Визуализация лесистой местности

Студент: Княжев Алексей Викторович, ИУ7-52Б

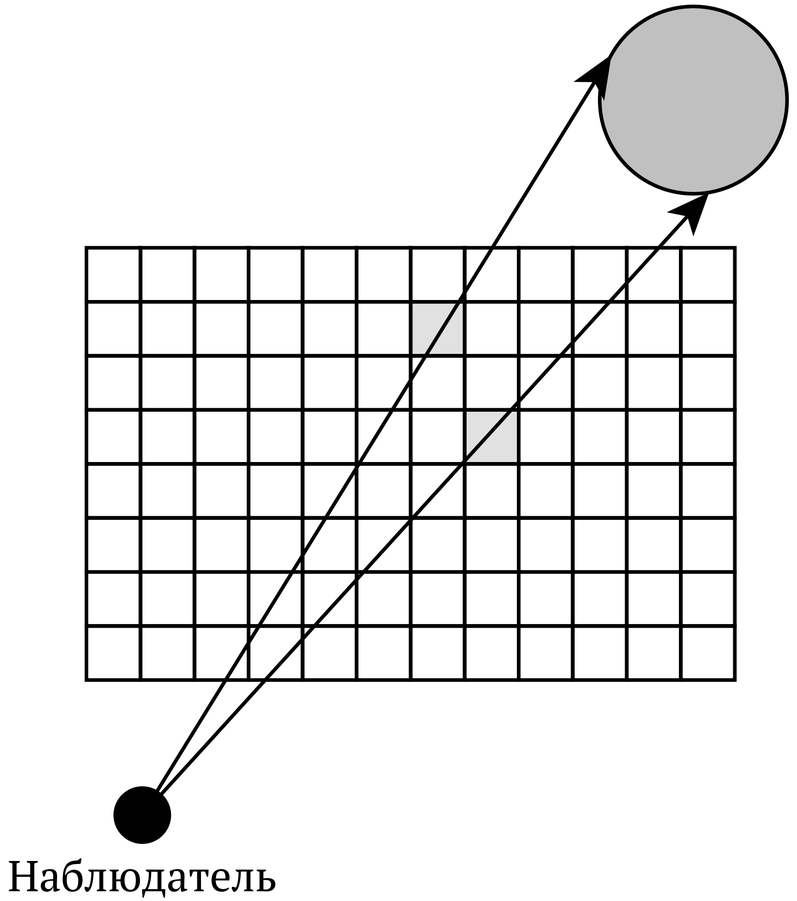
Руководитель: Кострицкий Александр Сергеевич

Постановка задачи

Разработать программу для построения трехмерного изображения лесистой местности. Количество деревьев, густота кроны, и расположение деревьев задается на этапе выполнения. Предусмотреть возникновение тени от кроны деревьев.


Построение реалистичного изображения
Удаление невидимых линий и поверхностей
Учет теней и освещения
Алгоритмы разбиения на окна
Алгоритмы построчного сканирования
Алгоритмы, использующие z-буфер
Алгоритмы трассировки лучей

В качестве алгоритма удаления невидимых линий и поверхностей был выбран алгоритм обратной трассировки лучей. Идея алгоритма заключается в отслеживании световых лучей в обратном направлении, то есть от наблюдателя к объектам, от объектов к источникам света. Это позволяет обрабатывать только видимые наблюдателем лучи.

