

Оберган Татьяна
ИУ7-55Б

Город, погода

Научный руководитель:
Романова Т.Н.

Цель работы

Цель данной работы – реализовать построение трехмерной сцены и визуализацию погодных эффектов в городском ландшафте.

1. описание структуры трехмерной сцены
2. выбор и/или модифицирование существующих алгоритмов трехмерной графики, которые позволят визуализировать трехмерную сцену;
3. реализация данных алгоритмов для создания трехмерной сцены;
4. разработка программного обеспечения, которое позволит отобразить трехмерную сцену и визуализировать погодные эффекты в городском ландшафте.

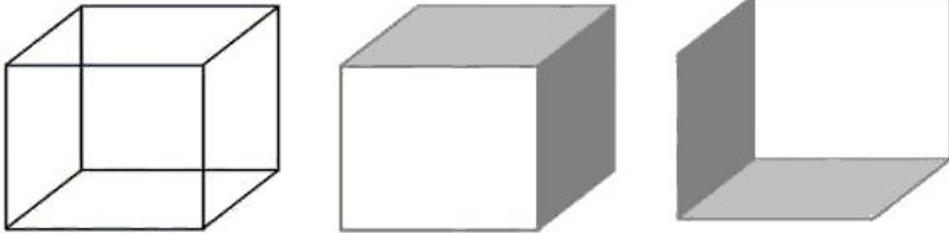
Формализация сцены

- Плоскость земли
- Здания
- Источник света
- Осадки
- Ветер
- Туман

Выбор алгоритмов

Алгоритмы удаления невидимых линий,
построения теней

- Обратная трассировка лучей
- Алгоритм Робертса
- Алгоритм Варнока
- ★ Z буфер



Методы закрашивания:

- ★ Простая закрашка - один уровень интенсивности на грань
- Закраска по Гуро - билинейная интерполяция интенсивностей
- Закраска по Фонгу - билинейная интерполяция векторов нормалей

Простой метод освещения

В простом методе освещения интенсивность рассчитывается по закону Ламберта:

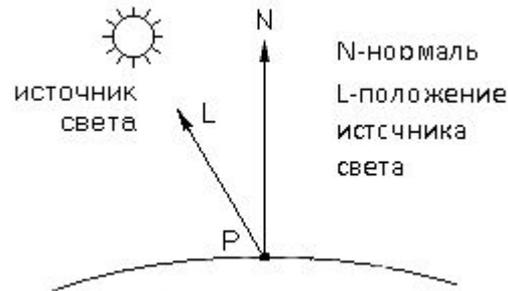
$$I = I_0 * \cos(\alpha), \text{ где}$$

I – результирующая интенсивность света в точке

I_0 – интенсивность источника

α – угол между нормалью к поверхности

и вектором направления света



Эффекты погоды

Для визуализации дождя:

- Система частиц
- ★ Метод Кшитиза и Шри

Во всех проанализированных
многочисленных алгоритмах туман
всегда зависит от расстояния
до объекта.



Общий алгоритм визуализации сцены

1. Получение информации о сцене
2. Выполнение преобразований и расчетов
3. Алгоритм Z-буфера для наблюдателя и источника света
4. Поиск теней
5. Добавление эффектов погоды

Алгоритм Z-буфера

1. Всем элементам буфера кадра присвоить фоновое значение
2. Инициализировать Z буфер минимальными значениями глубины
3. Выполнить растровую развертку каждого многоугольника сцены:
 - a. Для каждого пикселя, связанного с многоугольником вычислить его глубину $z(x, y)$
 - b. Сравнить глубину пикселя со значением, хранимым в Z буфере. Если $z(x, y) > z_{буф}(x, y)$, то $z_{буф}(x, y) = z(x, y)$, $цвет(x, y) = цветПикселя$.
4. Отобразить результат

Генерация осадков

1. Получение начальных данных
2. Пока не получена команда прекращения осадков:
 - 2.1 Обновление положения частиц по заданному закону
 - 2.2 Инициализация новых частиц
 - 2.3 Отображение частиц на дисплее
3. Пока система частиц не пуста
 - 3.1 Обновление положения частиц по заданному закону
 - 3.2 Отображение частиц на дисплее

Туман

Для того, чтобы создать эффект дымки или плотного тумана нужно знать удаленность от наблюдателя видимых пикселей.

