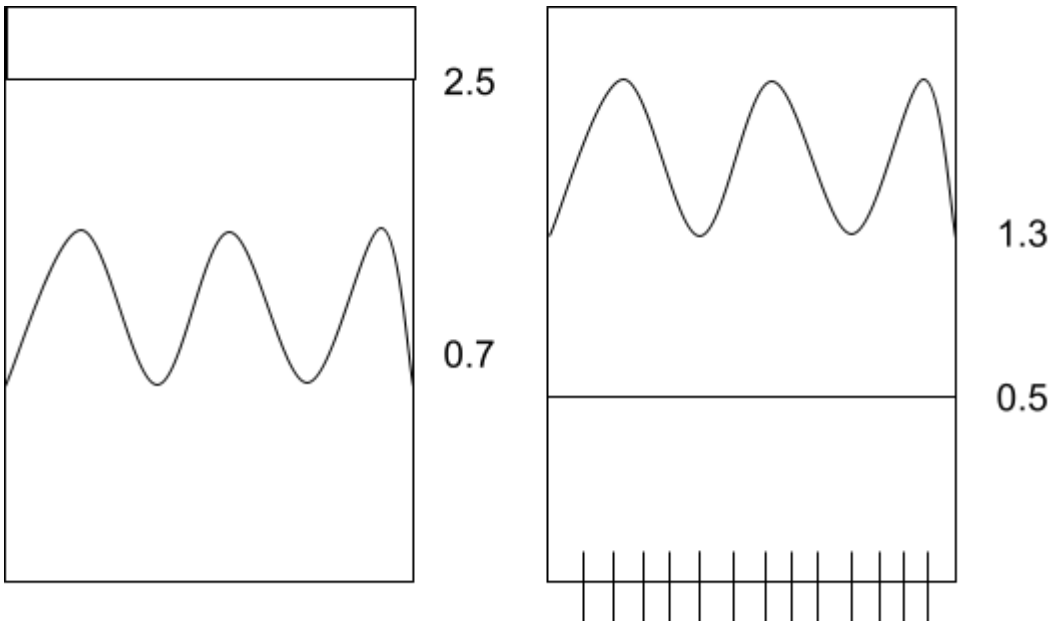


Методы построения нерегулярных(неструктурированных) сеток

В неструктурированных сетках, в отличие от структурированных, узлы могут располагаться где угодно, а их нумерация может быть какой угодно.

Поэтому методы их построения носят в основном геометрический характер.

Наиболее часто употребимыми являются метод построения триангуляций Делоне и метод движущегося фронта.

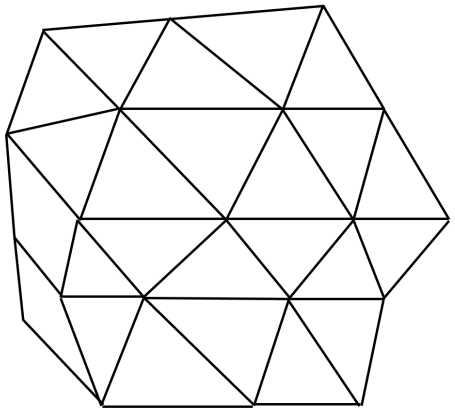
Ячейки в неструктурированных сетках могут быть произвольного характера, но мы в данном случае остановимся на случае треугольных сеток.

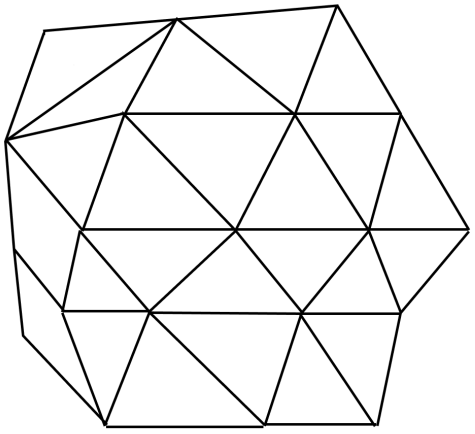
Треугольные сетки - двумерные сетки, у которых все ячейки являются треугольниками.

В двумерном случае достаточно трех- и четырехугольных ячеек для получения полноценной неструктурированной сетки.

Построение треугольной сетки по множеству точек называется триангуляцией. Первоначально, вокруг заданного множества точек мы можем построить некоторую оболочку. Далее нашей задачей будет покрыть всю эту выпуклую оболочку треугольной сеткой.

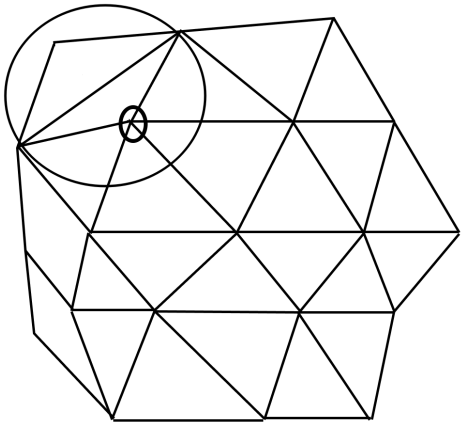
Данную задачу можно выполнить по-разному, например:





Однако, существует особая триангуляция - триангуляция Делоне, которая основывается на следующем:

Если мы проведем прямую, а затем опишем круг вокруг внутреннего треугольника и увидим, что этот круг содержит точку из заданного множества, данная прямая будет являться неверной.



Следовательно, нам необходимо, чтобы треугольники обладали таким свойством, что ни одна из точек множества внутрь описанных вокруг треугольников кругов не попадёт.

Такая триангуляция и называется триангуляцией Делоне

В трехмерном случае все аналогично, но вместо треугольников мы будем рассматривать тетраэдр.

Соответственно, тетраэдрация Делоне обладает следующим свойством:

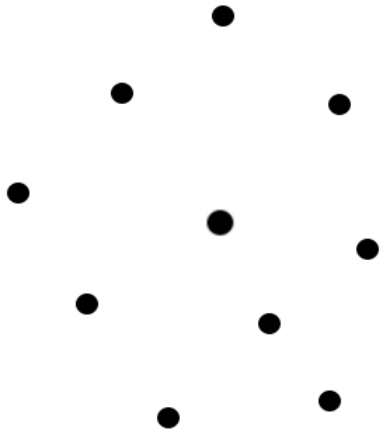
Ни одна из точек заданного множества не может попасть внутрь шара описанного вокруг тетраэдра.

Для построения триангуляции Делоне рассмотрим еще одну геометрическую характеристику заданного множества - ячейки Дирихле.

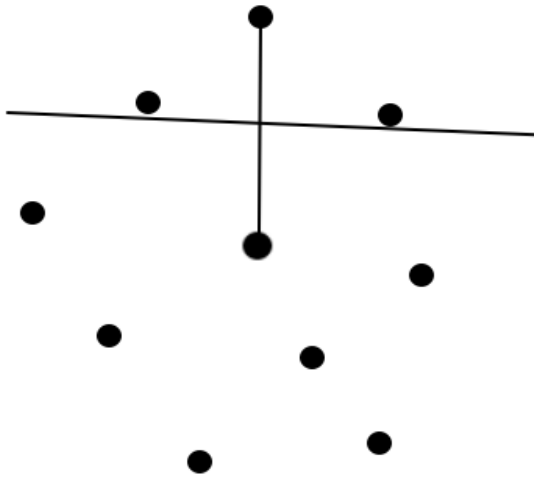
Выберем точку из множества, тогда остальные точки множества будут разбиты на 2 группы:

В первую группу, называемую ячейкой Дирихле, попадут точки, расстояние от которых до выбранной точки меньше или равно расстоянию до любой другой точки. Во вторую группу попадут все остальные точки.

Итак, возьмем набор точек и выберем одну из них в качестве исходной.

