Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

Визуализация озера с растительностью и фламинго

Студент: Пронина Лариса Юрьевна ИУ7-54Б
Руководитель: Кострицкий Александр Сергеевич

2023

Цель и задачи курсовой работы

Цель — разработка программного обеспечения для построения трехмерной сцены и визуализации озера с растительностью и фламинго.

Задачи:

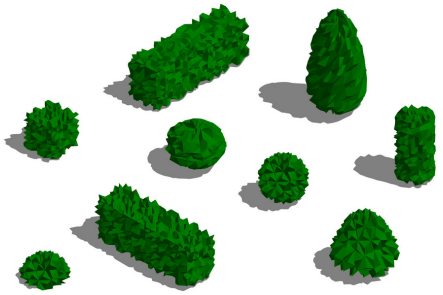
- формализовать задачу в виде IDEF0 диаграммы;
- провести анализ существующих алгоритмов компьютерной графики: удаления невидимых линий и поверхностей, построения теней, закраски и освещения;
- спроектировать программное обеспечение для построения трехмерной сцены и визуализации озера с растительностью и фламинго;
- выбрать средства реализации спроектированного программного обеспечения и разработать его;
- исследовать зависимость скорости генерации кадра разработанного программного обеспечения от числа объектов на сцене и количества источников света.

Формализация процесса создания видео

Объекты сцены

Фламинго

Невыпуклый объект. Задается множествами точек и полигонов. Для каждого множества определен материал.



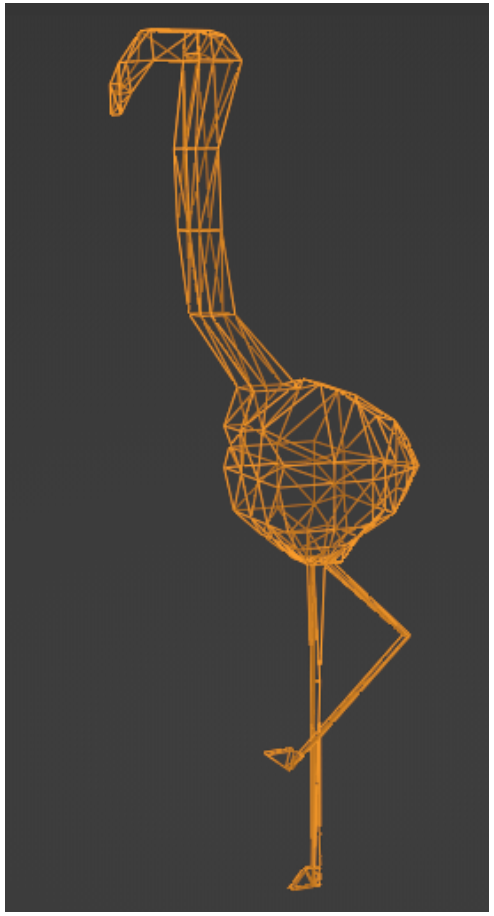
Растительность

Задается множеством точек и полигонов, которые определяют каждый куст на сцене.

Источник света

Невидимый точечный объект. Задается 3 координатами положения и коэффициентом освещенности.

Способ задания трехмерной модели



Была создана модель фламинго, которая задается полигональной сеткой, каждый полигон представляет собой треугольник. Полигональная сетка хранится в виде списка граней.