Если пиксель дальше последнего видимого z интенсивность тумана будет равна 1, иначе

$$k = \frac{(z_{\text{пикс}} - z_{\text{дальнее}})}{(z_{\text{наблюдателя}} - z_{\text{дальнее}})},$$

где k – интенсивность пикселя, $1 - k$ – интенсивность тумана.



Выбор языка программирования и среды разработки

В качестве языка программирования был выбран C#:

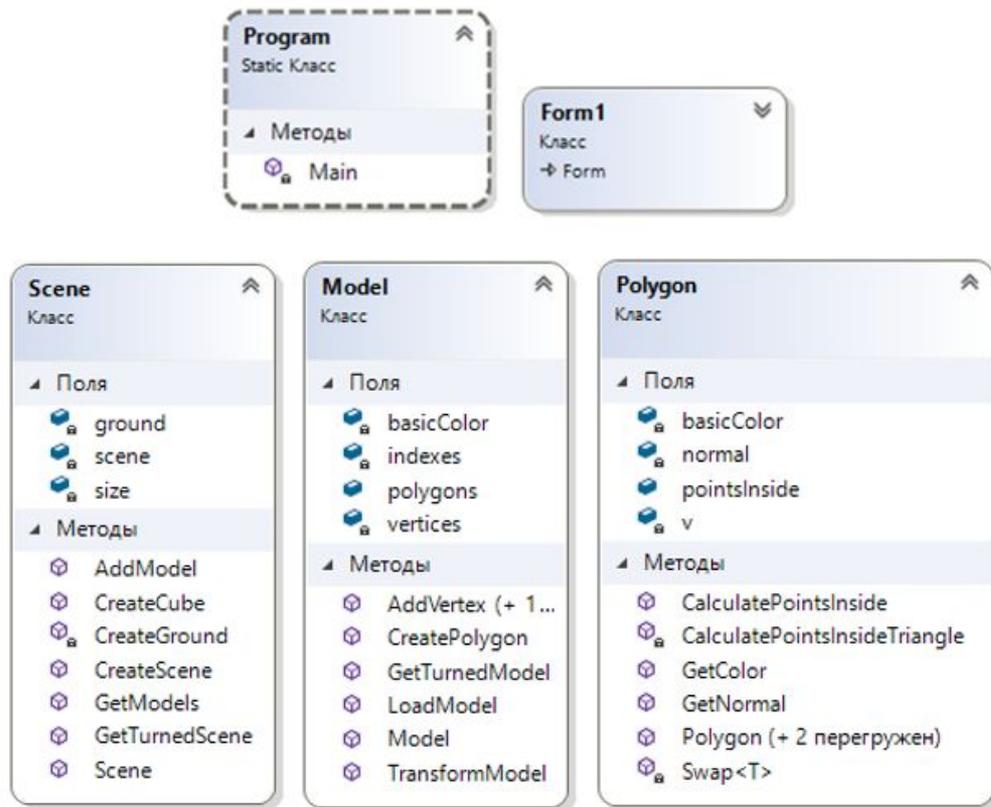
- ознакомилась с этим языком во время занятий по компьютерной графике
- ООП

В качестве среды разработки была выбрана Visual Studio 2017:

- бесплатна для студентов
- удобства отладки и написания кода

Для создания графического интерфейса был использован Windows Forms.

Структура и состав классов



Vector
Класс

Поля

- length
- x
- y
- z

Методы

- FindLength
- GetAngleXBetween
- GetAngleYBetween
- GetAngleZBetween
- GetLength
- RotateVectorX
- RotateVectorY
- ScalarMultiplication
- Vector (+ 2 переружен)
- VectorMultiplication

ParticleSystem
Класс

Поля

- direction
- rnd
- system
- xMax
- yMax

Методы

- DrawParticles
- InitParticles
- IsEmpty
- ParticleSystem
- ProcessSystem
- UpdateParticles

Drop
Класс

Поля

- x
- y
- z

Методы

- Draw
- Drop
- GetDepth
- GetNextPoint
- GetPoint
- IsBelow
- IsNextVisible
- IsVisible
- Update

