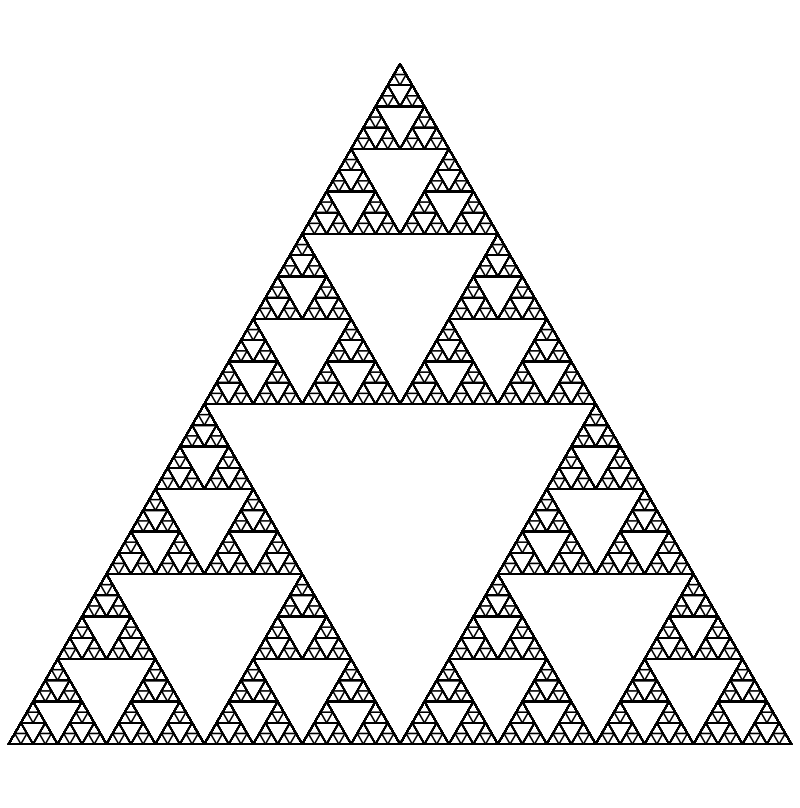
*Моделирование турбулентного течения жидкости, процессов диффузии-адсорбции, сжатие изображений, структуры пористой среды, описание систем внутренних органов, генерация ландшафтов, растительности, систем рек – всё это можно реализовать при помощи фракталов, чаще всего определяемых как геометрические фигуры, которые обладают свойством самоподобия, т.е. состоящих из частей, подобных всей фигуре в целом. В этой статье будет рассмотрено более точное определение фрактала, виды фракталов*

Как уже было сказано фрактал – это множество, обладающее свойством самоподобия. И это действительно так: например, треугольник Серпинского на рисунке 1 состоит из трёх меньших копий самого себя.



Однако под это же определение подойдёт также и любой квадрат, который можно разбить на четыре квадрата поменьше. Поэтому необходимо дать более точное определение: фрактал – это геометрическая фигура, имеющая дробную размерность.

К примеру, тот же треугольник Серпинского обладает размерность равной 1,585. Это означает то, что при увеличении длины его стороны в два раза площадь будет увеличиваться в 21.585, т.е. в 3 раза. Если обратиться к более привычным размерностям, то станет понятно что, если увеличить длину стороны квадрата в 2 раза, то его площадь изменится в 4 раза, а объём куба в 8 раз. Отсюда можно получить такие равенства: 22 = 4, 23 = 8. И как раз степень здесь является размерностью фигуры.

