ВЕСТНИК ЮГОРСКОГО ГОСУДАРСТВЕННОГО УНИВЕРСИТЕТА
2021 г. Выпуск 4 (63). С. 84–94

DOI: 10.17816/byusu20210484-94
УДК 004.45

Е. И. Сафонов, О. И. Соколков

РАЗРАБОТКА ПРОГРАММНОЙ СРЕДЫ ДЛЯ ГЕНЕРАЦИИ ЛАНДШАФТА ПО КАРТЕ БИОМОВ

В статье приводится описание процесса проектирования и создания программной среды, позволяющей в автоматизированном режиме создавать реалистичный ландшафт. Проводится обзор существующих подходов к генерации ландшафта, которые обладают набором недостатков, принятых во внимание при разработке программной среды. Описана диаграмма компонентов и основных классов. Разработанная подпрограмма реализующая алгоритм генерации сетки многоугольников предоставляет интерфейс создания и редактирования сетки шестиугольников на плоскости для упрощенной работы с биомами, а также детализации границ многоугольников для придания ландшафту элементов случайности и, в следствии, реалистичности. В процессе используется алгоритм генерации шумов Diamond Square. Алгоритм состыковки предназначен для уменьшения разрывов между высотами разных биомов. Алгоритм эрозии использует частицы, генерируемые на карте высот, переносящие частицы почвы в соответствии с физическими законами. Представлен пользовательский интерфейс работы приложения и результаты работы алгоритмов.

Ключевые слова: генерация ландшафта, карта высот, биомы, эрозия, программирование

E. I. Safonov, O. I. Sokolkov

DEVELOPMENT OF A VIRTUAL REALITY SIMULATOR REPLACING THE CHECK VALVE ON CHRISTMAS TREE

The article describes the process of designing and creating a software environment that allows in automatic mode to create a realistic landscape. A review of existing approaches to landscape generation is carried out, which have a set of disadvantages taken into account when developing a software environment. A diagram of components and main classes is described. The developed subroutine that implements the polygon mesh generation algorithm provides an interface for creating and editing a mesh of hexagons on a plane, used for simplified work with biomes, as well as detailing the boundaries of polygons to give the landscape elements of randomness and, as a result, realism. The process uses the Diamond Square noise generation algorithm. The docking algorithm is designed to reduce the gaps between the heights of different biomes. The erosion algorithm uses particles generated on a height map to carry soil particles in accordance with physical laws. The user interface of the application and the results of the algorithms are presented.

Keywords: terrain generation, height map, biomes, erosion, programming

Введение

Цифровое изобразительное искусство имеет большой спрос: в киноиндустрии, рекламе, компьютерных играх, при создании чертежей, карт и так далее. Речь может идти как о трехмерной, так и о двухмерной растровой или векторной графике.

Существует частная задача: моделирование рельефа и ландшафта. Построение природы в цифровом виде особенно актуально для проектов с открытым миром, там, как правило, требуется создавать большие природные пространства.

Рассмотрим подходы моделирования ландшафтов.

1. Вручную, в графическом редакторе можно, изображается карта высот – черно-белое цифровое изображение, белый цвет на котором – самая высокая точка местности, черный – самая низкая. Посредством мыши можно нанести в нужные области светлый или темный цвет. В целом, черно-белое изображение дает человеку примерное представление о рельефе местности. Основной минус в том, что карта высот сама по себе недостаточно наглядна, это лишь изображение, которое абстрагировано от всего остального.

2. Вручную в графической среде разработки, которая предоставляет наглядный интерфейс с возможностью редактирования, изменение уровня ландшафта, сглаживания и т.д., то есть непосредственно с картой высот мы не работаем.

Но даже в такой среде не исключаются трудности:

– водоёмы. Вручную несложно отразить несколько простых речных изгибов, круглых озер, но, если мы нацеливаемся на большую площадь и реалистичность, то человеку приходится имитировать случайность и при этом полагаться на интуицию, при этом постоянно следить за тем, чтобы определенные фрагменты ландшафта повторялись как можно меньше. Для того чтобы ландшафт выглядел удовлетворительно, приходится вносить неоднократное множество изменений, отнимает большое количество времени и сил;
– горы и холмы. В добавок к проблемам присущим моделированию водоёмов, поверхность гор и холмов подвергается выветриванию и эрозии. Также свои особенности вносит время года, материал поверхности, высота и т.д. Данные явления привносят истинную красоту ландшафту, и большие проблемы человеку, которые собирается их рисовать вручную.

3. Автоматизировано, в специализированных графических пакетах. Подавляющее большинство таких систем генерирует ландшафт используя принцип функциональных блоков, посредством которых пользователь последовательно задает поток данных и преобразований, определяющие выходное изображение. За счет подстановок и перестановок таких блоков достигается гибкость функционала.

Несмотря на преимущество описанного подхода, генерация ландшафта происходит случайным образом, а если в одном месте обязательно должна находиться равнина/гора/овраг, то не стоит ожидать того, что даже с двадцатой попытки все сгенерируется необходимым образом.

4. Биомный подход. Речь идет об интерфейсе, где можно было бы определить области (далее биомы), границы между ними и собственные параметры генерации (рис. 1).

