Geometri Komputasional

Tim Olimpiade Komputer Indonesia

Pendahuluan

Melalui dokumen ini, kalian akan:

- Memahami konsep vektor
- Memahami konsep poligon
- Memahami konsep convex hull
- Memahami konsep line sweep

Vektor

- **Vektor** adalah objek matematika yang memiliki besaran dan arah.
  - Sebagai contoh, "10 langkah ke utara" pada instruksi "berjalan 10 langkah ke utara" adalah sebuah vektor, karena "10 langkah" merupakan besaran dan "ke utara" merupakan arah.
- Dua vektor dibandingkan dengan besaran dan arahnya.
  - Sebagai contoh, "10 langkah ke utara" pada instruksi "berjalan 10 langkah ke utara dari titik A" dan "berjalan 10 langkah ke utara dari titik B" adalah dua vektor yang sama.
  - Namun, "10 langkah ke utara" dan "10 langkah ke selatan" adalah dua vektor yang berbeda.

Notasi Vektor

- Sebuah vektor biasanya dinotasikan dengan tanda panah di atas variabelnya (contoh: $\overrightarrow{v}$).
- Besaran vektor $\overrightarrow{v}$ dinotasikan dengan $\|\overrightarrow{v}\|$.
- Pada ruang Euklides n dimensi, sebuah vektor $\overrightarrow{v}$ biasanya direpresentasikan dalam n bilangan: $(v_1, v_2, \ldots, v_n)$, dengan v_i adalah besaran vektor pada arah dimensi ke-i.
  - Menggunakan teorema Pythagoras, $\|\overrightarrow{v}\| = \sqrt{\sum_{i=1}^{n} v_i}$.
  - Pada dokumen ini, kita akan mengasumsikan seluruh vektor berada pada ruang Euklides n dimensi.

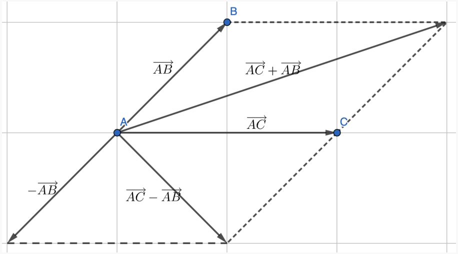
Operasi Vektor

- Penjumlahan dua vektor dinotasikan dengan $\overrightarrow{v} + \overrightarrow{w}$ dan dapat dihitung dengan menjumlahkan besaran pada setiap dimensinya.
- Negasi vektor dinotasikan dengan $-\overrightarrow{v}$ dan dapat dihitung dengan menegasikan besaran pada setiap dimensinya.
- Perkalian vektor dan skalar dinotasikan dengan $k\overrightarrow{v}$ dan dapat dihitung dengan mengkalikan besaran pada setiap dimensinya dengan k.
- **Dot product** dua vektor adalah sebuah skalar yang dinotasikan dengan $\overrightarrow{v} \cdot \overrightarrow{w}$ dan dapat dihitung dengan menjumlahkan perkalian besaran pada setiap dimensinya.
- **Cross product** dua vektor 3 dimensi adalah sebuah vektor 3 dimensi yang dinotasikan dengan $\overrightarrow{v} \times \overrightarrow{w} = (v_2w_3 - v_3w_2, v_3w_1 - v_1w_3, v_1w_2 - v_2w_1)$.

Operasi Vektor: Contoh

Sebagai contoh, jika $\overrightarrow{AB} = (1, 1), \overrightarrow{AC} = (2, 0)$.

- $\overrightarrow{AC} + \overrightarrow{AB} = (2 + 1, 0 + 1) = (3, 1)$
- $-\overrightarrow{AB} = (-1, -1)$
- $\overrightarrow{AC} - \overrightarrow{AB} = (2 - 1, 0 - 1) = (1, -1)$
- $4\overrightarrow{AB} = (4 \times 1, 4 \times 1) = (4, 4)$
- $\overrightarrow{AB} \cdot \overrightarrow{AC} = 1 \times 2 + 1 \times 0 = 2$