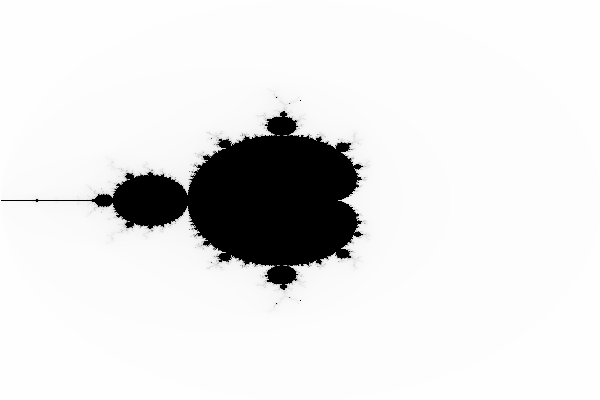
Применяя тот же способ к треугольнику Серпинского, можно сказать: если увеличить длину его стороны в два раза, то его площадь увеличится в 3 раза, т.к. три меньшие копии станут в два раза больше и, соответственно, станут соразмерными изначальному треугольнику. Получается что 2D = 3, отсюда . Также можно увидеть, что при количестве итераций его площадь стремится к бесконечности, т.к. становится всё больше пустых мест, а периметр стремится к бесконечности. Отсюда становится понятно, что этот фрактал недостаточно двумерный чтобы описать его натуральной площадью и слишком одномерный чтобы описать его одной линией.

Фракталы подразделяются на:

* Алгебраические;
* Геометрические;
* Стохастические.

