平面几何

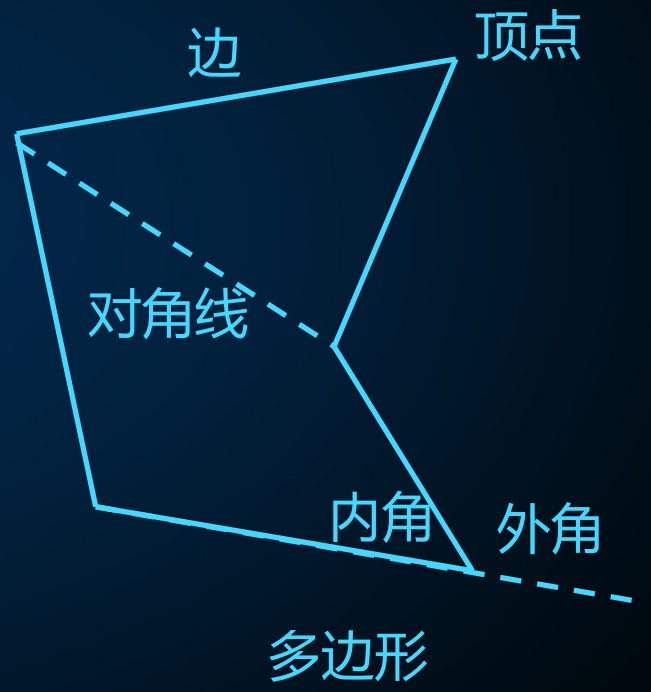
应用

[1] 凸包

1 凸包

多边形：由至少**三条**线段**首尾相接**组成的图形

- 线段被称为**边**，线段的**公共端点**被称为**顶点**。
- 两条**邻边**形成的角被称为**内角**，
  内角的一边与另一边**反向延长线**组成的角被称为**外角**。
- n边形的内角和为(n - 2)π，外角和为2π。



1 凸包

简单多边形：没有**自交**的多边形

- 顶点不重合。
- 顶点不在边上。
- 边与边只会在顶点处相交。

简单多边形

非简单多边形

1 凸包

正多边形：所有边都**等长**的多边形

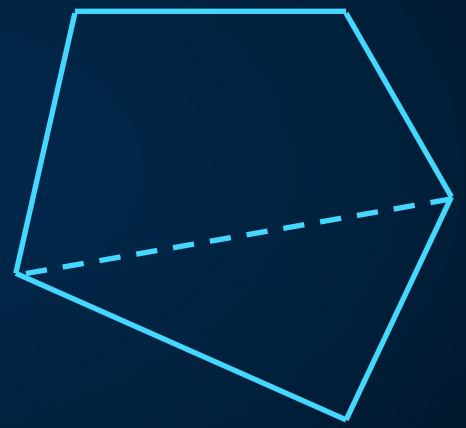
- 所有内角都**相等**。
- 一定是简单多边形。
- 随着边数增加，正多边形**无限**接近**圆形**。

正五边形

1 凸包

凸多边形：所有内角均**小于**π的简单多边形

- 任意**对角线**均在凸多边形的内部。
- 正多边形和三角形都是凸多边形。

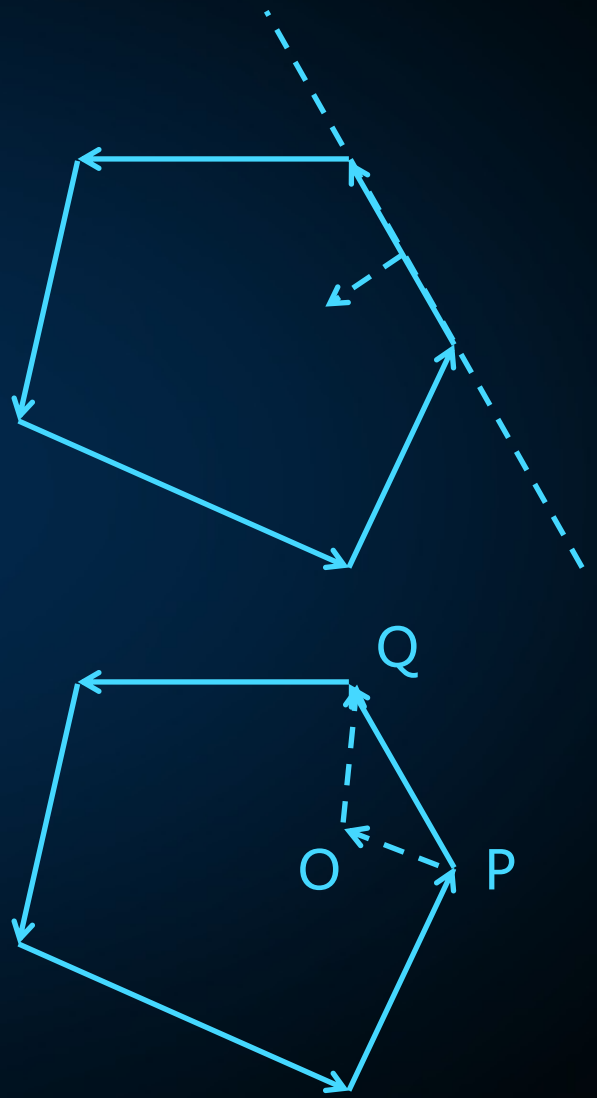


凸多边形

1 凸包

凸多边形特性

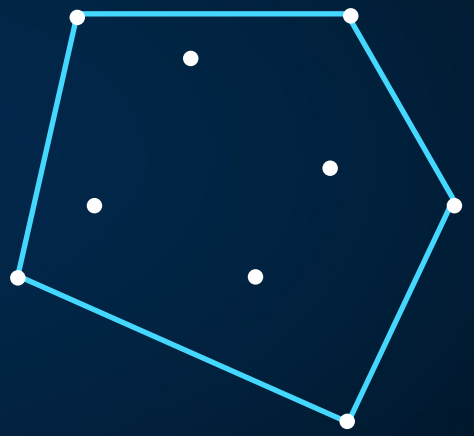
- 所有顶点均在任意边所在直线的**同侧**。
- 如果按照**逆时针**顺序查看边，则边只会不断**左旋**，所有顶点均在任意边的**左侧**。
- 对于凸多边形**内**任意点O，与凸包上相邻点PQ，必然有$\overrightarrow{PO}$和$\overrightarrow{OQ}$构成**右旋**。



