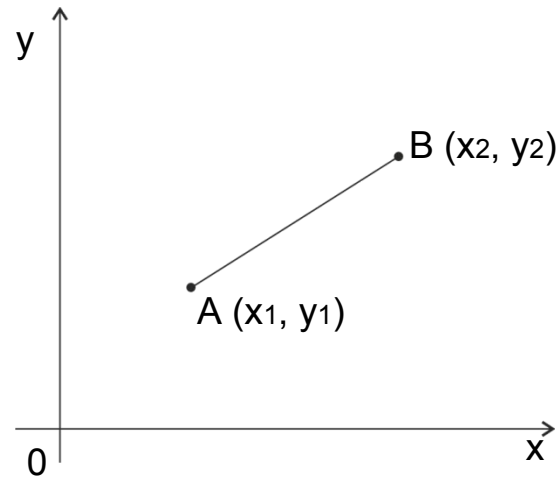


**АЛГОРИТМЫ И ФОРМУЛЫ
АНАЛИТИЧЕСКОЙ ГЕОМЕТРИИ**

На алгоритмах вычислительной (аналитической) геометрии основаны решения многих задач, являющихся стандартными функциями программ КГ вообще и геоинформационных систем в частности. Мы рассмотрим основные, наиболее употребимые формулы и алгоритмы. Начнем с тех, на которых основаны картометрические функции.

1. Расстояние между двумя точками.

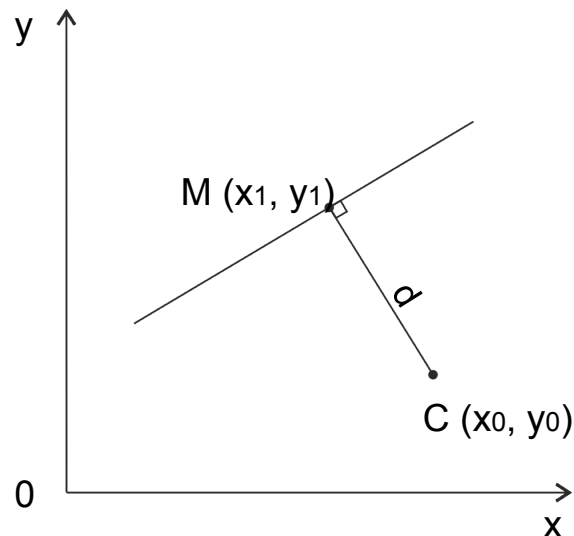


Как найти расстояние?

В прямоугольной системе координат оно вычисляется по теореме Пифагора:

$$S = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Расстояние от точки до прямой



По т. Пифагора
$$d = \sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2} \quad (1)$$

Координаты x_1 и y_1 находятся из решения системы уравнений:

$$\left[\begin{array}{l} A_x + B_y + C = 0 \\ A(y - y_0) - B(x - x_0) = 0 \end{array} \right. \text{ - ур-е прямой, проходящей через точку } M \text{ и перпендикулярной к исходной}$$

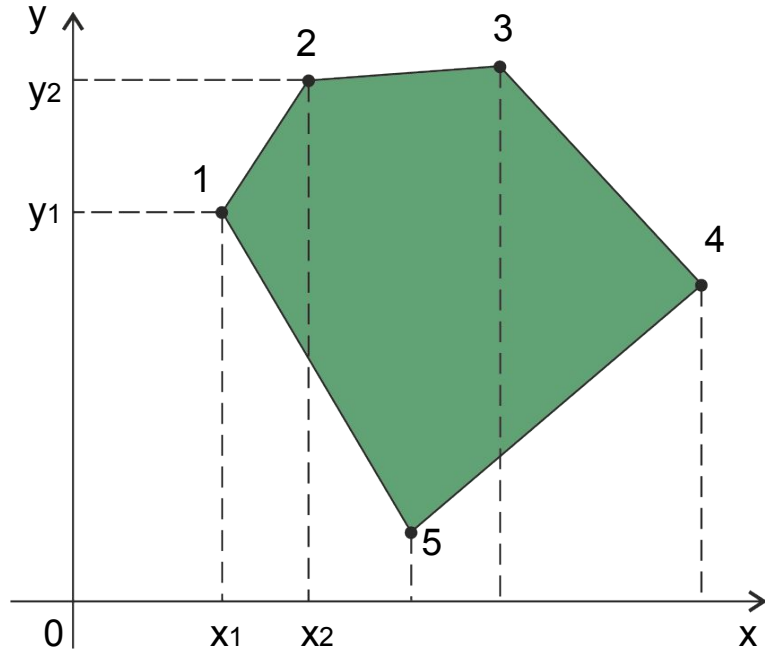
Решив систему и подставив в (1) получаем:

$$d = \left| \frac{A_{x_0} + B_{y_0} + C}{\sqrt{A^2 + B^2}} \right|$$

Площадь многоугольника

Вычисление площади многоугольника произвольной формы основано на аппроксимации простыми фигурами, чьи площади находятся по элементарным формулам.

Обычно используют аппроксимацию трапециями.



Чему равна площадь трапеции?

$$s = \frac{a + b}{2} h$$

$$S_{1,2,x_2,x_1} = \frac{(x_2 - x_1)(y_2 + y_1)}{2}$$

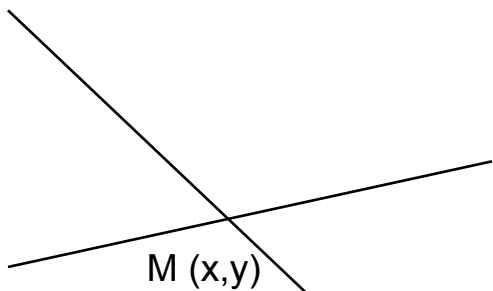
Двигаясь по часовой стрелке находят площади всех трапеций. При этом часть трапеций имеет отрицательную площадь (при $x_{i+1} < x_i$).

Площадь всего многоугольника равна сумме площадей всех получившихся трапеций:

$$S = \frac{1}{2} \sum (x_{i+1} - x_i)(y_{i+1} + y_i)$$

Формула справедлива при нумерации вершин по часовой стрелке.