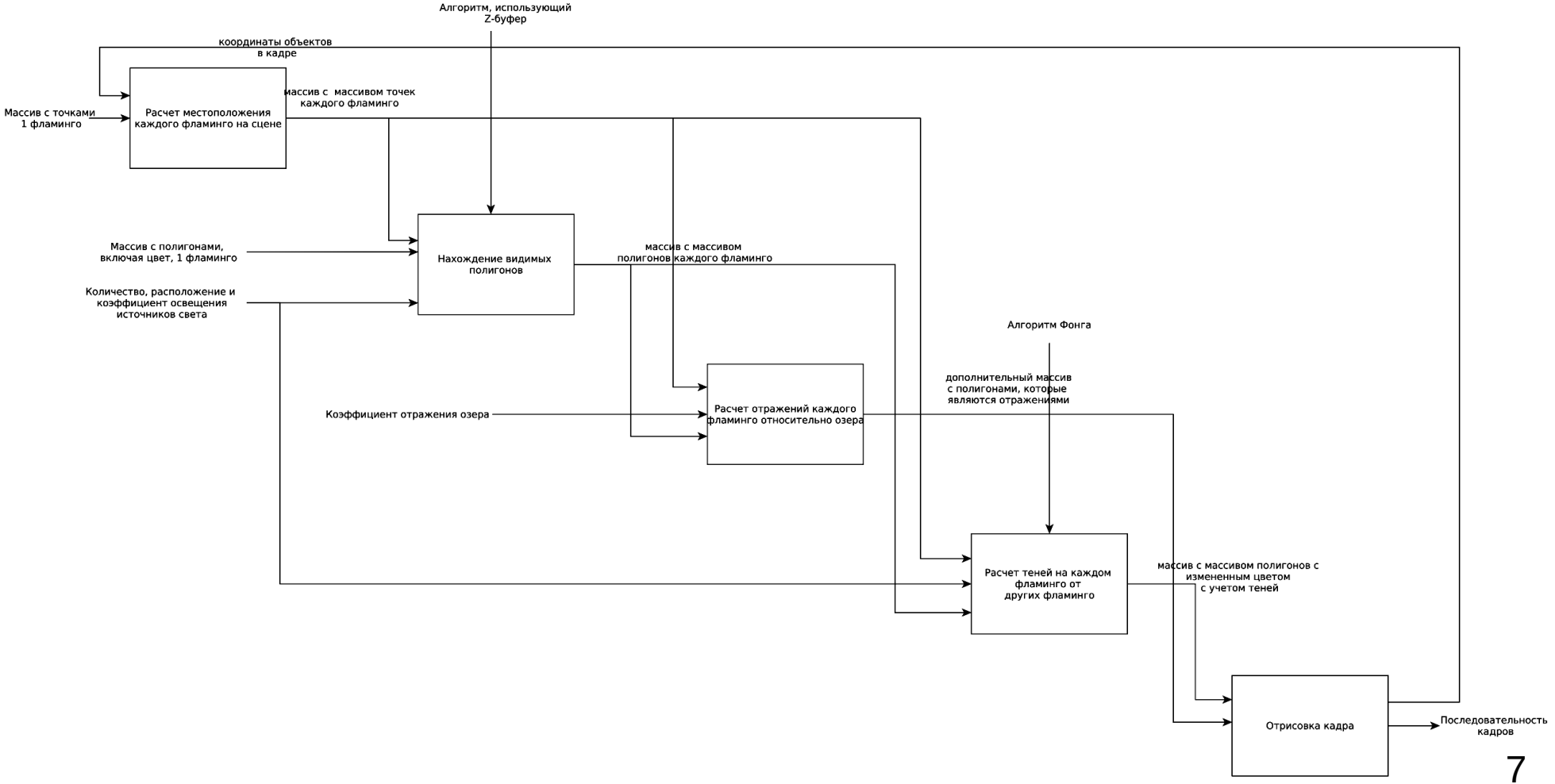
Алгоритмы для генерации кадра

Сравнительная таблица алгоритмов удаления невидимых линий и поверхностей

Критерии	Алгоритмы			
	z-буфер	Робертса	Обратная трассировка	Варнока
Вычислительная трудоемкость	**O(N*n)**	$O(n^2)$	O(N*n)	O(N*n)
Работает с невыпуклыми	**да**	нет	да, с доп. проверками	да, с доп. обработкой сцены
Рабочее пространство	**изображение**	объект	изображение	изображение
Применение для сцен в реальном времени	**может быть эффективным, но потреблять больше памяти**	используется, но может быть не эффективным для сложных сцен	обеспечивает высокое качество изображений, но может работать дольше других при сложных сценах	используется и позволяет обрабатывать большие сцены за счет эффективной сортировки и однократного прохода по полигонам

В качестве алгоритма удаления невидимых линий был выбран алгоритм, использующий Z -буфер, построение теней выполнялось с помощью модифицированного алгоритма z-буфера, освещение и закраска с помощью алгоритма Фонга.

Формализованная модель визуализации трехмерной сцены



Выбранные средства реализации

Язык программирования — C++
Среда разработки — Qt Creator
Фреймворк для тестирования — Google Test

Модульное тестирование

В качестве меры, используемой при тестировании программного обеспечения выбрано покрытие кода. Созданный набор модульных тестов имеет покрытие, равное 40%.

