|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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)** |

ФАКУЛЬТЕТ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Информатика и системы управления\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА \_\_\_Программное обеспечение ЭВМ и информационные технологии\_\_\_\_\_\_\_\_\_\_\_

**ОТЧЕТ ПО ПРОИЗВОДСТВЕННОЙ ПРАКТИКЕ**

Студент\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Малышев Иван Алексеевич\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*фамилия, имя, отчество*

Группа\_\_\_\_ИУ7-41Б\_\_\_\_\_

Тип практики \_\_\_\_\_\_\_\_\_\_\_\_\_ Технологическая\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Название предприятия\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ кафедра\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Студент **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_**Малышев И. А.**\_\_\_\_**

*подпись, дата фамилия, и.о.*

Руководитель практики **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_**Куров А. В.**\_\_\_\_\_\_\_\_**

*подпись, дата фамилия, и.о.*

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*2021 г.*

**Индивидуальное задание**

Разработать программу генерации трёхмерного ландшафта в дополненной реальности. Программа должна строить трёхмерную модель ландшафта на основе двумерной карты ландшафта и совмещать построенную модель с видеоизображением, получаемым с веб-камеры. Программа должна предоставлять возможность вращать модель вокруг оси, перпендикулярной плоскости основания модели, перемещать её параллельно этой плоскости, изменяя видимую часть модели, и масштабировать.

**Оглавление**

**Введение**

В настоящее время многие компании заинтересованы в технологии дополненной реальности и стараются активно её использовать, так как это очень наглядный, интерактивный метод представления цифровой информации в контексте физического пространства. Новая виртуальная среда образуется путём наложения запрограммированных виртуальных объектов поверх видеосигнала с камеры, и становится интерактивной путём использования специальных маркеров.

Дополненная реальность уже много лет используется в медицине, в образовании, в рекламной отрасли, в ландшафтном дизайне, в военных технологиях, в игровой индустрии, для мониторинга объектов и в мобильных устройствах.

Задачи генерации ландшафта актуальны для создания полностью или частично виртуальных ландшафтов, которые выглядят правдоподобно. Подобные ландшафты используются в фильмах с компьютерной графикой, трёхмерных играх и в качестве демонстрационного материала для различных архитектурных объектов или дизайнерских решений по ландшафту.

Процесс моделирования вручную даже с помощью вспомогательного ПО крайне трудоёмкий, неудобный и неблагодарный, по сравнению с моделированием других трёхмерных моделей. Основой виртуального мира должен быть ландшафт значительных размеров, а для реалистичности он должен быть детализированным и разнообразным. Кроме того, чтобы увидеть эту детализацию, нужна система освещения, а значит понадобятся корректные, реалистичные карты нормали и в большом количестве. Очевидно, что создавать всё это вручную нецелесообразно, а результат будет, в лучшем случае, посредственным.

Альтернативой ручного моделирования ландшафта является процедурная генерация с использованием генераторов случайных чисел, что включает в себя построение математической модели ландшафта и реализации алгоритма генерации. Именно этот способ моделирования ландшафта и будет рассмотрен в работе.

Применение технологии дополненной реальности к системам трёхмерного моделирования ландшафта даёт более наглядное представление о виртуальной сцене, объектах на ней и позволяет интерактивно взаимодействовать с ней. Подобная интеграция технологий может быть полезна для более доступной демонстрации архитектурных объектов или проектов ландшафтных дизайнеров: достаточно будет одного смартфона, чтобы самому увидеть планируемое.

Целью работы является проектирование программного обеспечения для генерации трёхмерного ландшафта и отображения его в дополненной реальности. Таким образом, необходимо решить следующие задачи:

1. Формализовать объекты синтезируемой сцены и преобразования над ней;
2. Провести анализ существующих алгоритмов синтеза ландшафта и отображения виртуальной сцены в дополненной реальности, обосновать оптимальность выбранных алгоритмов;
3. Реализовать выбранные алгоритмы;
4. Разработать программный продукт для визуализации и преобразования виртуальной сцены в дополненной реальности.
5. **Аналитическая часть**
   1. **Формализация задачи**

Сцена содержит следующие объекты:

* Модель ландшафта – трёхмерный объект, состоящий из следующих частей:
* Полигональная сетка – совокупность вершин, рёбер и граней, которые определяют форму многогранного объекта в трёхмерной компьютерной графике и объёмном моделировании. В данном случае сетка будет определять форму ландшафта.
* Карта нормалей – растровое изображение в формате RGB, каждый пиксель которого несёт информацию о нормали. Нормали, полученные из карты, используются для искажения уже имеющихся нормалей, которые используются в расчётах освещения. Так, в результате получаются по-разному освещённые и затенённые участки. Как итог, абсолютно плоская поверхность визуально кажется неровной.
* Карта освещения – структура данных, хранящая информацию об освещённости (яркости) поверхности трёхмерной сцены. Карты освещения рассчитываются предварительно для неподвижных объектов, ускоряя тем самым рисование освещённой сцены.
* Текстуры ландшафта – набор растровых изображений, накладываемые на поверхность полигональной модели ландшафта для придания ей цвета, окраски или иллюзии рельефа.
* Источник света – точка пространства, подобная точке положения наблюдателя. Принимает ортогональную проекцию визуализируемой сцены из своего положения с некоторым ограниченным обзором. В зависимости от расположения источника и направления распространения лучей света, определяет тень от объектов, расположенных на сцене. Положение источника света задаётся относительно текущей точки наблюдения последовательными поворотами по осям X и Y.
  1. **Анализ способов представления данных о ландшафте**

Существует несколько основных принципов представления данных для хранения информации о ландшафтах:

* Первый — использование регулярной сетки высот (или ещё другое название Карта Высот — HeightMap).
* Второе — использование иррегулярной сетки вершин и связей, их соединяющих (т.е. хранение простой триангулизированной карты).
* Третий — хранение карты ландшафта, но в данном случае хранятся не конкретные высоты, а информация об использованном блоке. В этом случае создаётся некоторое количество заранее построенных сегментов, а на карте указываются только индексы этих сегментов.
  + 1. **Карта высот**

В первом способе данные представлены в виде двумерного массива. Индексы массива задают две координаты вершины, X и Y, а третья координата задаётся конкретным значением в данной ячейке, то есть высота точки относительно плоскости XY. Это и есть карта высот. Обычно её представляют в виде монохромного изображения, что даёт нам предел высоты от 0 до 255. Это позволяет легко вносить изменения и более-менее наглядно просматривать данные. Тогда двумя координатами будет положение конкретного пикселя на картинке, а третья координата будет представлена цветом (чем выше значение, прямая зависимость от яркости пикселя — тем больше значение высоты для этой точки). С помощью этого способа можно представить достаточно обширные пространства.

Плюсы данного подхода:

1. Наглядность данных: в любой программе просмотра графических файлов можно сразу увидеть всю информацию о форме ландшафта;
2. Простота реализации: лёгкость нахождения координат вершины на карте;
3. Эффективность: данный способ позволяет эффективно описывать неровности поверхности, в отличие от аналогичных способов – расчёт освещённости остаётся неизменным.

Минусы:

1. Избыточность данных: особенно при построении поверхностей, близкие к плоским.
   * 1. **Иррегулярная сетка**

Во втором способе форма ландшафта представляется в виде трёхмерной модели. Это даёт основной выигрыш против карты высот в плане экономии памяти. Но у этого способа есть множество недостатков.

Плюсы данного подхода:

1. Используется значительно меньше информации для построения ландшафта. Нам необходимо хранить только значения высот каждой вершины и связи эти вершины соединяющие. Это даёт нам выигрыш в скорости при передаче огромных массивов информации в видеокарту, в процессе визуализации ландшафта.

Минусы:

1. Алгоритмы построения ландшафтов в основном предназначены для регулярных карт высот. Оптимизация таких алгоритмов под этот способ потребует значительных усилий;
2. Хранение, просмотр, модификация такого ландшафта также представляет сложности. При использовании карт высот вы пользуетесь достаточно простыми и "стандартными" средствами пиксельной графики. Хотя бы тот же MS Paint. Тут же вам потребуются более навороченные и "весомые" пакеты.
   * 1. **Посегментная карта высот**

В третьем способе также используются карты высот. Только вместо высот в ней хранятся индексы ландшафтных сегментов. Как эти сегменты представлены – не имеет значения. Они могут быть и регулярными, и иррегулярными (причём можно использовать и те и другие одновременно).

Плюсы данного подхода:

1. Кроме самих ландшафтов в таких блоках можно хранить и информацию о зданиях, строениях, растениях, специфических ландшафтных решениях (например, пещеры или скалы, нависающие друг над другом);
2. Возможность создания нескольких вариантов одного и того же сегмента, но при разной степени детализации. В зависимости от скорости или загруженности компьютера можно выбирать более или менее детализованные варианты (так называемые LOD ландшафты — LOD — Level Of Detail).

Минусы:

1. Проблема стыковки разных сегментов: не понятно, как состыковывать регулярные и иррегулярные сетки;
2. Неочевидность данных: взглянув на картинку, вы не сможете моментально представить, как это должно будет выглядеть в игре;
3. Проблема модификации: для разных сегментов используются разные инструменты редактирования.