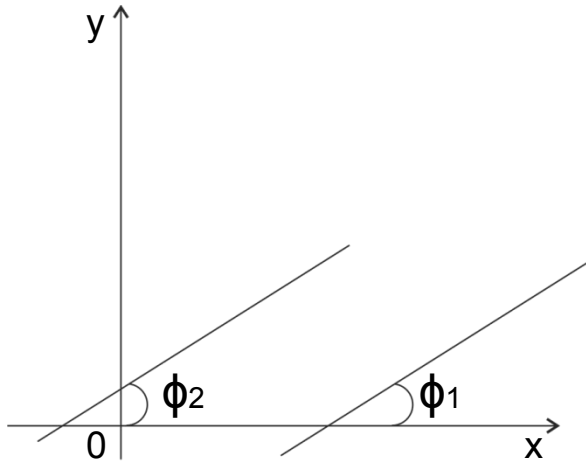
Нахождение точки пересечения двух прямых



$$\begin{cases} y = k_1x + b_1 \\ y = k_2x + b_2 \end{cases}$$

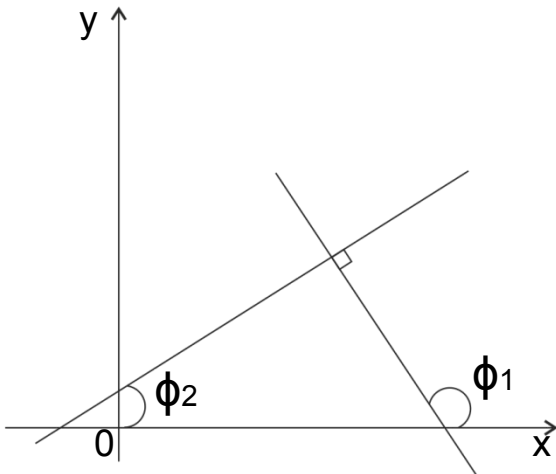
Для нахождения точки пересечения прямых надо решить систему линейных уравнений

Условие параллельности и перпендикулярности двух прямых



$$\begin{cases} y = k_1 x + b_1 \\ y = k_2 x + b_2 \end{cases}$$

Прямые \parallel при $\mathbf{tg \phi_1 = tg \phi_2}$ или $\mathbf{K_1 = K_2}$

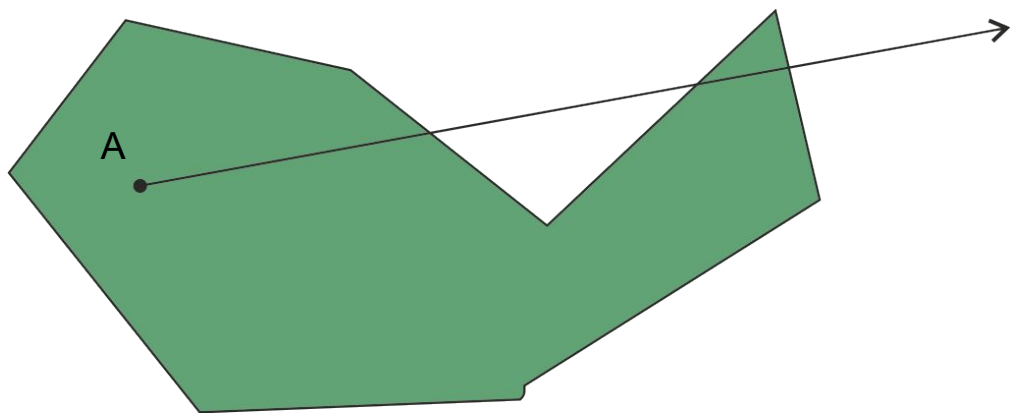


В случае \perp прямых $\phi_2 - \phi_1 = \pi/2$

$$\phi_2 = \phi_1 + \pi/2 \text{ или } \mathbf{tg \phi_2 = tg (\phi_1 + \pi/2) = -ctg \phi_1}$$

$$\mathbf{tg \phi_1 tg \phi_2 = -1} \text{ или } \mathbf{K_1 K_2 = -1}$$

Условие принадлежности точки полигону



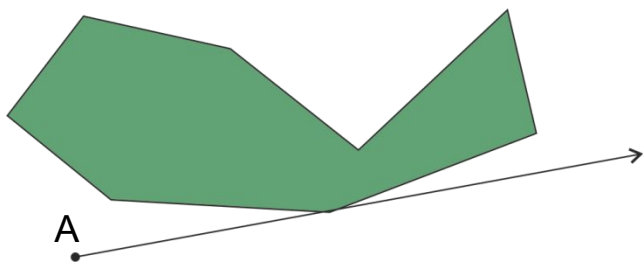
Находится точка внутри или снаружи?
Машина «не видит». Как определить?

Выпускается произвольный луч из точки A и считается количество пересечений:

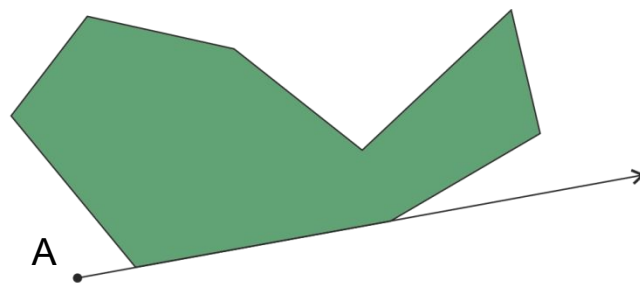
Если четное – снаружи

Если не четное - внутри

На практике возможны неоднозначные случаи:

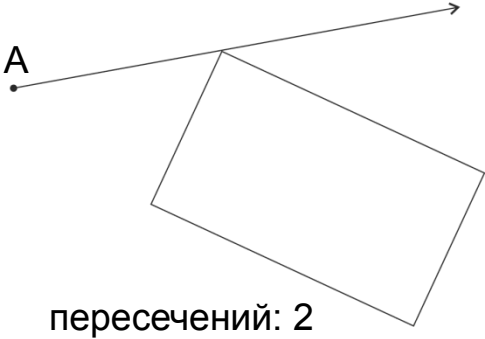
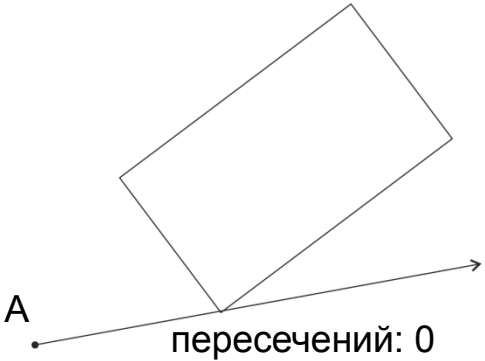
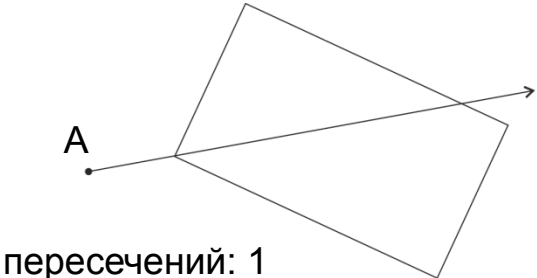
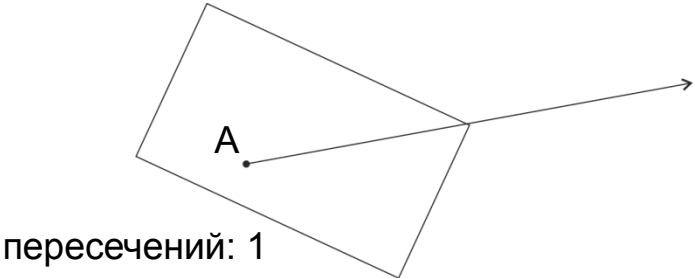


Луч попал в вершину



Луч попал в ребро

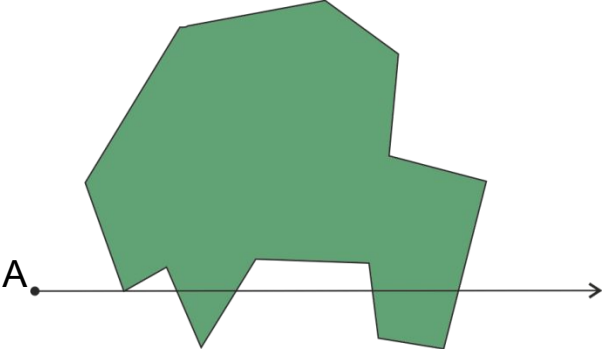
Правило: если луч попал в вершину, то засчитываются пересечения только тех ребер, для которых эта вершина является верхней.



Чтобы не разбираться с попаданием луча строго в ребро, выпускают не произвольный луч, а горизонтальный, а все горизонтальные ребра игнорируют.

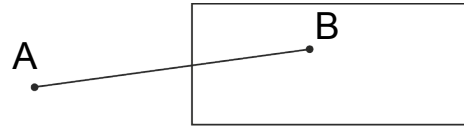


Сколько пересечений засчитывается?

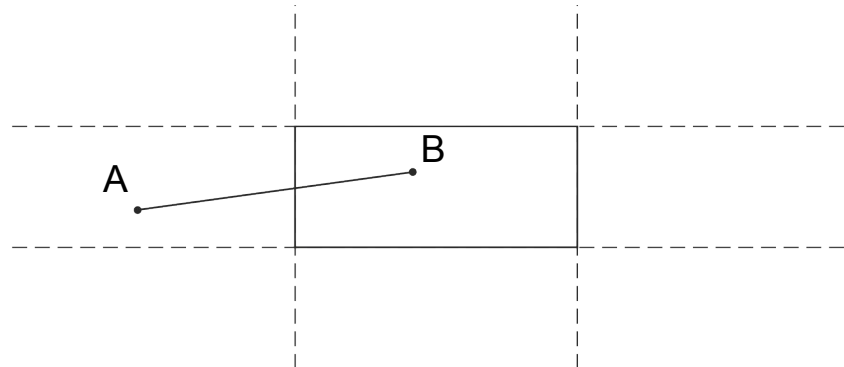


Отсечение отрезка. Алгоритм Сазерленда – Кохена (или Когана)

Простейший случай – часть отрезка отрезать
прямоугольником.



Проблема в том, что машина не видит пересекает ли сторона прямоугольника отрезок и если да, то какая.



Стороны прямоугольника продлеваются прямыми в бесконечность

Каждой получившейся части плоскости ставится в соответствие 4-битовый код:

0011	0010	0110
0001	0000	0100
1001	1000	1100

бит 0 – находится или нет левее прямоугольника
бит 1 – находится или нет выше прямоугольника
бит 2 – находится или нет правее прямоугольника
бит 3 – находится или нет ниже прямоугольника
Порядок: 3-й, 2-й, 1-й, 0-й.
1 – да, 0 – нет.

Алгоритм определяет код конечных точек отрезка.

Тривиальные случаи:

- а). если оба кода = 0000, то отрезок полностью находится в прямоугольнике;
- б). если один = 0000, а другой нет, то по коду определяется секущая сторона и находится точка пересечения;
- в). если битовое **И** обоих кодов не равно нулю, то отрезок не пересекает прямоугольник (так как это значит, что обе конечные точки отрезка находятся с одной стороны прямоугольника).

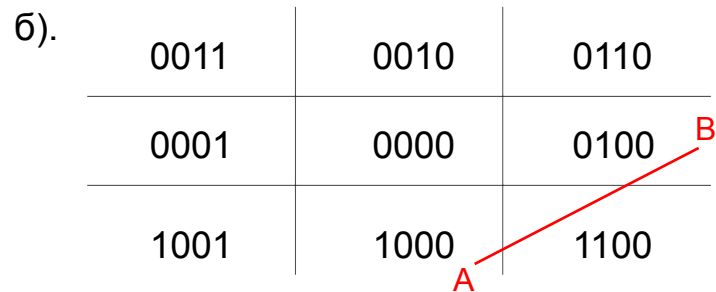
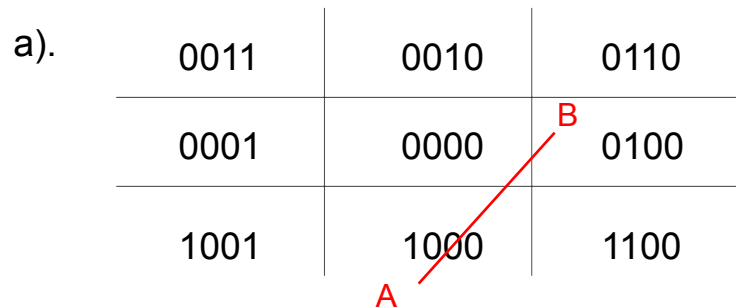
0011 B	0010	0110
0001	0000	0100
A 1001	1000	1100