Пример модульного теста:

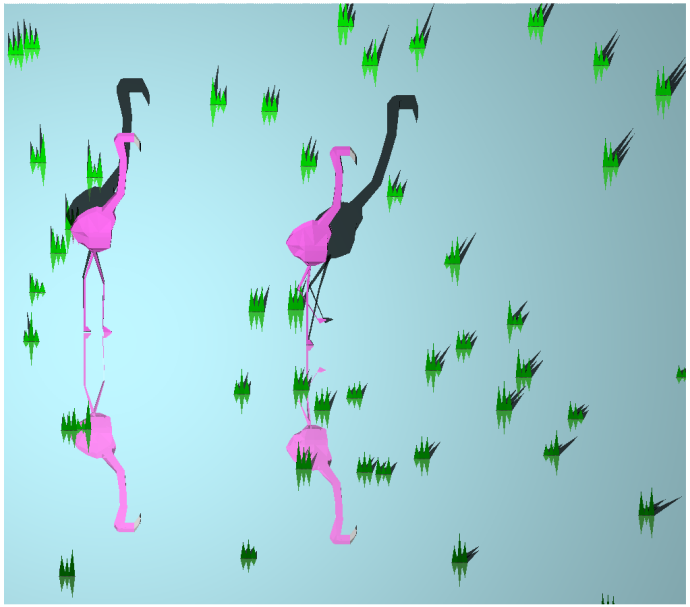
```
TEST(IsInsideTest, IsInsideTest) {
    drawer dr;
    std::vector<QVector3D> points = {
        QVector3D(0, 0, 0),
        QVector3D(2, 1, 0),
        QVector3D(3, 5, 0)
    };
    int x = 1;
    int y = 2;
    bool result = dr.is_inside(x, y,
points);
    EXPECT_TRUE(result);
}
```

Пример вывода результата тестирования:

```
GOOGLE TESTS:
[==========] Running 9 tests from 3 test suites.
[----------] Global test environment set-up.
[----------] 4 tests from DrawerTest
[ RUN      ] DrawerTest.BorderTest
[       OK ] DrawerTest.BorderTest (0 ms)
[ RUN      ] DrawerTest.BorderTest2
[       OK ] DrawerTest.BorderTest2 (0 ms)
[ RUN      ] DrawerTest.PutShadowPolygonTest
[       OK ] DrawerTest.PutShadowPolygonTest (5 ms)
[ RUN      ] DrawerTest.PutShadowBufferTest
[       OK ] DrawerTest.PutShadowBufferTest (0 ms)
[----------] 4 tests from DrawerTest (5 ms total)
...
[  PASSED  ] 9 tests.
```

Функциональное тестирование

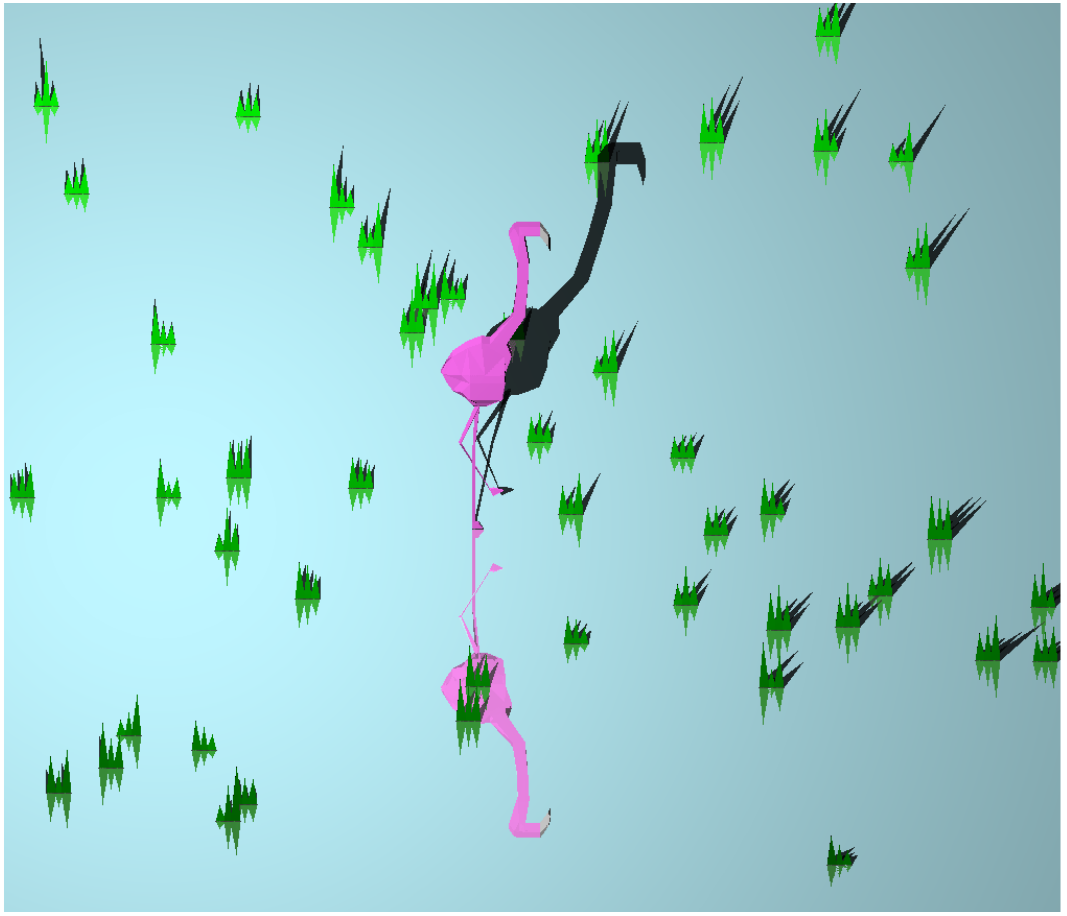
Алгоритм проведения функционального тестирования:
1) разработать тестовые случаи для программы;
2) составить входные данные для каждового тестового случая;
3) Получить картинку для этих входных данных;
4) визуально оценить результат.