1 凸包

凸包：包含**所有**点的**最小**凸多边形

- 是点集中**最外层**点连接起来的图形。
- 可以想象成包围所有点的橡皮筋。



凸包

1 凸包

【例1】凸包模板

给定平面上的n个点，求凸包的周长。

先求出凸包，再计算周长。

https://www.luogu.com.cn/problem/P2742

1 凸包

求解凸包的步骤

- 先找到凸包上的**任意**一点。
- 再由该点开始**依次**找到凸包的每条边。
- 可以根据凸包的性质来求解。

1 凸包

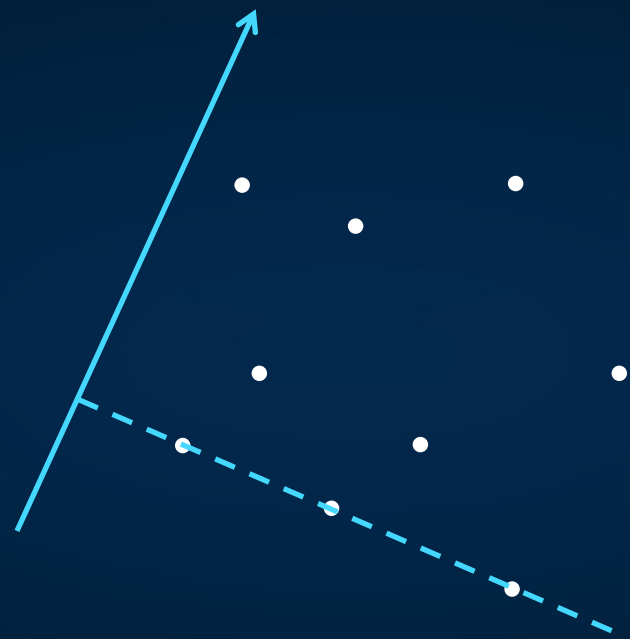
找到凸包上任意点

方法一：任选**参照点**，离参照点**最远**的点肯定在凸包上。

同时有多个点距离最远时，选择**任意**一个。

1 凸包

方法二：任选**参照方向**，在该方向上**最小**的点肯定在凸包上。



有多个点最远时，选择**垂直**方向**最小**或者**最大**的，
避免选到凸包**边上**的点。

1 凸包

哪种方法更简单？

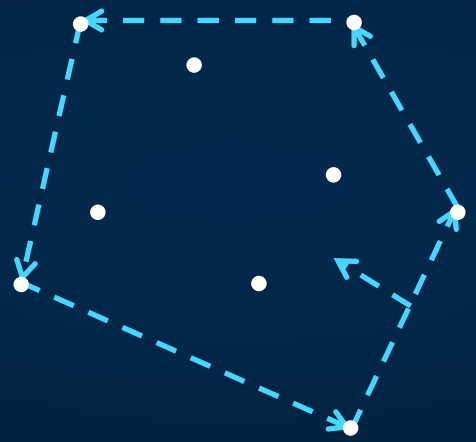
方法二更简单，可以选择**x轴正方向**。

此时x最小，其次y最小的点必定在凸包上。

1 凸包

寻找凸包的边

不妨按照**逆时针**的顺序寻找凸包的边。

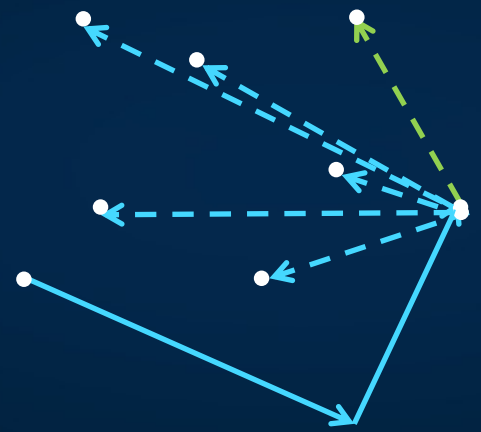


逆时针寻找

此时所有点都在任意边的**左侧**。

1 凸包

对于凸包上任意点，
它与下个点组成的向量一定是最**靠右**的。



可以**枚举**寻找最靠右的向量。

分析上述方法的时间复杂度

- 【T】寻找起始点O(n)。
- 【T】寻找邻点O(n)，凸包上的点数O(n)，总共$O(n^2)$。
- 【T】总时间复杂度为**$O(n^2)$**。

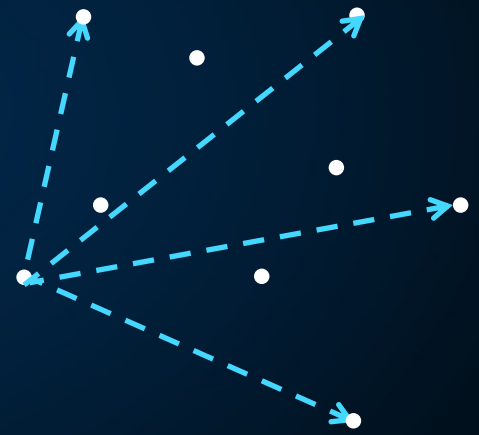
上述算法有什么缺点？

没有考虑点之间的**关系**，导致每次都需要枚举寻找。

1 凸包

凸包上点的顺序

- 从起点按照**逆时针**顺序依次向凸包上的点做向量，
这些向量也是不断**左旋**的。



- 将所有点按照**极角序**排序，
则凸包上的点必然按照这个顺序依次出现。

- 如果有**共线**的点，应该**较近点**靠前，
因为旋转角范围为[0, π)。

1 凸包

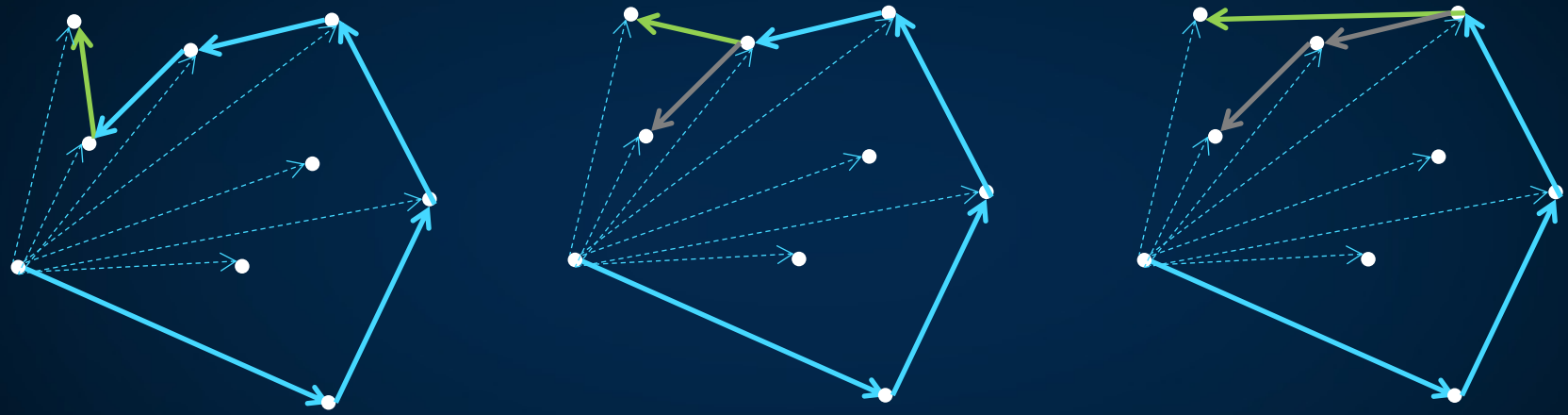
按照点的顺序寻找凸包的边

- 每次加入新点前，**不断**对**新边**与**尾边**进行判断。

- 如果两条边**不**构成**左旋**，则说明尾点在凸包**内部**或者**边上**，应当删除。

1 凸包

可能会需要连续删除多个点。



直到构成左旋时再加入新点。

分析改进方法的时间复杂度

- 【T】寻找起始点O(n)。
- 【T】对点进行排序O(n log n)。
- 【T】按顺序选点、删点O(n)。
- 【T】总时间复杂度为**O(n log n)**。