A = 1001

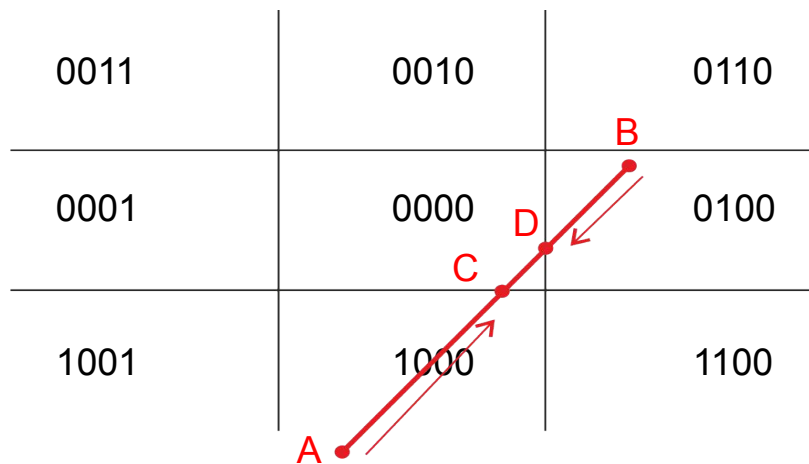
B = 0011

битовое И обоих кодов $\neq 0$

Не тривиальные случаи:



Алгоритм выбирает конечную точку, находящуюся вне прямоугольника, находит ближайшую к ней точку пересечения отрезка с одной из линий, образующей стороны прямоугольника, и использует эту точку пересечения как новую конечную точку отрезка. Оставшийся отрезок снова пропускается через алгоритм.

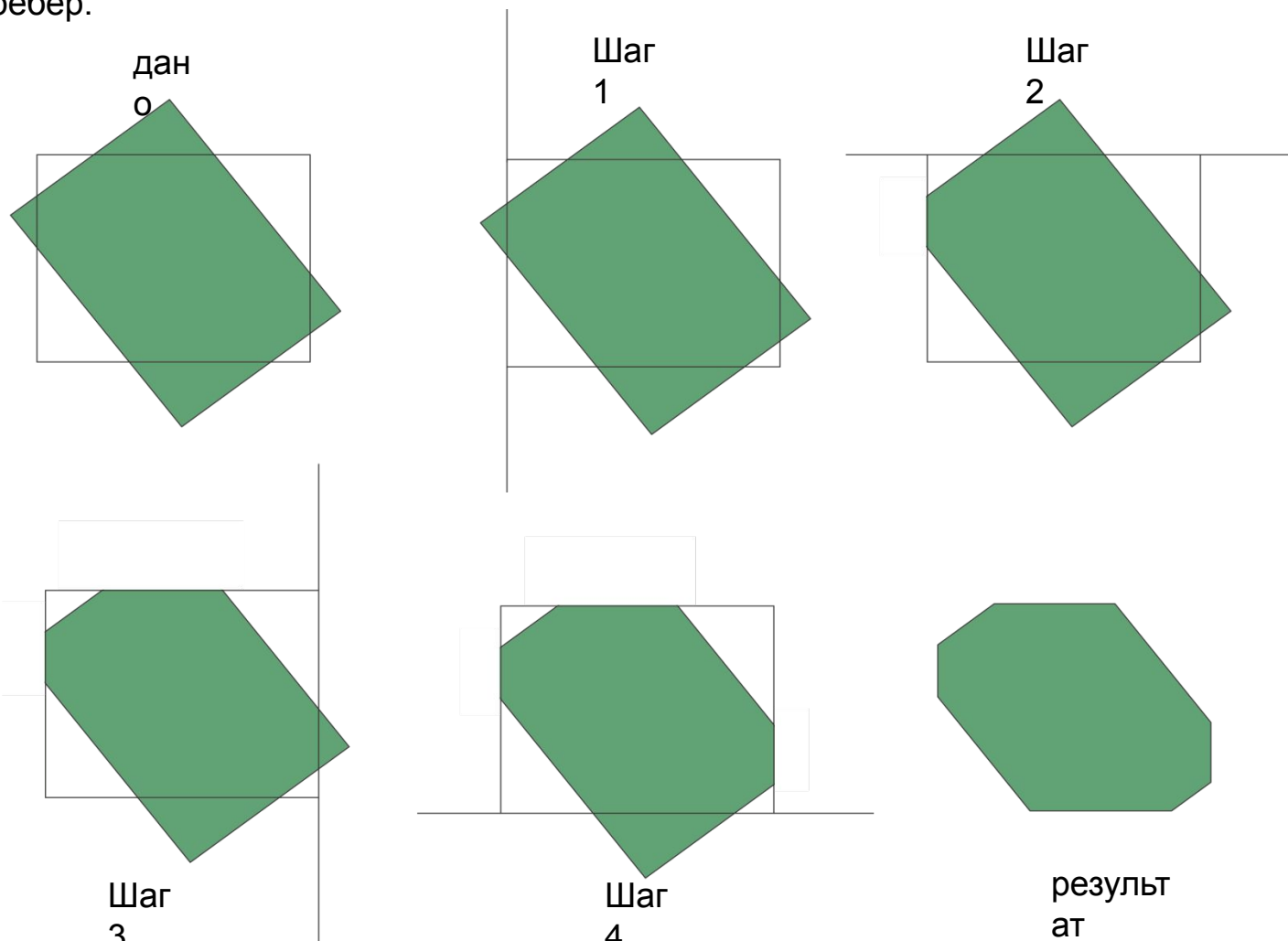


Коды концов отрезка CD = 0000, т.е. это часть исходного отрезка, попавшая внутрь прямоугольника.

В случае б). действия аналогичные, но поскольку коды ни одного из концов $\neq 0000$, отсечения нет.

Пересечение двух полигонов. Алгоритм Сазерленда – Ходжмана.