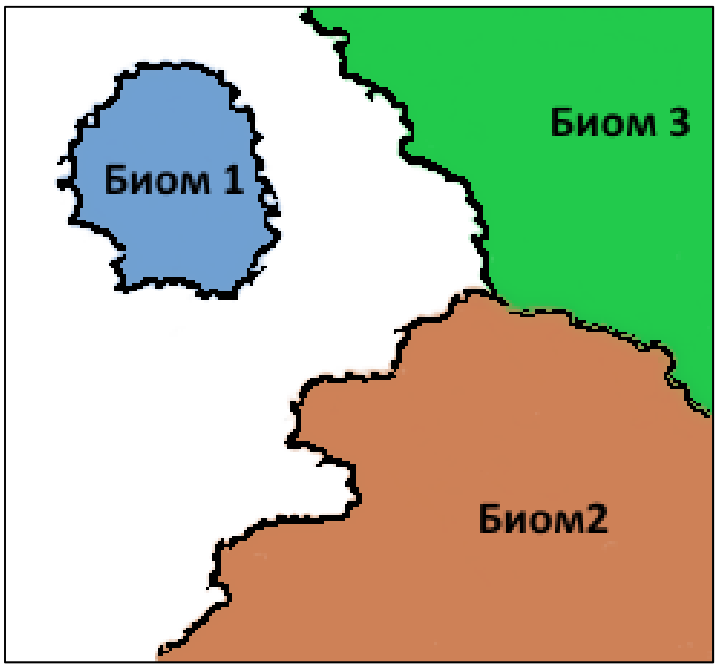

Рисунок 1. Карта биомов

В целом, такой подход, можно наблюдать в компьютерных играх. Данный подход имеет распространение, например, можно рассмотреть статьи [1] и [2]. В них описываются генерация полигонов, на их основе которых строятся регионы, реки и так далее. Но в описанных статьях не предполагается явное задание биомов: их количество, свойства не определяются напрямую пользователем, то есть, в них результат определяется фактором случайности, что усложняет получение необходимого результата.

Подход, описываемый в данной работе нацелен на уменьшение фактора случайности. При нем пользователь будет иметь четкое, ясное представление о том, что он получит в итоге.

Проектирование компонентов

До разработки трехмерного интерфейса редактирования сетки шестиугольников было принято решение разработать простое и небольшое ПО, которое позволит создавать файлы с данными сетки описанных в подпрограмме «HexGridEditor». Данное решение разрешит проблему независимости разработки в виду того, что на трехмерный интерфейс нужно больше времени, больше вероятность багов и прочих проблем. Имея простой двухмерный интерфейс можно быстро и удобно отработать, протестировать логическое поведение сетки, создать множество входных файлов для тестов.

Были спроектированы компоненты и связи между ними (рис. 2).

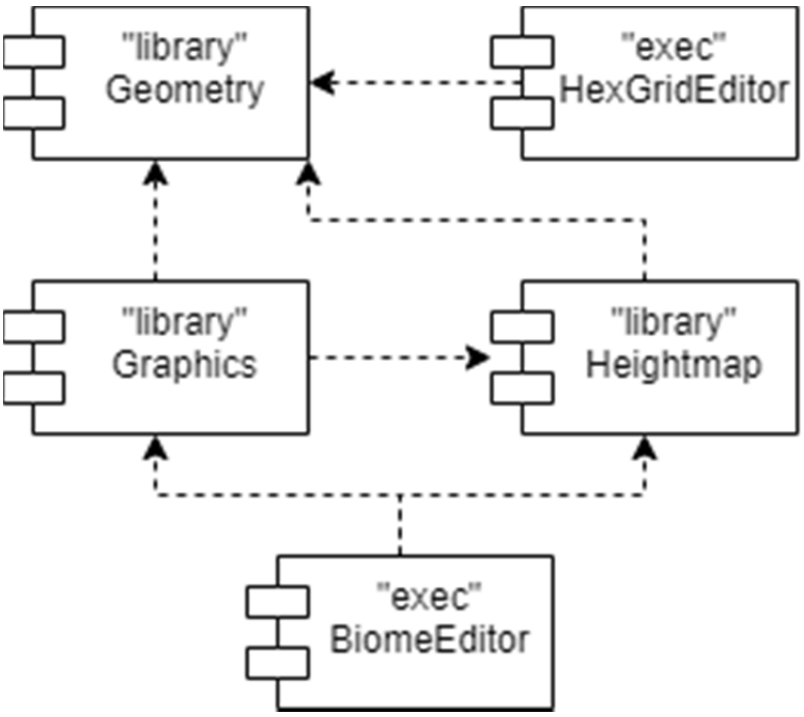

Рисунок 2. Диаграмма компонентов

Компонент «Geometry» будет содержать в себе функционал шестиугольной сетки, полигональной сетки, детализации отрезков и все что тематически связанно с этим.

Компонент «Heightmap» будет хранить классы генерации, состыковки и эрозии карты высот.

Компонент «Graphics» содержит графический функционал трехмерной графики, модуль отображения ландшафта и шестиугольной сетки.

Программа «BiomeEditor» содержит пользовательский интерфейс для взаимодействия с трехмерными элементами для генерации ландшафта.

Компонент «Geometry»

Диаграмма классов компонента «Geometry» представлена на рисунке 3.