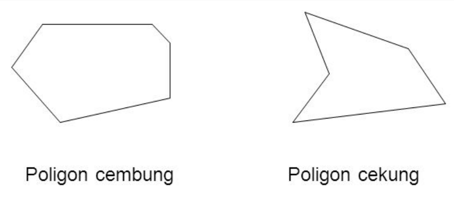


Penggunaan Cross Product

- Pada bidang 2 dimensi, jika kita memiliki tiga titik $O = (x_O, y_O), A = (x_A, y_A), B = (x_B, y_B)$, kita dapat menghitung apakah sudut yang dibentuk oleh ketiga titik tersebut positif menggunakan *cross product*.
  - Hitung besaran pada arah dimensi ketiga dari vektor $\overrightarrow{OA}$ dan $\overrightarrow{OB}$. Untuk lebih mudahnya, anggap nilai P adalah $(x_A - x_O)(y_B - y_O) - (y_A - y_O)(x_B - x_O)$.
  - Jika $P > 0$, maka sudut OAB berlawanan arah jarum jam. Dengan kata lain, B berada di kiri garis OA.
  - Jika $P < 0$, maka sudut OAB searah jarum jam. Dengan kata lain, B berada di kanan garis OA.
  - Jika $P = 0$, maka titik O, A, dan B berada pada garis yang sama.

Poligon

- **Poligon** adalah bentuk datar yang terdiri dari titik (disebut titik sudut) serta garis yang menghubungkan titik sudut dan membentuk lintasan tertutup.
- **Poligon cembung** (*convex polygon*) adalah poligon dengan besar seluruh sudut dalam kurang dari 180°.
  - Dengan kata lain, tidak terdapat segmen garis di antara dua titik pada batas poligon yang keluar dari poligon.
- **Poligon cekung** (*concave polygon*) adalah poligon dengan besar setidaknya satu sudut dalam tidak kurang dari 180°.



Luas Poligon

- **Shoelace formula** adalah rumus yang dapat digunakan untuk menghitung luas poligon.
- Poligon dengan titik sudut $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ memiliki luas

$$\frac{1}{2}\left|\sum_{i=1}^{n} x_i y_{i+1} - \sum_{i=1}^{n} x_{i+1} y_i\right|$$

$$= \frac{1}{2}\left|x_1 y_2 + x2 y_3 + \cdots + x_n y_{n+1} - x_2 y_1 - x_3 y_2 - \cdots - x_{n+1} y_n\right|$$

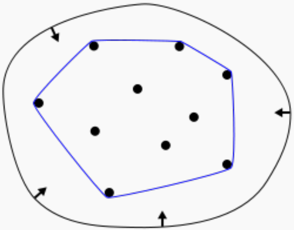
dengan $(x_{n+1}, y_{n+1}) = (x_1, y_1)$.

Lokasi Titik pada Poligon

- Untuk poligon cembung, jika terdapat titik $P = (x, y)$, maka kita dapat menentukan lokasi titik berada pada di dalam atau di luar poligon menggunakan *cross product*.
  - Untuk semua titik bersebelahan A dan B pada poligon, hitung arah dari sudut PAB.
  - Jika semua sudut PAB berlawanan arah jarum jam atau berada pada garis yang sama, maka titik berada di dalam poligon.
  - Jika semua sudut PAB searah arah jarum jam atau berada pada garis yang sama, maka titik berada di dalam poligon.
  - Jika terdapat sudut PAB yang searah dan berlawanan arah jarum jam, maka titik berada di luar poligon.

Convex Hull

- Pada 2 dimensi, **convex hull** dari himpunan titik adalah poligon cembung terkecil yang mencakup seluruh titik tersebut.
- *Convex hull* dapat dibayangkan seperti karet yang cukup besar dan terdapat paku pada setiap titik, kemudian karet tersebut dilepaskan. Bentuk akhir dari karet membentuk *convex hull*.
- Pada pembahasan ini, kita ingin mencari titik-titik yang membentuk *convex hull* (himpunan titik-titik yang disentuh oleh karet).



Graham Scan

- Salah satu metode untuk mencari *convex hull* adalah menggunakan algoritme **Graham scan**.
- Algoritme ini mengambil titik yang berada di paling bawah (koordinat y terkecil) sebagai *pivot* dan bagian dari *convex hull*.
- Lakukan pengurutan titik-titik lainnya berdasarkan sudut polar terhadap *pivot*. Dengan kata lain, titik A lebih kecil dari titik B jika titik B berada di kiri garis OA dengan titik O adalah *pivot*.

Graham Scan (lanj.)