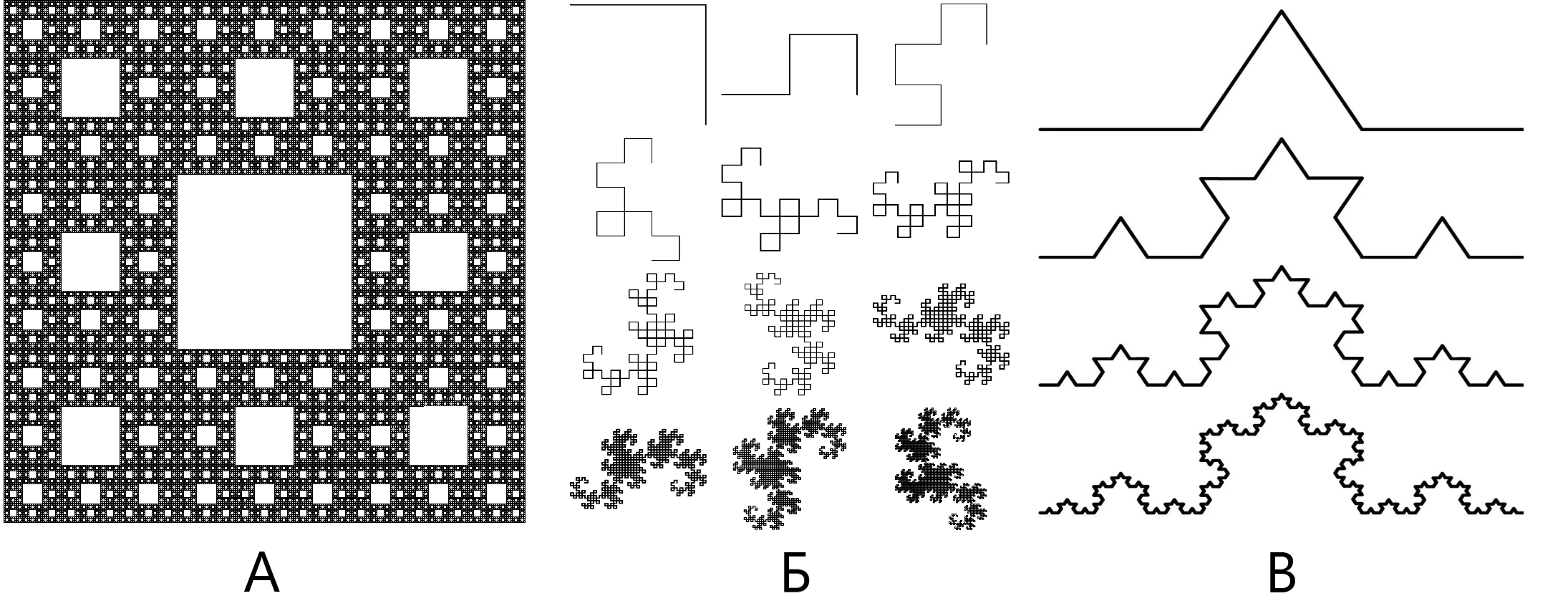
**Алгебраические фракталы**

Алгебраические фракталы основываются на математических формулах, чаще всего на комплексной динамике. Например, множество Мандельброта, которое описывается очень простым уравнением: zn + 1 = zn2 + с, где с это точка, которая проверяется на принадлежность к множеству. Если точка с осталась в заданных пределах, при определённом количестве шагов, то она входит в множество и окрашивается в чёрный цвет, иначе – в белый. Пределом обычно считают момент, когда модуль очередного числа zn превышает 2. Чтобы сгладить края множества, точки, не вошедшая в множество, окрашиваются в тона серого таким образом, чтобы более близкие к вхождению имели более тёмный цвет.



**Геометрические фракталы**

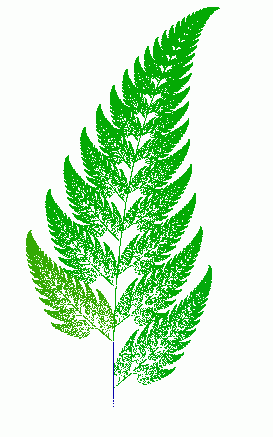
Геометрические фракталы строятся следующим образом: берётся основа, некоторые части которой затем заменяется каким-либо фрагментом, далее фрагменты преобразовываются подобно основе до того момента, когда визуально различить вносимые изменения будет невозможно. После этого общая форма станет ясна. Но следует заметить, что настоящий фрактал подразумевает бесконечное количество итераций. Примерами геометрических фракталов являются: снежинка Коха, кривая дракона, ковёр Серпинского.



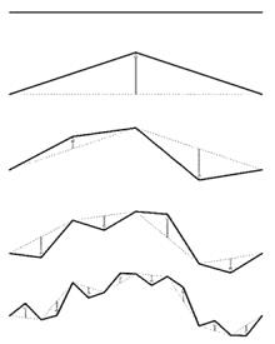
Геометрические фракталы используются в компьютерной графике для построения моделей листьев, растительности, береговых линий.

**Стохастические фракталы**

Их построение проходит с постоянным изменением случайным образом параметров, определяющих форму фрактала.

