Geometry

2020.08.21

CONTENTS

- Geometry

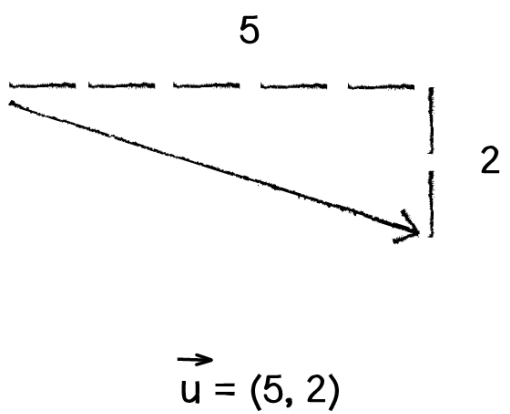
Geometry

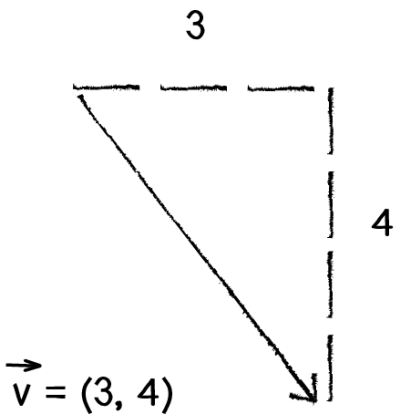
계산기하란?

점, 선, 다각형과 원 등을 다루는 알고리즘을 계산 기하 알고리즘이라고 한다.
흔히 수학의 Vector를 사용하여 구하고 있다.

벡터?

크기와 방향을 모두 가지는 양
일반적으로 벡터는 화살표로 나타낸다.





Dot & Cross Product

[내적]
- 벡터의 사이 각 구하기
- 벡터의 직각여부 확인

- 벡터의 내적(Dot Product) -

두 벡터 $\vec{a}=\langle a_1,a_2,a_3\rangle$, $\vec{b}=\langle b_1,b_2,b_3\rangle$ **의 내적은**

$\vec{a}\cdot\vec{b}=a_1b_1+a_2b_2+a_3b_3$

으로 정의한다.

[외적]
- 면적 구하기
- 벡터의 방향 판별
  각각의 벡터 값이 음수이냐 양수이냐에 따라서 벡터의 상대적인 위치를 알 수 있게 된다.

*vectorA*vectorB=ax*by−ay*bx* 로 볼 때

- 양수일 경우

vectorB가 A보다 시계방향으로 위치해 있다.

- 음수일 경우

vectorB가 A보다 반시계방향에 위치해 있다.

- 벡터의 외적(Cross Product) -

두 벡터 $\vec{a}=\langle a_1,a_2,a_3\rangle$, $\vec{b}=\langle b_1,b_2,b_3\rangle$ **에 대하여**

$\vec{a}\times\vec{b}=\langle a_2b_3-a_3b_2,a_3b_1-a_1b_3,a_1b_2-a_2b_1\rangle$

를 벡터의 외적(Cross Product) 이라고 한다.

Angle Sort

- 모든 벡터에서 한 벡터가 이루는 각을 구하여 오름차순, 내림차순 하는 정렬 알고리즘
- Convex Hull 알고리즘에 사용됨

[arctan 소스]

```
public double getAngle() { return Math.atan2(y, x); }
```

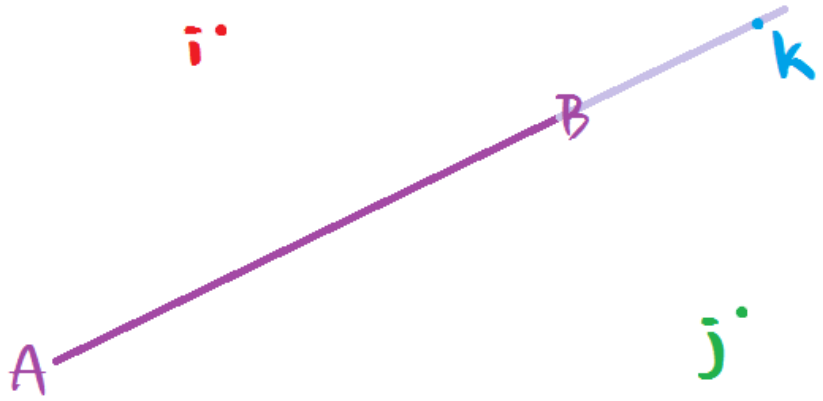
CW & CCW

- 점 3개의 위치관계를 구하여 시계방향[CW], 반시계방향[CCW] 순대로 구하는 정렬 알고리즘
- Convex Hull 알고리즘에 사용됨

res = (A.x * B.y + B.x * C.y + C.x * A.y) − (A.x * C.y + B.x * A.y + C.x * B.y)