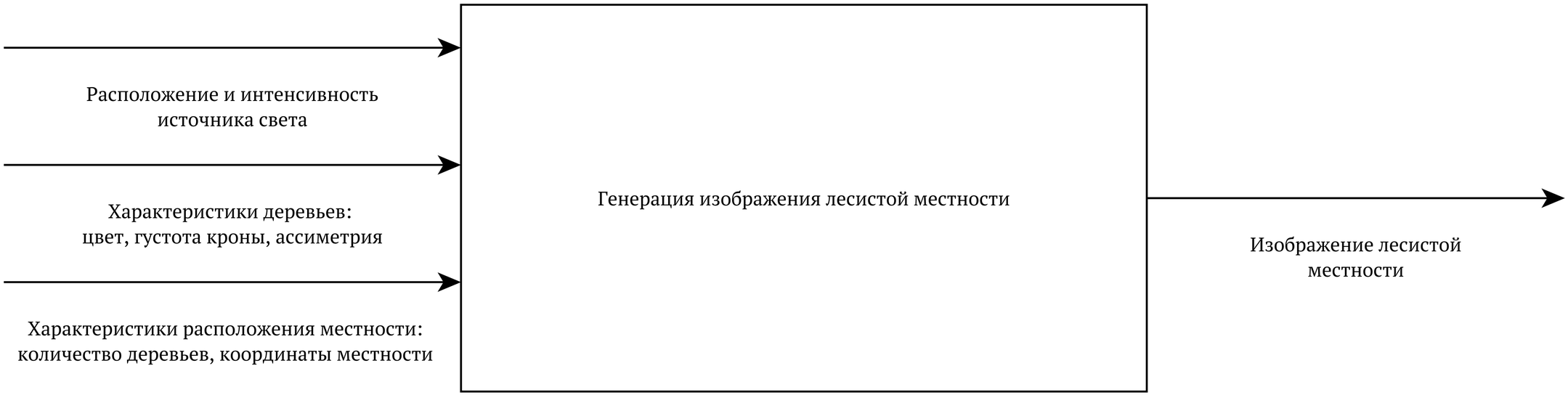


Для создания деревьев используется метод процедурной генерации. Учитываются следующие особенности. Во-первых, ветка разделяется на две ветки-потомка. Во-вторых, после разделения, длина ветки не изменяется, изменяется только ее толщина. Таким образом, если к разделению ветки повесить качели, то со временем они будут оставаться на месте.

В связи с данными фактами, растение в программе представлено в виде бинарного дерева. У этого дерева имеется метод получения питательных веществ, который увеличивает размеры текущей ветки, и передает остаток питания потомкам.