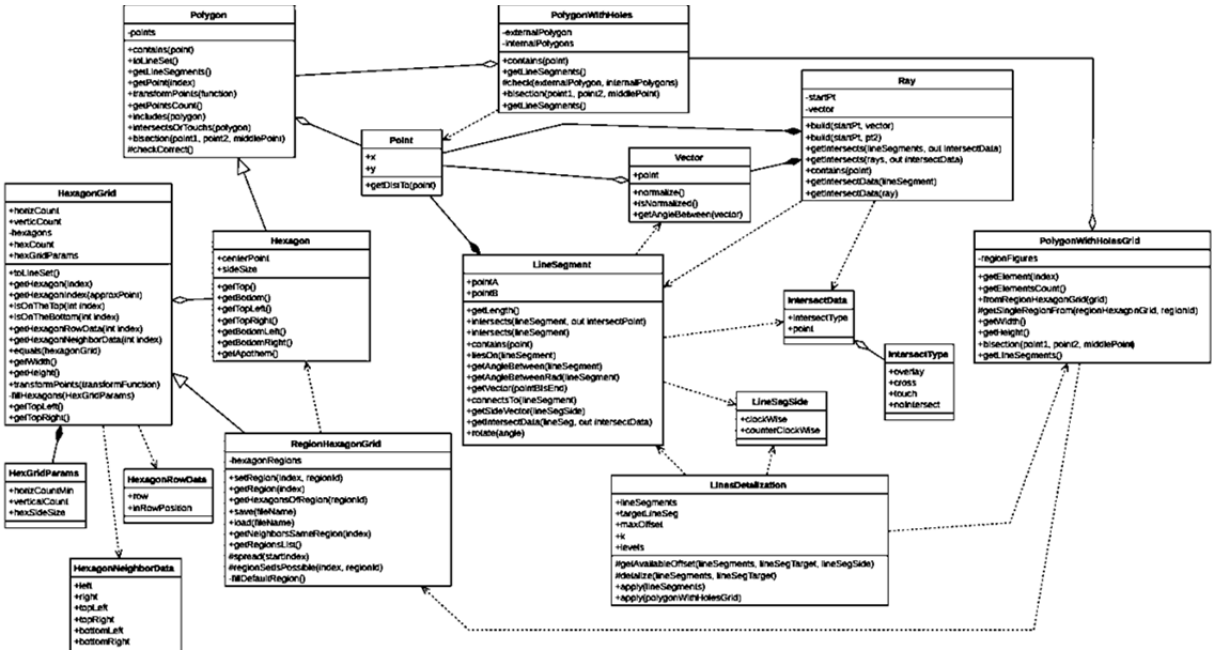

Рисунок 3. Диаграмма классов компонента «Geometry»

Опишем классы компонента «Geometry».

Класс «HexagonGrid» реализует сетку шестиугольников, методы получения ширины, высоты, преобразования точек.

Класс «HexGridParams» содержит параметры отображаемой сетки, количество по горизонтали, вертикали и размер стороны шестиугольника.

Класс «HexagonNeighborData» содержит индексы соседних шестиугольников для каждого шестиугольника.

Класс «RegionHexagonGrid» наследуется от «HexagonGrid» добавляет функционал разметки пространства. За счет метода setRegion(index, regionId) мы указываем, что шестиугольник с данным индексом теперь будет относиться к биому с данным id.

Класс «Hexagon» является интерпретацией шестиугольника, в нем реализованы методы для получения его точек, что может оказаться полезным в случае, если мы будем работать непосредственно с точками.

Класс «Polygon» интерпретирует полигон – набор точек и является родительским классом для «Hexagon».

Класс «PolygonWithHolesGrid» описывает сетку полигонов, основанную на сетке биомов класса «RegionHexagonGrid».

Класс «LineSegment» определяет отрезок соединяющий две точки, определяет наличие и угол пересечения с точками и отрезками.

Класс «LinesDetalization» предназначен не напрямую для сетки полигонов, а для набора отрезков. Данное решение делает этот класс более универсальным: можно детализировать не только сетку, но и любой произвольный набор отрезков. Для детализации сетки необходимо получить набор её отрезков. Это было реализовано посредством метода «getLineSegments».

Класс «PolygonWithHoles» определяет вложенность полигонов на плоскости.

Класс «Point» описывает точку в пространстве.

Класс «Vector» описывает нормаль, проведенную из точки.

Класс «Ray» описывает вектор, проведенный из точки по направлению экземпляра класса «Vector».

Компонент «Heightmap»

Диаграмма классов компонента «Heightmap» представлена на рисунке 4.

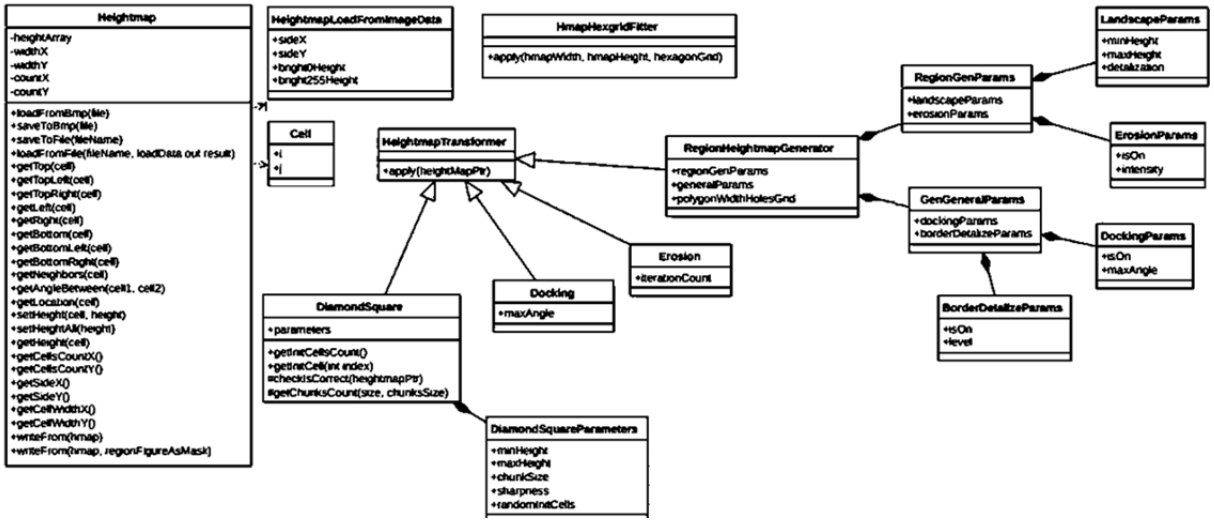

Рисунок 4. Диаграмма классы компонента «Heightmap»

При описании компонента «Heightmap» опишем только часть классов.