1 凸包

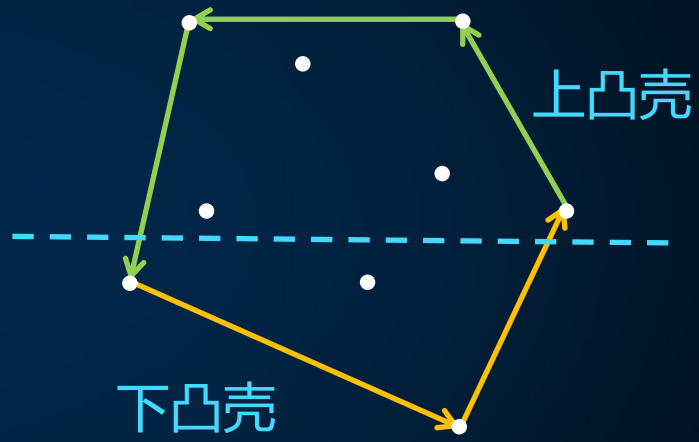
改进的关键在于确定点的顺序

- 如果更换点的顺序，也可以得到相似的做法。
- 不局限于极角序，可以让凸包的性质有更广泛的应用。

1 凸包

凸包的**拆分**

- 选择一个**方向**，可以将凸包拆分成**两半**，半边凸包被称为**凸壳**。
- 通常选择**水平**方向，将凸包拆分成**上凸壳**和**下凸壳**。
- 按照**逆时针**顺序访问凸包的点，下凸壳中的点x坐标是**递增**的，上凸壳中的点x坐标是**递减**的。



1 凸包

应用

- **垂直**边理论上不属于任意凸壳，
  但此时的凸包是不完整的。

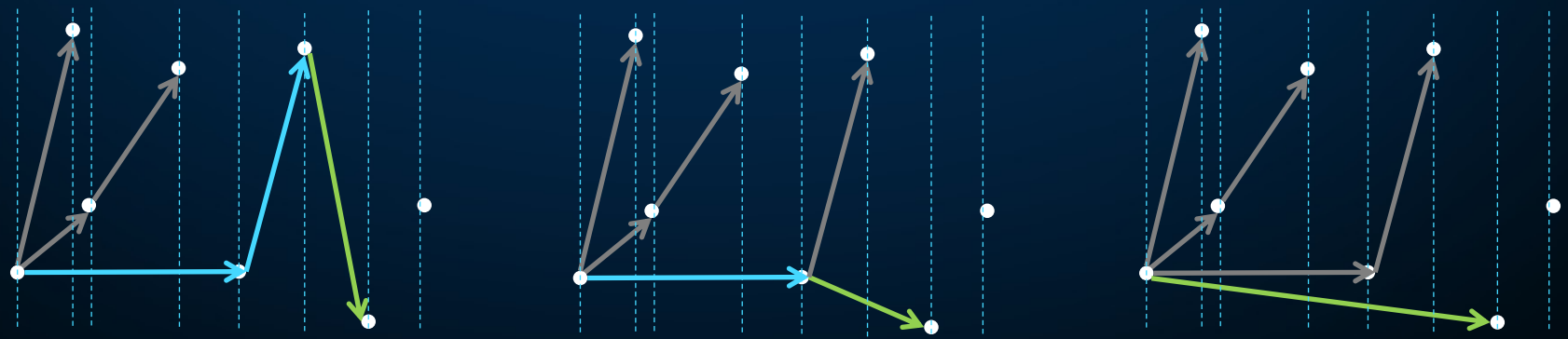
- 求解时可以认为**左**垂直边属于**上**凸壳，
  右垂直边属于**下**凸壳。

- x坐标最**小**，其次y坐标最**小**的点是下凸壳的起点，
  x坐标最**大**，其次y坐标最**大**的点是上凸壳的起点。

1 凸包

直角坐标序求凸包

- 将所有点按照x坐标由小到大，其次y坐标由小到大排序。
- **顺序**寻找凸包的边，可以得到下凸壳。
  逆序寻找凸包的边，可以得到上凸壳。
- 加入点的规则不变，要求每条边都是左旋。



1 凸包

对比两种序求凸包

- 排序方面：
  极角序需要自己实现比较器；
  直角坐标序可以借用STL。

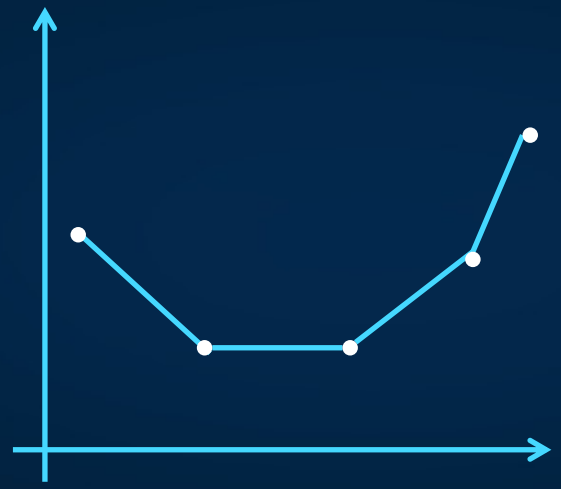
- 寻找起点方面：
  极角序需要寻找起点；
  直角坐标序排序后即可直接得到获得起点。

