Теория фракталов

Кащенко И.С.

Ярославский государственный университет
Математический факультет
Кафедра математического моделирования

Фрактальные алгоритмы генерации ландшафтов

Задача

Цель:

Получить реалистичный ландшафт (поверхность)

Части алгоритма

1. Случайная часть
2. Детерминированная часть (диаграммы Вороного, фрактальные идеи)

Задача

Цель:

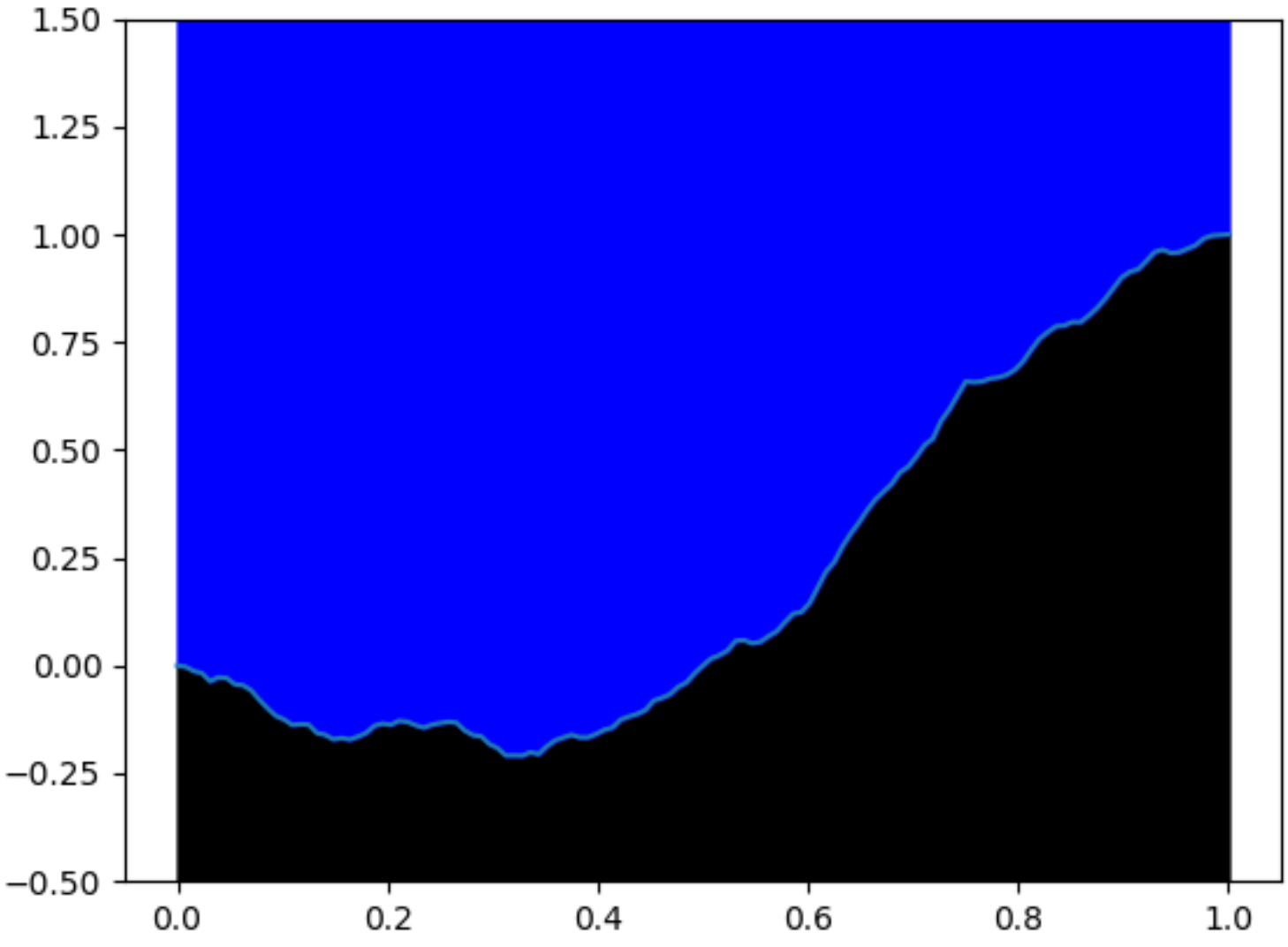
Получить реалистичный ландшафт (поверхность)

Части алгоритма

1. Случайная часть
2. Детерминированная часть (диаграммы Вороного, фрактальные идеи)

Содержание

Алгоритм „Midpoint displacement“



Построение линии горизонта

Цель

Сгенерировать реалистичную линию горизонта (линию высот)

Midpoint displacement

Идея: последовательное (рекурсивное) перемещение средней точки.