Класс «Heightmap» в основном служит для получения и записи высоты в нужную ячейку, методы «getHeight» и «setHeight» соответственно, а также получение размера массива карты высот, методы «getCellsCountX» и «getCellsCountY».

Методы «saveToBmp» и «loadFromBmp» необходимы для сохранения и загрузка карты в виде изображения. В данном случае – происходит сохранение в файл набора вещественных чисел, но, если нам нужен неограниченный набор значений высот, требуется сохранять данные по-другому.

Определен ввод и получение физических размеров карты, методы «getSideX», «getSideY», функционал которых может понадобиться для некоторых алгоритмов, использующих законы физики и основанных на размерах, скоростях и расстояниях между объектами.

Так же подготовлены вспомогательные методы для удобного написания кода: получение соседних ячеек относительно выбранной, методы «getTop», «getTopLeft», «getBottom», «getNeighbors» и другие.

Из-за того, что класс генерации, состыковки и эрозии имеют общее свойство – они изменяют входную карту высот, было принято решение сделать для них общий родительский класс «HeightmapTransformer».

При генерации разных участков по разным параметрам, на границах в большинстве случаев будут резкие перепады высот. Класс «Docking» определен для решения данной проблемы. На вход будем подавать максимальный угол, т.е., например, если подадим 45 градусов, не будет ни одной местности на карте, где угол между соседними точками будет больше 45.

Класс «RegionHeightmapGenerator» предназначен для объединения функционала вышеописанного: то есть, программисту не придется явно взаимодействовать с классами эрозии, генерации и состыковки.

Набор классов «RegionGenParams», «GenGeneralParams», «LandscapeParams», «ErosionParams», «DockingGenParams», «BorderDetalizeParams» хранят параметры для генерации ландшафта.

Реализация алгоритма генерации ландшафта

Были определены этапы реализации алгоритма:

1. Сгенерировать сетки шестиугольников и многоугольников.
2. Повысить реалистичность путем генерации шума для каждого биома с индивидуальными параметрами минимальной и максимальной высоты.
3. Сгладить все резкие перепады на карте, так как нет никакой гарантии, что после генерации шума с разными параметрами на границах биомов не будет не состыковок.

4. Применить ко всей карте алгоритм эрозии для повышения уровня детализации, придания реалистичности.

5. Просмотр и редактирование сетки шестиугольников в трехмерном пространстве, установка параметров высоты для каждого биома, запуск генерации и просмотр результата.

Алгоритм генерации сеток многоугольников

Была разработана небольшая подпрограмма, предоставляющая интерфейс создания и редактирования сетки шестиугольников на плоскости. В ней мы можем выделить биомы, сохранить результат в виде файла, а затем загрузить с целью изменения.

В полученной карте биомов на шестиугольной сетке предстоит совершить детализацию границ, то есть, увеличить число входящих в них отрезков, чтобы добиться большей реалистичности. Опишем общий принцип данной операции:

1. Определяются отрезки, которые есть в сетке.
2. Каждый отрезок разбивается на два.
3. Их середина перемещается перпендикулярно изначальному отрезку случайным образом в одну из двух сторон.
4. При всем вышеописанном отслеживается, чтобы новые отрезки не пересекались с остальными в сетке: т.е. рассчитывается максимальное расстояние, на которое можно передвинуть среднюю точку.
5. При необходимости повторить пункты 1–4.

Последовательность операций представлена на рисунке 5.