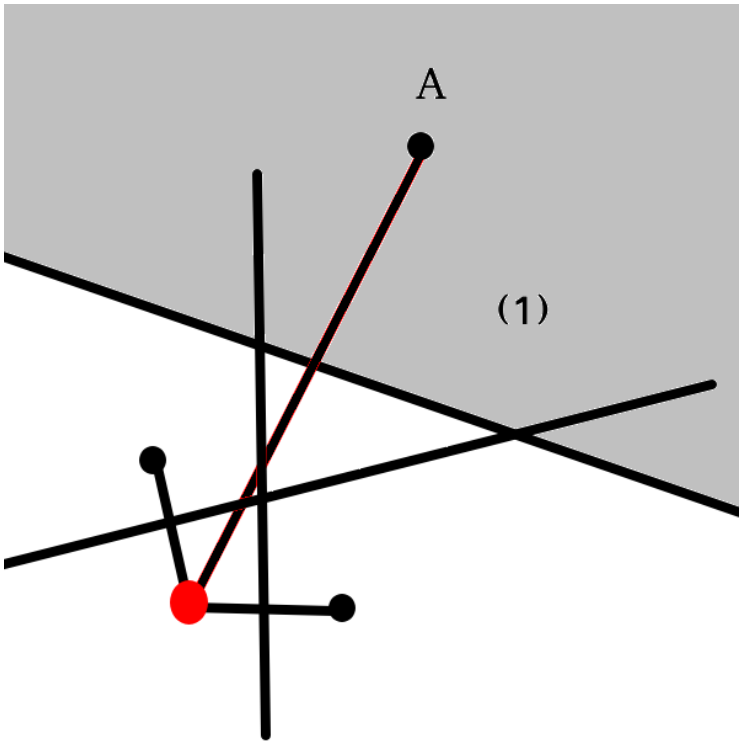


Далее возьмем какую-то другую точку и соединими их линией, а затем проведём серединный перпендикуляр к этой линии



следовательно, расстояние до точки в полуплоскости над серединным перпендикуляром будет меньше, чем расстояние до исходной точки. А пересечение всех полуплоскостей, удовлетворяющих заданному условию и будет являться ячейкой Дирихле.

Для простоты возьмем меньшее количество точек, соединим их все с исходной точкой и проведём серединные перпендикуляры.



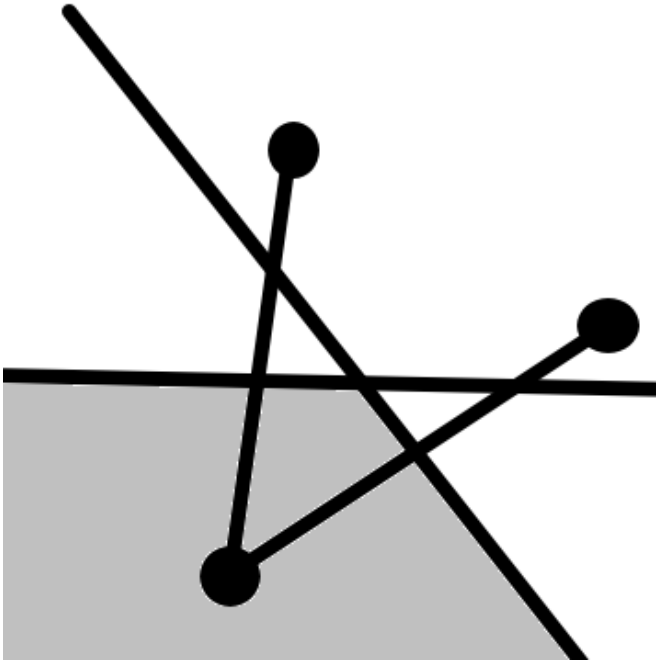
Заметим, что область (1) полностью лежит в не удовлетворяющих условию полуплоскостях других серединных перпендикуляров, а значит, точка A не будет участвовать в формировании ячейки Дирихле.

Получим 2 типа точек, одни будут отсечены прочими полуплоскостями, а другие будут формировать границу ячейки Дирихле.

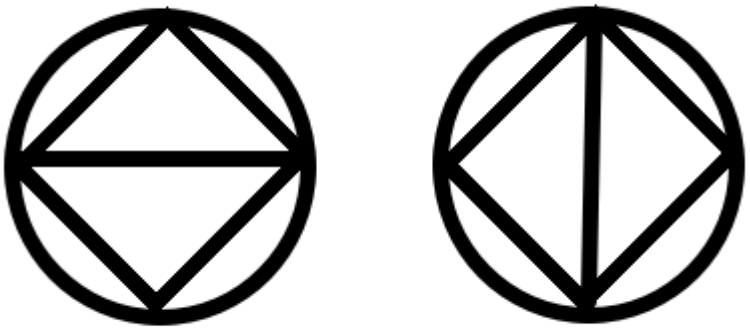
Эти точки, каждая из которых формирует участок границы ячейки Дирихле будут называться соседними для нашей исходной точки.

Утверждение.
Соединив между собой все соседние точки, мы получим триангуляцию Делоне.

Кроме того, встречаются точки, у которых границы ячейки Дирихле уходят в бесконечность, такие точки являются границей выпуклой оболочки нашего набора точек.

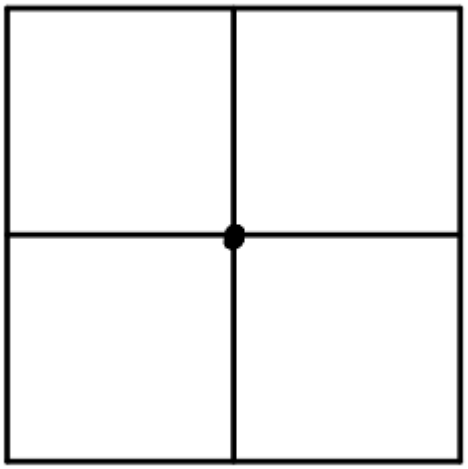


Также возможен случай, когда 4 точки лежат на одной окружности, тогда триангуляция Делоне будет неоднозначной.



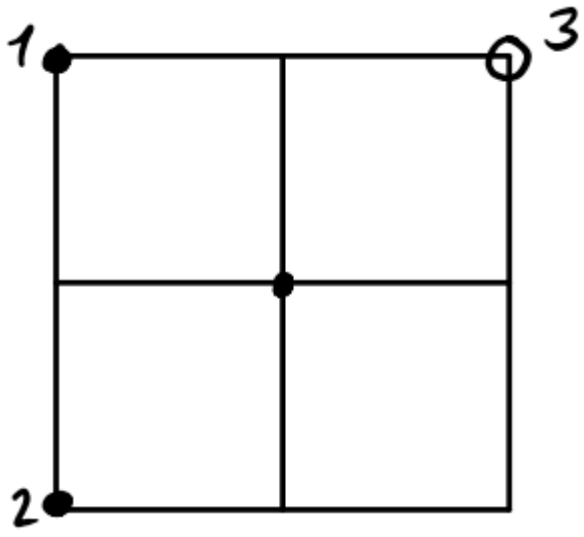
Посмотрим, что случится с ячейкой Дирихле в этом случае.

Нарисуем ячейку дирихле для квадратного расположения точек.



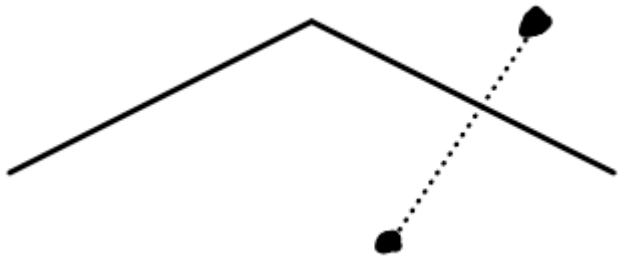
Для того чтобы выяснить лежат ли точки на одном круге, достаточно выяснить пересекаются ли 4 линии в 1 точке. Казалось бы простая задача для решения с помощью линейных уравнений, однако точно решать линейные уравнения мы “не умеем”, мы всегда делаем погрешности из-за которых мы можем установить неправильный результат.

Теперь посмотрим что произойдет с сечениями дирихле:

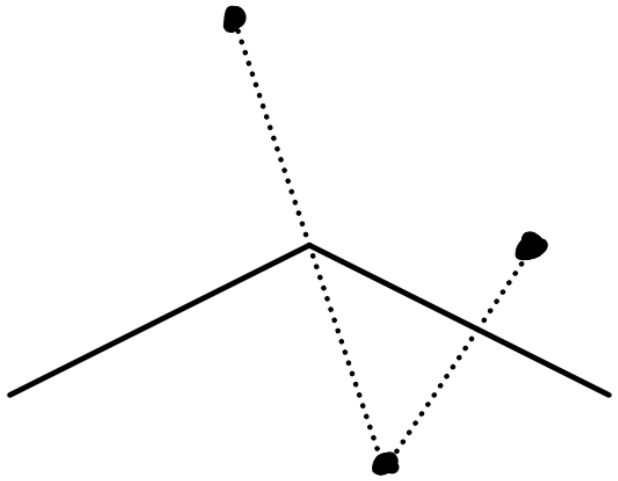


В любом случае точки 1 и 2 являются соседями. Но является ли соседом точка 3? Если мы хотим сделать триангуляцию, мы должны как-то разрезать. Можем ли мы решить эту задачу точно? Для этой цели удобно использовать арифметику отличающуюся от решения разностных схем.