Начальные данные

Надо знать горизонтальные размеры рисунка и значения высоты в крайних точках.
При необходимости создания специфической линии (гора, яма и т.п.) можно задать также значения высот и в других точках.

Построение линии горизонта

Цель

Сгенерировать реалистичную линию горизонта (линию высот)

Midpoint displacement

Идея: последовательное (рекурсивное) перемещение средней точки.

Начальные данные

Надо знать горизонтальные размеры рисунка и значения высоты в крайних точках.
При необходимости создания специфической линии (гора, яма и т.п.) можно задать также значения высот и в других точках.

Построение линии горизонта

Цель

Сгенерировать реалистичную линию горизонта (линию высот)

Midpoint displacement

Идея: последовательное (рекурсивное) перемещение средней точки.

Начальные данные

Надо знать горизонтальные размеры рисунка и значения высоты в крайних точках.

При необходимости создания специфической линии (гора, яма и т.п.) можно задать также значения высот и в других точках.

Построение линии горизонта

Цель

Сгенерировать реалистичную линию горизонта (линию высот)

Midpoint displacement

Идея: последовательное (рекурсивное) перемещение средней точки.

Начальные данные

Надо знать горизонтальные размеры рисунка и значения высоты в крайних точках.
При необходимости создания специфической линии (гора, яма и т.п.) можно задать также значения высот и в других точках.

Шаг алгоритма

На входе:

Отрезок (x_L, y_L)–(x_R, y_R)

Шаг:

Среднюю точку отрезка переставляем вертикально в положение

$$y_c = \frac{y_l + y_R}{2} + R \cdot a \cdot Random(-1, 1)$$

a – длина отрезка (длина проекции $|x_R - x_L|$)
R – *roughness* – *шероховатость*

Этот шаг проделываем с каждым отрезком (в том числе с вновь построенными)

Окончание работы

Все отрезки имеют длину меньше заданного порога (обычно 1px)

Шаг алгоритма

На входе:

Отрезок (x_L, y_L)–(x_R, y_R)

Шаг:

Среднюю точку отрезка переставляем вертикально в положение

$$y_c = \frac{y_l + y_R}{2} + R \cdot a \cdot Random(-1, 1)$$

a – длина отрезка (длина проекции $|x_R - x_L|$)
R – *roughness* – *шероховатость*

Этот шаг проделываем с каждым отрезком (в том числе с вновь построенными)

Окончание работы

Все отрезки имеют длину меньше заданного порога (обычно 1px)

Шаг алгоритма

На входе:

Отрезок (x_L, y_L)–(x_R, y_R)

Шаг:

Среднюю точку отрезка переставляем вертикально в положение

$$y_c = \frac{y_l + y_R}{2} + R \cdot a \cdot Random(-1, 1)$$

a – длина отрезка (длина проекции $|x_R - x_L|$)

R – roughness – шероховатость

Этот шаг проделываем с каждым отрезком (в том числе с вновь построенными)

Окончание работы

Все отрезки имеют длину меньше заданного порога (обычно 1px)

Шаг алгоритма

На входе:

Отрезок (x_L, y_L)–(x_R, y_R)

Шаг:

Среднюю точку отрезка переставляем вертикально в положение

$$y_c = \frac{y_l + y_R}{2} + R \cdot a \cdot Random(-1, 1)$$

a – длина отрезка (длина проекции $|x_R - x_L|$)
R – *roughness* – *шероховатость*

Этот шаг проделываем с каждым отрезком (в том числе с вновь построенными)

Окончание работы

Все отрезки имеют длину меньше заданного порога (обычно 1px)

Шаг алгоритма

На входе:

Отрезок (x_L, y_L)–(x_R, y_R)

Шаг:

Среднюю точку отрезка переставляем вертикально в положение

$$y_c = \frac{y_l + y_R}{2} + R \cdot a \cdot Random(-1, 1)$$

a – длина отрезка (длина проекции $|x_R - x_L|$)
R – *roughness* – *шероховатость*

Этот шаг проделываем с каждым отрезком (в том числе с вновь построенными)

Окончание работы

Все отрезки имеют длину меньше заданного порога (обычно 1px)

Шаг алгоритма

На входе:

Отрезок (x_L, y_L)–(x_R, y_R)

Шаг:

Среднюю точку отрезка переставляем вертикально в положение

$$y_c = \frac{y_l + y_R}{2} + R \cdot a \cdot Random(-1, 1)$$

a – длина отрезка (длина проекции $|x_R - x_L|$)
R – *roughness* – *шероховатость*

Этот шаг проделываем с каждым отрезком (в том числе с вновь построенными)

Окончание работы

Все отрезки имеют длину меньше заданного порога (обычно 1px)

Midpoint displacement

Постобработка

1. Ко всем высотам можно прибавить константу (и сделать их положительными)
2. Ко всем высотам можно применить функцию $f(x)$. Например, $f(x) = x^2$.