- 求解流程方面：
  极角序一次性得到凸包；
  直角坐标序分别得到上下凸壳。

- 总体来说差别不大。

1 凸包

凸壳上边的**斜率**是**单调**的



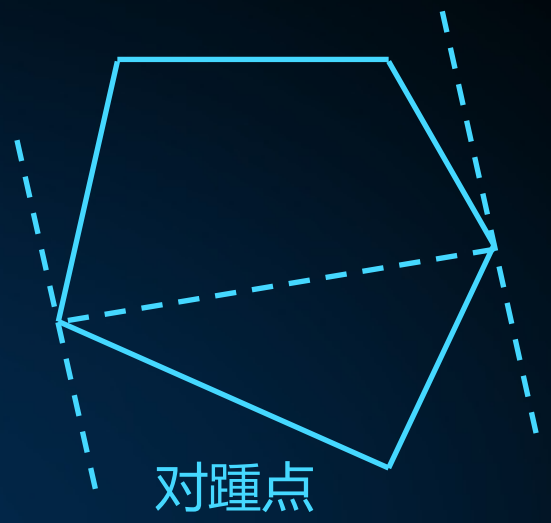
利用单调性可以进行**斜率优化**。

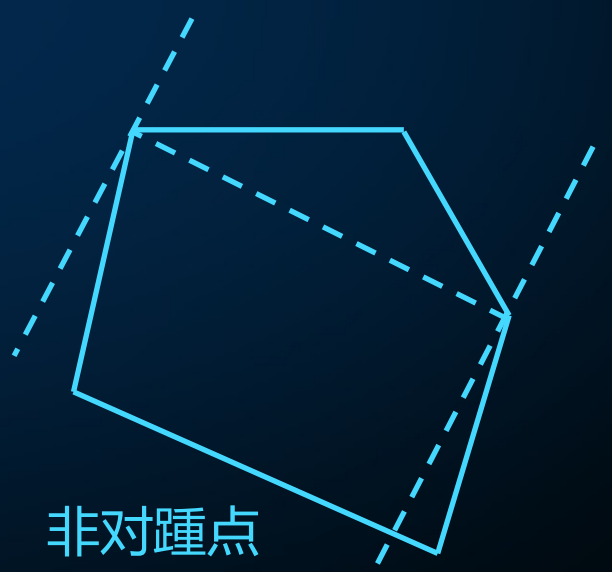
[2] 旋转卡壳

2 旋转卡壳

凸包上的**对踵点**

- 对于凸包上的一对点，
  将其连线，并在两点作**垂线**。
- 如果凸包上**所有**点都在这对平行线之间，
  则将这对点称为对踵点。
- 凸包上可能有**多对**对踵点。
- 可以想象成凸包在管道中滑动，
  对踵点是**最宽**的一对点。