**Итог**

Проанализировав все способы представления данных о ландшафте, можно прийти к следующим выводам:

* Первый способ для решения задачи является самым простым в обращении и для него уже существует множество алгоритмов построения ландшафтов.
* Второй способ для решения задачи не подойдёт – алгоритмы генерации ландшафта в основном созданы для первого способа, поэтому придётся потратить силы для модификации под второй способ.
* Третий способ для решения задачи тоже не подойдёт, так как возможность содержания иррегулярных сеток влечёт за собой проблемы второго способа.

На основе этого приходим к выбору первого способа, карты высот.

* 1. **Анализ алгоритмов процедурной генерации ландшафта**

За последние годы в этой области было описано множество методов генерации случайных карт высоты, и большинство из них сводилось к единственному простому условию: создать случайный набор значений и фильтровать его значения до тех пор, пока ландшафт не станет достаточно гладким, то есть смежные элементы высотной карты содержат значения, отличные на некую величину. Рассмотрим несколько из них.

* + 1. **Алгоритм Diamond-Square**

Этот алгоритм основан на алгоритме *midpoint displacement* и имеет всего два шага, «diamond» и «square»:

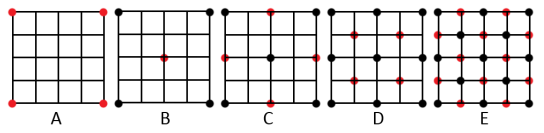
1. «Diamond»: берётся квадрат из четырёх точек. Вычисляется центральная точка этого квадрата. Её значение вычисляется как среднее значений вершин квадрата с небольшим случайным смещением;
2. «Square»: вычисляется значения точек, лежащие на середине рёбер квадрата как среднее значений центра и вершин квадрата, принадлежащие данным рёбрам.
3. Процесс повторяется с новыми квадрантами до тех пор, пока не установятся все значения карты высот

Рис. 1.1. Два этапа работы алгоритма diamond-square

Плюсы:

1. Простота: алгоритм довольно простой с точки зрения реализации;
2. Является довольно быстрым по сравнению с другими алгоритмами.

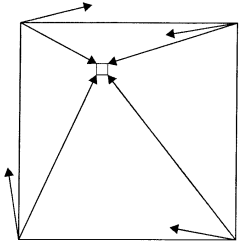
Минусы:

1. Есть вероятность появления артефактов на ландшафте, напоминающие всплески;
2. Сложно контролировать рельеф местности.
   * 1. **Шум Перлина**

В общем виде является *n*-мерной функцией шума. В данном разделе будет рассмотрена двухмерная версия шума Перлина.

Данный алгоритм использует два инструмента: шумовую функцию и интерполяцию. Основная идея такова:

1. *Задать случайные данные на основе данных ячейки сетки.* Для этого нужно определить сетку поверх карты высот и в каждой точке сетки определить случайный градиент единичной длины, который указывает в случайном направлении в пределах каждого из квадратов. Для каждого пикселя изображения определяется ячейка, где он находится и строится значение высоты, которое основано на векторах узлов этой ячейки и 4-х диагональных векторах, соединяющих углы ячейки с текущим пикселем;

Рис. 1.2. Расчёт значения высоты пикселя при помощи градиентов и векторов, соединяющих точки сетки с расчётной позицией

1. *Интерполировать полученные данные для вычисления значения высоты пикселя.* Вычисляем скалярные произведения векторов, лежащие на узлах сетки. Имея четыре значения, нужно объединить их, произвести три смешивания. Для этого нужно вычислить веса смешиваний на основе положения текущего пикселя в ячейке. Веса определяются функцией *smootherstep* (полином , где вместо *t* подставляются значения *x* и *y*). Дальше смешиваются значения скалярных произведений в верхних и нижних углах с помощью линейной интерполяции с использованием веса по *x*. Эти результаты также смешиваются через линейную интерполяцию, но уже с использованием веса по *y*. Данный результат и будет высотой пикселя.

Чтобы контролировать генерацию шума, существует набор параметров:

* Масштаб(scale) – число пикселей на единицу длины сетки;
* Октавы(octaves) – количество уровней детализации шума;
* Лакунарность(lacunarity) – множитель, который определяет изменение частоты с ростом октавы (по умолчанию равен двум, что соответствует определению октавы);
* Стойкость(persistence) – множитель частотной амплитуды, который определяет изменение амплитуды с ростом октавы.

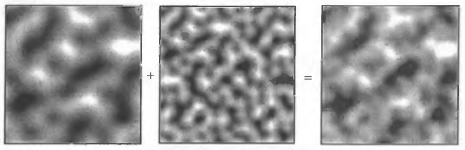
Для достижения качественных ландшафтов, нужно комбинировать шумы разных масштабов и количества октав, складывая или перемножая их. Таким образом, можно добиться ландшафтов, сочетающие разные типы местности: луга, горы, берега и т. д.