Классы эквивалентности функционального тестирования:
1) на сцене 1 фламинго с растительностью на озере;
2) на сцене одновременно несколько фламинго (2, 6);
3) на сцене ни одного фламинго;
4) нескольких источников света;
5) установка значения прозрачности фламинго в 0.5;
6) разная плотность растительности (0, 0.4).

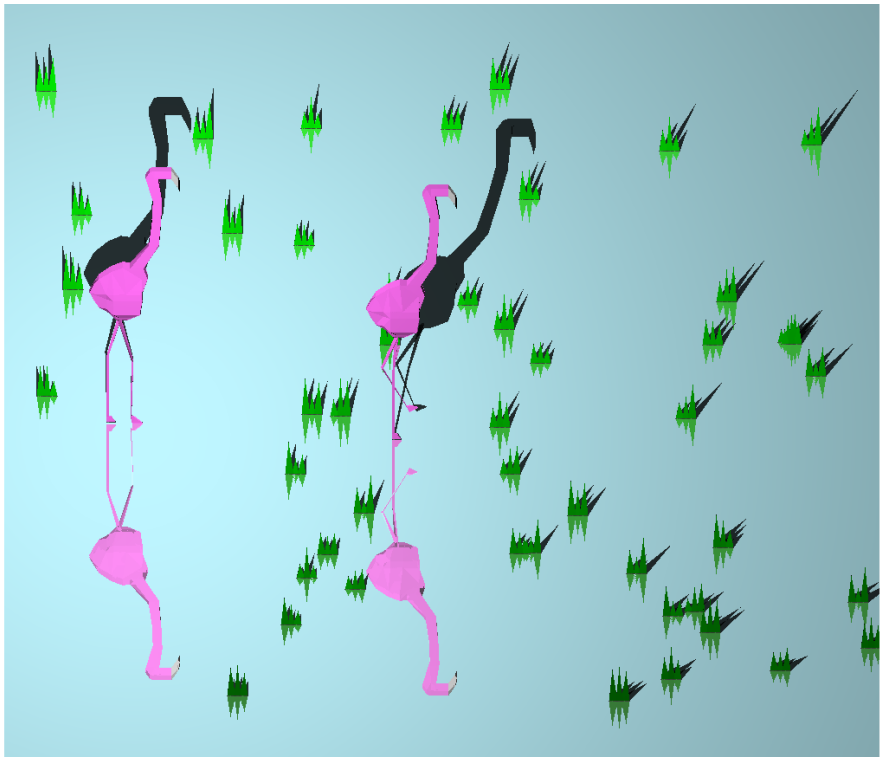
Примеры работы программы, которые подтверждают выполнение всех разработанных функциональных тестов, представлены на следующих слайдах.