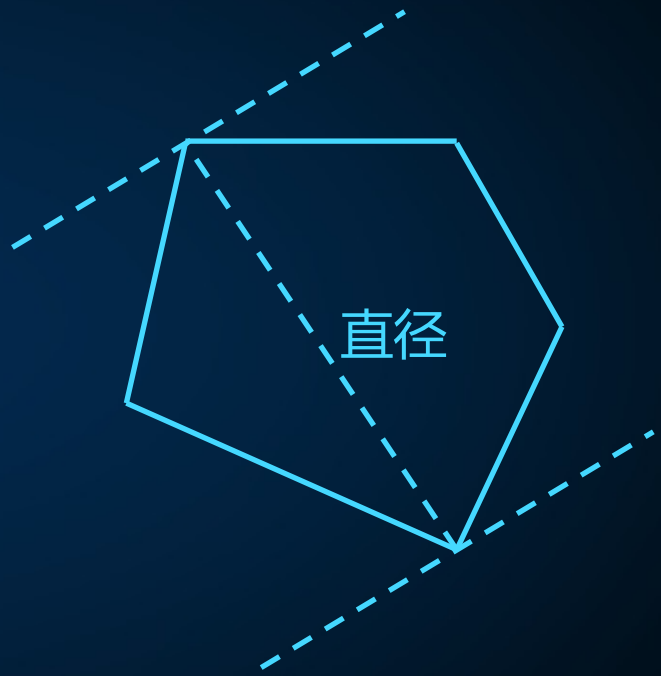
2 旋转卡壳

凸包的**直径**：凸包上**最远**两点的距离

- 直径上的两点一定是对踵点。

- 直径表示能够让凸包
  从**任意**方向滑动通过的**最小**宽度。

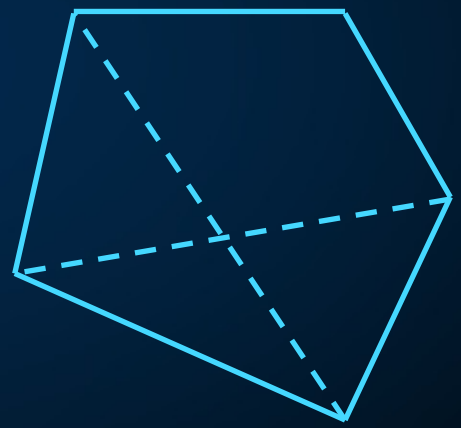
- 凸包的直径**不等价于**凸包**外接圆**的直径。



2 旋转卡壳

寻找凸包的直径

- 可以**枚举**点对寻找直径，但这样做效率不高。
- 容易想到，最远点总是在**相对**的位置上。



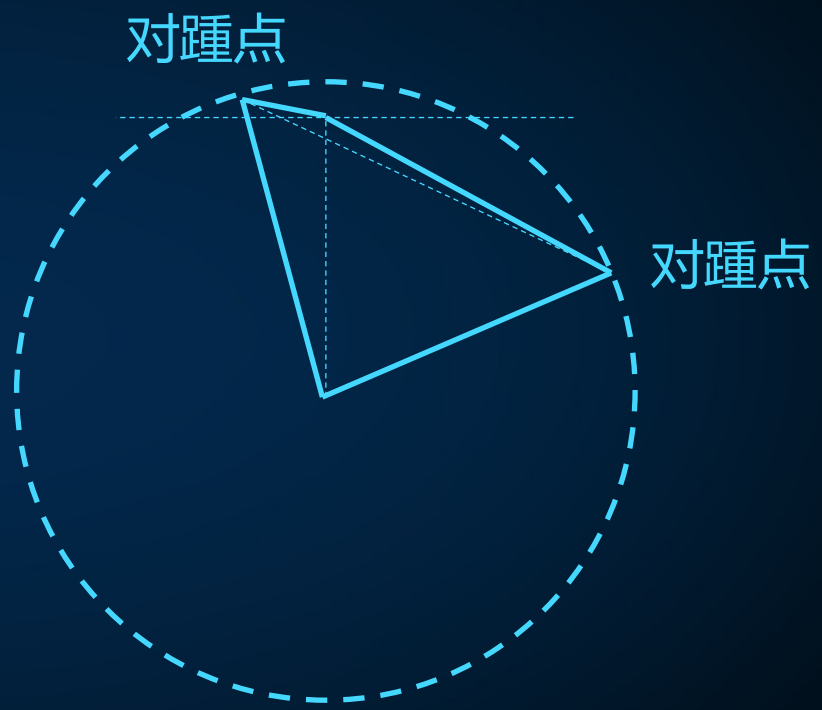
2 旋转卡壳

- 考虑按**逆时针**顺序枚举点，
  期望其最远点也会按照逆时针顺序移动。

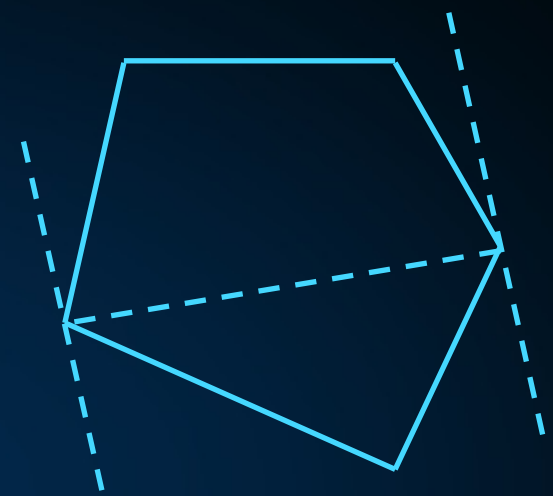
- 但这个想法是不正确的，
  距离的变化**不具有**单调性。

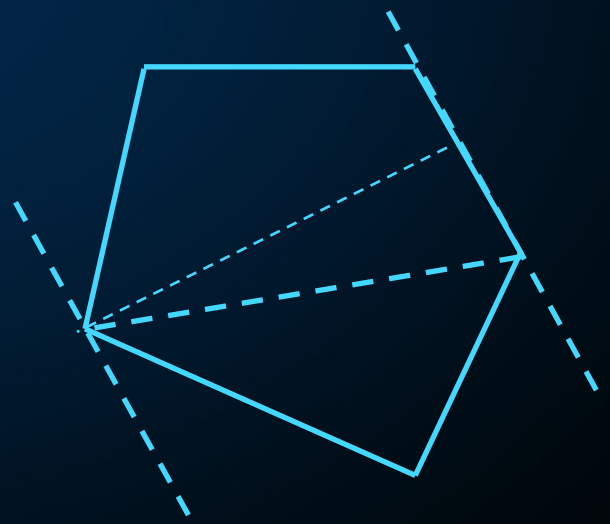
2 旋转卡壳

- 对踵点也**不是连续**出现的。
- 可以尝试挖掘更多对踵点的性质。



2 旋转卡壳





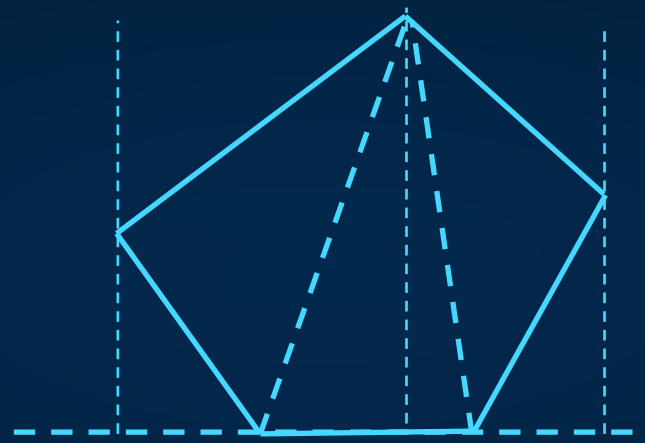
旋转卡壳：通过对踵点寻找凸包直径

- 对于任意一对对踵点，
  将一对平行线分别绕点**同步**旋转，
  直到**任意**一条线与边**共线**。

- 另一点一定是距离该条边**最远**的点。

2 旋转卡壳

- 对于**任意**边，按顺序查看**点**到**边**的距离，存在**单峰**。

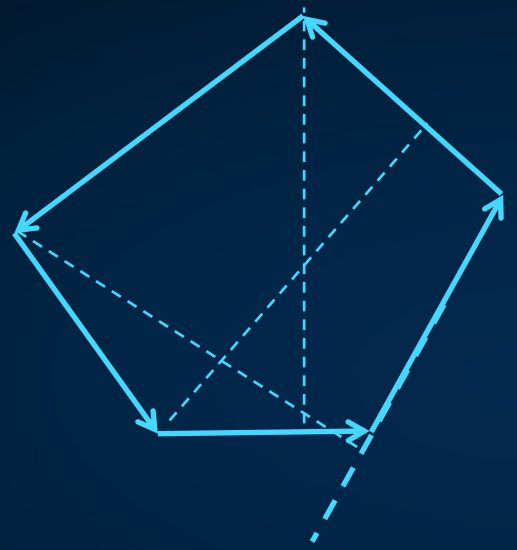


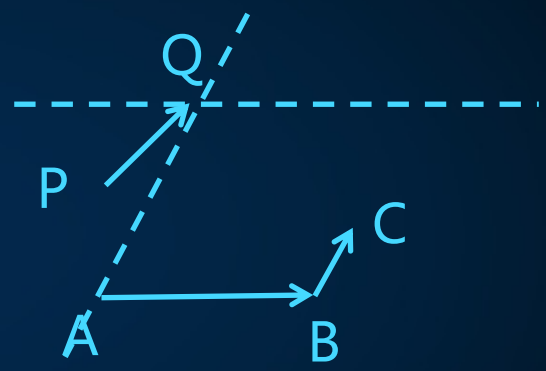
- 对踵点只可能是**最远点**和边的**两个端点**之一。

- 归根结底，凸包是点与边的关系。

2 旋转卡壳

- 按照**逆时针**顺序枚举边，则最远点也会逆时针旋转。





- 否则存在相邻的边AB和BC，以及边PQ，
使得Q离AB更远，而P离BC更远。

- $\overrightarrow{PQ}$在$\overrightarrow{AB}$ 和$\overrightarrow{BC}$之间，与凸包的定义矛盾。

2 旋转卡壳

分析旋转卡壳的时间复杂度

- 【T】枚举边O(n)。
- 【T】跟着边调整最远点O(n)。
- 【T】总时间复杂度为**O(n)**。

2 旋转卡壳

【例2】最远点对

给定平面上的n个点，求最远点对。

- 任意点的最远点肯定在凸包上，
  因为圆内不存在凸包能够包含圆上的点。
- 因此最远点对肯定都在凸包上。
- 可以先求凸包，再在凸包上求最远点对。

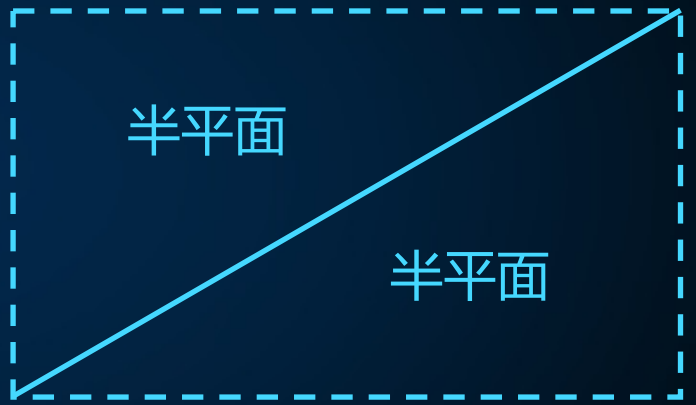
https://www.luogu.com.cn/problem/P1452

[3] 半平面交

3 半平面交

一条直线可以将平面划分成两半

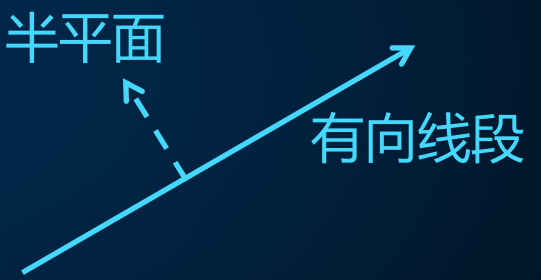
- 直线一侧的**半个**平面被称为**半平面**。
- **包含**边界直线的半平面被称为**闭**半平面，
  不包含边界直线的半平面被称为**开**半平面。
- 可以用解析式Ax + By + C ≥ 0来表示闭半平面，其中A、B、C为参数。



