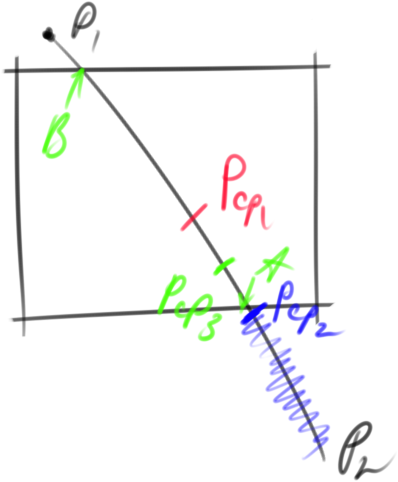
23.04.14-----------------------------------------------------------------------

**Алгоритм отсечения средней точкой**

Основное отличие – точка пересечения находится не аналитически. Уравнение, фактически, решается численно – методом деления отрезка пополам. Деление на 2 – сдвиг, выполняется достаточно быстро, посему алгоритм имеет право на существование. Точность или . Нахождение точек пересечения отрезка со стороной отсекателя.



В качестве текущей всегда рассматриваем первую точку и ищем точку пересечения отрезка со стороной отсекателя. Точка пересечения A – наиболее удалённая от вершины Р1, но ещё видимая. Находим середину отрезка Р1Р2 (Рср1) и проверяем видимость\невидимость от-резка Рс1Р2. В данном случае отрезок не является невидимым – ищем его середину Рср2. Отрезок Рср2Р2 – невидимый, отбрасываем его; наиболее удалённая видимая от Р1 точка расположена на отрезке Рср1Рср2. Ищем нужную точку методом половинного деления, с точностью . Можно проверять, не стала ли одна из проекций отрезка (постоянно уменьшающегося) <=1. (возможно – нужно проверять что ОБЕ стали <=1, у нас ведь может быть ситуация когда dy=1 и dx=100 для почти параллельного отрезка..)

В общем, процесс деления повторяется пока не окажется что |P1-P2|<=eps. После нахождения А, аналогично находится Б. В данном случае, при половинном делении мы ориентируемся не на значение функции, а на приснопамятный код точки, .

Алгоритм:

1. ввод исходных данных Хл, Хпр, Ун, Ув, Р1, Р2

2. вычисление кодов концов отрезка, Т1 и Т2

3. вычисление сумм кодов концов С1 и С2

4. проверка отрезка на полную видимость, С1=0 и С2=0; если видимый - высвечиваем и заканчиваем

5. если не полностьювидимый - проверка на невидимость; если невидимый - заканчиваем

если же частично видимый - пошло-поехало

6. запоминание Р1 (Т=Р1)

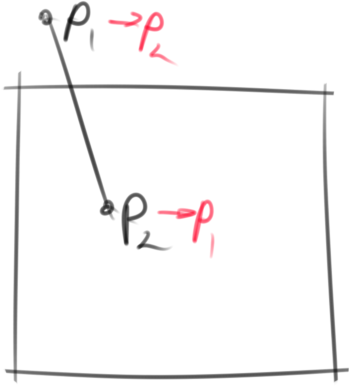
7. определить номер искомой точки пересечения i=1

8. если i>2, то анализируем полученный отрезок на невидимость (произведение кодов концов, если равно нулю - то видим)

9. проверка S2 = 0. Если S2=0, то Res[i] = P2;

i+=1

Р1 = P2; P2 = T



10. если S2 != 0 , то ищется точка пересечения:

10.1. если |P1-P2|>E то

ищется средняя точка Рср = (Р1+Р2)/2

Pm = P1

P1 = Рср

определение невидимости Р1Р2, предварительно вычислить код Р1

если Р1Р2 невидим, то Р1 = Pm (вернуть Р1 на прежнее место)

P2 = Рср

(иначе работа продолжается с Р1Р2)

Иначе, если же |P1-P2|<=E, то

Res[i] = P2

P1 = P2

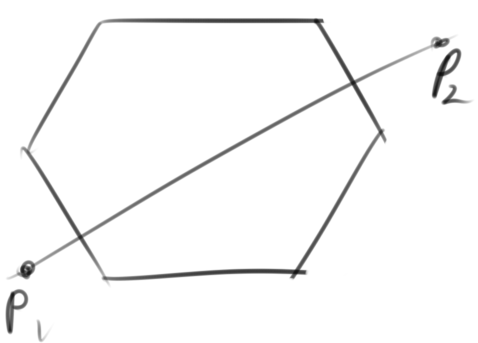
P2 = T

i+=1

переход к 8.