선분의 각 끝 점을 각각 A와 B에 대입하고, 구하려고 하는 점을 C에 대입한 후 res값을 통해 위치관계를 알 수 있습니다.

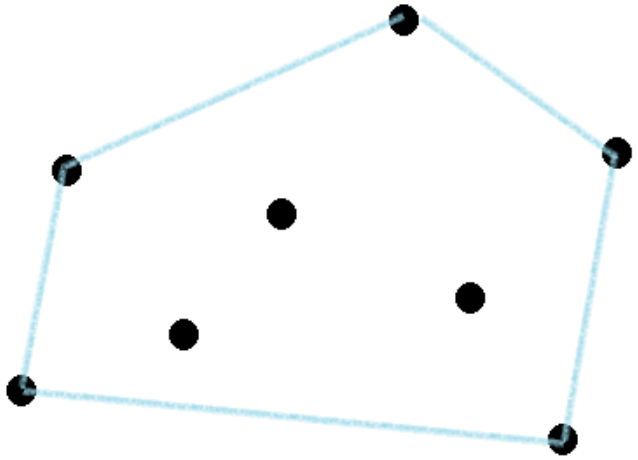
res == 0 : 점 C는 선분 AB의 일직선상에 있습니다. (세 점 모두 일직선상에 있습니다.)
res > 0 : 점 C는 선분 AB의 반시계방향[CCW]에 있습니다.
res < 0 : 점 C는 선분 AB의 시계방향[CW]에 있습니다.



Convex Hull

2차원 평면상에 여러개의 점이 있을 때 그 점 중에서 일부를 이용하여 볼록 다각형을 만드는 알고리즘 단, 볼록 다각형 내부에 모든 점을 포함시키는 것을 의미한다.

이를 만드는 알고리즘 중 **그라함 스캔(Graham's Scan) 알고리즘이 있다.**



출처: https://www.crocus.co.kr/1288 [Crocus]

Graham's Scan

성능 : **O (N log N)**
N은 점의 수

1. 우선 기준점을 잡는다. (보통 기준점은 y좌표가 가장 작은 것을 기준으로 한다.)

2. 그리고 이 기준점으로 하여 다른 점들을 반시계 방향[CCW]으로 정렬한다.
(각에 따라 정렬하면 된다.)

3. 그라함 스캔(Graham's Scan) 알고리즘을 이용한다.

Graham's
Scan : https://www.crocus.co.kr/1288 [Crocus]