К примеру. Имеется ломаная. Пересекает ли отрезок по двум точкам эту ломаную?

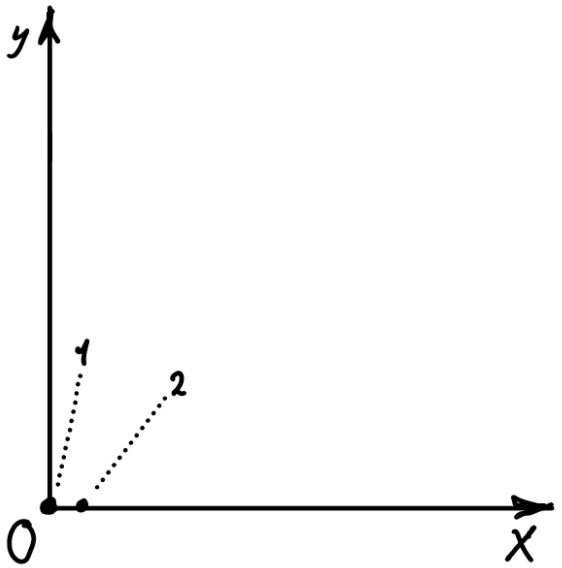


Казалось бы простая задача - проверяем на пересечение отрезок со всеми отрезками составляющими ломанную и если нет пересечения с составляющими, то нет пересечения и с самой ломаной. Но это все хорошо только в случае когда отрезок не проходит вблизи узла.



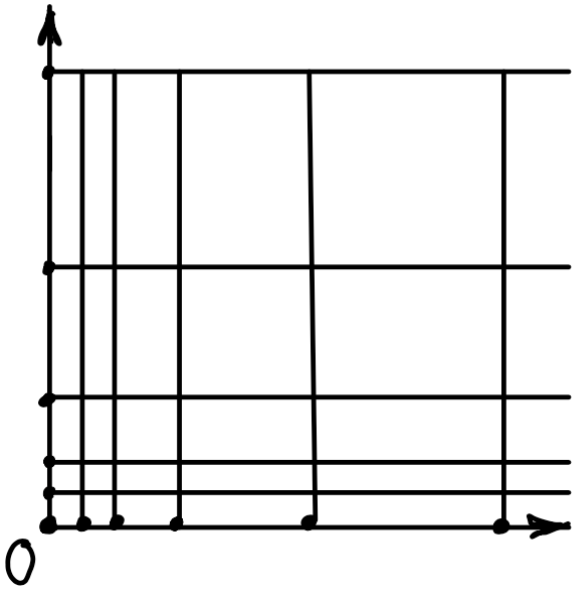
Тогда при вычислении пресечения, из-за погрешности, можно получить что отрезок не пересекает ни одну составляющую. Это следствие использование чисел с плавающей запятой.

Теперь рассмотрим что из себя представляет точка на плоскости.



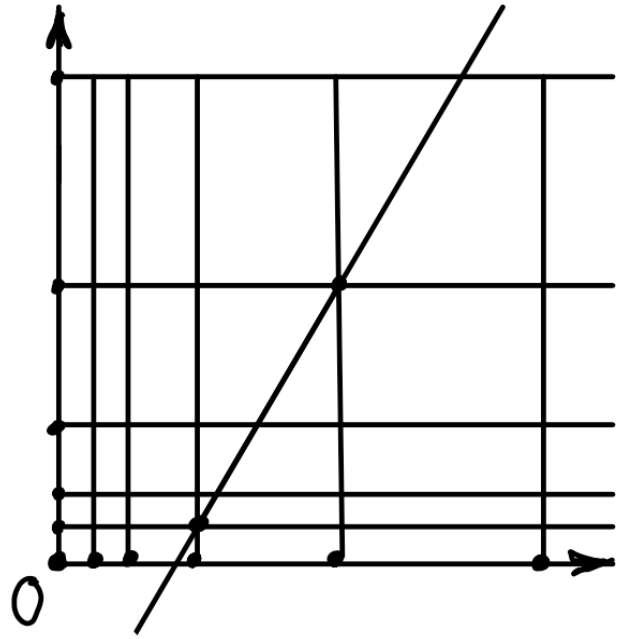
Точка 1 имеет координаты (0, 0) которые мы можем точно представить. Точка 2 имеет координаты (e, 0), где e-самое маленькое число после 0 которые мы можем представить переменной с плавающей запятой.

Отметим на осях еще несколько последовательных чисел с плавающей запятой и сделаем из них сетку:

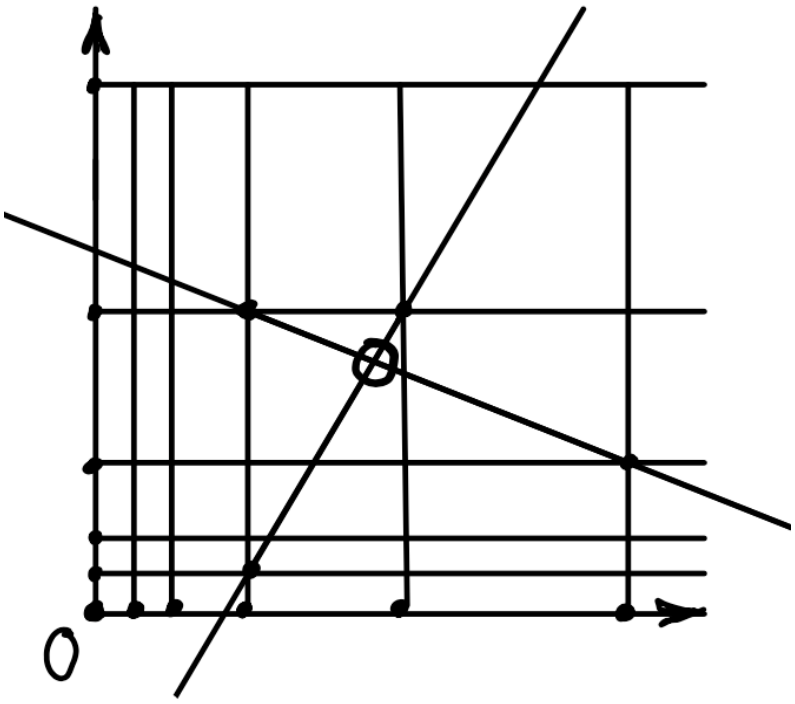


Получается, как мы будем ее называть, вся машинная плоскость, состоящая из конечного числа точек, представляемых двумя переменными с плавающей запятой.

Тогда что такое прямая на этой плоскости? Это прямая проходящая через заданные точки на этой плоскости:

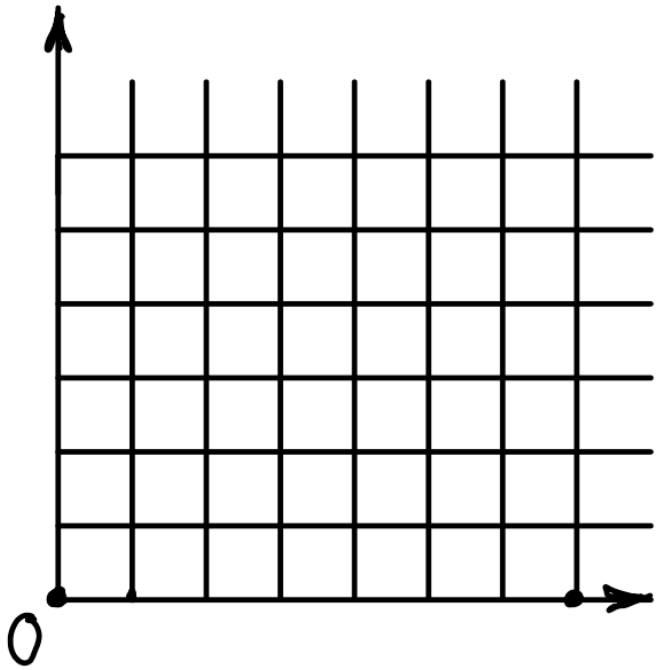


Тогда может образоваться ситуация, что если у нас есть другая прямая, которая пересекает данную:

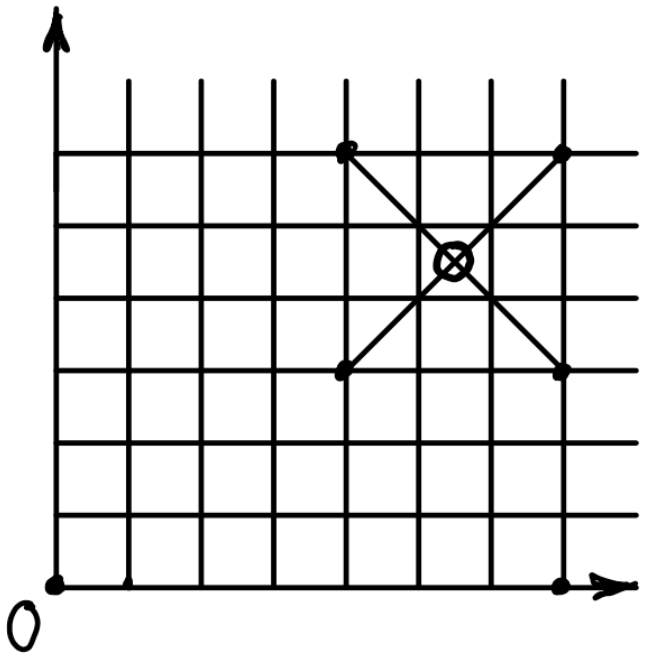


их точки пересечения может не оказаться на машинной плоскости (как показано на рисунке). Поэтому при работы с представлением точки через плавающую запятую мы должны так или иначе использовать округление и перенос точки к ближайшей на машинной плоскости. Точка в которую будет перенесена оригинальная зависит от многих причин, таких как положение точки и алгоритм округления. Кроме того чем больше абсолютная величина точки, тем грубее получаются ячейки.

Поэтому при описании геометрических объектов при работе в примерно одинаковом масштабе, целесообразно использовать целочисленную равномерную решетку:



Казалось бы мы не ушли от проблемы, так как можно найти 2 линии, которые не пересекаются ни в одной точке машинной плоскости.



Но здесь ситуация другая. Примем шаг наших координат за Δx и Δy. Тогда координаты точек это $i_1\Delta x,\ j_1\Delta y;\ \ i_2\Delta x,\ j_2\Delta y$. Формула прямой для этих точек:

$$\frac{x - i_1\Delta x}{i_2\Delta x - i_1\Delta x} = \frac{x - j_1\Delta y}{j_2\Delta y - j_1\Delta y}$$

Δx и Δy можно сократить:

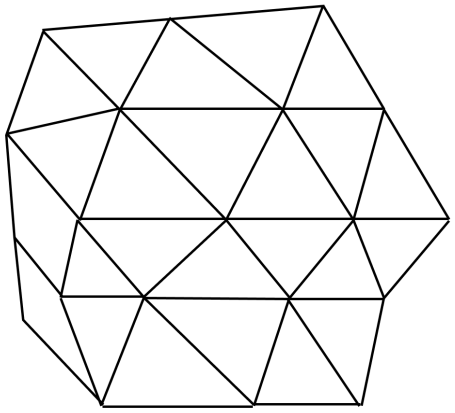
$$\frac{x - i_1}{i_2 - i_1} = \frac{x - j_1}{j_2 - j_1}$$

Получаем уравнение с рациональными коэффициентами. При умножении на $(i_2 - i_1)(j_2 - j_1)$ получим уравнение с целыми коэффициентами, систему которых можно точно и однозначно решить. Теперь если у нас есть несколько прямых пересекающихся в 1 точке, для каждой прямой можно составить уравнение с целыми коэффициентами и найти точки пересечения решая с ними систему уравнений с целыми коэффициентами. Такое решение будет давать однозначный ответ о том совпадают точки или нет. То же применимо и для трехмерного случая. Такой подход с целочисленной сеткой решает проблему описанную нами выше.

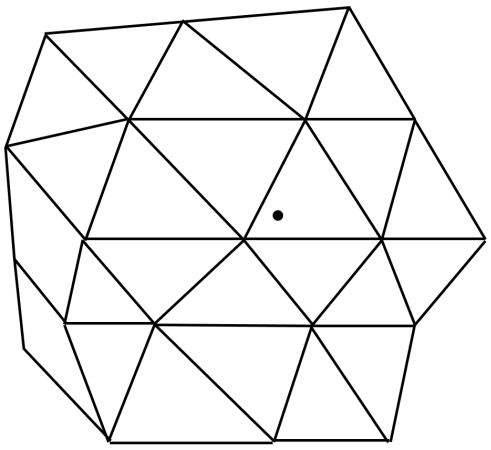
Как уже говорилось, мы построить триангуляцию поиском пересечения полуплоскостей, однако на построение ячайки дирихле в данном случае уйдет n операций, перебор всех точек нашей сетки. Это не очень эффективно, так как большинство точек будет отброшено, поскольку мы берем только ближайшие точки.

Рассмотрим следующий алгоритм оптимизации:

Предположим что у нас уже построена триангуляция для некоторых точек:



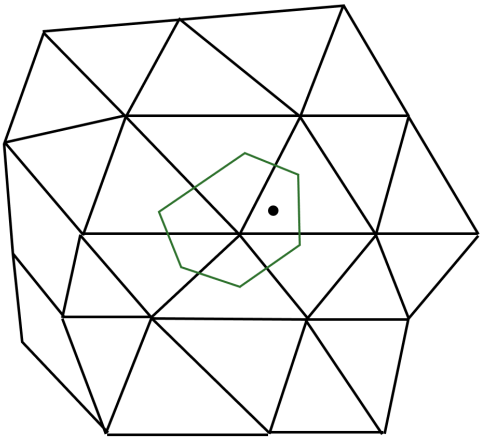
И теперь мы хотим добавить в триангуляцию еще одну точку.
Предположим, что ее добавили так:



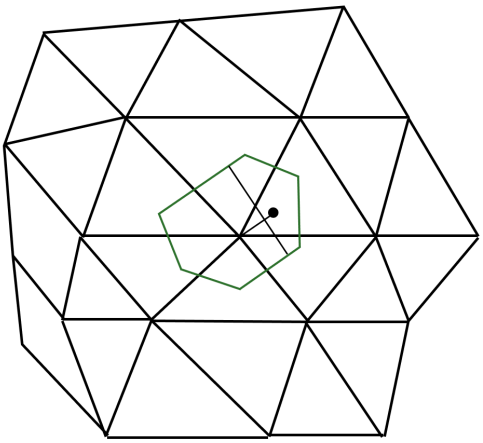
Перестройка триангуляции займет почти конечное число операций. То есть нам не потребуется n^2 операций для того, чтобы ввести еще одну точку.

Первое, что мы должны сделать - найти ближайшую точку из нашего множества этой точки. Эта операция кажется сравнительно безобидной, на

самом деле она в этом и других алгоритмах будет решающей. Здесь понадобится переборка n^2. Сейчас будем считать, что мы ее нашли. Далее нам потребуется конечное число операций для того, чтобы все выяснить. На рисунке представлена ячейка Дирихле данной точки:



Мы соединим эту точку с нашей точкой и проведем серединный перпендикуляр:



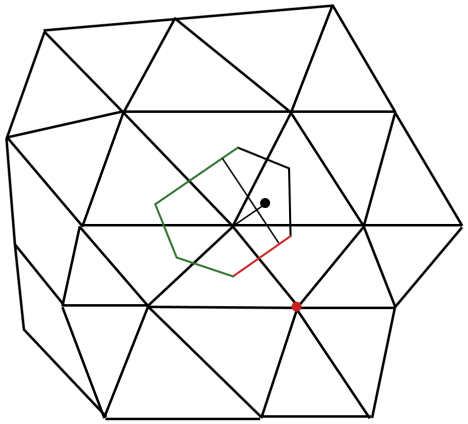
Как мы говорили, ячейка Дирихле представляет собой выпуклый многоугольник, поэтому серединный перпендикуляр разрежет ее на две части. При этом он начнет формировать список соседей нашей новой точки и будет изменять списки соседей некоторых точек. В данном случае мы изменим список соседей ближайшей точки, куда попадет, конечно, новая точка, потому что до нее будет самое близкое расстояние.