LightSource
Класс

Поля

- color
- direction

Методы

- LightSource

Colors
Static Класс

Методы

- Mix

Fog
Класс

Поля

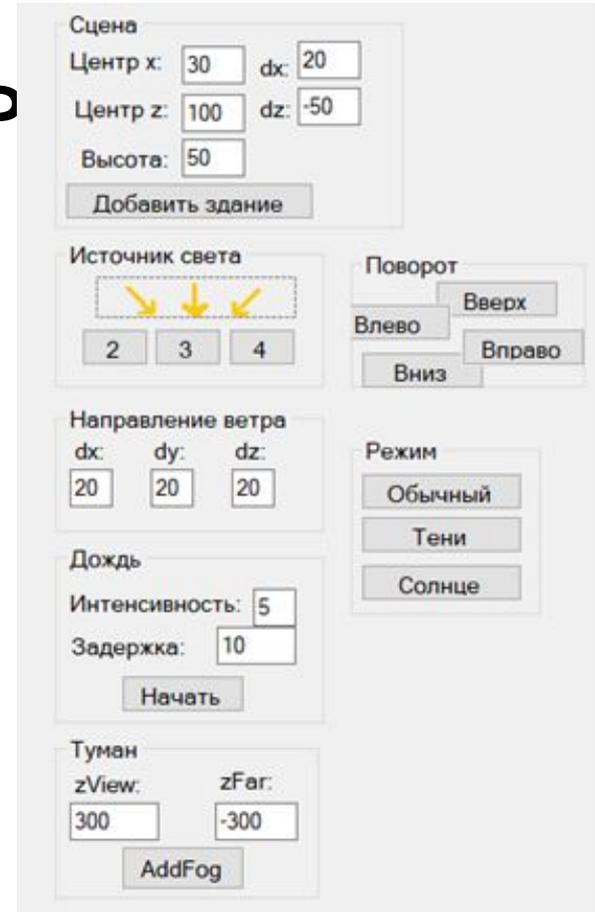
- fogColor

Методы

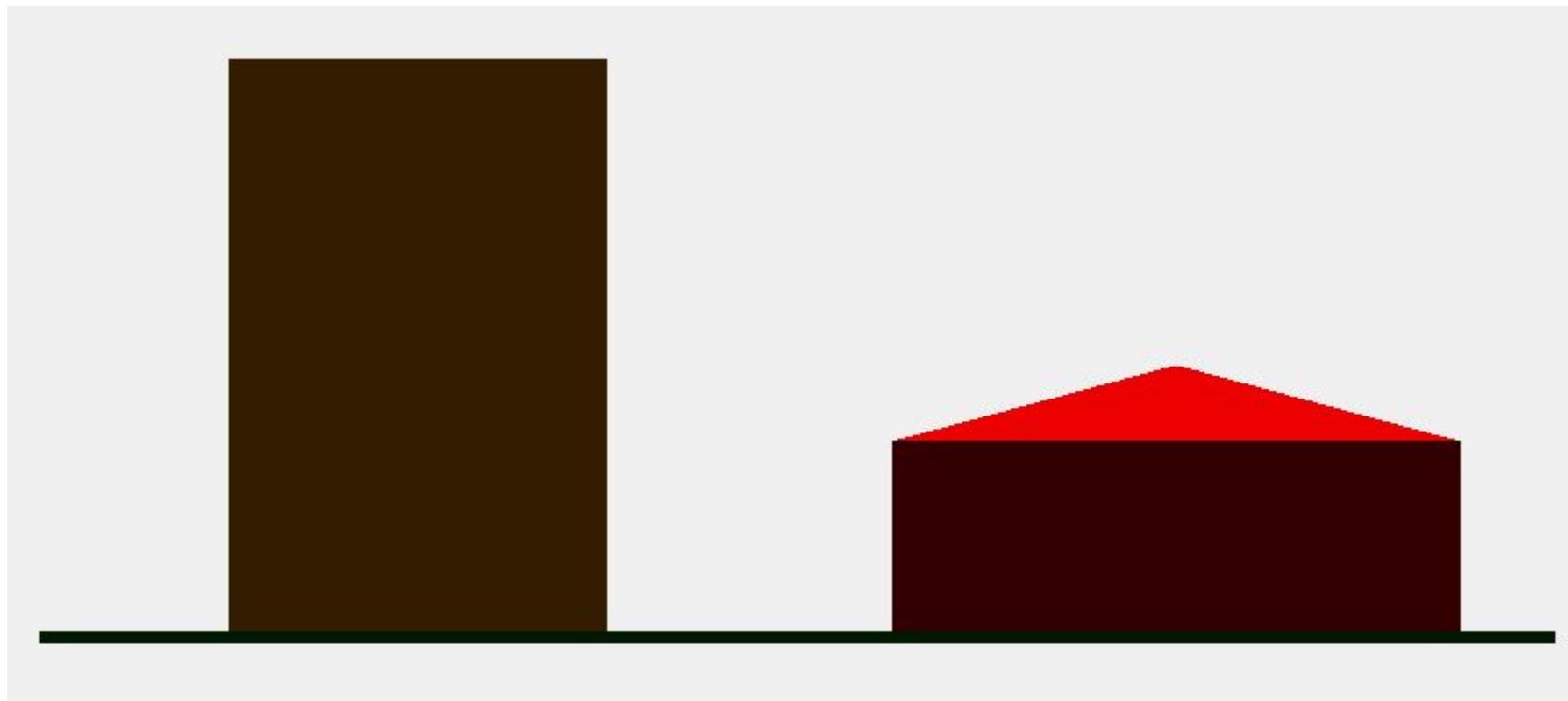
- AddFog

Интерфейс программы