3 半平面交

可以用**有向线段**表示半平面

- 通过有向线段的**起点**、**方向**确定半平面，
  半平面与有向线段的**长度无关**。



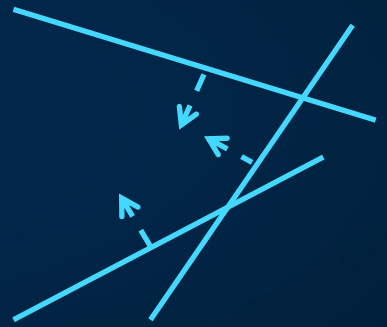
- 默认半平面在有向线段的**左边**，
  也可以将有向线段统称为**向量**，

! 这里的向量与**标准**定义的向量是有差别的。

3 半平面交

半平面交：半平面的交集

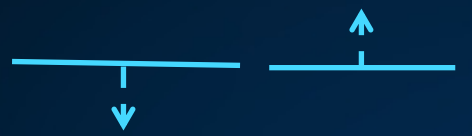
- 半平面交得到的多边形**必然**为凸多边形，此时交集**面积非零**。

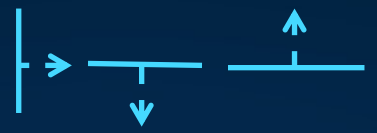


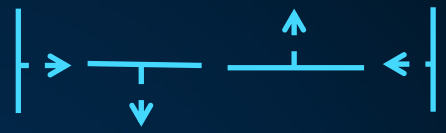
- 也可能得到**不完整**的多边形，此时交集**面积无限**。

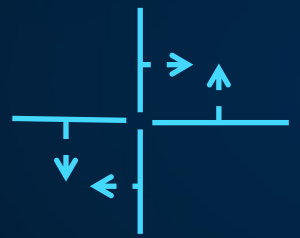
3 半平面交

半平面交可能是直线、射线、线段或者点，此时**面积为零**。









半平面交可能为**空**，即不存在交集。

面积为零与空是不同的。

3 半平面交

可以对简单多边形求半平面交

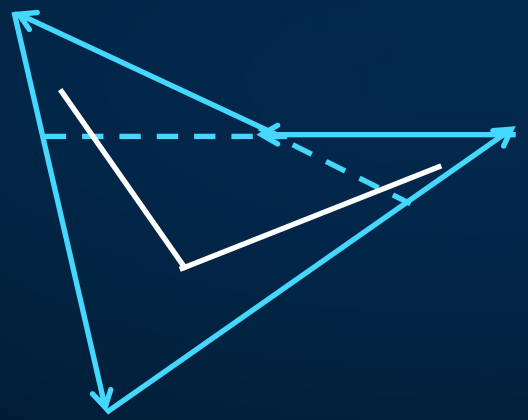
- **逆时针**查看简单多边形的每条边，求边**左侧**的半平面的交。
- 半平面交被称为多边形的**核**。

- 多边形的核可能不存在，凸多边形的核等于本身。

3 半平面交

简单多边形的核的特性

核内**任意**点，与多边形内**任意**点连成的线段，
一定在多边形内部，即**不会**与多边形的边有交点。



也可以理解成，核内的点能够看见多边形内的每一个点。

3 半平面交

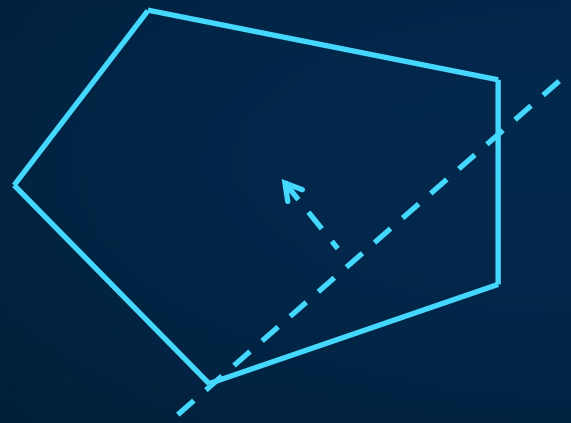
求半平面交的直观方法

- 初始化一个**足够大**的矩形平面，保证交一定在这个平面中。
- 不断加入新的半平面，并维护半平面交。

3 半平面交

在半平面交中加入新的半平面

- 新的半平面会对当前半平面交进行切割，
可能将当前半平面交分成两部分。



- 按**顺序**检查当前半平面交的顶点，保留在新半平面内的顶点。

- 如果当前半平面交中的**相邻**顶点分别在新半平面的**两侧**，
则会在新半平面交中产生一个新的顶点。

3 半平面交

分析上述方法的时间复杂度

- 令半平面的数量为n。
- 【T】加入半平面时检查点O(n)，总共**$O(n^2)$**。

3 半平面交

上述算法有什么缺点？

- 没有考虑半平面之间的**关系**，每个半平面都是**单独**加入的。
- 应当按照**顺序**来加入半平面。

3 半平面交

半平面交中半平面的顺序

- 半平面交中的每条边属于一个半平面，
  按照**逆时针**顺序访问这些边，是不断**左旋**的。

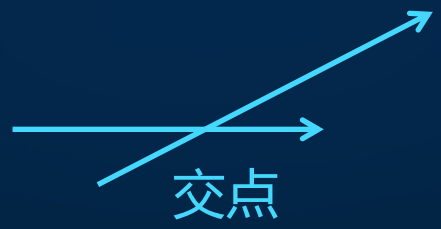
- 将所有半平面按照**极角序**排序，
  则半平面交中的边必定按照这个顺序出现。

- 如果有**同向**的半平面，应该**保留靠内**的，**舍弃靠外**的，
  同时应当**去重**。

3 半平面交

增量算法：按顺序加入半平面

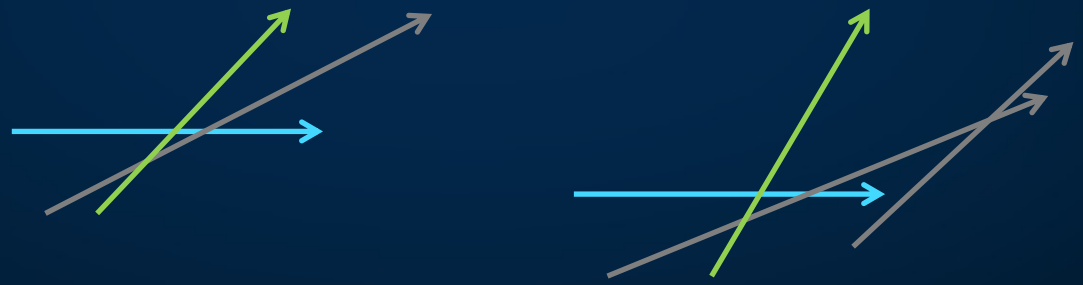
- 除了**首个**半平面，每个半平面都会与**上个**半平面产生一个交点。



- 这个交点就是当前半平面交的一个顶点。

3 半平面交

- 加入新的半平面时，考虑新半平面对当前半平面交产生的影响。
- 如果上个交点**在**新半平面的**右边**，
  此时上个半平面**不会**对结果产生影响，应该**删除**。