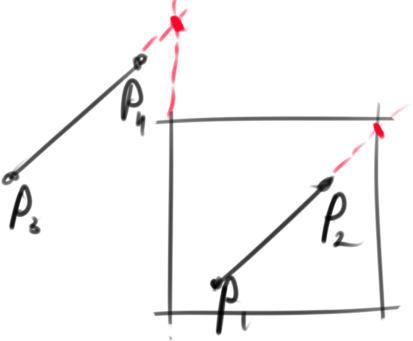
**Отсечение отрезка произвольным выпуклым отсекателем**

Алгоритм Кируса-Бека.



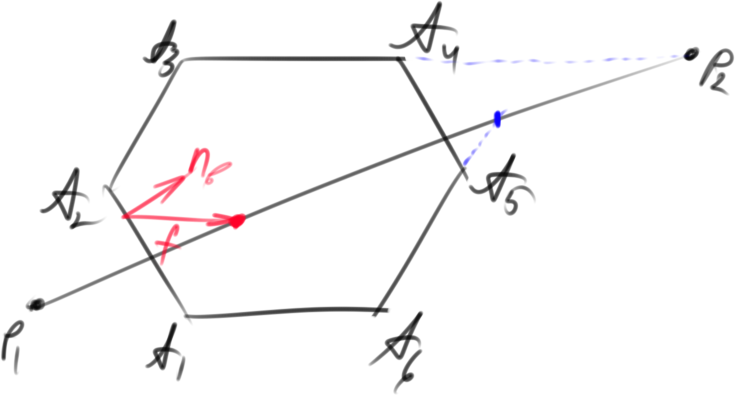
*.* Здесь t – параметр для параметрической формы задания отрезка. Де факто, имеем два уравнения: y(t) = y1 +…, x(t) = x1 + ….

При попытке найти точки пересечения для полностьювидимого отрезка, можно получить точку, находящуюся за пределами отсекателя – не проходит проверка на корректность. Раньше мы видимые отрезки отсеивали, а здесь нет. Для полностью невидимого отрезка можно получить аналогичную картину.



Простыми средствами отсеять эти отрезки невозможно, если полная видимость\невидимость и обнаруживается, то только в процессе работы.

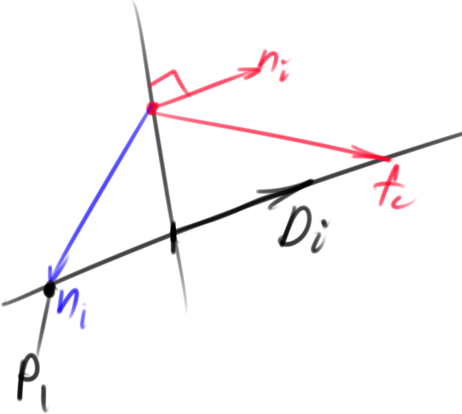
Прямая пересекает выпуклый многоугольник только в двух точках.



Используем скалярное произведение векторов – вектора нормали к стороне, и вектора от точки на ребре к точке на отрезке (красные).

*. //fi –* просматриваем очередную i-ю сторону. В качестве произвольной f удобно задавать вершину отсекателя.

Вектор директрисы, направления отрезка - ;

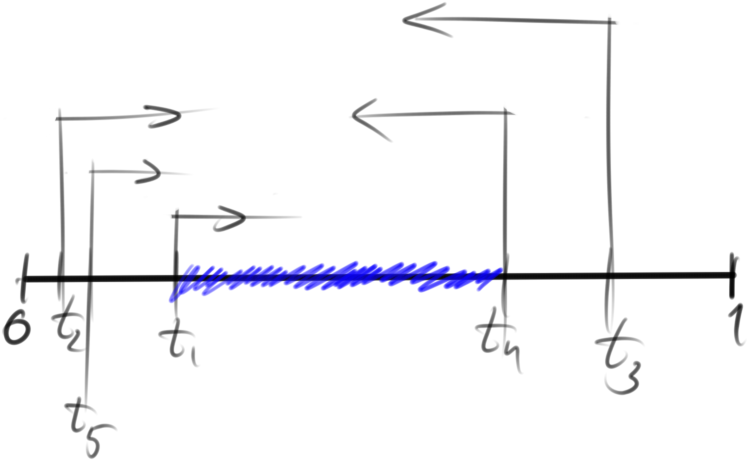
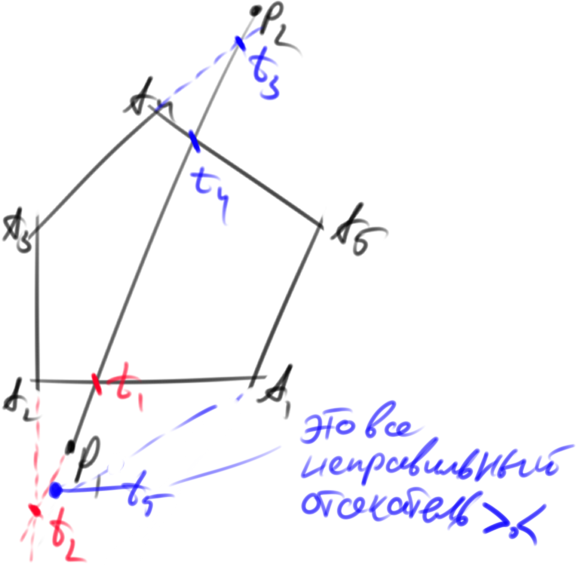