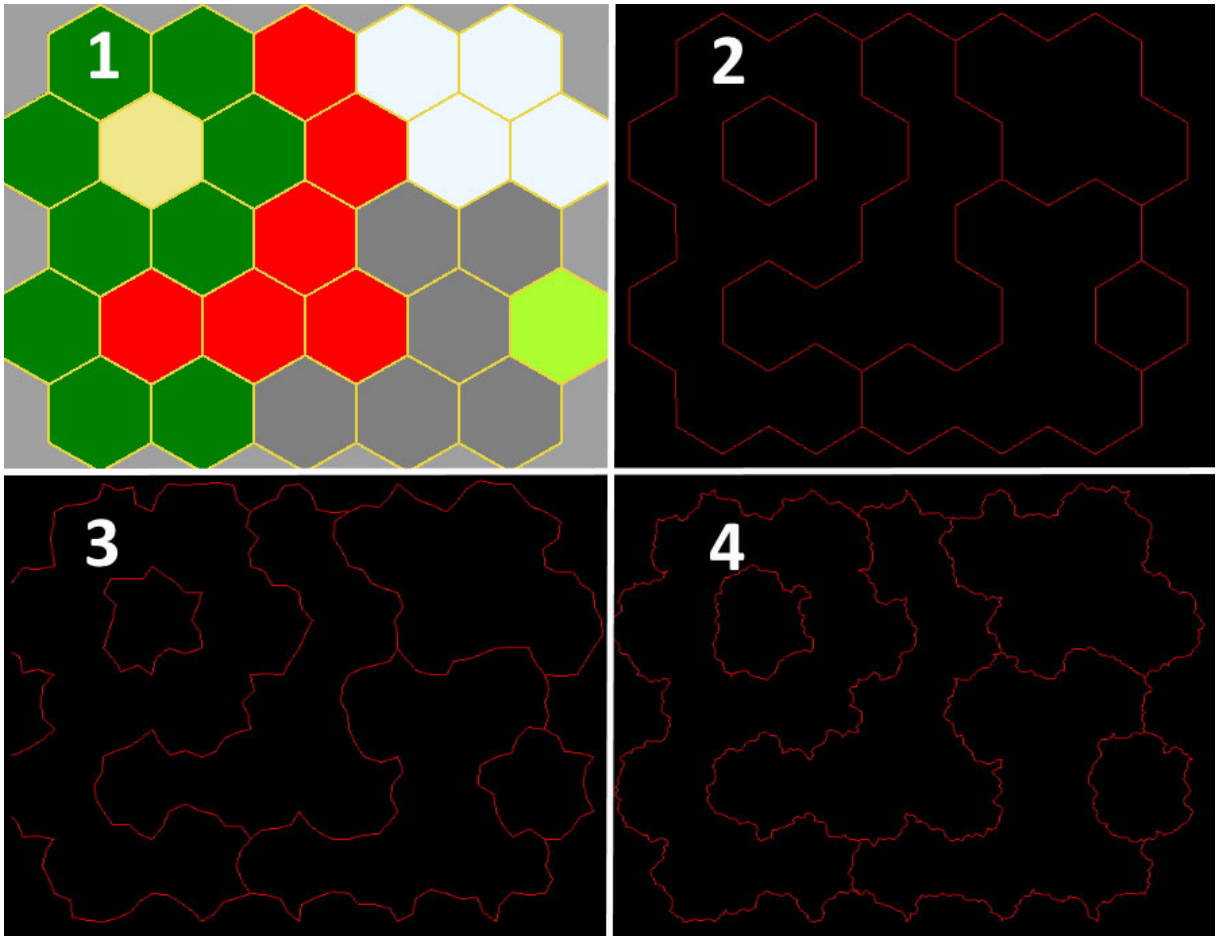

Рисунок 5. Сетка шестиугольников (1), получение полигонов из сетки (2), детализация границ (3, 4)

Алгоритм генерации шума

Далее необходимо добавить алгоритм Diamond Square, пример работы алгоритма представлен на рисунке 6.

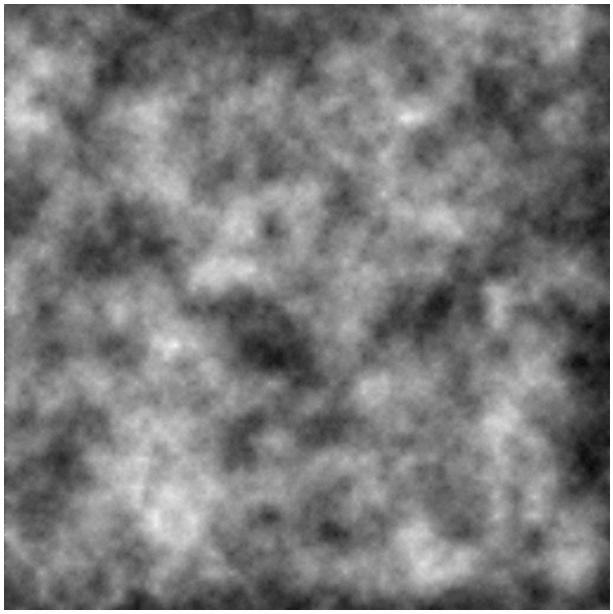

Рисунок 6. Сгенерированная карта высот алгоритмом «Diamond Square»

Данный алгоритм необходим для генерации карты высот для каждого биома в соответствии с их параметрами: минимальная и максимальная высота, то есть один биом в среднем располагается пониже, другой – повыше. Полученный результат применения алгоритма показан на рисунке 7.

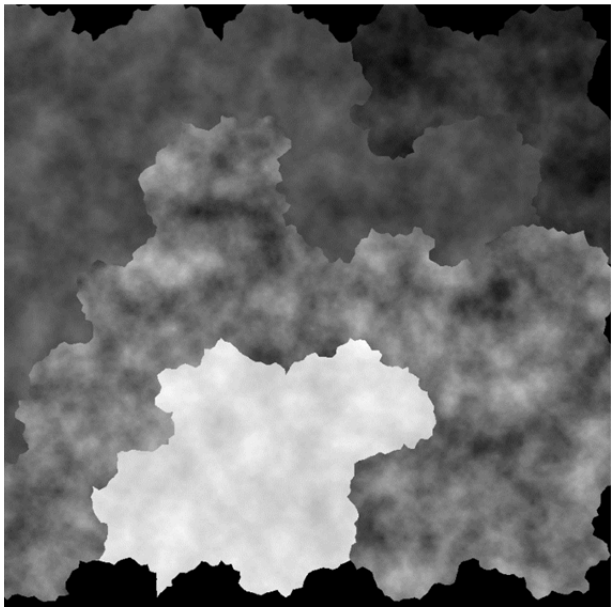

Рисунок 7. Карта высот, сгенерированная по разным параметрам

Как видно на границах существуют разрывы, их надо сгладить или удалить. Чтобы решить данную задачу, был разработан алгоритм состыковки ландшафта.