Теперь рассмотрим следующую ситуацию: серединный перпендикуляр отрезал 2 участка ячейки Дирихле, соответствующие соседи, которые

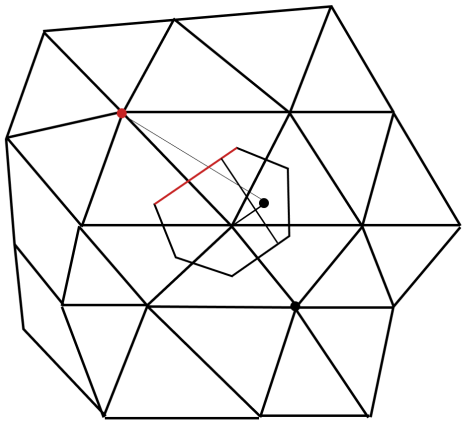
были у этой точки, выбрасываются из списка соседей, вместо них вставляется эта точка.

Если ситуация другая, когда точка попала в ячейку Дирихле, но немного дальше, и серединный перпендикуляр не отрезал ни одной точки, то точка вставляется между теми соседями, границы которых она пересекла.

Далее мы смотрим, с какими гранями пересекся наш серединный перпендикуляр, например, возьмем нижнюю грань - она соответствует данной точке (выделены красным цветом):

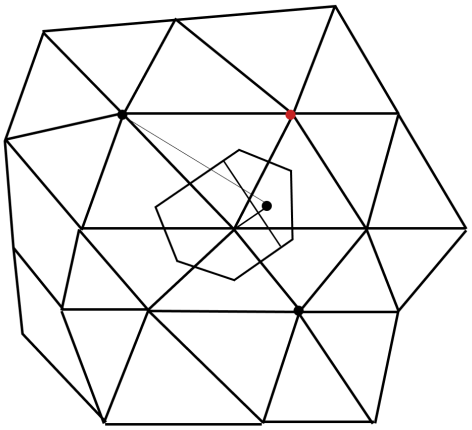


Эта точка также попадает в список соседей нашей точки. Если двигаться против часовой стрелки от нашей точки, то мы возьмем верхнюю грань:



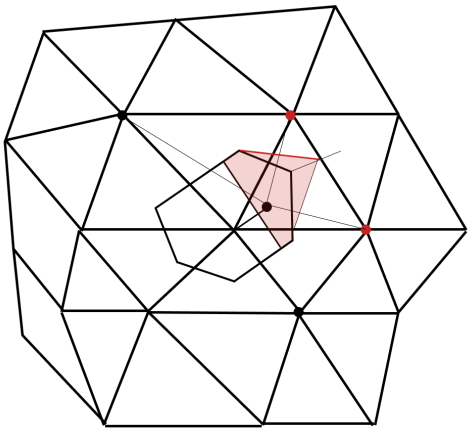
Данная точка попадает в список соседей нашей точки при соединении.

Проведем еще один серединный перпендикуляр - он обязательно попадет в данную точку. Далее мы попадем на границу ячейки Дирихле с данной точкой:



У нас продолжается процедура таким образом. В данном случае точка ничего не отсекла, и у нее появился дополнительный сосед, дополнительное ребро ячейки Дирихле.

Соединим данную точку с нашей и проведем еще один серединный перпендикуляр:



Процесс замкнулся, и у нас получилась новая ячейка Дирихле. Очевидно, что у нас число операций на каждого вновь появляющегося соседа конечное: мы должны вычислить точку пересечения, проверить список соседей, переставить и т.д, если число соседей конечно. Среднее число соседей в заполненной области будет равно 6.

Теперь рассмотрим количество вершин $N_{\text{В}}$, количество ребер $N_{\text{Р}}$ и количество узлов $N_{\text{У}}$. Есть формула Эйлера, которая для произвольного графа, а триангуляция является плоским графом, связывает между собой число вершин, ребер и узлов:

$$N_{\text{РЕБЕР}} = N_{\text{УЗЛОВ}} + N_{\text{ЯЧЕЕК}} - 1$$

Если брать выпуклую оболочку нашего множества, то есть узлы, которые лежат на границе оболочки, а есть узлы, которые лежат внутри. Будет считать, что число узлов, лежащих внутри, намного больше числа узлов снаружи. Грубо говоря, сетка равномерная. Если взять квадратную сетку, то число граничных узлов будет $4N - 4$, а число внутренних узлов будет $(N - 1)^2$.

Теперь запишем такую формулу: если каждое ребро будет внутренним, то оно обязательно граничит с двумя ячейками. Если перебрать все ячейки, то каждая ячейка имеет 3 ребра, а если просуммировать все это количество ребер по всем ячейкам, то каждое ребро встретится 2 раза, т.е.

$N_{\text{Р}} = \frac{3}{2} N_{\text{Я}}$.

подставим в формулу выше, получим:

$\frac{1}{2} N_{\text{Я}} = N_{\text{у}}$.

Пусть у каждой точки n_i соседей, тогда среднее число соседей будет равняться сумме n_i, разделенной на $N_{\text{Я}}$:

$$\frac{\sum n_i}{N_{\text{у}}} = n_{\text{ср}}.$$

Выясним число вершин. У каждой ячейки 3 узла, просуммируем и получим $3N_{\text{Я}}$. Значит каждый узел будет считаться со своей кратностью, т.е.

$$3N_{\text{Я}} = \sum n_i \Rightarrow N_{\text{Я}} = \frac{\sum n_i}{3}$$

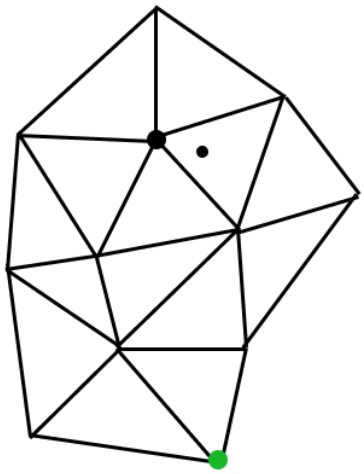
$$3 = n_{\text{ср}} \cdot N_{\text{у}}$$

$$\frac{\sum n_i}{6} = N_{\text{у}} \Rightarrow \frac{\sum n_i}{N_{\text{у}}} = 6$$

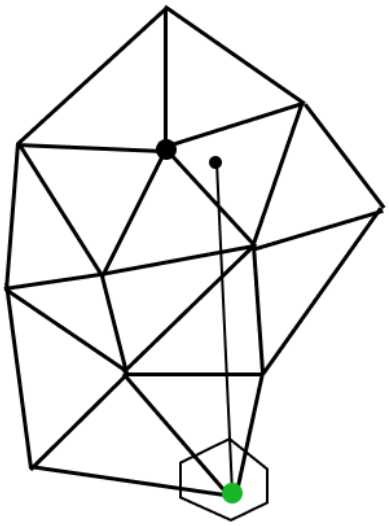
То есть в среднем мы получим 6 соседей на каждый узел.

Теперь разберемся с тем, как найти ближайшую точку. Воспользуемся тем, что у нас уже задана триангуляция, когда мы вводим дополнительную точку.

Возьмем произвольную точку в качестве первого приближения. Будем считать, что зеленая точка - ближайшая:

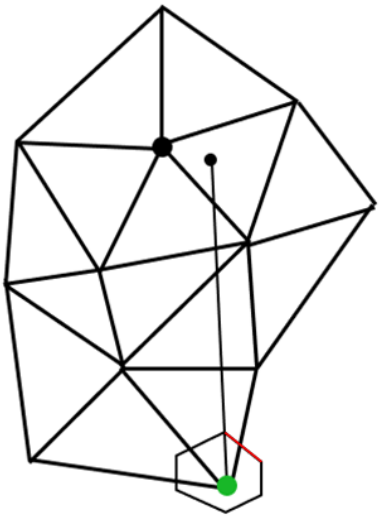


Мы должны проверить, ближайшая она или нет. Мы не будем перебирать все точки. Вместо этого возьмем ячейку Дирихле этой точки, соединим 2 точки и выясним, пересекает ли эта линия соответствующие ячейки Дирихле.

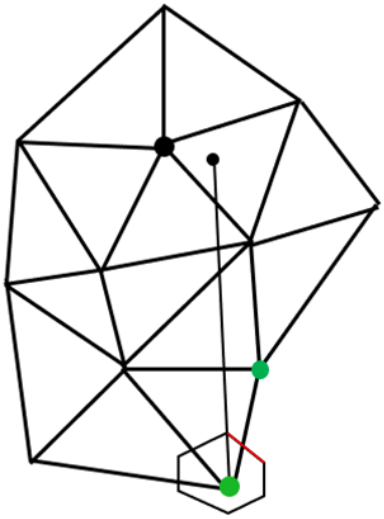


Для этого нам потребуется конечное число операций.