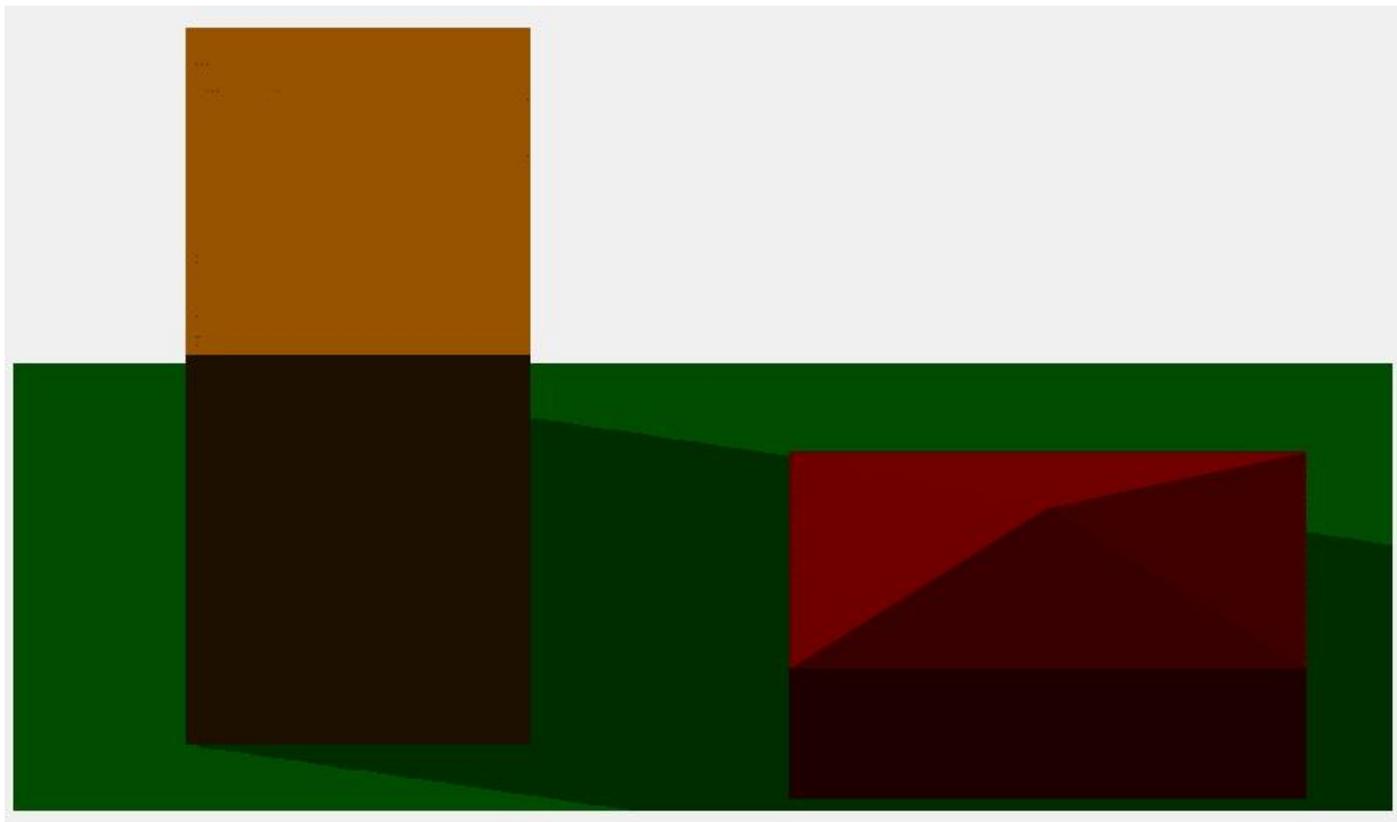
- Сцена - добавление зданий
- Источник света - изменение положения источника
- Направление ветра
- Дождь - наложение эффекта дождя
- Туман - наложение эффекта тумана
- Поворот - поворот сцены
- Режим - выбор вида изображения