Преимущества

1. Легко программируется, быстро работает.
2. Позволяет менять масштаб
3. Позволяет делать „скроллинг“ в сторону. Нужно только „придумать“ значение в крайней новой точке.

Midpoint displacement

Постобработка

1. Ко всем высотам можно прибавить константу (и сделать их положительными)
2. Ко всем высотам можно применить функцию $f(x)$.
Например, $f(x) = x^2$.

Преимущества

1. Легко программируется, быстро работает.
2. Позволяет менять масштаб
3. Позволяет делать „скроллинг“ в сторону. Нужно только „придумать“ значение в крайней новой точке.

Midpoint displacement

Постобработка

1. Ко всем высотам можно прибавить константу (и сделать их положительными)
2. Ко всем высотам можно применить функцию $f(x)$.
Например, $f(x) = x^2$.

Преимущества

1. Легко программируется, быстро работает.
2. Позволяет менять масштаб
3. Позволяет делать „скроллинг“ в сторону. Нужно только „придумать“ значение в крайней новой точке.

Midpoint displacement

Постобработка

1. Ко всем высотам можно прибавить константу (и сделать их положительными)
2. Ко всем высотам можно применить функцию $f(x)$.
Например, $f(x) = x^2$.

Преимущества

1. Легко программируется, быстро работает.
2. Позволяет менять масштаб
3. Позволяет делать „скроллинг“ в сторону. Нужно только „придумать“ значение в крайней новой точке.

Midpoint displacement

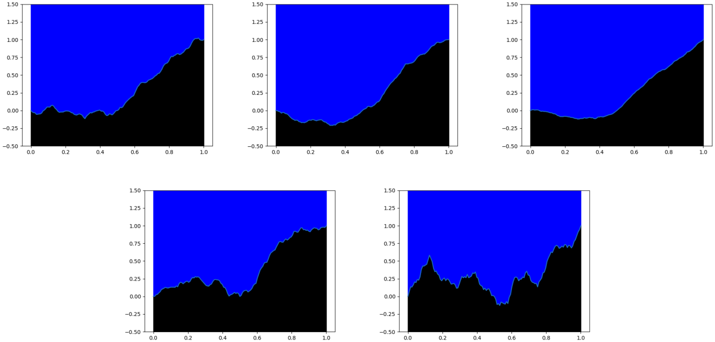
Постобработка

1. Ко всем высотам можно прибавить константу (и сделать их положительными)
2. Ко всем высотам можно применить функцию $f(x)$.
Например, $f(x) = x^2$.

Преимущества

1. Легко программируется, быстро работает.
2. Позволяет менять масштаб
3. Позволяет делать „скроллинг“ в сторону. Нужно только „придумать“ значение в крайней новой точке.

Примеры



Лабораторная работа

Реализуйте алгоритм Midpoint displacement.
Сделайте изменение начальных условий, параметра R.
Сделайте „бесконечный скроллинг“.