Алгоритм состыковки

Общий принцип заключается в следующем: одни ячейки изменяют высоту других ячеек в случае, если разница между ними слишком велика. Затем управление переходит изменен-

ным ячейкам. Они, в свою очередь, изменяют высоту своих соседей (так же, в случае максимальной разницы, выше установленной). Процедура повторяется до тех пор, пока не останется ячеек, которые могли мы изменить высоту других.

Разработано два варианта алгоритма:

- понижение: когда более низкие ячейки понижают высоту более высоких в случае большой разницы;
- повышение: наоборот, более высокие повышают более низкие ячейки.

Понижающий вариант оказался более предпочтительным для состыковки воды и остального ландшафта. Так, если вода будет на самом нижнем уровне высоты, в случае понижения ландшафта появится плавный переход между реками и землей: речное пространство не будет затронуто. Если же включить алгоритм на повышение: окружающие, более высокие ячейки просто поднимут все то место, которое должно было быть под водой.

Далее представлены результаты работы алгоритма в обработке сформированного ландшафта, рисунок 8.

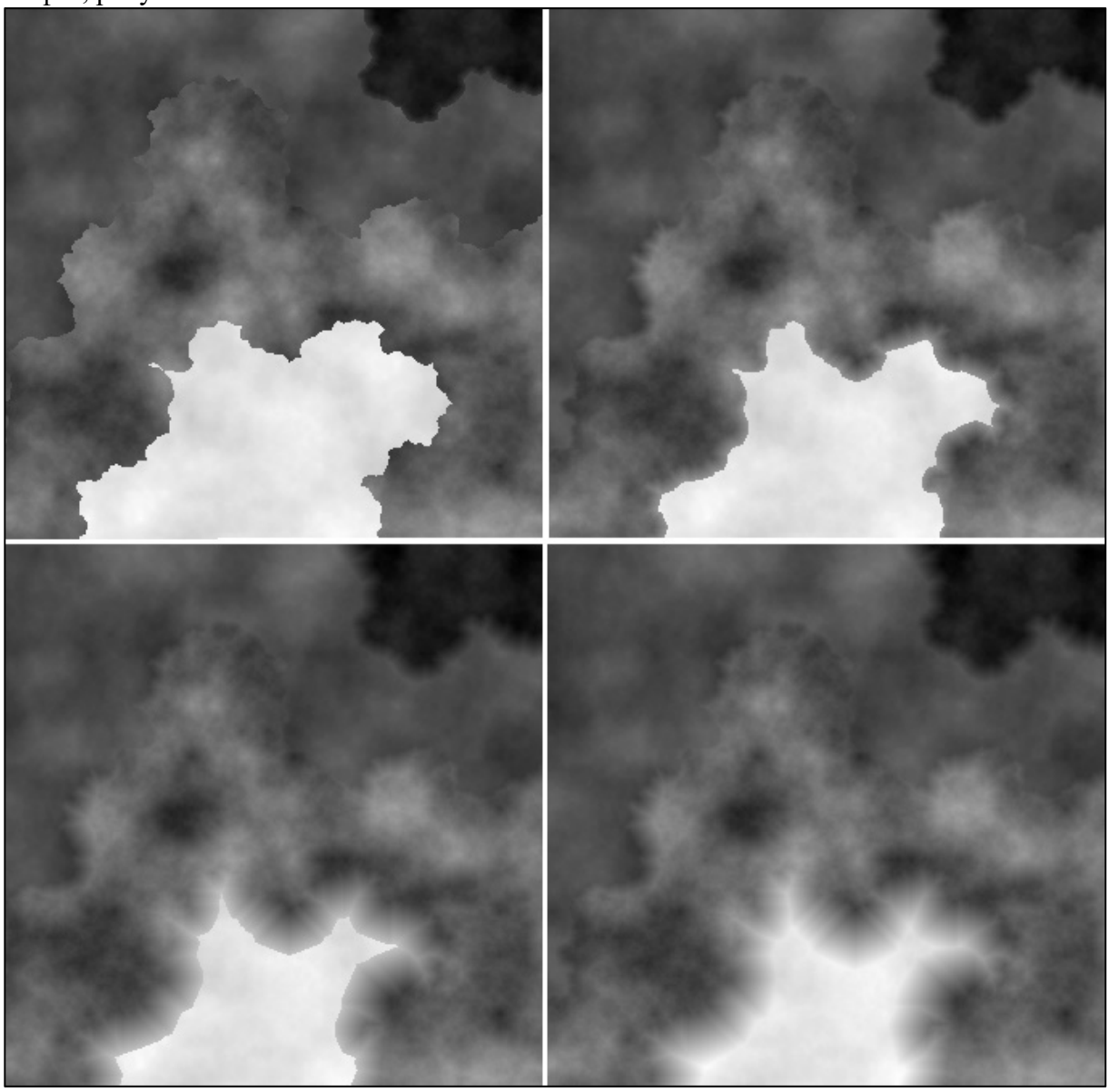

Рисунок 8. Последовательная работа алгоритма состыковки

Как можем видеть ландшафт постепенно шлифуется, начиная от самых нижних областей до самых высоких.

Изначально в алгоритме задается единый максимальный угол состыковки, то есть, каждая ячейка «руководствуется» одним значением максимального угла. В рамках улучшения работы алгоритма можно задать разные значения максимального угла. Это поднимет уровень реалистичности итоговой картины после стыковки.