Алгоритм разбивает задачу на ряд более простых задач об отсечении полигона прямой, пересекающей одно из ребер.



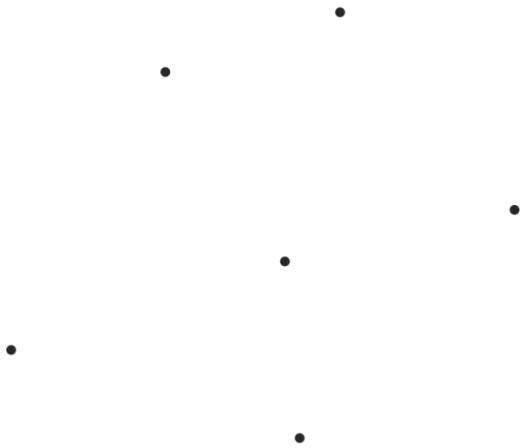
На каждом шаге выбирается очередное ребро отсекающего полигона и проверяется положение всех вершин относительно прямой, проходящей через это ребро. При этом в полученный многоугольник добавляется 0, 1 или 2 вершины.

Триангуляция Делоне

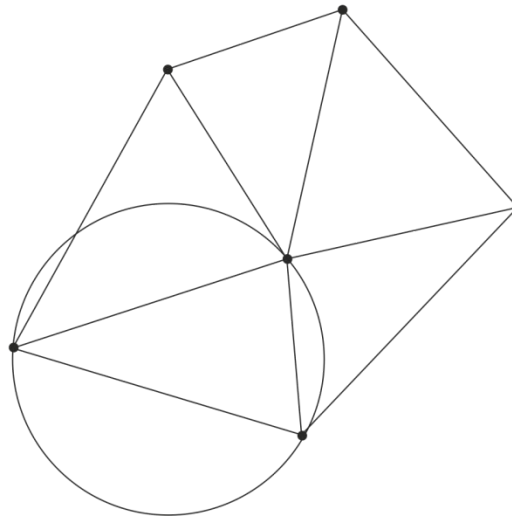
Названа так в честь русского математика Бориса Николаевича Делоне.

Является наиболее сбалансированной (или наименее вырожденной) из всех возможных на данном наборе точек.

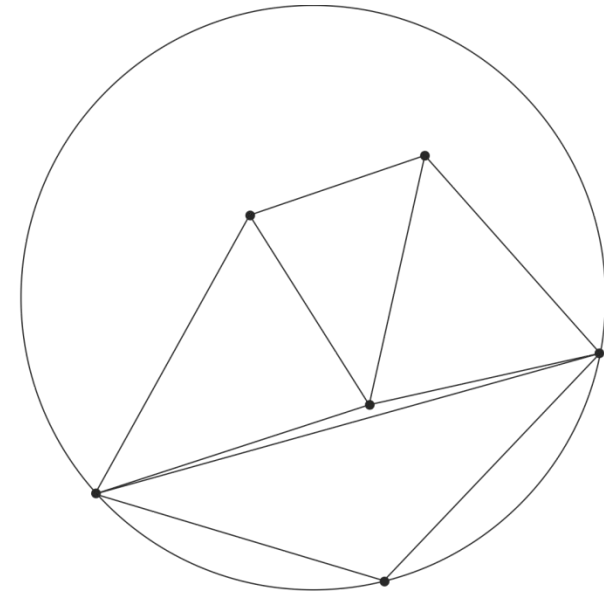
Триангуляцией Делоне называется триангуляция набора точек S , если окружность, описанная вокруг каждого из треугольников, не будет содержать внутри себя точек набора S .