Содержание

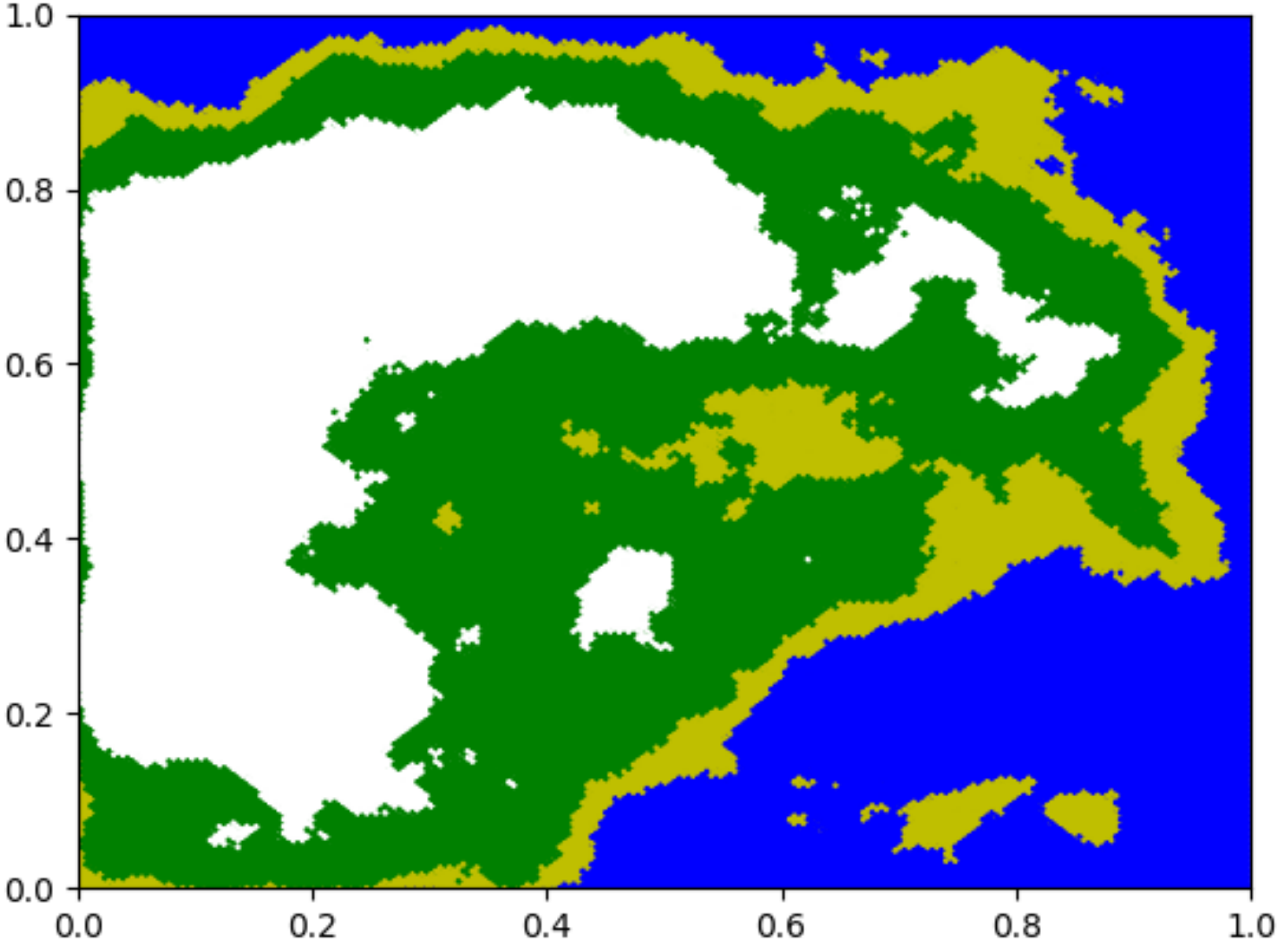


Рис.:

Алгоритм „Diamond-square“

Цель

Сгенерировать реалистичный ландшафт (карту высот)

Начальные данные

Квадрат со стороной 2^N

Значения высот в углах

Значения вне квадрата: нули, периодический закон, ...

Алгоритм „Diamond-square“

Цель

Сгенерировать реалистичный ландшафт (карту высот)

Начальные данные

Квадрат со стороной 2^N

Значения высот в углах

Значения вне квадрата: нули, периодический закон, ...

Алгоритм „Diamond-square“

Цель

Сгенерировать реалистичный ландшафт (карту высот)

Начальные данные

Квадрат со стороной 2^N

Значения высот в углах

Значения вне квадрата: нули, периодический закон, ...

Алгоритм „Diamond-square“

Цель

Сгенерировать реалистичный ландшафт (карту высот)

Начальные данные

Квадрат со стороной 2^N

Значения высот в углах

Значения вне квадрата: нули, периодический закон, ...

Diamond-square

Square

Поучаем значение высоты в центре квадрата – среднее арифметическое вершин сдвинутое на случайную величину:

$$h_c = \frac{h_1 + h_2 + h_3 + h_4}{4} + R \cdot a \cdot Random(-1, 1)$$

Diamond

Определяем значения высот на серединах сторон – используем четыре точки: две вершины, центр квадрата и симметричную центру относительно стороны (эта точка, возможно, лежит вне квадрата!)