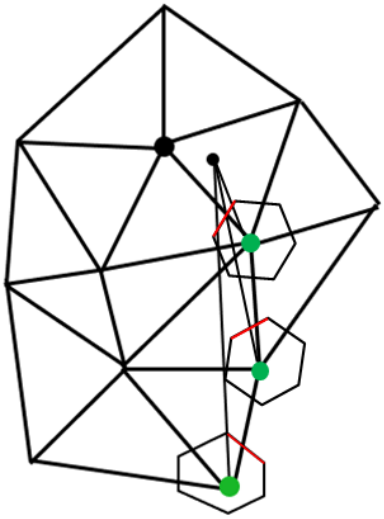
Если она не пересекает, то она находится внутри ячейки Дирихле, и задача решена. Если же она пересекает, значит эта точка не ближайшая, в этом случае мы выясняем, какую границу она пересекла. Предположим, что она пересекла следующую границу:



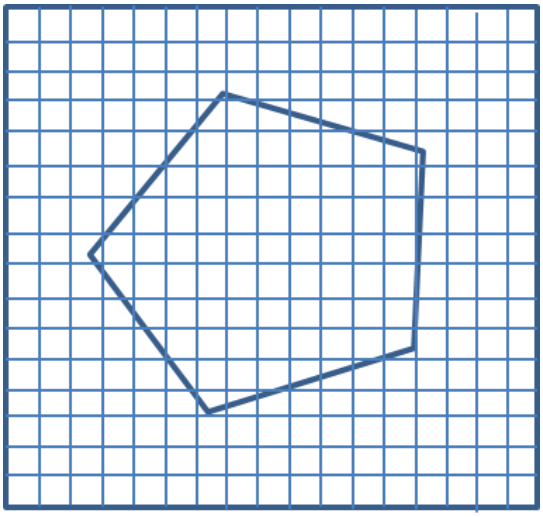
Тогда в качестве следующего приближения выбираем следующую точку:



И с ней опять повторяем операции:



И вот мы нашли ближайшую точку. В среднем такая операция займет порядка N шагов. Для того, чтобы быстрее угадывать концы, нужно проделать такую вещь:

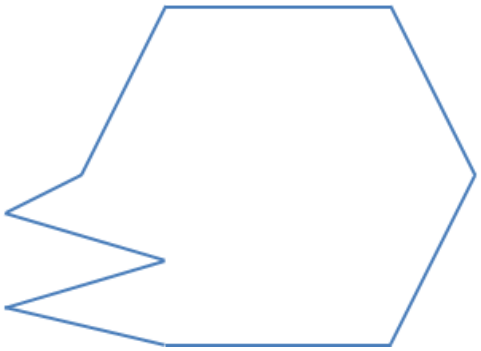


Помещаем область внутрь прямоугольника и вводим в нем равномерную прямоугольную сетку. Каждая точка попадет в соответствующий квадрат сетки. Мы можем составить список по квадратам за N операций. Чтобы выяснить нужный квадрат достаточно двух операций: деление на ∂x и деление на ∂y. И тогда, если мы возьмем достаточно малое разбиение, то в каждый квадрат попадет мало точек. Теперь в качестве начального приближения будем брать точки из квадрата, куда попала добавляемая точка. Если в нем не оказалось точек, то берем ближайшие, и так далее. Такой процесс заставляет нас ввести общее число операций $NlnN$.

Метод движущегося круга

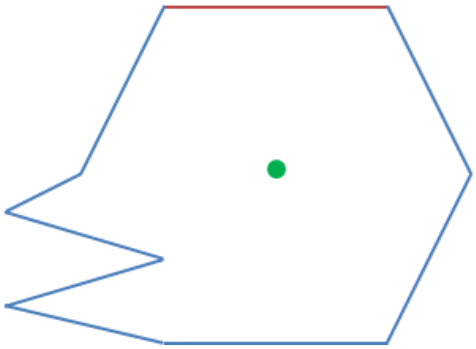
В первом случае никаких нюансов не возникает - все те же геометрические задачи переходят на уравнения 3×3.

Теперь рассмотрим задачу:

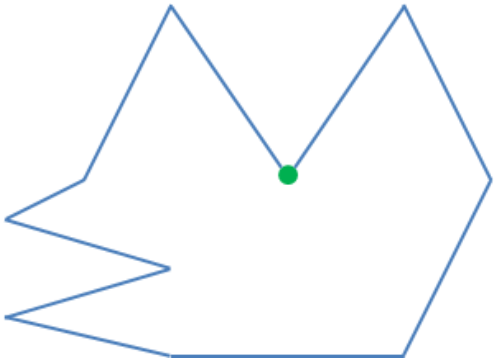


Нам дана область, необязательно выпуклая, и надо построить внутри области треугольную сетку.

Для решения существует следующий алгоритм: выберем произвольное ребро этой области. Некоторым способом построим точку, которая будет называться подходящей точкой.



Возьмем треугольник, опирающийся на подходящую точку и ребро и вырежем ее из нашей области.



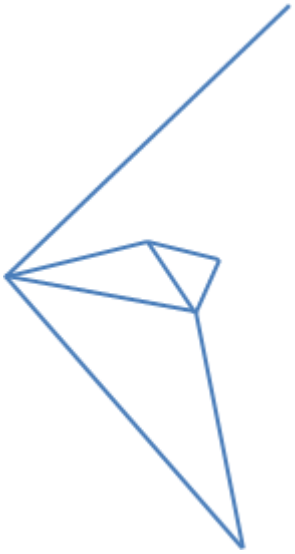
Таким образом задача сводиться к прежней с более малой областью. Повторяя операцию снова и снова, задача будет решена.

Подходящая точка находится следующим образом: строится равнобедренный треугольник, в основании которого выбранное ребро. В случае, когда стороны треугольника пересекают границу, вместо подходящий точки берется одна из граничных.

Существует теорема, которая гласит, что для любой области мы всегда можем отрезать от нее треугольник, который построен на двух соседних граничных точках и ещё одной граничной точки, и этот треугольник не будет пересекать границы ломанной и будет находиться целиком внутри.

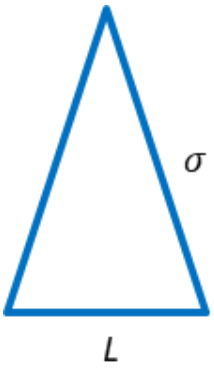
Поэтому всегда такой треугольник построить можно, либо опираясь на наш равнобедренный треугольник, либо перебрав все границы.

Неправильный выбор подходящей точки может привести к сходимости алгоритма к некоторому куску области, но не к области целиком.



Существует способ выбора подходящей точки, при котором алгоритм обязательно сработает до конца. Используем некоторую адаптацию. Если задана функция $S(x, y) > S_0$, которая будет характеризовать подходящий размер ячеек с координатами x и y. Введем некий параметр r и подходящую точку будем рассматривать только в том случае, если круг радиуса r не пересекает границу. Тогда мы может гарантировать, что расстояние между точками не станет слишком маленьким. Для каждой точки строится свой круг, в зависимости от функции $S(x, y)$ и некоторых других вещей.

Стороны равнобедренного треугольника выбираются следующим образом:



$$\sigma = max\left(\frac{5L}{9}, S(x, y)\right)$$

Такой выбор подходящей точки обеспечивает полное выполнение алгоритма в том случае, если $r = \frac{h}{2}$, где h - высота треугольника.

Поскольку $\sigma > \frac{5}{9}L$, делаем вывод, что

$$h = \sqrt{\sigma^2 - \frac{L^2}{4}} \geq \sqrt{\sigma^2 - \frac{81}{100}\sigma^2} = \frac{\sqrt{19}}{10}\sigma$$

И значит

$$\frac{\sqrt{19}}{10}\sigma \geq \frac{\sqrt{19}}{18}l$$

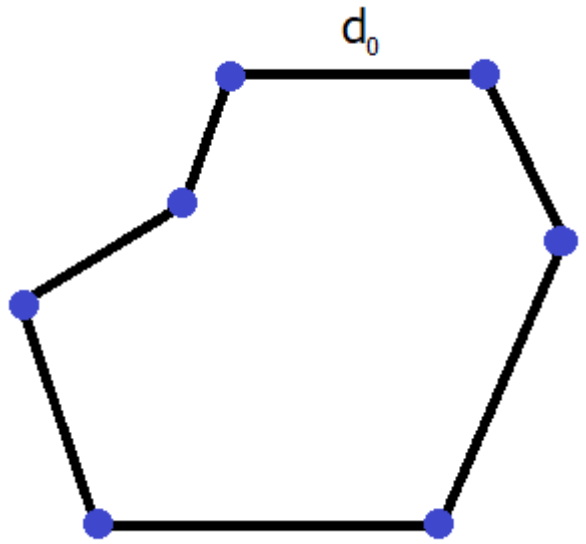
То есть h будет одновременно больше $\frac{\sqrt{19}}{10}\sigma$ и $\frac{\sqrt{19}}{18}l$.