Набор точек S

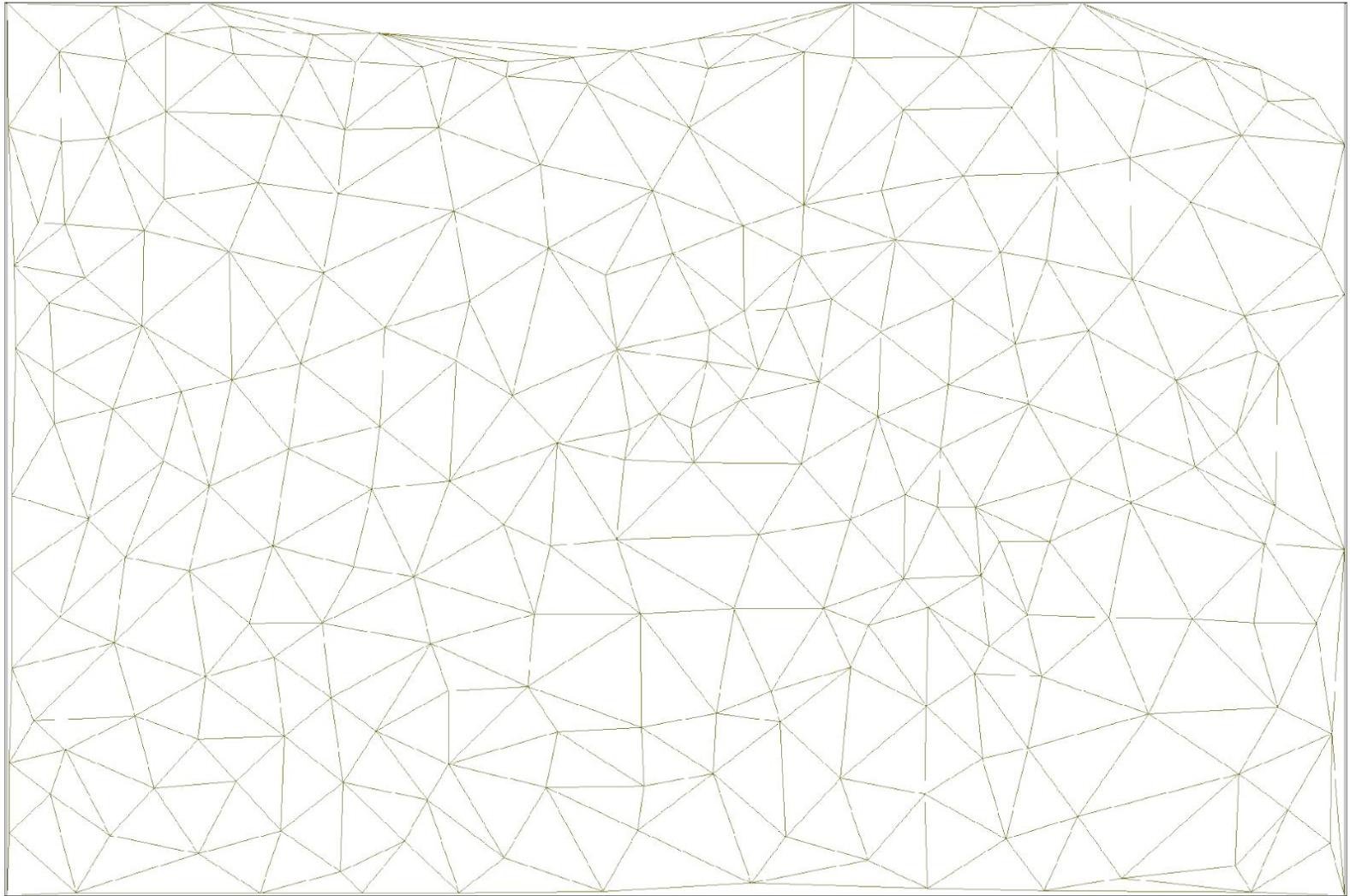


Триангуляция Делоне



Не триангуляция Делоне

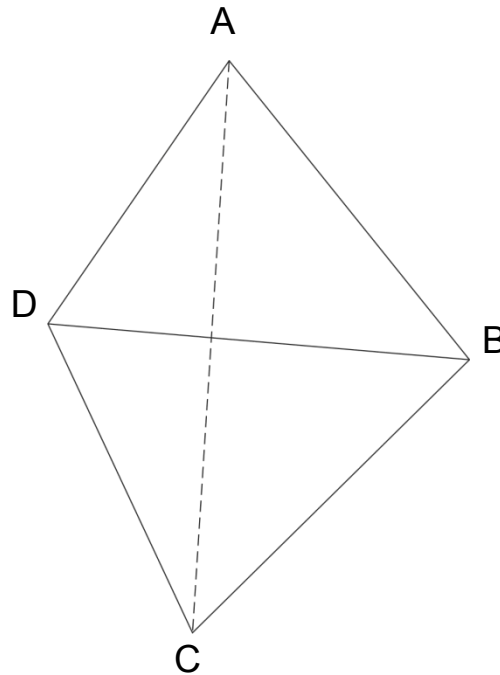
Триангуляция Делоне



Есть множество алгоритмов построения, они делятся на декрементные и инкрементные.

Декрементные – строят непосредственно сразу триангуляцию Делоне. Они не эффективны при большом количестве точек (много операций – много времени).

Инкрементные – базируются на перестройке существующей триангуляции флипами. Флипом называется операция переброски диагонали выпуклого четырехугольника.



Флип четырехугольника: диагональ AC заменяется на BD.

Общая схема инкрементных алгоритмов:

1. Сначала строится произвольная триангуляция
2. Для каждой точки последовательно выбирается гнездо треугольников, которые имеют эту точку в качестве вершины
3. Для каждого треугольника гнезда последовательно: по или против часовой стрелки ищется сопряженный треугольник
4. Каждая полученная пара треугольников проверяется на соответствие триангуляции Делоне. Если соответствует – помечается как проверенная и двигается дальше, если нет, то в 4-угольнике, образованном парой сопряженных треугольников делается флип с образованием двух новых.

Алгоритм работает до тех пор, пока все треугольники не будут удовлетворять условиям триангуляции Делоне.

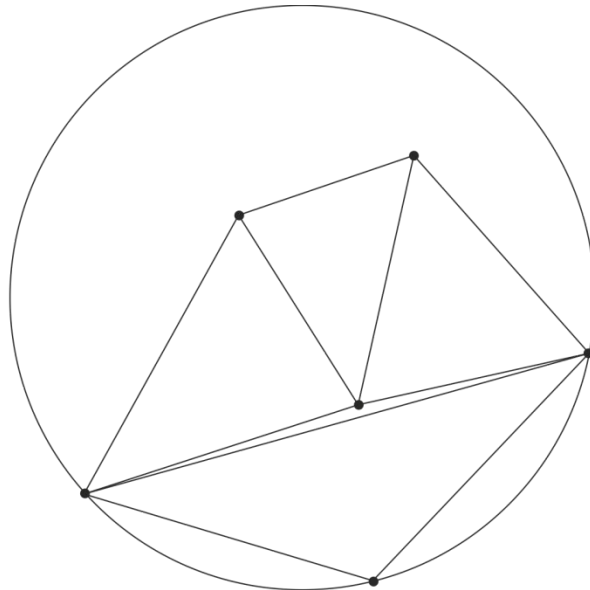


Диаграмма Вороного

Названа так в честь русского математика Георгия Феодосьевича Вороного. Также в литературе может называться: полигоны Тиссена, ячейки Дирихле или области Дирихле-Вороного, зоны Вигнера-Зейтца.

Диаграмма Вороного – геометрическая конструкция, образованная относительно исходных точек таким образом, что границы полигонов являются отрезками срединных перпендикуляров, восстановленных к сторонам треугольников триангуляции Делоне.