Diamond-square

Square

Поучаем значение высоты в центре квадрата – среднее арифметическое вершин сдвинутое на случайную величину:

$$h_c = \frac{h_1 + h_2 + h_3 + h_4}{4} + R \cdot a \cdot Random(-1, 1)$$

Diamond

Определяем значения высот на серединах сторон – используем четыре точки: две вершины, центр квадрата и симметричную центру относительно стороны (эта точка, возможно, лежит вне квадрата!)

Diamond-square

Порядок работы

Обработка квадратов сначала ведется в ширину, только потом в глубину.

На k-м шаге сначала запускаем Square для всех квадратов размера 2^{n-k}, и только потом Diamond.

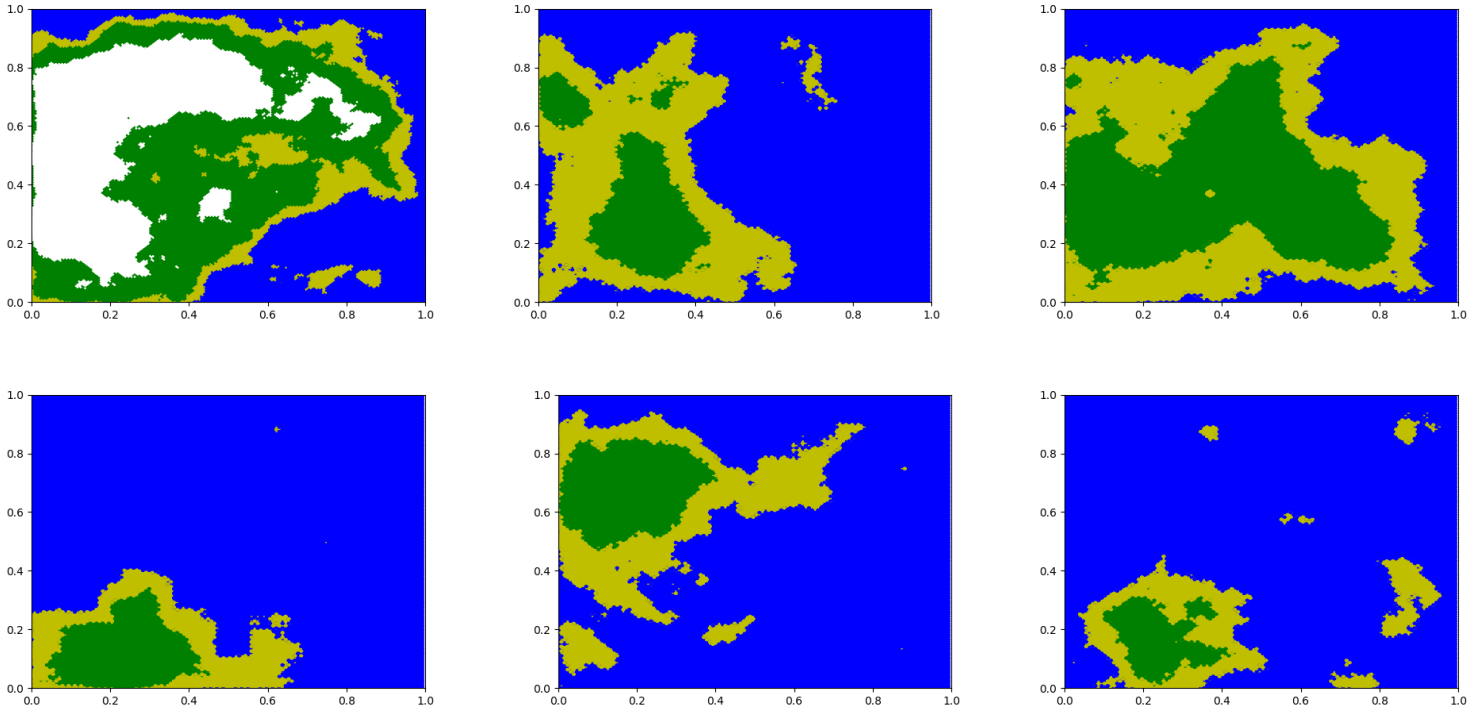
Diamond-square

Порядок работы

Обработка квадратов сначала ведется в ширину, только потом в глубину.

На k-м шаге сначала запускаем Square для всех квадратов размера 2^{n-k}, и только потом Diamond.

Примеры



Лабораторная работа

Реализуйте алгоритм Diamond-Square
Сделайте изменение начальных условий, параметра R, параметров отображения
Сделайте „бесконечный скроллинг“.