Но ландшафт может потерять свой облик в случае большого значения угла. Если не внести дополнительные изменения в алгоритм, нужную стыковку не провести без повреждения облика всего ландшафта. Для этого на границах областей можно усилить влияние стыковки (то есть, уменьшить максимальный угол), а в других местах – уменьшить. Следует отметить, что данный подход можно объединить с предыдущим, то есть максимальные углы могут быть случайными, но в среднем ниже – на границах, выше – в остальных местах.

Алгоритм эрозии

Далее в проект был внедрен алгоритм эрозии, представленный в статье [3]. Алгоритм не был подвернут изменениям. Результаты можно наблюдать на рисунке 9.

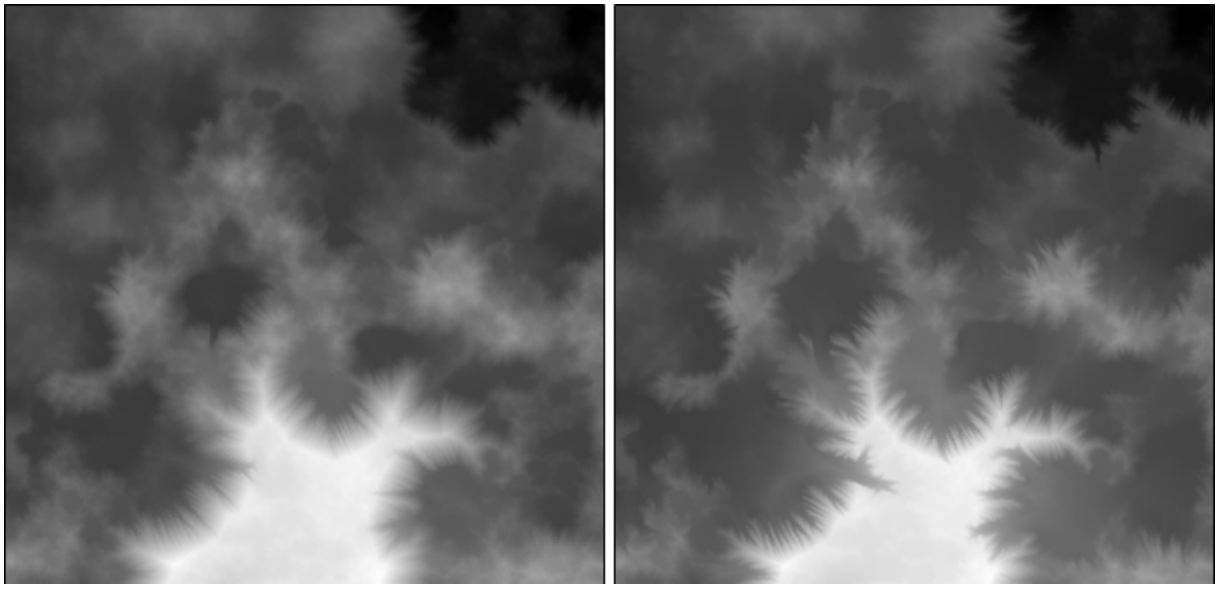

Рисунок 9. Работа алгоритма эрозии (средняя эрозия – слева, сильная – справа)

Можем наблюдать, как карта приобрела более реалистичный вид.

Принцип работы алгоритма аналогичен частицам воды, которые переносят в себе частицы земли и горных пород:

1. На карте высот случайным образом генерируются частицы.
2. Частицы перемещаются по физическим законам по карте, «забирая» с собой часть высоты и переносят ее в другое место.
3. В итоге частицы «испаряются» со временем.

Пользовательский интерфейс

Был реализован оконный пользовательский интерфейс для нанесения биомов на сетку шестиугольников, задания настроек и определения к какому биому принадлежит закрашенный шестиугольник.

В поле «Параметры биомов» выбирается порядковый номер биома, после чего в нижней части окна отобразятся поля для ввода минимальной и максимальной высоты для данного биома. Далее в процессе перемещения в трехмерном пространстве, наводя курсор на нужные шестиугольники, пользователем создается разметка карты, рисунок 10.