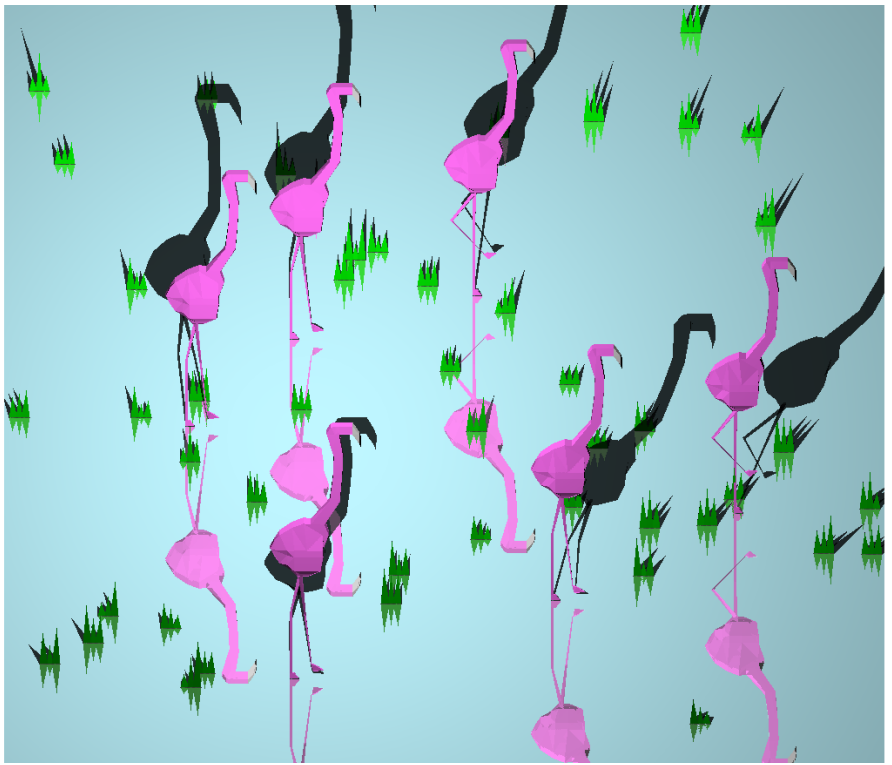
Функциональное тестирование



Функциональный тест для определения реалистичности фламинго и растительности на озере, а также определяющий возможность задания 1 фламинго на сцене.