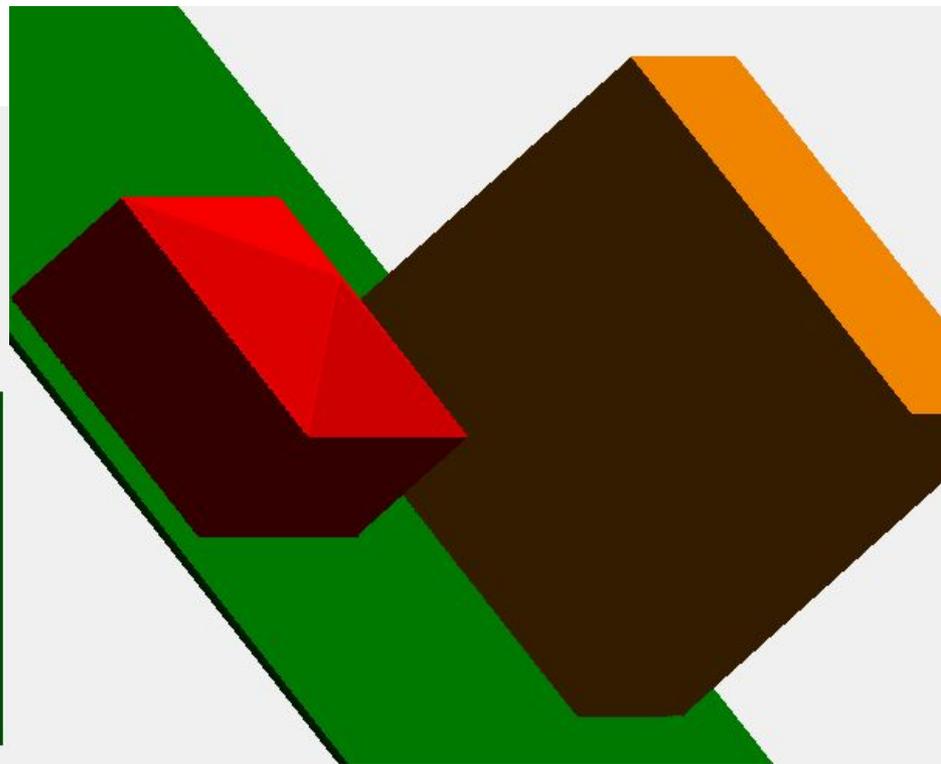
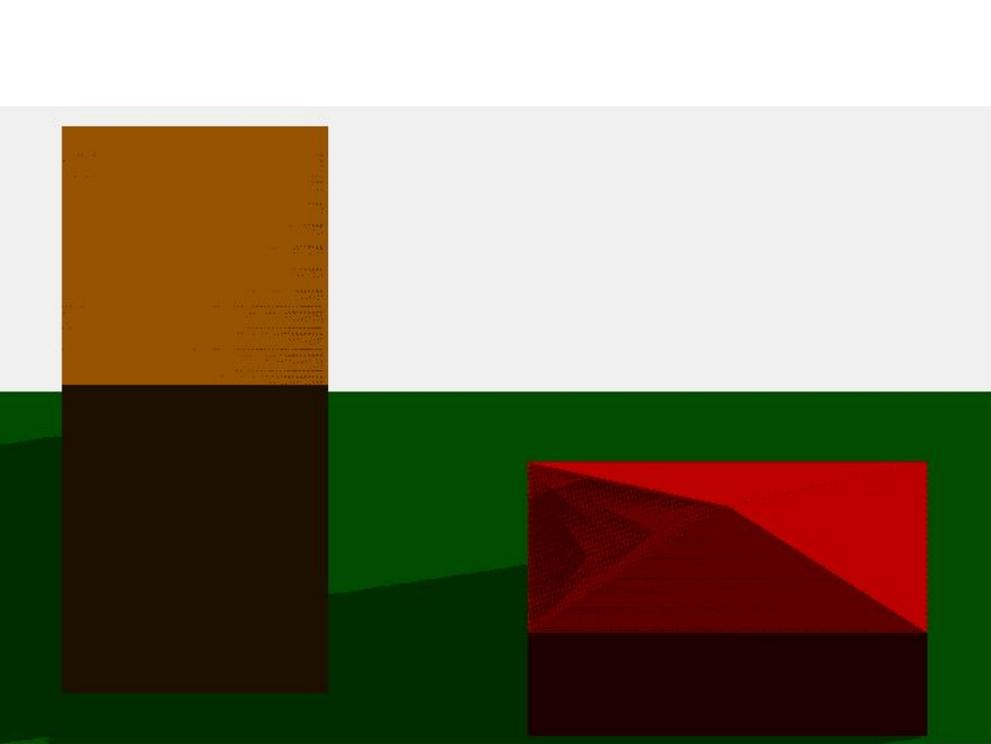
Сцена