Рис. 1.3. Пример изображения шума Перлина, полученный сложением двух октав.

Плюсы:

1. Является относительно быстрым и детерминированным: зная лишь параметры генерации, можно получить значение высоты любой точки мира, не зная значений высоты соседних точек;
2. Благодаря механизму комбинаций октав, позволяет более тщательно настроить детали ландшафта.

Минусы:

1. Без комбинаций разных октав, ландшафт получается однообразным и не детализированным.
   * 1. **Холмовой алгоритм**

Это простой итерационный алгоритм, основанный на нескольких входных параметрах. Основная идея алгоритма проста:

1. Создаём двухмерный массив и инициализируем его нулевым уровнем (заполняем все ячейки нолями);
2. Берём случайную точку на ландшафте или около его границ (за границами), а также берём случайный радиус в заранее заданных пределах. Выбор этих пределов влияет на вид ландшафта — либо он будет пологим, либо скалистым;
3. В выбранной точке "поднимаем" холм заданного радиуса;
4. Возвращаемся ко второму шагу и так далее до выбранного количества шагов. От него потом будет зависеть внешний вид нашего ландшафта;
5. Проводим нормализацию ландшафта;
6. Проводим "долинизацию" ландшафта.

Поднять холм - это создать параболоид вращения на сетке ландшафта. Выбранный радиус задаёт радиус основания холма, а квадрат выбранного радиуса – высоту холма. Уравнение «холма» выглядит так:

Здесь (xc, yc) - заданная точка (центр холма), R - выбранный радиус, z – высота точки, принадлежащая холму.

Чтобы сгенерировать ландшафт полностью, нужно построить множество таких холмов. Но есть ещё две вещи на которые необходимо обратить внимание:

1. Целесообразно игнорировать отрицательные значения высоты холма.
2. При генерации последующих холмов лучше добавлять полученное значение для данного холма к уже существующим значениям. Это позволяет построить более правдоподобный ландшафт, нежели правильно очерченные округлые холмы.

Полученный ландшафт будет иметь слишком крутые горы и слишком мало равнин, поэтому нужен этап долинизации. Долинизация – процесс возведения в квадрат каждого значения высоты. Это эффективно снизит значения высоты среднего диапазона, не слишком сильно влияя на максимумы. Таким образом, мы сгладим местность, не сглаживая горы.

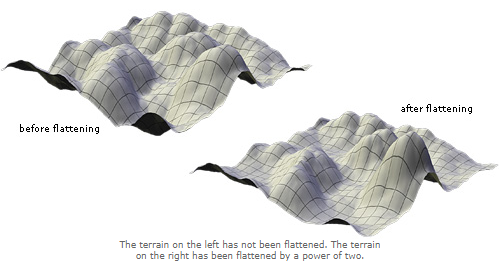
 На следующем изображении показано, как долинизация(выравнивание) влияет на сгенерированный ландшафт:

Рис. 1.4. Разница между ландшафтом до долинизации(слева) и после(справа)

Плюсы:

1. Есть возможность контролировать «гористость» ландшафта;
2. Легко модифицируется под другие типы ландшафта (одинокий остров, озёрные ландшафты и т.д.).

Минусы:

1. Является самым медленным из всех представленных алгоритмов: многократное обращение к точкам и изменение их высоты;
2. В некоторых случаях требует слишком много корректировок для того, чтобы добиться правдоподобности ландшафта.

**Итог**

Проанализировав несколько алгоритмов генерации ландшафта, можно привести краткую характеристику в виде таблицы:

|  |  |  |
| --- | --- | --- |
| Название | Скорость | Качество ландшафта |
| Diamond-Square | Высокая | Среднее |
| Шум Перлина | Средняя | Высокое |
| Холмовой алгоритм | Низкая | Среднее |

Таблица 1. Сравнение алгоритмов генерации

В данной таблице не было рассмотрено потребление памяти алгоритмами, так как данная характеристика для решения задачи не важна: результат сохраняется в файле-изображении, поэтому памяти везде используется одинаковое количество.

Главным критерием алгоритма является качество ландшафта, так как для решения задачи важно создать правдоподобный рельеф.

Из всех представленных самым подходящим алгоритмом для решения задачи является шум Перлина: он может обеспечить высокое качество ландшафта засчёт механизма комбинаций разных октав шумов. При этом является не самым медленным из представленных, что тоже важно.

* 1. **Анализ алгоритмов удаления невидимых линий**