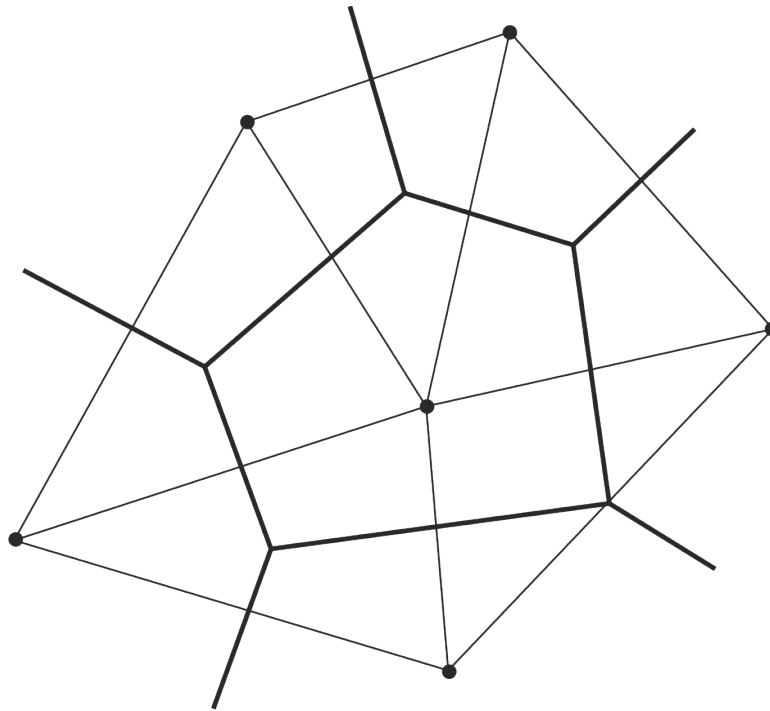
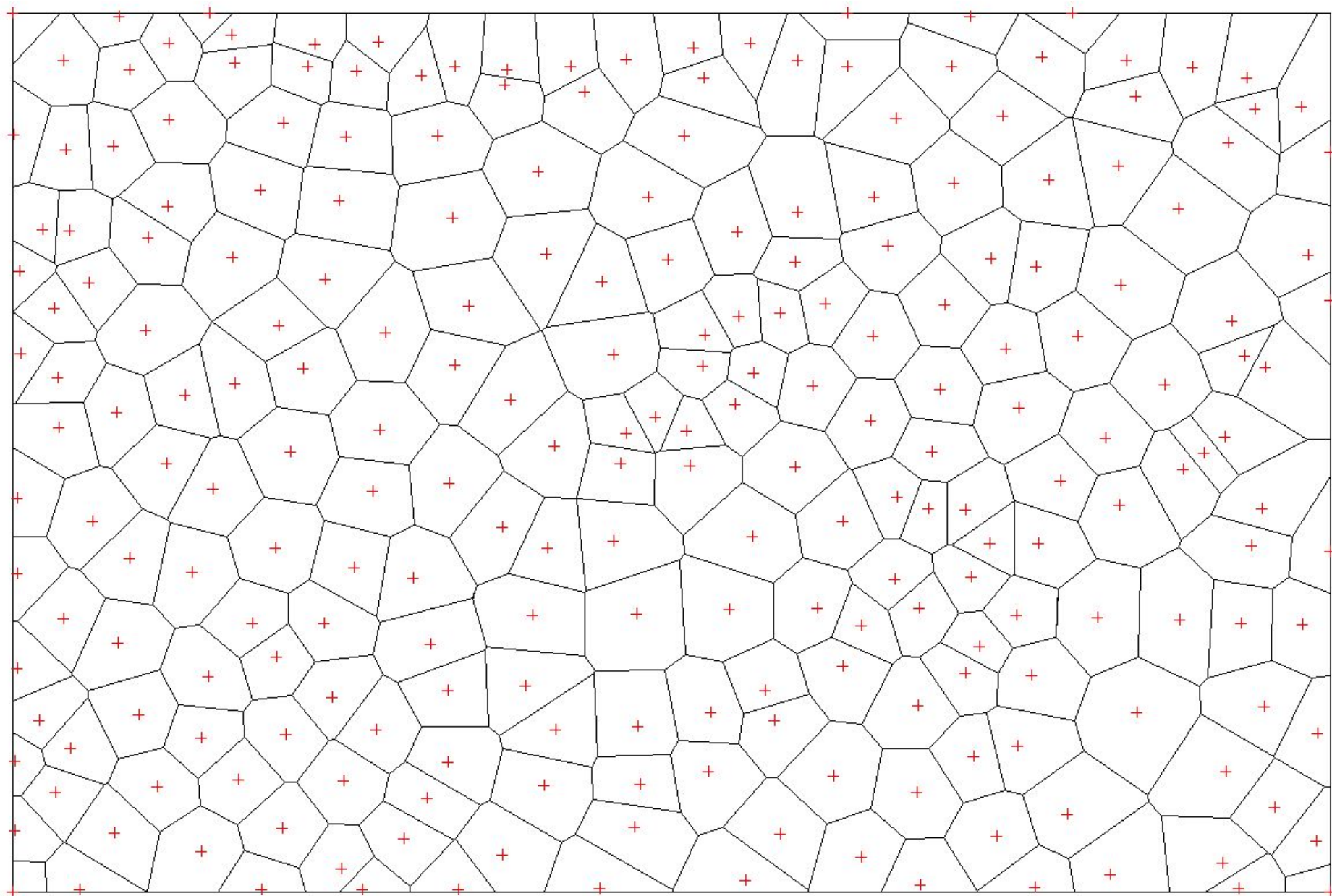
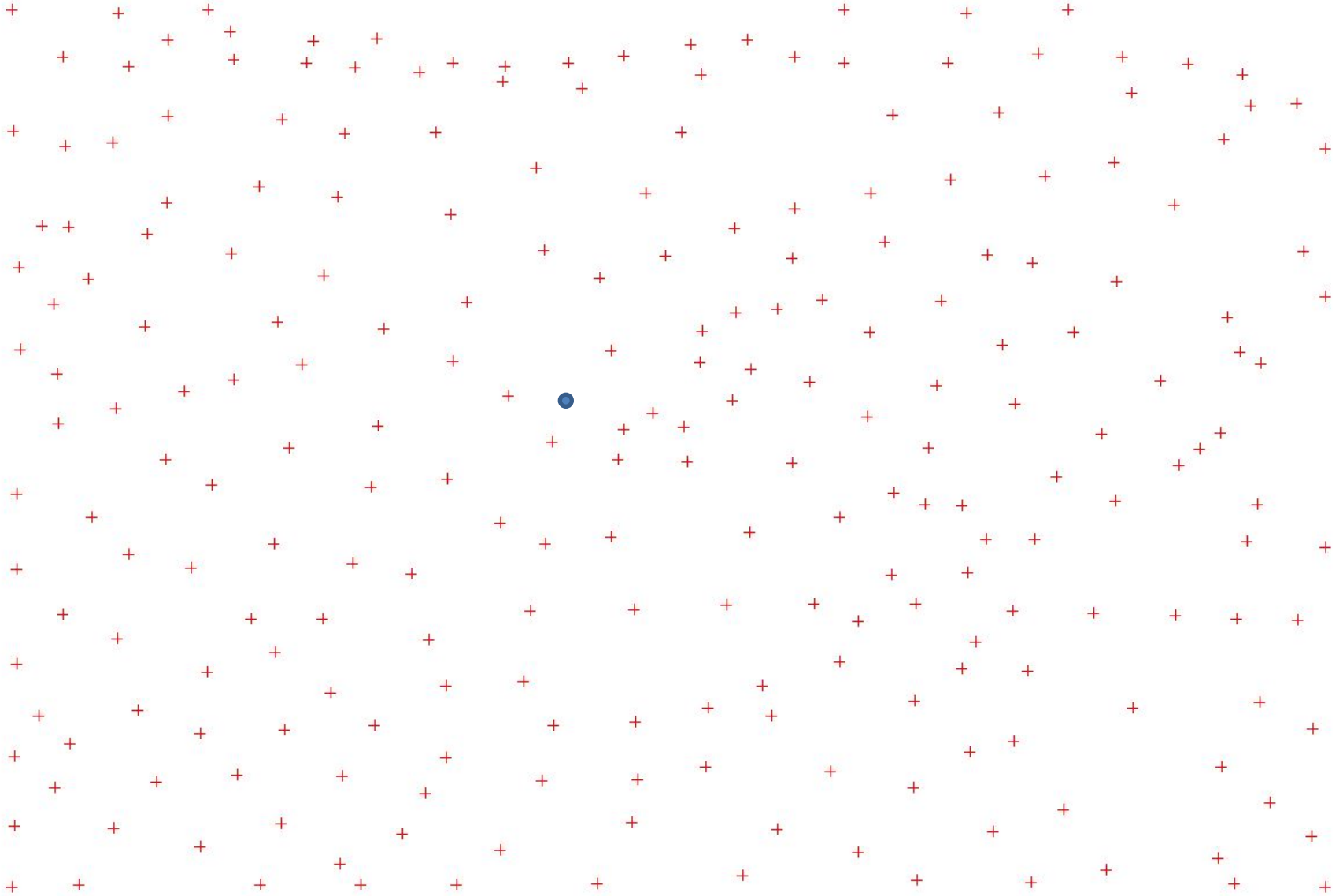