сигма? во всяком случае, будет больше чем S (поэтому нам и достаточно сигмы), а S больше чем S_0, то есть мы можем записать

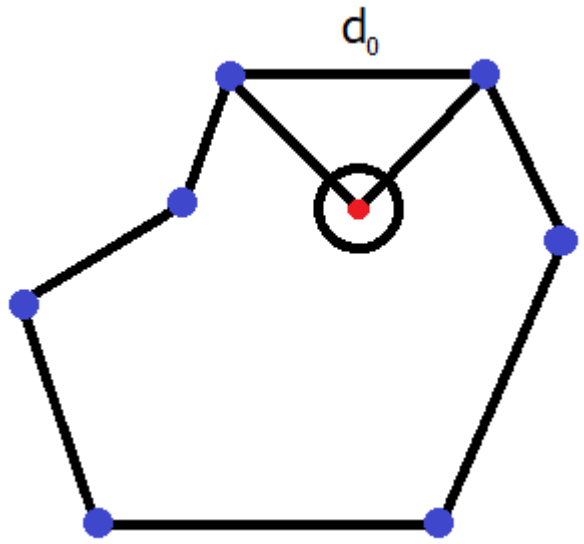
$$h > \frac{\sqrt{19}}{10}S_0$$

То есть какие мы бы точки не ставили, у нас высота треугольника будет всегда больше чем $\frac{\sqrt{19}}{10}S_0$ и тогда $r > \frac{\sqrt{19}}{20}S_0$.

То есть наши точки будут считаться подходящими только в том случае, если они находятся на фиксированном расстоянии от границы. Теперь, что из этого следует. Предположим начальный размер меньше всего минимальной длины ребра d_0



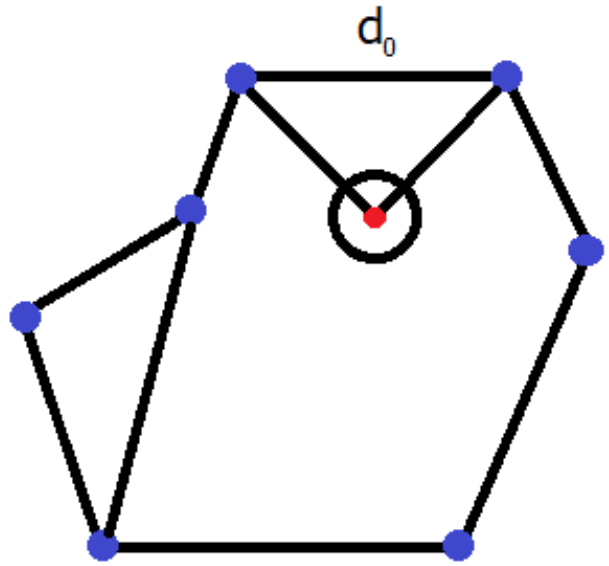
из конечного числа звеньев. Мы можем вычислить число, которое характеризует минимальную длину ребра. Если мы добавим точку



то расстояние от этой точки до всех остальных точек будет больше $\frac{\sqrt{19}}{20}S_0$, потому что круг не пересекается с границей и соответственно расстояние до его граничных точек будет больше чем радиус этого круга. Но тогда d_0, если мы возьмем теперь:

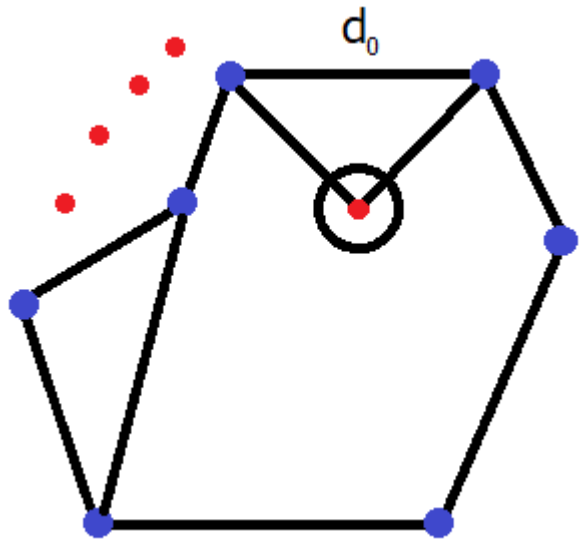
$d_1 = min\{d_0, \frac{\sqrt{19}}{20}S_0\}$ это будет у нас новое расстояние между выбранной новой границей.

Таким образом, если мы соединяем эти точки между собой, то d_0 не меняется:

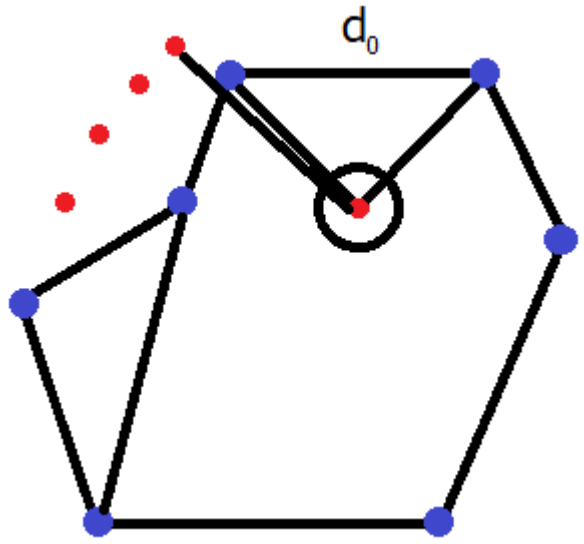


И даже если мы добавим ещё одно точку оно не изменится. d_1 будет минимальное расстояние между любой из двух точек.

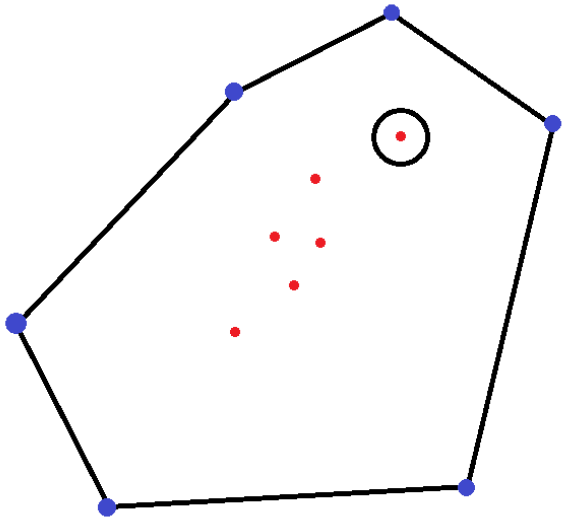
Те точки которые находятся за границей:



Они будут тоже на расстоянии большем, чем d_0, потому что, если проведём звено:



то длина его будет больше, чем длина этой линии и будет меньше чем расстояние до границы, если не пересекает границу, то тогда он не пересекает радиус этой точки.
И таким образом мы получаем.