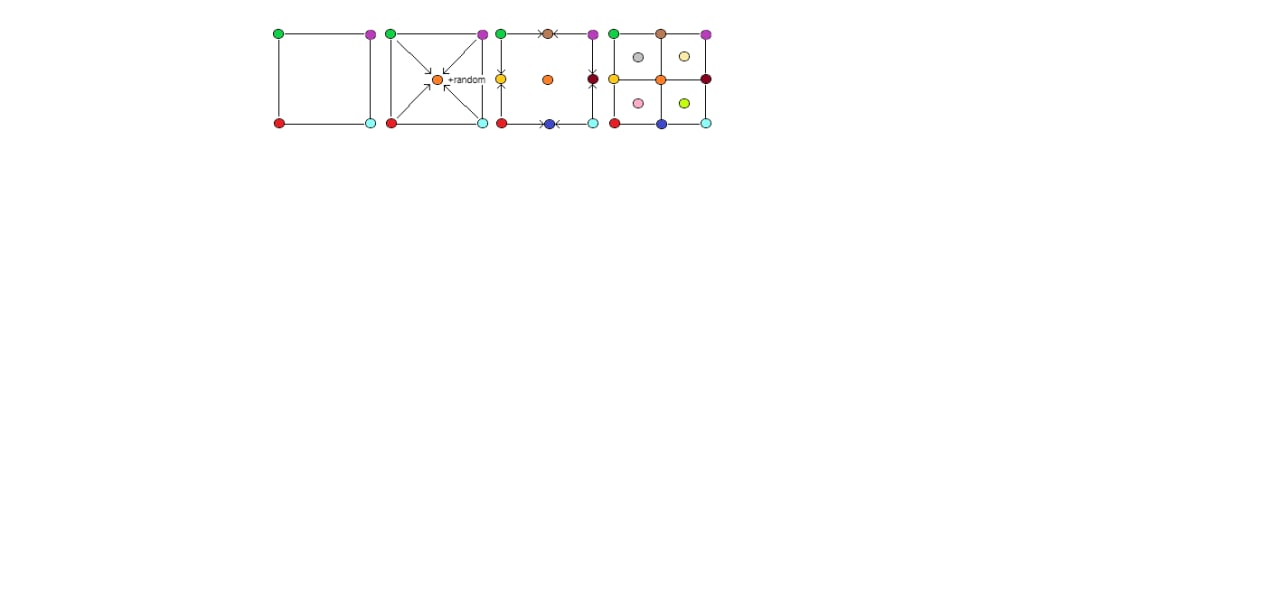


Геометрические фракталы способны повторять многие природные формы и структуры, однако получившиеся изображения выглядят неестественно. Причиной этого является то, что в природе множество внешних факторов влияют на развитие той или иной структуры: горы подвергаются эрозии, листва на деревьях растёт обильнее с солнечной стороны и так далее. Поэтому построить такие объекты с помощью одних лишь формул будет очень сложно. Видимость влияния внешних факторов может создать случайное изменение параметров на каждой итерации. Для более натурального влияния случайный фактор должен иметь предопределённые пределы. Установка этих пределов поможет задать направление развития структуры сохранив природную асимметрию.

**Midpoint Displacement**

Алгоритм Midpoint Displacement изначально предполагал генерацию кривых в одном измерении. Изначально берётся отрезок между точками со случайными высотами. Далее берётся центральная точка на отрезке и случайным образом меняется её высота, после чего образуются два отрезка с общей точкой. На следующих итерациях такая же операция производится с получившемися отрезками.

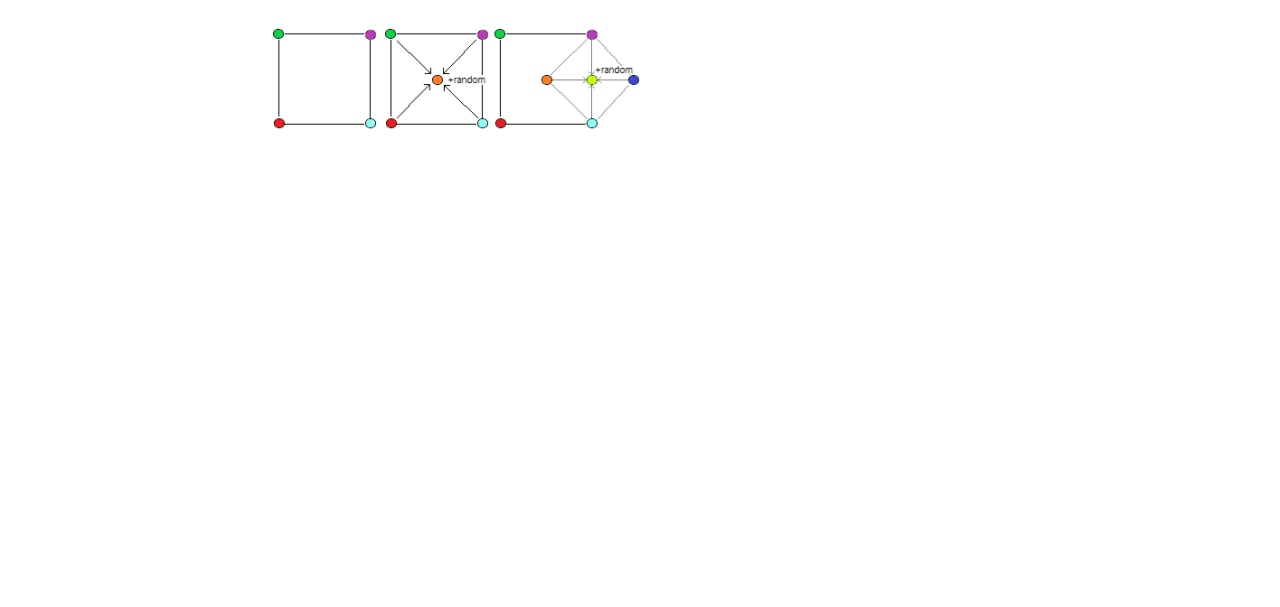
Детали алгоритма могут варьироваться, например, центральную точку можно двигать не перпендикулярно оси абсцисс, а перпендикулярно отрезку, к которому эта точка принадлежала.