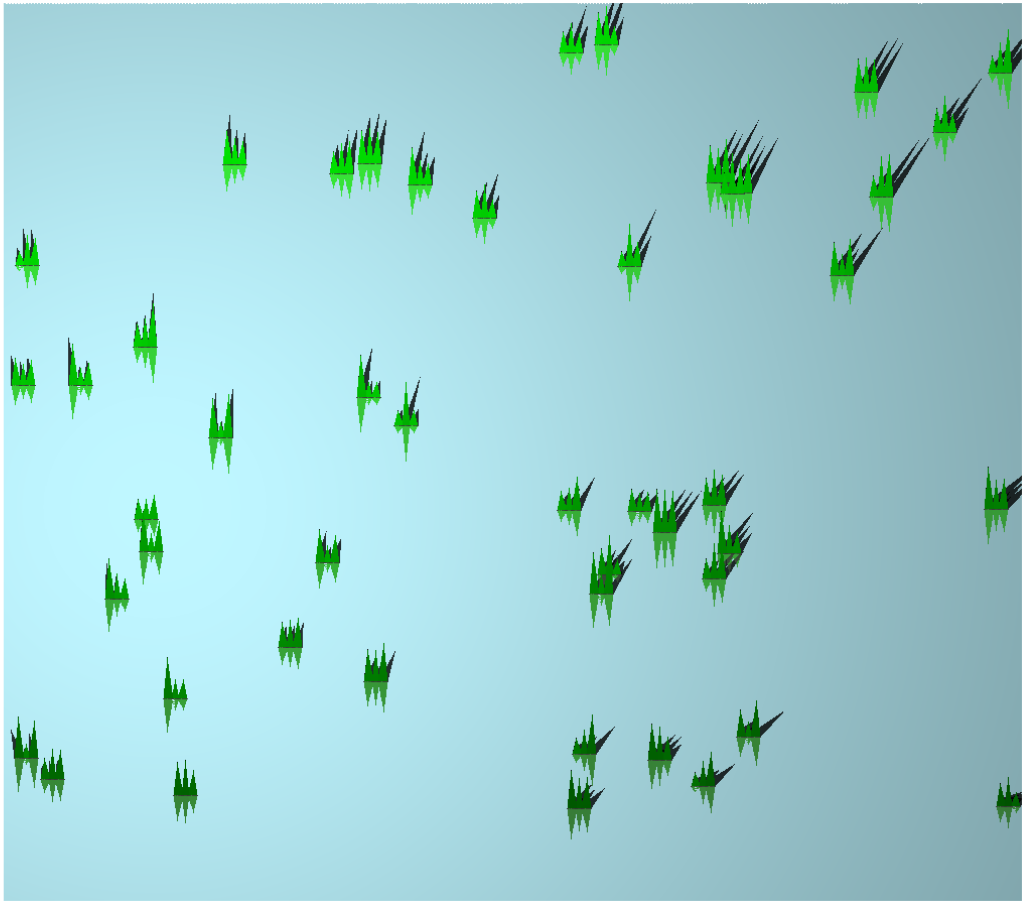
Функциональное тестирование





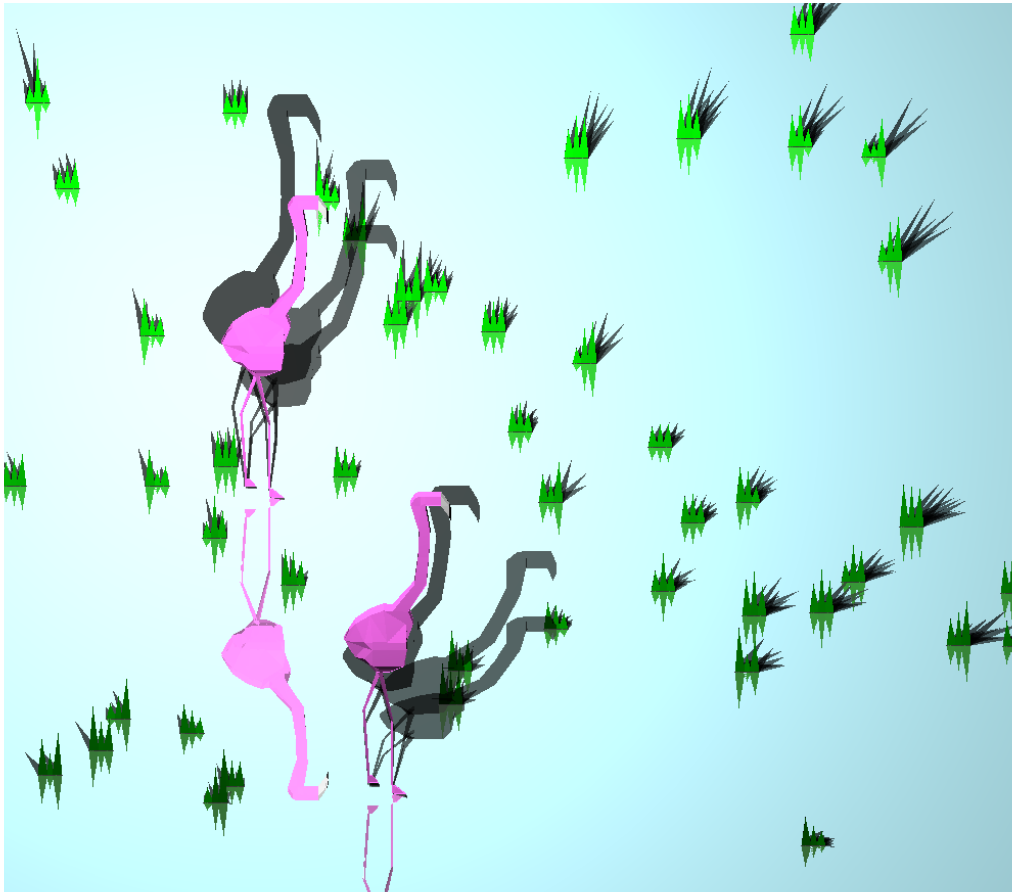
Тест для определения возможности задать несколько фламинго на сцене.