Имеем сумму скаляров: , отсюда . Анализируем знаменатель на равенство 0: может быть Д=0 – отрезок вырождается в точку, которую анализируем на видимость. Знаменатель = 0, когда нормаль перпендикулярна директрисе, а значит сам отрезок паралеллен стороне отсекателя.

Видимость точки Р1 определяется с помощью скалярного произведения вектора внутренней нормали n на вектор первой вершины W, . <0 – Р1 невидима, >0 – видима, =0 – лежит на границе.

*–* отрезок параллелен стороне отсекателя, нужно проверить его видимость. Отрезок может лежать по видимую или по невидимую сторону. Сделать это можно, проверив видимость любой его точки, например Р1 (спомощью nW). Если относительно текущей границы отрезок видим, то переходим к следующему шагу – ищем точки пересечения отрезка с рёбрами. Если же отрезок невидим отноеительно текущей границы, то он невидим и относительно остального многоугольника.

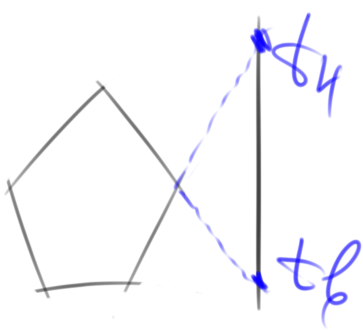
Вообще, отрезок может пересекаться с продолжениями всех сторон отсекателя, а нам надо выбрать всего две точки.



Имеем .

Т1, Т2, Т5 – расположены ближе к началу отрезка, Т3,Т4 – ближе к концу. Среди точек пересечения, одни лежат ближе к началу отрезка, другие ближе к концу, достаточно очевидный факт. Из «начальных» точек надо выбрать максимальное значение – нижняя граница видимости. Из «конечных» - минимальное, нижняя граница.  
 – точки, расположенные ближе к началу отрезка, аналогично .

Тем не менее существует частный случай, когда отрезок не может быть распознан как невидимый.



Нужна проверка на то что tн <= tв – если нет, то отрезок заведомо невидим.

1. начало

2. ввод Р1,Р2,n вершин, A[n] массив вершин

//анализ н-угольника на выпуклость

3. вычисление D = (P2-P1)\bar

4. изначально считаем что весь отрезок видимый, tн=0, tв=1

5. цикл отсечения по всем сторонам отсекателя, i=1..n

6. вычислить нормаль n\_внi; задание точки f\_i (можно взять очередную вершину)

7. вычислить вектор Wi = (P1-fi)\bar

8. вычисление скалярных произведений Dsk=nD и Wsk=nWi

9. если Dsk==0, то //вектор параллелен стороне

10. если Wsk>0, то

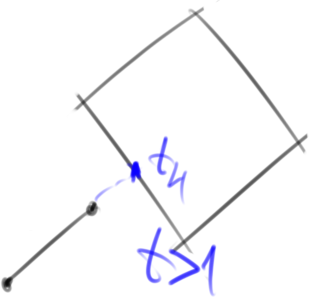
11. i++; continue;

иначе //отрезок невидим

12. конец

иначе //можно поделить

13. t = -Wsk / Dsk

14. если Dsk > 0 //проверяем расположение - "видимая часть отрезка" начинается за границей пересечения 

15. если t>1, то //точка пересечения относится к нижней границе, но при этом парамтр >1

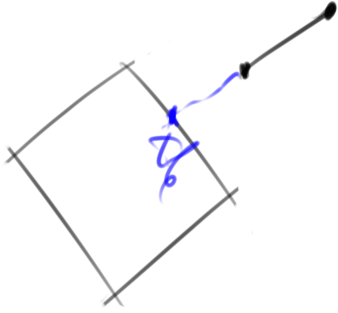
гото 12 //отрезок невидимый

иначе

16. tn = max(tn, ti);

гото 11

иначе

17. если t<0, то //"видимая часть отрезка" за границей 

гото 12 //отрезок невидимый

иначе

18. tв = min(tв, ti)

гото 11

19. если tН <= tВ, то

20. отрисовать отрезок P(tн) до P(tв)

иначе

гото 12 //отрезок невидимый