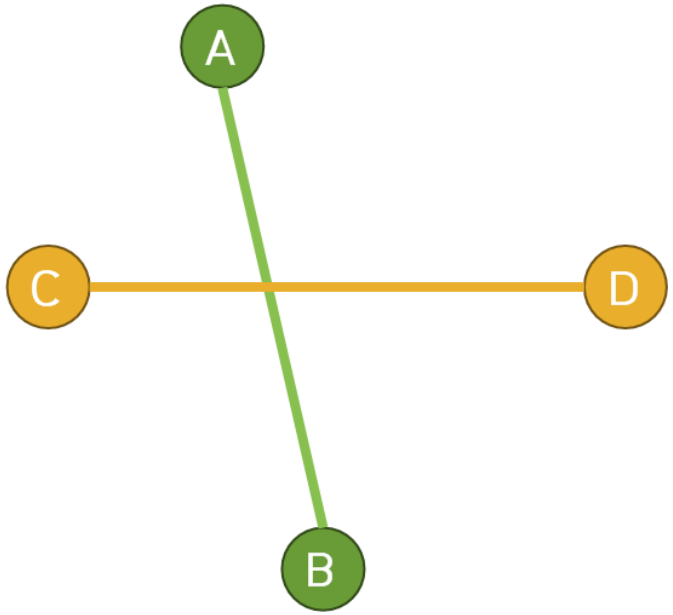
Line Intersection

한 직선 위에 두 선분이 있을 때, 이 선분들이 교차하는가를 판별하는 알고리즘

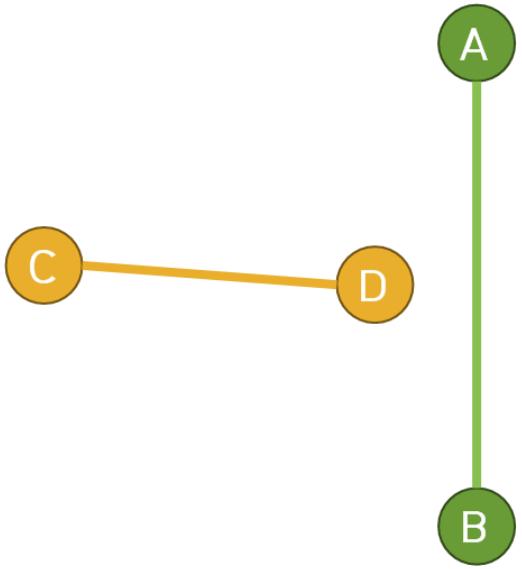
- CCW를 사용하여 판별이 가능하다.
- 선분의 교차를 정확하게 판별하는게 목적

[참고] https://jason9319.tistory.com/358

1. 선분끼리 교차하는 경우



2. 두 선분 교차하지 않는 경우



3. 일직선상에 있을 경우

Plane Sweeping

직사각형이 겹쳐졌을 때 색칠된 부분의 넓이 합을 구하는 것

https://codedoc.tistory.com/421

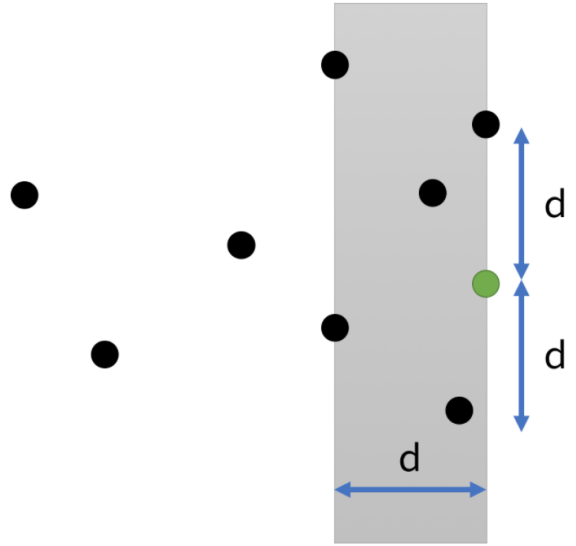
Line Sweeping

수 많은 점들 중에서 가장 가까운 두 점을 찾는 문제

모든 점들을 확인할 시 시간복잡도는 O(N^2)이므로 많은 시간이 걸린다.
따라서 Line Sweeping을 이용해 O(N Log N)으로 줄일 수 있다.

[알고리즘 과정]

1. 한 점(p)을 기준으로 각 p.x, p.y 좌표에서 임의의 기준으로 최단 거리(d)를 -+ 영역에 존재하는 후보들을 추출한다. 최단 거리보다 큰 점들이 있을 시 후보자에서 걸러준다.
2. 후보들을 추출하는 과정에서 후보자가 없을 시 최단 거리(d)를 갱신한다.



Rotating Calipers

수 많은 점들 중에서 가장 먼 점을 찾는 문제
- 볼록 다각형의 지름을 구하여 두 점의 가장 지름이 긴 점을 찾아 해결함

성능 : O(n log n)

[알고리즘 과정]
- Graham's Scan 알고리즘을 진행한다. -> 볼록 껍질을 만들기 위함
- [참고] https://hellogaon.tistory.com/40

Reference

Cross & dot Product : http://blog.naver.com/mindo1103/90103350914 [네냐플]

CW & CCW : https://cookyworld.tistory.com/49 [CookyWorld]

Convex Hull : https://www.crocus.co.kr/1288 [Crocus]

Graham's Scan : https://www.crocus.co.kr/1288 [Crocus]

Line intersection: https://jason9319.tistory.com/358 [ACM-ICPC 상 탈 사람]

Plane sweeping : https://codedoc.tistory.com/421 [codedoc]

Rotating Calipers : https://hellogaon.tistory.com/40 [hellogaon]