Функциональное тестирование



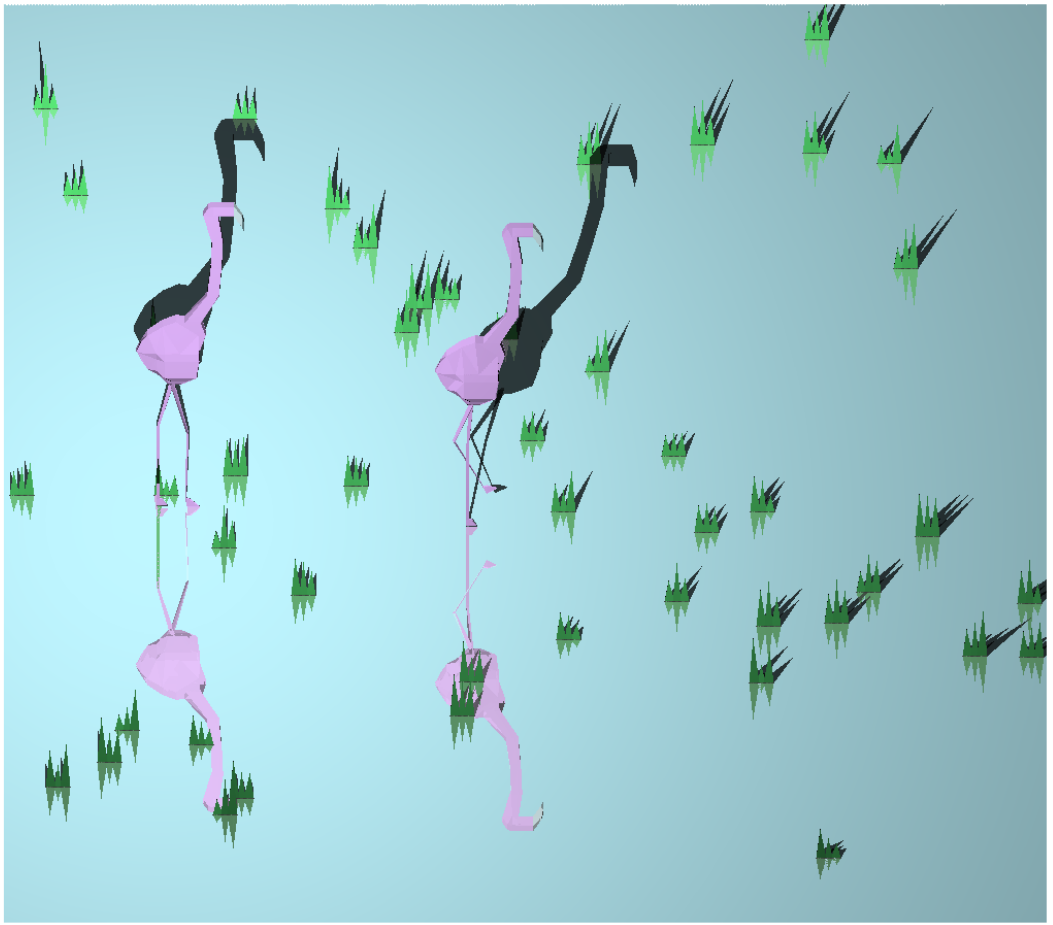
Тест для определения возможности не задавать ни одного фламинго.

Функциональное тестирование



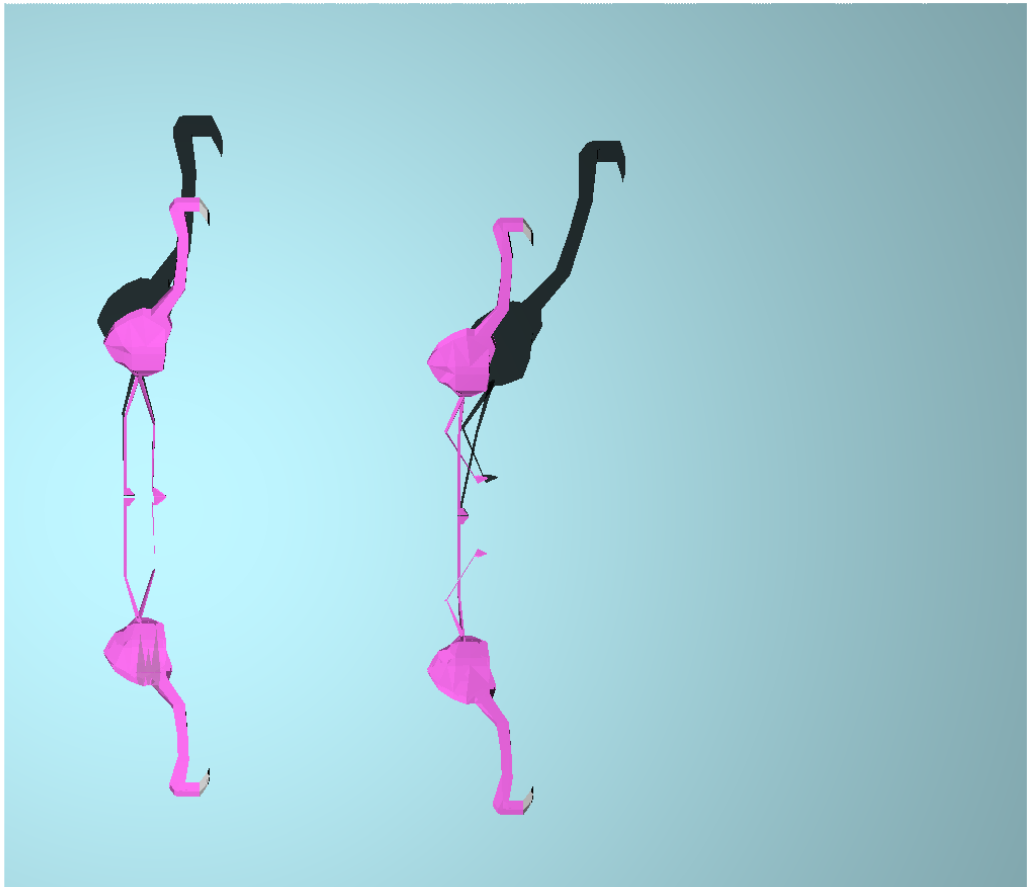
Тест для подтверждения того, что можно задать несколько источников света.