Этот алгоритм можно применить и к двум измерениям. В таком случае четырём углам изначальной карты присваивается случайная высота. Центральной точке этого квадрата присваивается высота равная усреднённому значению высот угловых точек плюс случайная величина. Далее серединным точкам на сторонах квадрата присваивается усреднённая высота угловых точек, которых касается сторона. На следующих итерациях такая же операция производится с получившемися четырьмя квадратами.

**Diamond-square**

Предыдущий алгоритм способен генерировать приемлемые результаты, однако процесс создания ландшафта можно внести ещё больше случайности. Алгоритм diamond-square таким же образом высчитывает высоту центральной точки – шаг square. Второй шаг – diamond – изменяет формулу расчёта серединных точек из-за чего их высоты зависят от большего количества переменных. Здесь, чтобы получить высоту серединной точки необходимо найти среднее между высотами центра своего квадрата, центра пограничного квадрата и точек, являющимися концами отрезка на котором лежит исследуемая точка. К полученному среднему значению также прибавляется случайная величина.



Для реализации данного алгоритма следует отметить, что для второго этапа необходимо знать высоты всех центральных точек, поэтому этап diamond не должен наступать до того, как этап square не будет выполнен для всех возможных квадратов. Также при расчёте высот серединных точек у краёв карты будет происходить выход за границы. Для того, чтобы предотвратить это необходимо либо заранее определить константы, которые будут использоваться вместо несуществующих точек, либо использовать точки, лежащие на противоположной стороне карты.

Используя эти алгоритмы можно получить карту высот, которые далее можно интерпретировать в необходимое представление: погружать под воду участки с высотой ниже уровня моря, накладывать двумерные структуры друг на друга и получать карту подземелий, на основе высоты и отдалённости от воды создавать различные климатические условия, от которых будет зависеть дальнейшая детализация карты и так далее.

Данные алгоритмы являются одними из самых простых. Для генерации ландшафта можно использовать, например, шумы, клеточные автоматы или комбинировать различные алгоритмы друг с другом, используя дополнительные инструменты оптимизации и систематизации.

**Сжатие изображения**

Фрактальное сжатие изображения – это способ чрезвычайно сильно уменьшить количество памяти, требуемой для хранения изображения, с приемлемым уровнем потерь в качестве. Алгоритм основан на системах итерируемых функций и предполагает поиск самоподобных участков изображения.

Система итерируемых функций (СИФ) – набор функций, который позволяют отобразить одно множество на другое.

СИФ – ещё один способ описания фракталов. Наиболее простая СИФ состоит из аффинных преобразований на плоскости.

Нахождение коэффициентов аффинного преобразования для фрактала не составляет проблем, так как всё изображение есть самоподобная фигура. Для подавляющего же количества изображений найти такие коэффициенты намного сложнее, или же вовсе невозможно из-за ограничений, заложенных в аффинных преобразованиях. Однако, несмотря на то, что в произвольном изображении отдельные его части не являются подобными всему изображению в целом, возможно нахождение фрагментов подобных другим фрагментам. С этого шага и начинается алгоритм.

Перед сжатием картинка разбивается на неперекрывающиеся ранговые блоки и на перекрывающиеся доменные блоки. Далее, для каждого рангового блока выбирается наиболее похожий доменный блок и находятся коэффициенты аффинного преобразования.

….