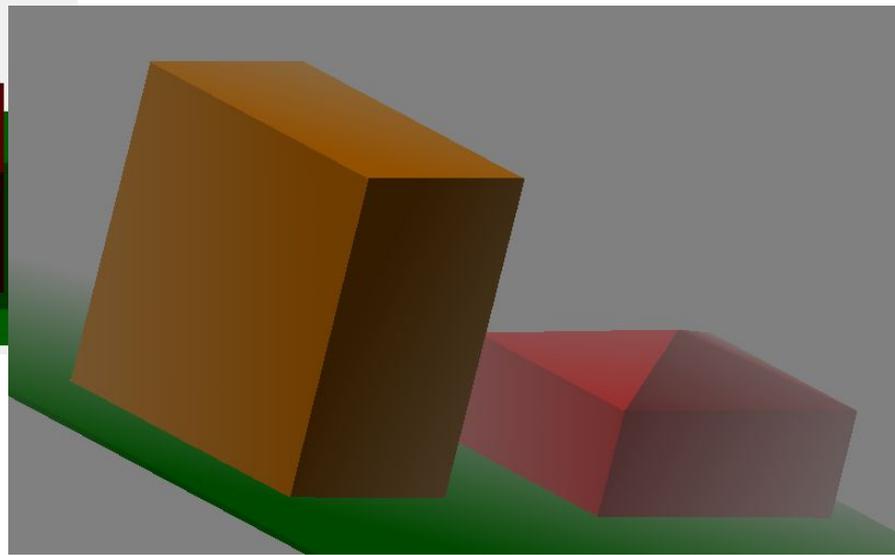
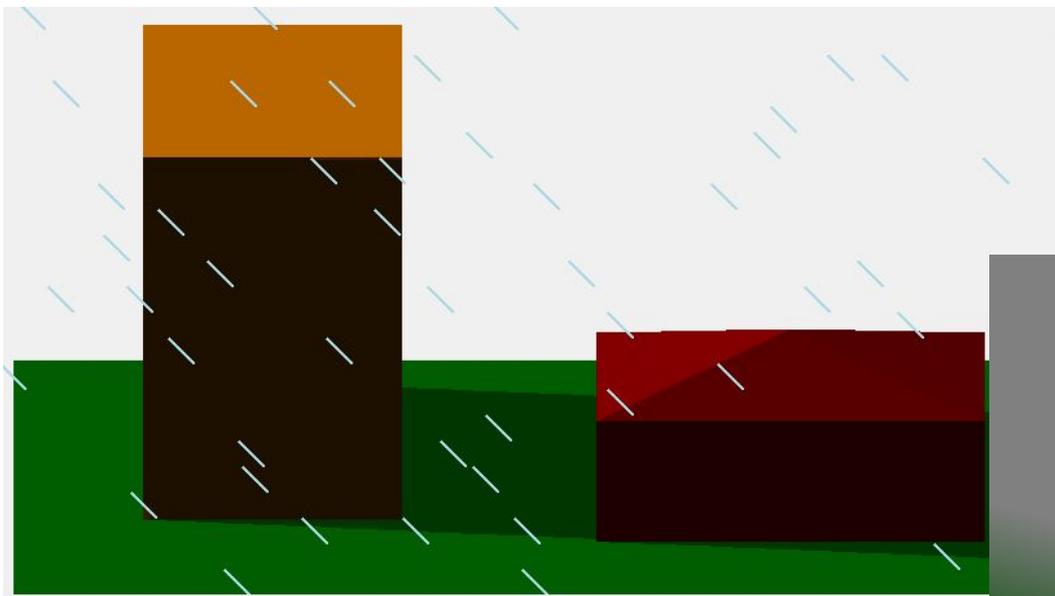
Тени