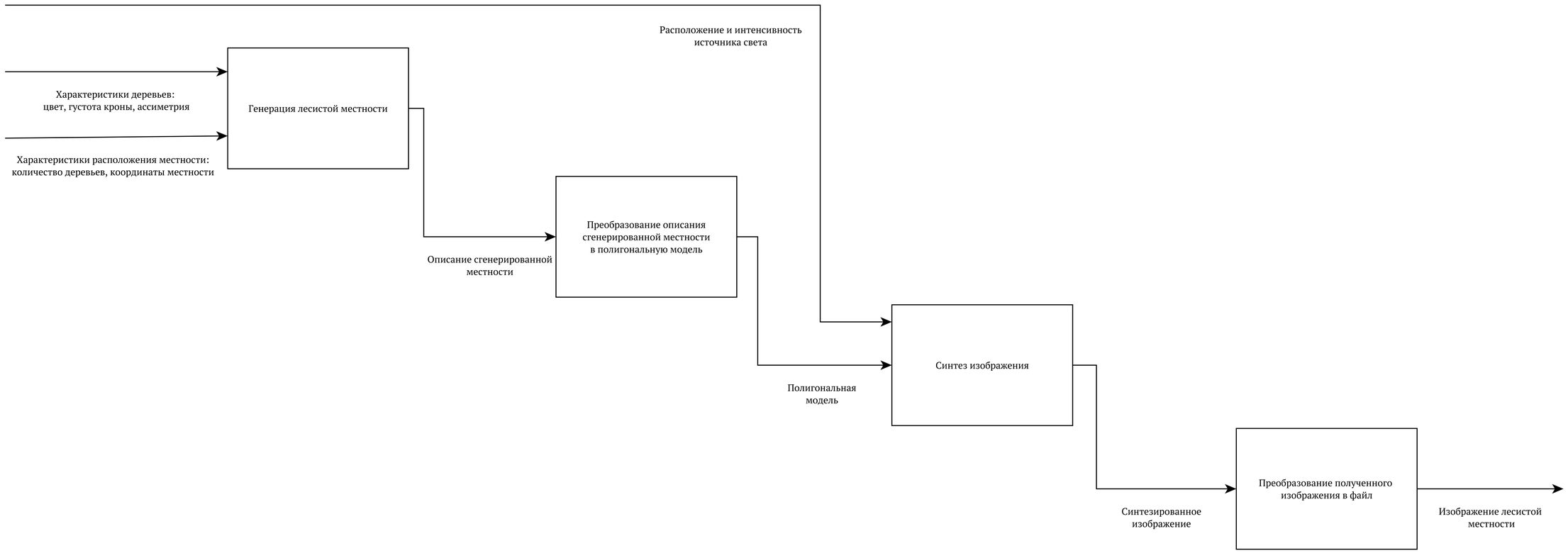
Фунциональная модель программы

Диаграмма IDEF0



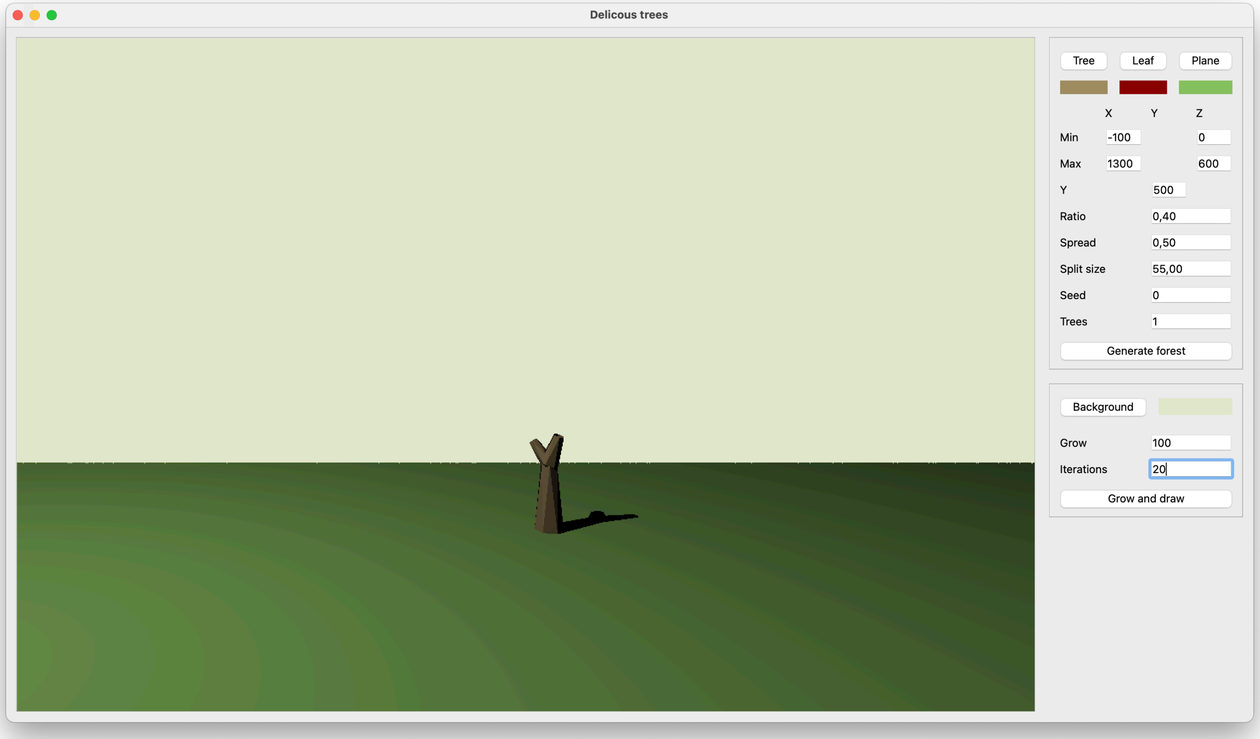
Фунциональная модель программы

Диаграмма IDEF0



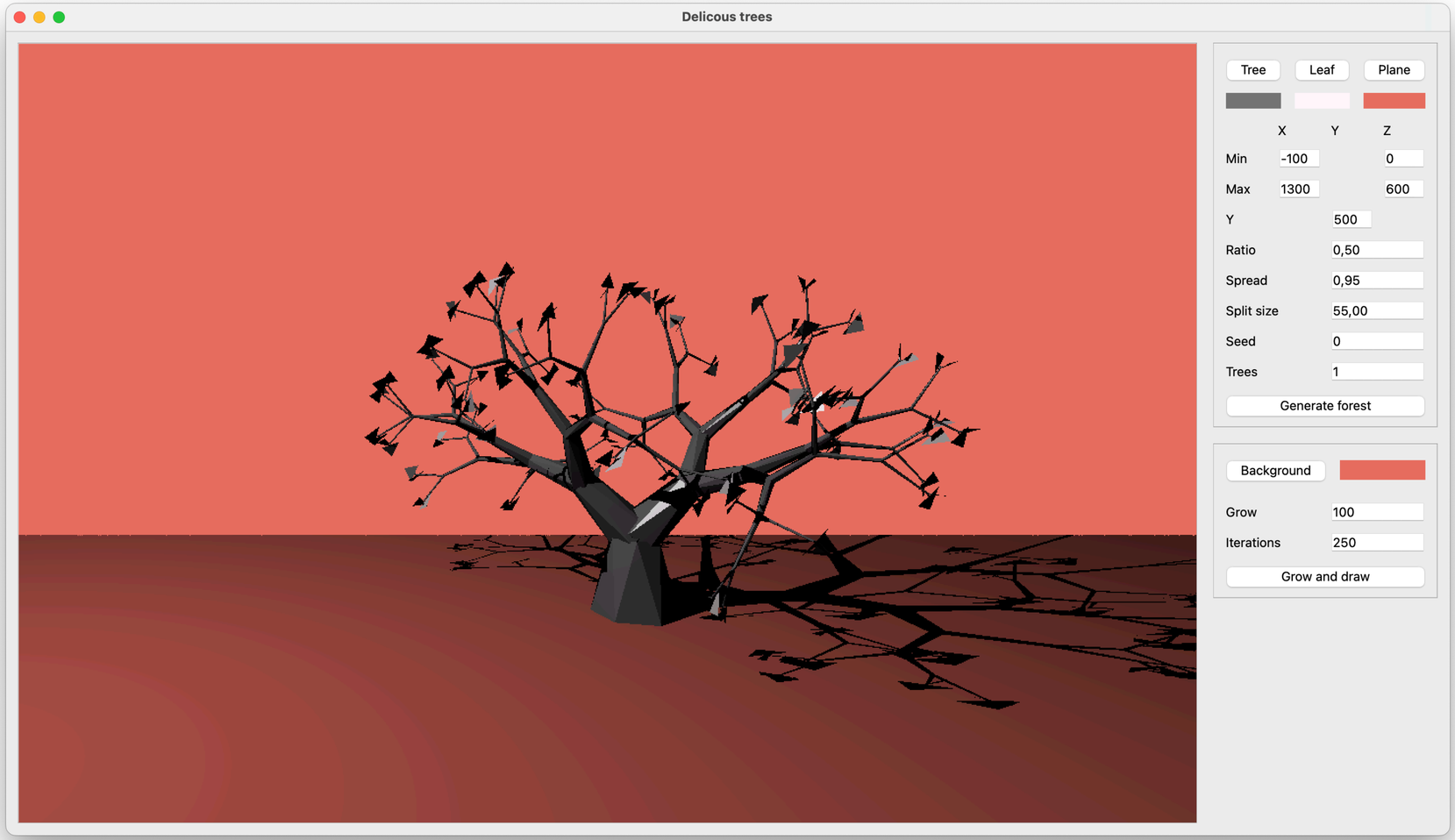
Интерфейс программы

Программа представляет собой оконное приложение, включающее в себя область вывода изображения, а также область ввода параметров генерации изображения, такие как количество деревьев, границы области местности, цвета деревьев, густота кроны и т. д.