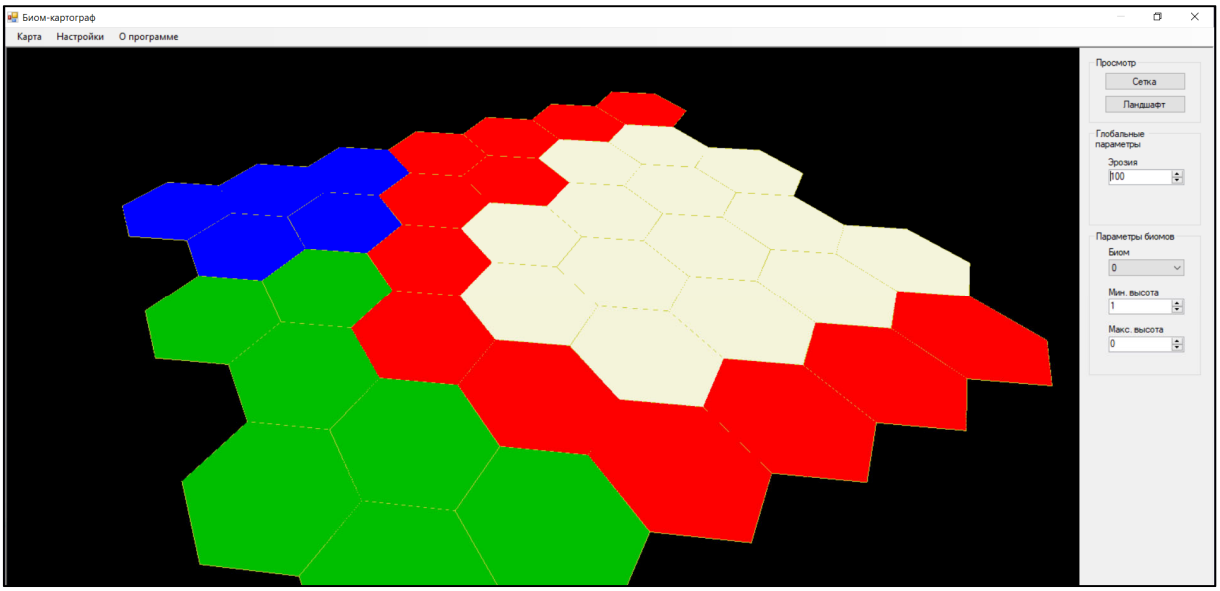

Рисунок 10. Нанесение биомов на шестиугольную сетку

В описываемом примере для «белого» биома указаны максимальная высота выше, для «синего» – ниже. У «зеленого» были указаны средние значения минимальной и максимальной высоты.

Далее, для запуска генерации следует перейти в меню «Карта» и нажать на соответствующую кнопку. После ожидания, в случае успешного завершения работы автоматически откроется окно просмотра ландшафта, рисунок 11.

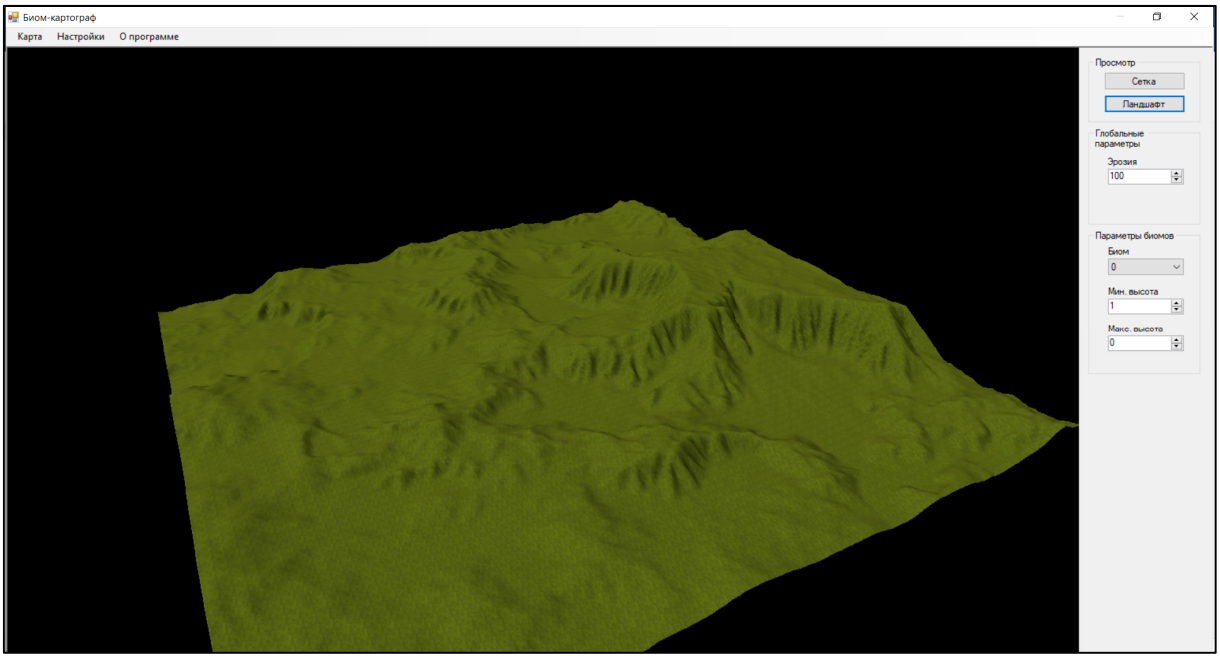

Рисунок 11. Сгенерированный ландшафт

Можем наблюдать сгенерированную карту высот по разметке биомов. В случае, если результат генерации пользователя не устроил, он может перейти обратно к редактированию сетки нажав на кнопку «Сетка» в верхнем правом углу экрана и повторить описанный раннее процесс.

Заключение

Была разработана программная среда для автоматизированной генерации ландшафтов по карте биомов. На итоговом ландшафте на сетке полигонов после детализации можно наблюдать острые углы. На основании углов между соседними отрезками можно рассчитать, куда следует перемещать среднюю точку, чтобы не появлялись острые углы, либо способствовать уменьшению этой остроты.

Алгоритм эрозии требует доработки, в виду того, что все ячейки на карте изначально обладают одинаковой «твердостью» и, следовательно, размываются одинаково. Так как в природе есть камни, почва и песок, то можно сделать так, чтобы одни области размывались сложнее, другие – проще.

Литература

1. AutoBiomes: procedural generation of multi-biome landscapes – Текст : электронный / R. Fisher, Ph. Dittman, R. Wellwr, G. Zachmann // The visual Compuret. – 2020. – Vol. 36. – URL: https://link.springer.com/article/10.1007/s00371-020-01920-7 (date of application : 26.08.2021).

2. Алгоритм «diamond–square» для построения фрактальных ландшафтов. – Текст : электронный // Хабр. – URL: https://habr.com/ru/post/111538/ (дата обращения: 26.08.2021).

3. Симуляция эрозии рельефа на основе частиц. – Текст : электронный // Хабр. – URL: https://habr.com/ru/post/496762/ (дата обращения: 26.08.2021).