- Kemudian, algoritme ini mencoba untuk memasukkan titik-titik lainnya satu per satu ke dalam *convex hull*, diurutkan dari yang paling kecil.
  - Misalkan titik R adalah titik yang sekarang sedang diperhatikan, Q adalah titik terakhir pada *convex hull* sementara, dan titik P adalah titik sebelum titik Q pada *convex hull* sementara.
  - Jika titik R berada di kanan garis PQ, maka dapat dipastikan titik Q tidak berada di dalam *convex hull*. Karenanya, kita dapat mengeluarkan titik Q pada *convex hull* sementara.
  - Perbaharui titik P dan titik Q menggunakan definisi di atas.
  - Kita dapat melakukan hal yang sama sampai titik R berada di kiri garis PQ, atau terdapat hanya satu titik pada *convex hull*.
- Kompleksitas algoritme ini adalah $O(N \log N)$, dengan N adalah banyaknya titik.

Graham Scan (lanj.)

```
typedef pair<int, int> Point;
#define x first
#define y second

bool turn_right(Point o, Point p, Point q) {
  return (p.x - o.x) * (q.y - o.y) < (p.y - o.y) * (q.x -
    o.x);
}

vector<Point> convex_hull(vector<Point> P) {
  // Letakan pivot di P[0].
  for (int i = 1; i < P.size(); ++i) {
    if (P[i].y < P[0].y) {
      swap(P[0], P[i]);
    }
  }

  sort(P.begin() + 1, P.end(), [&] (Point p, Point q) {
    return turn_right(P[0], q, p);
  });

...
```

Graham Scan (lanj.)

```
  vector<Point> convex_hull = {P[0]};

  for (int i = 1; i < P.size(); ++i) {
    while (convex_hull.size() > 1) {
      Point p = convex_hull[convex_hull.size() - 2];
      Point q = convex_hull[convex_hull.size() - 1];
      if (turn_right(p, q, P[i])) {
        break;
      }
    }
    convex_hull.push_back(P[i]);
  }

  return convex_hull;
}
```

Algoritme Convex Hull lain

- Sebagai tambahan informasi, selain algoritme *Graham scan* terdapat beberapa algoritme lain yang cukup umum:
  - Algoritme **Gift wrapping** (sering juga disebut **Jarvis march**) mencari *convex hull* dalam waktu $O(NH)$, dengan H adalah banyaknya titik pada *convex hull*.
  - Algoritme **Monotone chain** (sering juga disebut **Andrew's algorithm**) mencari *convex hull* dalam waktu $O(N \log N)$.
- Kita tidak akan membahas algoritme tersebut pada diskusi ini.

Line Sweep

- Algoritme **line sweep** menyapu (*sweep*) garis untuk menyelesaikan beberapa soal pada bidang 2 dimensi.
- Contoh soal: diberikan himpunan A yang berisi N segmen garis yang sejajar dengan sumbu x dan himpunan B yang berisi N segmen garis yang sejajar dengan sumbu y. Tentukan apakah terdapat segmen garis pada himpunan A yang berpotongan dengan segmen garis pada himpuan B.
- Soal ini dapat diselesaikan dengan menyimpan koordinat x dari setiap **peristiwa**, yang merupakan salah satu dari:
    - ujung kiri dari segmen garis pada himpunan A,
    - ujung kanan dari segmen garis pada himpuan A,
    - segmen garis pada himpunan B.

Line Sweep (lanj.)

- Peristiwa diproses satu per satu dengan urutan koordinat x menaik, sehingga seperti penyapuan garis. Kita juga membutuhkan sebuah struktur data D.
  - Jika peristiwa merupakan ujung kiri dari segmen garis pada himpunan A, maka tambahkan koordinat y garis tersebut pada D.
  - Jika peristiwa merupakan ujung kanan dari segmen garis pada himpunan A, maka hapus koordinat y garis tersebut pada D.
  - Jika peristiwa merupakan segmen garis pada himpunan B, maka periksa apakah terdapat bilangan pada D yang berada di antara y_{min} dan y_{max}.
    - Jika ada, maka segmen garis ini berpotongan dengan segmen garis pada himpunan A.
- Kita dapat menggunakan struktur data C++ **set** agar seluruh penambahan dan pengecekan pada D membutuhkan waktu $O(\log N)$.
  - Sehingga, total kompleksitas waktu solusi ini adalah $O(N \log N)$.