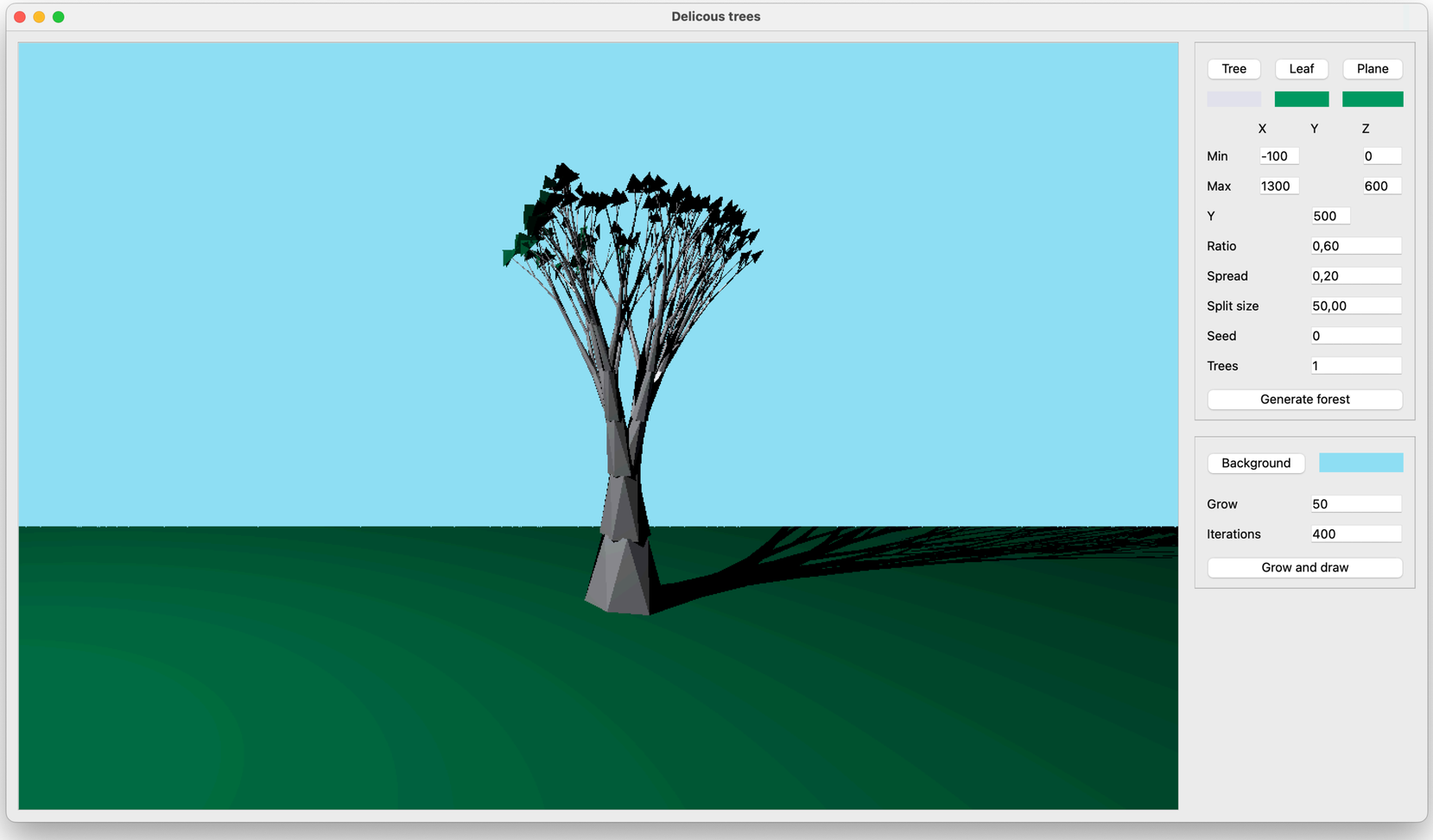


Примеры работы программы

Примеры работы программы

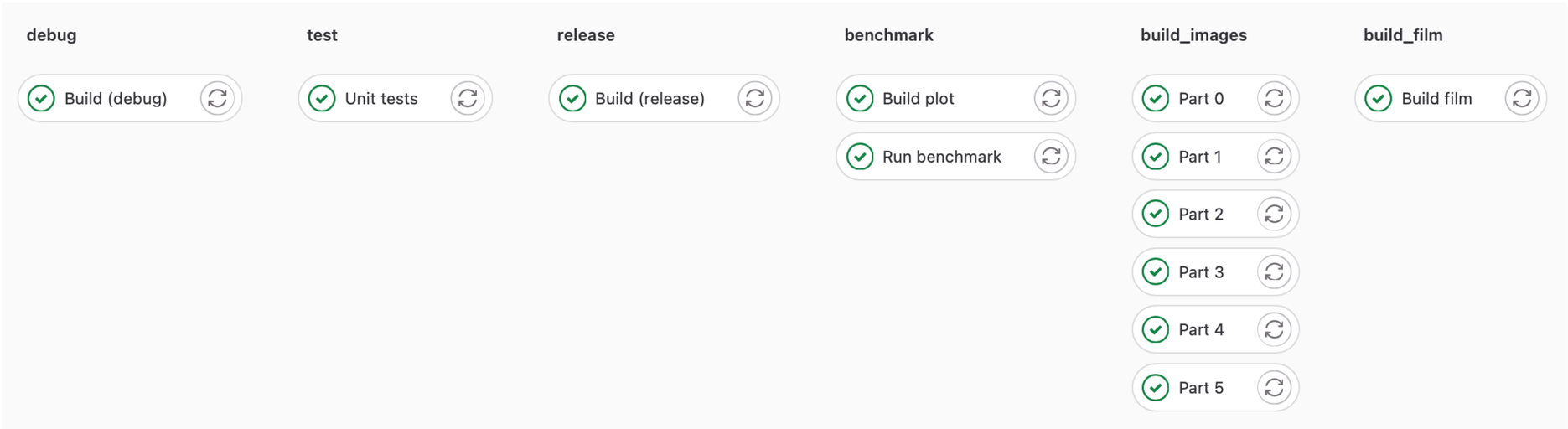


Примеры работы программы



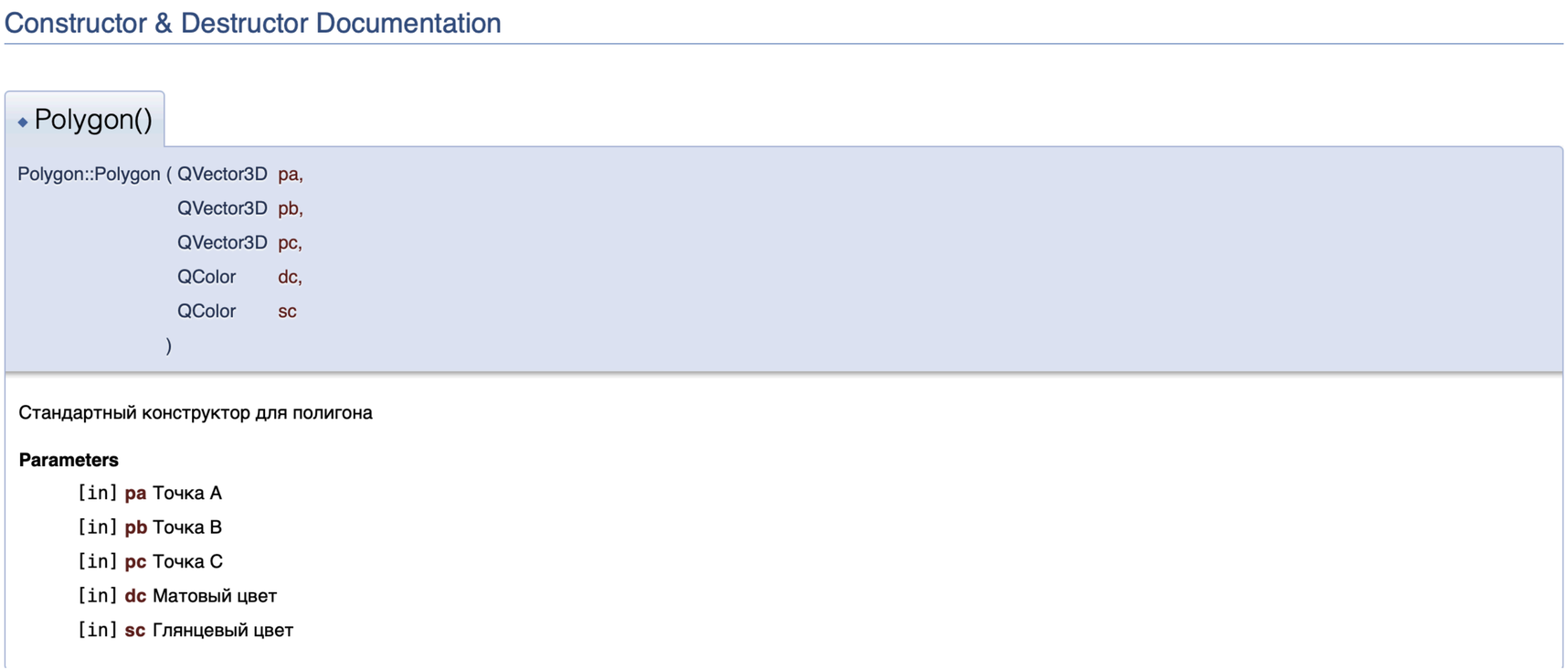
CI/CD

Для программы настроен конвейер, включающий в себя отладочную сборку, тестирования, выпускаемую сборку, замеры времени и генерацию фильма. При очередном коммите в git-репозиторий, конвейер запускается, обеспечивая контроль качества и работы программного продукта.



Документация

Программные модули были продокументированы с использованием средства для документации кода Doxygen. Пример сгенерированной страницы с документацией:



Constructor & Destructor Documentation

◆ Polygon()

```
Polygon::Polygon ( QVector3D  pa,
                   QVector3D  pb,
                   QVector3D  pc,
                   QColor     dc,
                   QColor     sc
                 )
```

Стандартный конструктор для полигона

Parameters

[in] **pa** Точка A

[in] **pb** Точка B

[in] **pc** Точка C

[in] **dc** Матовый цвет

[in] **sc** Глянцевый цвет

Исследование

На основе полученного программного продукта было проведено исследование: сравнение скорости работы алгоритма обратной трассировки лучей от количества потоков, используемых для обработки сцены.

По результатам замеров можно сделать следующие выводы: заметное ускорение скорости работы алгоритма наблюдается при повышении числа потоков до количества ядер процессора. При дальнейшем повышении количества потоков, повышения производительности не наблюдается.

Исследование

Исследование проводилось на процессоре Intel Xeon с 96 ядрами.

При количестве потоков более ста, график скорость работы алгоритма практически вырождается в прямую, параллельную оси X графика.