Дефекты



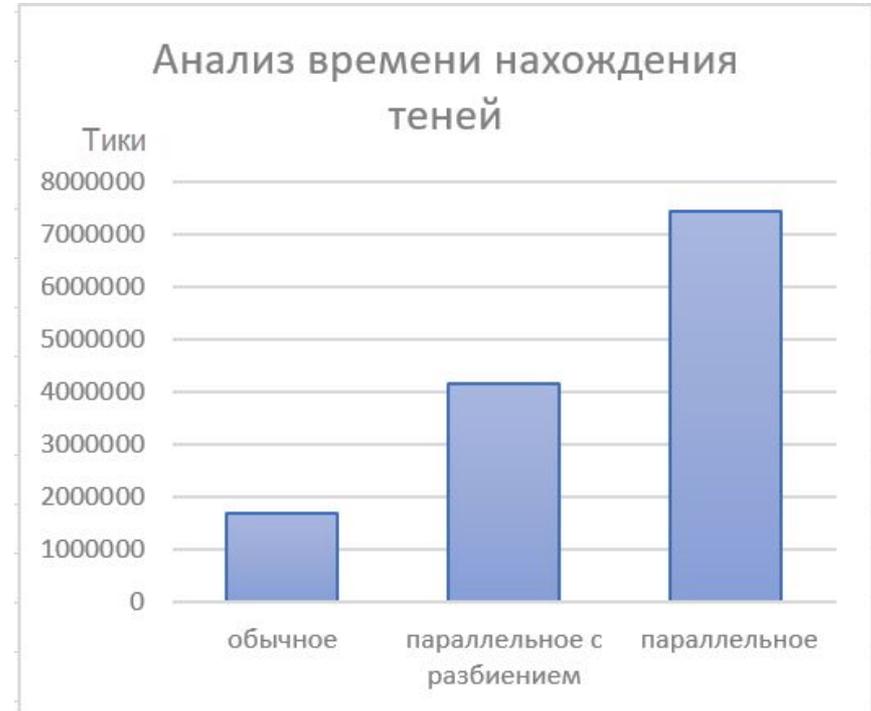
Погода



Эксперимент

Параллельное нахождение теней с помощью функции `Parallel.For` показало увеличение времени работы в 2.5 раза.

- 1666536 тиков – обычное нахождение теней;
- 4169953 тиков – параллельное нахождение теней, с разбиением изображения;
- 7452667 тиков – параллельное нахождение теней.



Спасибо за внимание

Москва 2019