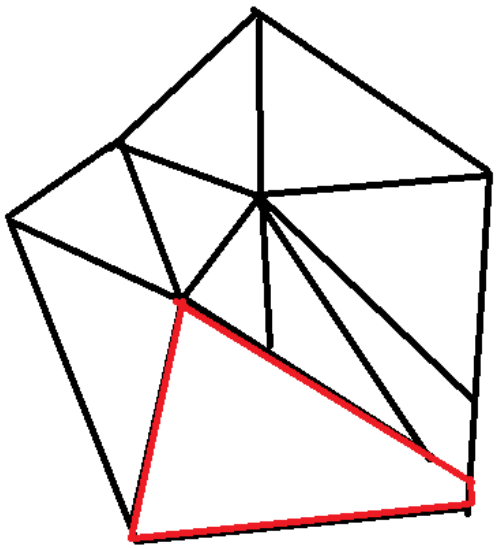
Все точки находятся в этой области, которую мы добавили, расстоянием большим, если мы нарисуем круг у каждой точки радиусом d_1, то эти все круги не будут пересекаться.
Если общая площадь области обозначить через Σ, то

$$\Sigma > N\pi d^2$$

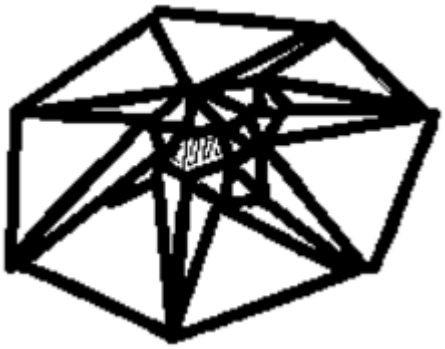
тогда число внутренних точек

$$N < \frac{\Sigma}{\pi d^2}$$

Если мы к этому количеству добавим ещё одну точку у нас это не получится. То есть в какой момент возникнет ситуация



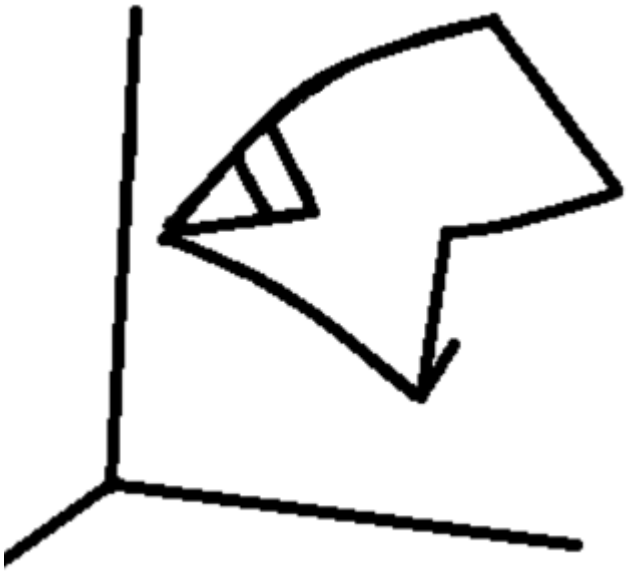
Поскольку остался только непокрытый треугольник, мы всё равно можем проделать операцию, если область ещё не покрыта треугольниками



Остался у нас кусочек непокрытый, в который мы уже ни одной точки добавить не можем, так как расстояние должно быть больше чем S_0, а S_0 будет больше, чем эта оставшаяся область. И тогда мы вынуждены будем отрезать так

В конце концов, так как здесь конечное число граничных точек, по крайней мере, меньше, чем число внутренних точек. Постепенно одна граничная точка отрезается и остаётся один там. Это показывает, что алгоритм работает конечным фронтом.

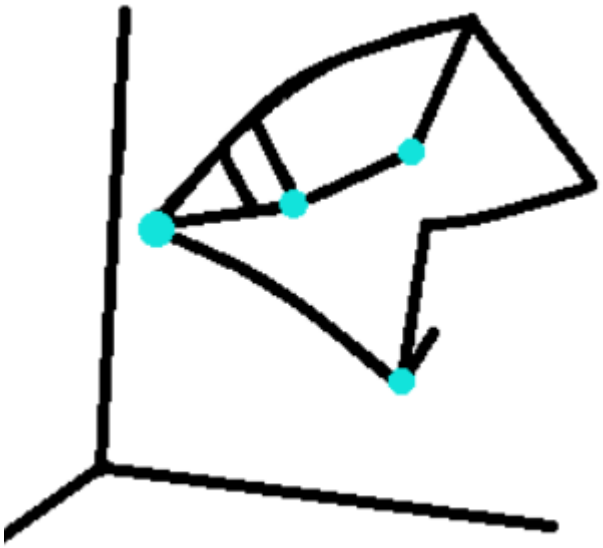
Если у нас есть поверхность, трехмерный случай, потребуется построить сетку треугольников на поверхности.



Как описывается поверхность? Поверхность можно описать разбиением на простые куски. То есть на каждом простом куске поверхность описывается, как функция двух переменных. В общем случае представление параметрическое.

Каждой точке на плоскости углы соответствует точка на поверхности. Если мы теперь возьмем все точки, мы тоже здесь получим некоторую область. Соединений никогда не будет.
Дальше. Перенесённая цепь уже плоская. Мы построим триангуляцию методом продвигающегося фронта.

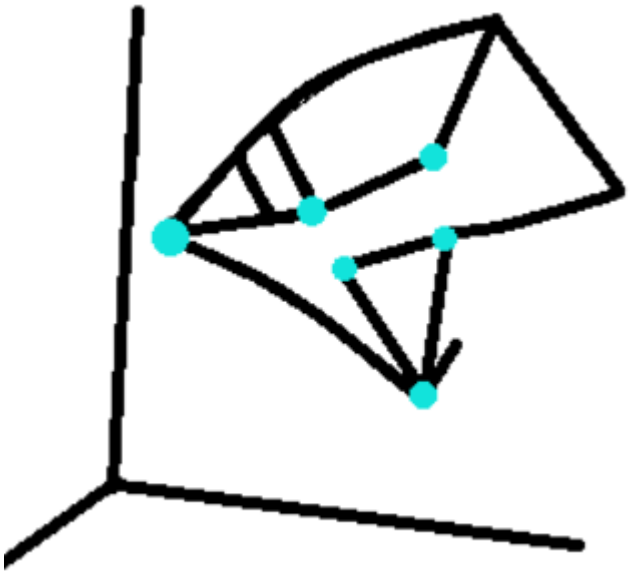
Теперь мы соединяем эти точки и на поверхности



В трёхмерном пространстве уже соединили прямыми линиями, мы получим некоторую триангуляцию близкую к нашей поверхности.
Если поверхность криволинейная

Мы заменим её на треугольник, который, так сказать, внизу лежит и будем его использовать.

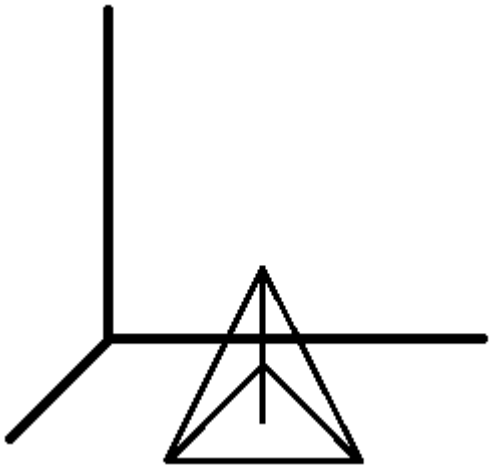
Можем поступить ещё более хитрым образом.



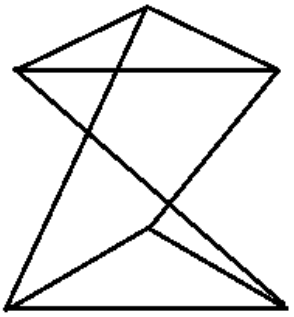
На каждый предмет поверхности строить свою подходящую точку, строить свой треугольник, строить кривые линии, но здесь есть некоторые неприятности. Всегда можно отрезать кусок от ломанной, но от ломанной на плоскости, а от ломанной на поверхности не всегда. Вопрос в том, какие линии проводить, так называемые геодезические линии, но это непростая задача.

А так, из ранее сказанного можно построить не очень хорошую сетку, так как это зависит от параметризации, но если она выбрана правильно, так что значения изменения параметров на параметрической плоскости не соответствуют большему изменению на поверхности, большему изменению точности, и наоборот.

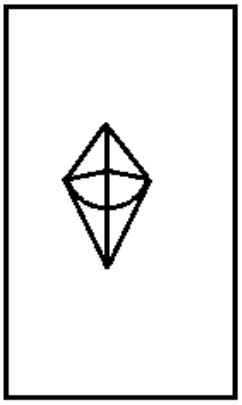
Замечание по поводу трёхмерного случая. Здесь мы можем построить тетраэдр.



Здесь также как и на поверхности не будет всегда находится подходящая точка.



Например, в скрученной призме можно доказать, что ни одного тетраэдра из него нельзя отрезать. Поэтому алгоритм будет затыкаться. В этом случае, если вдруг какая-то грань окажется без подходящей точки, мы её просто помечаем, что она не подходит и продолжаем действовать дальше. Обычно таких граней не так много. И тогда построенная театрализация, она будет встречаться мгновенно. Такие пустоты



где сетка не построена. Мы не можем это сделать. Эти пустоты нужно триангулировать используя триангуляции Делане. Там возникнут пересечения, так как мы будем триангулировать не всю призму, а выпуклую оболочку этих точек. Дальше мы вычисляем пересечения этих новых граней со старыми, построим дополнительные узлы и доведем реализацию до конца.