- 删除后可能情况不变，此时需要**连续**进行删除。

3 半平面交

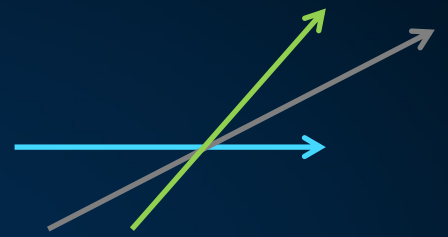
- 直到上个交点不在新半平面的右边，
  此时新半平面应当加入到当前半平面交中。

- 新半平面与上个半平面产生一个新的交点，
  是当前半平面交的新顶点。

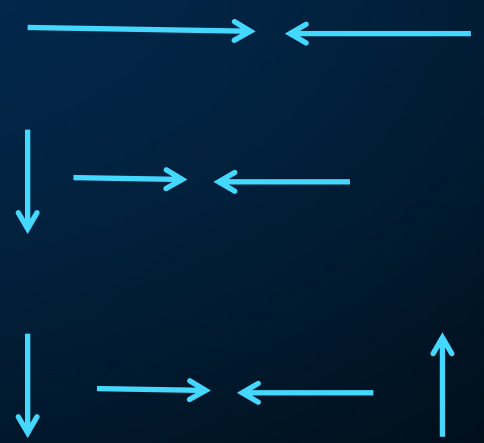
3 半平面交

交点在新半平面**边界上**时该如何处理？

看上去可以删除上个半平面，
因为这个交点是**重复**的。



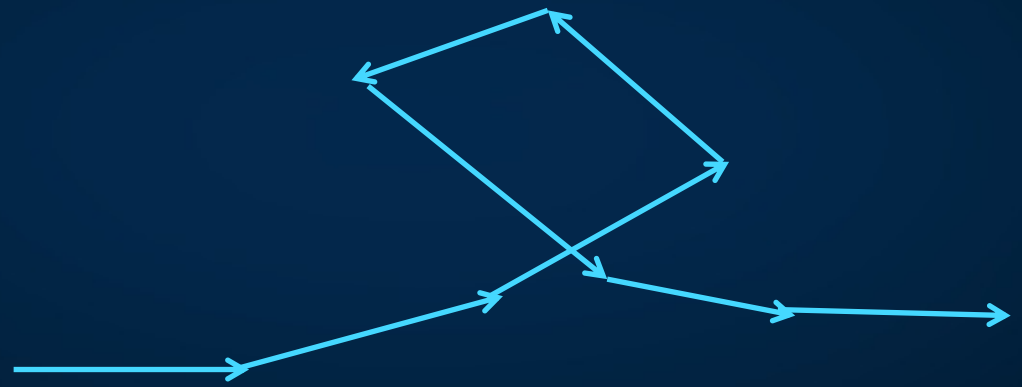
但如果有**平行**且**反向**的半平面，
则需要重合的交点给半平面交**定形**，
此时半平面交面积为零。