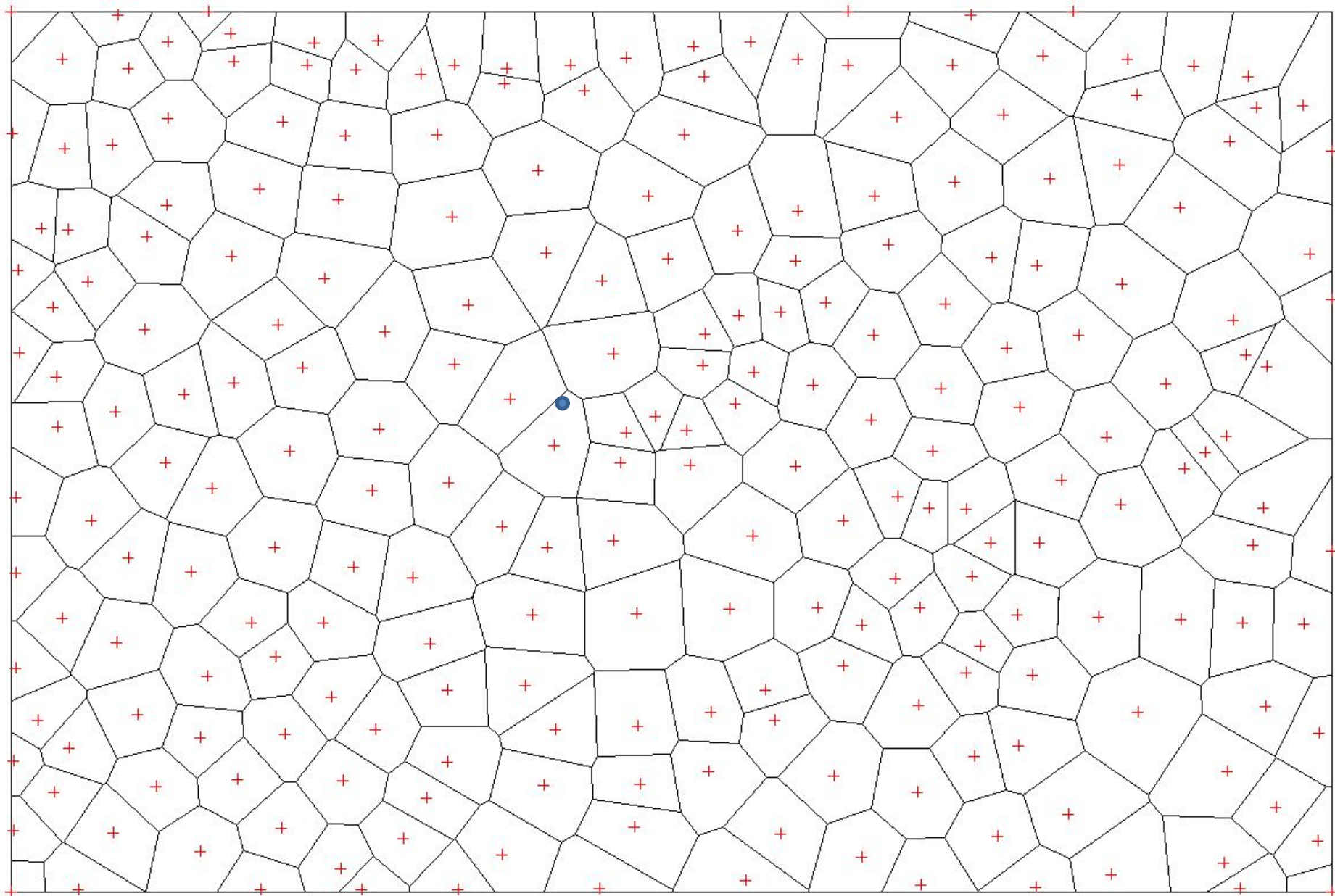
Диаграмма Вороного



Области применения:

Основное использование – построение зон тяготения, например, так называемая «задача почтовых отделений»





Другая область применения – использование весовых диаграмм.