3 rotation) {
    mRotation = rotation;
}

glm::vec3 OpenGLModel::getPosition() {
    return mPosition;
}
```

Листинг 2: Класс Модели игрового мира

Чтобы добавить новую модель достаточно лишь унаследовать класс OpenGLModel и определить конструктор с координатами:

```
Cube::Cube(std::shared_ptr<Texture> texture): OpenGLModel(texture) {
    std::vector<float> coordinates = {
        -1.0f, -1.0f, -1.0f,  0.0f,  0.0f, -1.0f, 0.0f, 0.0f,
...
    };
    initVBO(coordinates);
}
```