3 半平面交

从半平面序列中获取半平面交

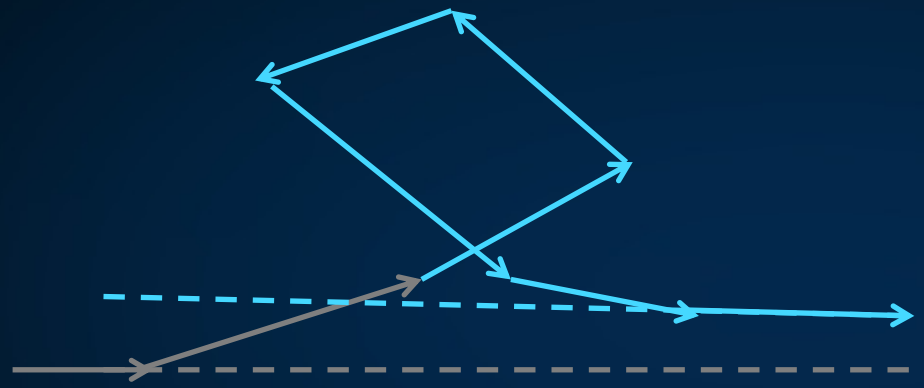
- 序列中的半平面可能产生**交叉**，需要找到半平面交的**闭合点**。

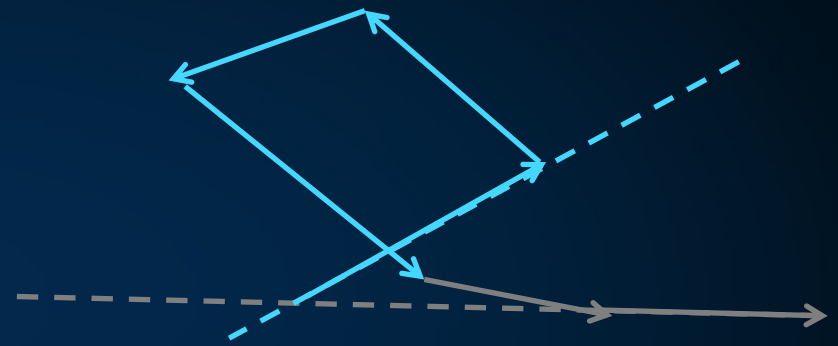


- 交叉的原因是只对序列中**连续**的半平面进行了检查，
但**半平面交**是环形的。

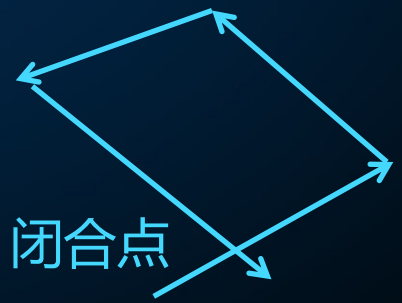
3 半平面交

- 可以不断用首尾半平面**互相**检查交点，将**多余**的半平面删除。





- 检查首部时应当检查与**下个**半平面的交点。
- 清理之后，首尾半平面的交点就是半平面交的闭合点。



3 半平面交

首尾检查时交点在半平面**边界上**该如何处理？

此时半平面是完全多余的，不影响半平面交，所以**可以**删除。

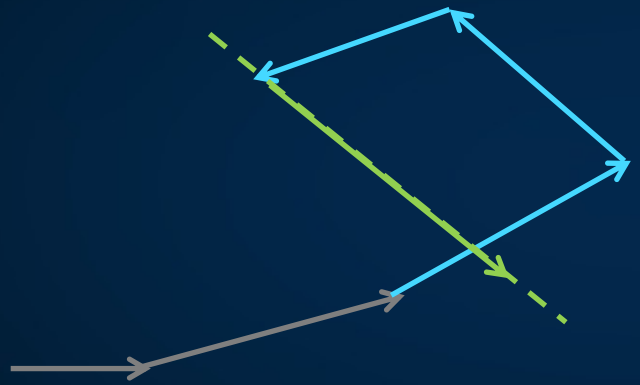
如果不删除，则会产生**重复**的交点。

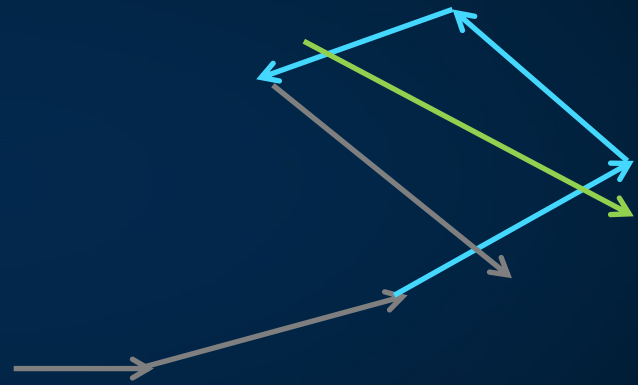
但是算法其他流程也有可能产生重复交点。

3 半平面交

可以提早删除**首部**半平面

- **每次**加入新半平面后，都可以对首部半平面的交点进行检查，判断是否可以删除首部半平面。



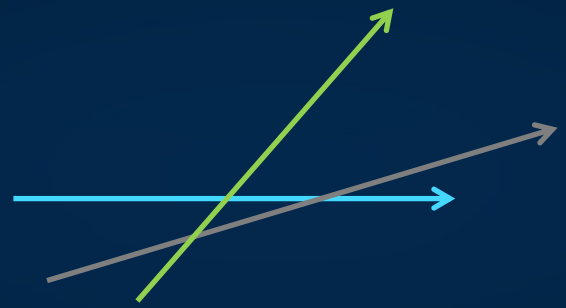


- 任意时刻首部半平面被删除，都意味着此时半平面**已经**闭合，删除掉的首部半平面不会再对结果产生影响。

- 即便用于删除首部半平面的半平面之后也被删除，也不影响。

3 半平面交

- 当序列中只有两个半平面时，首部交点和尾部交点是**同一个**交点。

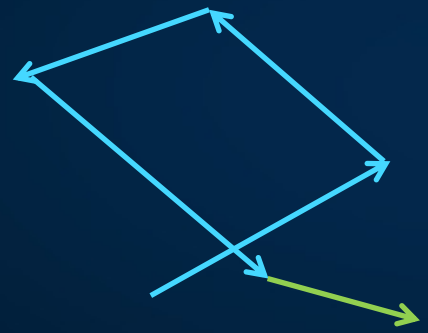


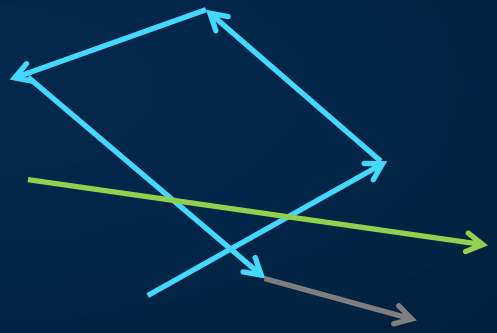
- **通常**认为新半平面与尾部半平面的关系更加紧密，
  应当优先删除尾部交点，再删除首部交点。

3 半平面交

可以提早结束尾部半平面的**添加**吗？

- 考虑当前新加入的半平面的交点，已经**不在**首部半平面的**左边**。





- **不能**提前结束，因为当前半平面可能被删除，
  不能确保后续半平面的交点也不在首部半平面的左边。

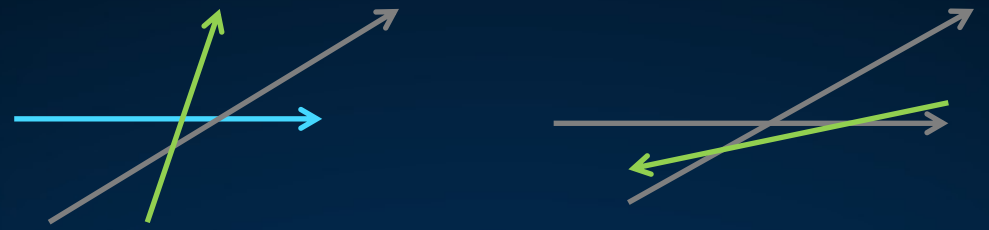
3 半平面交

判断半平面交为空

- **任意**时刻的半平面交为空，则表示最终的半平面交也为空。
- 当前半平面的交点必然都**不在**新半平面中，
  加入新半平面前**所有**交点都会被删除。

3 半平面交

只剩一个半平面时如何判断半平面交为空?



假设首个半平面在**足够远**的边界处有一个交点。

当这个交点也被删除时，说明半平面交为空。

考虑半平面的**倾斜角**，足够远交点的坐标可能达到**平方级**。

至此，**所有**半平面都统一拥有交点。

3 半平面交

当前算法有什么缺陷？

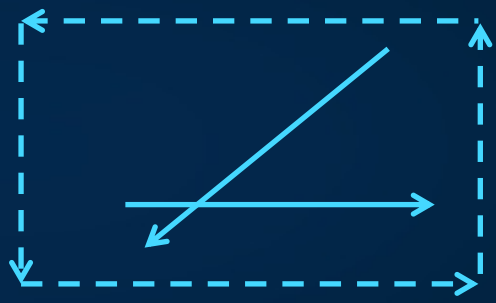
- 算法的思路基于半平面交结果为凸多边形。
- 当**连续**两个半平面转角**不小于**π时，无法正确进行处理。

- 此时的尾部交点实际上属于首部交点的处理范围。

3 半平面交

判断半平面交面积无限大

- 当**连续**两个半平面转角**不小于**π时，将无法形成**外包围**，使得半平面交的面积无限大。



- 添加一个**足够大**的外包围矩形，确保半平面交**有限**时一定**严格**在内部。此时面积必定有限，同时上述情况也不可能出现。

- 如果半平面交有顶点在边界上，且面积**不为**零，说明面积无限大。

3 半平面交

增量算法实现半平面交的思路

- 预处理部分。
- 1. 添加四个半平面形成外包围矩形。
- 2. 对半平面进行极角排序，去重、去除靠右的平行半平面。
- 3. 加入首个半平面，设置交点在边界处。

3 半平面交

- 求半平面序列。
- 4. 按顺序考虑之后每个半平面，维护半平面序列。
  - 4.1. 不断删除尾部半平面，直到交点不在新半平面右边。
  - 4.2. 判断当前半平面交是否为空。
  - 4.3. 将当前半平面加入序列，计算交点。
  - 4.4. 不断删除首部半平面，直到交点在新半平面左边。

3 半平面交

- 求半平面。
- 5. 不断删除尾部半平面，直到交点在首部半平面左边。
- 6. 求首尾半平面的交点，得到完整的半平面交。
- 7. 去除半平面交中重复的交点。
- 8. 检查半平面交的面积是否有限。

3 半平面交

分析增量算法求半平面交的时间复杂度

- 令半平面的数量为n。
- 【T】排序O(n log n)。
- 【T】按顺序处理半平面O(n)。
- 【T】总时间复杂度为**O(n log n)**。

3 半平面交

【例3】凸多边形交

给定n个凸多边形，求交。

- 可以将凸多边形看成多个半平面的交。
- 凸多边形的交就是对应半平面的交。

https://www.luogu.com.cn/problem/P4196

3 半平面交

【例4】监视摄像机

求简单多边形的内核是否为空。

求简单多边形的半平面交，并判断交是否为空。

https://www.luogu.com.cn/problem/P5696

谢 谢

在狼

WILD WOLF