Функциональное тестирование



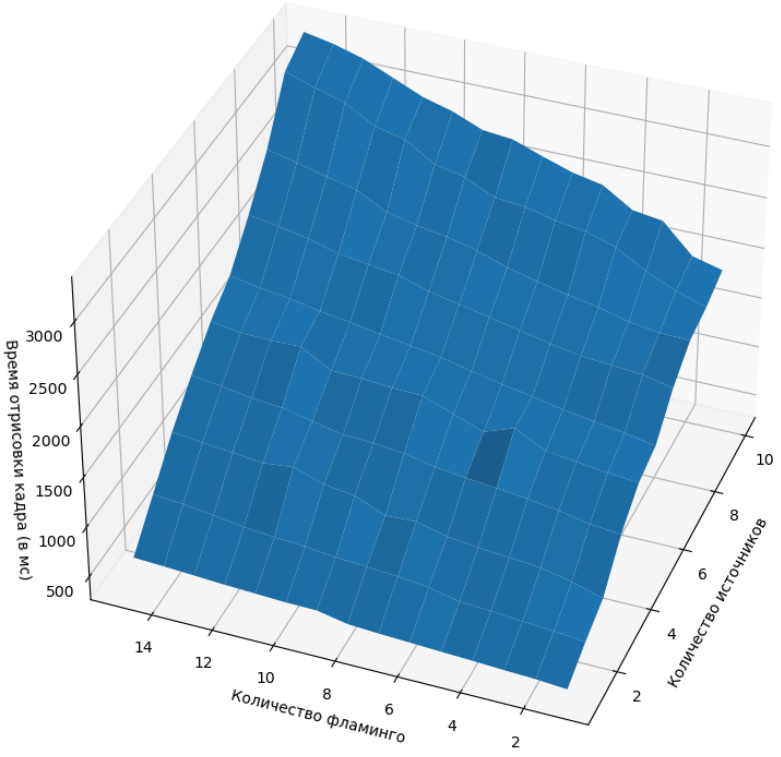
Функциональный тест, определяющий можно ли задать оптические свойства поверхностей фламинго.

Функциональное тестирование

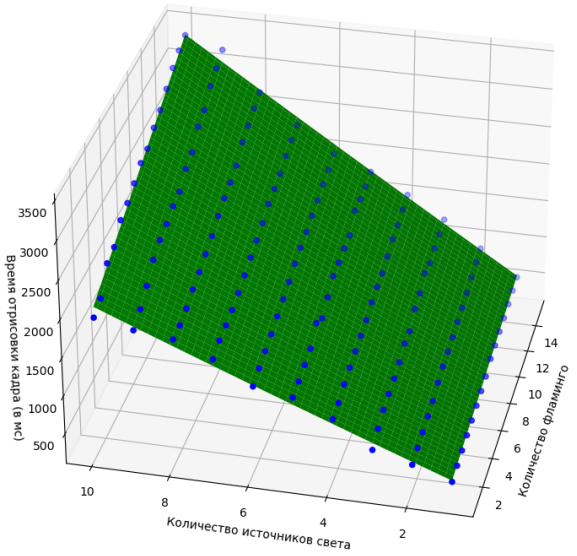


Функциональный тест, подтверждающий, что можно задать плотность растительности 0.

Зависимость времени генерации кадра от количества фламинго и источников света



Зависимость времени от количества фламинго и источников света на сцене



Функция, аппроксимирующая эту поверхность является полиномом 2 степени:

$f(x, y) = 0.833*y^2 + 0.038*x^2 + 8.598*x*y + 169.9*y + 8.7*x + 153.6$

Заключение

Поставленная цель была достигнута: было разработано программное обеспечение для построения трехмерной сцены и визуализации озера с растительностью и фламинго.