LAPORAN
TUGAS KECIL 1
IF2211 Strategi Algoritma
"Penyelesaian Persoalan Convex Hull dengan Algoritma Brute Force"

Disusun oleh:

- Jonathan Yudi Gunawan 13518084

Prodi Teknik Informatika

Sekolah Teknik Elektro dan Informatika

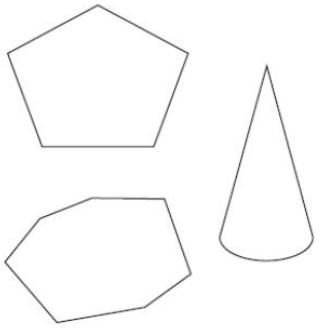
Institut Teknologi Bandung

2020

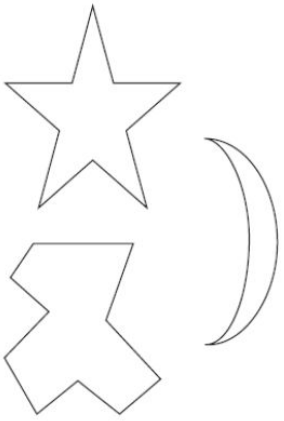
DAFTAR ISI

BAB I
DESKRIPSI MASALAH

Salah satu hal penting dalam komputasi geometri adalah menentukan convex hull dari kumpulan titik. Himpunan titik pada bidang planar disebut convex jika untuk sembarang dua titik pada bidang tersebut (misal p dan q), seluruh segmen garis yang berakhir di p dan q berada pada himpunan tersebut. Contoh gambar 1 adalah poligon yang convex, sedangkan gambar 2 menunjukkan contoh yang non-convex.

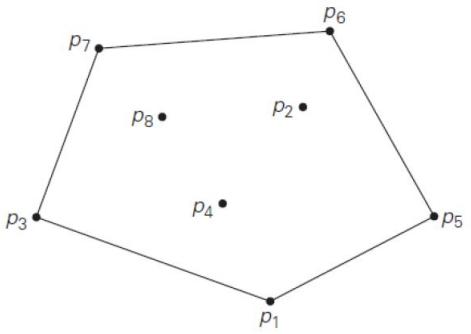


Gambar 1 convex



Gambar 2 non-convex

Convex Hull dari himpunan titik S adalah himpunan convex terkecil yang mengandung S. Untuk dua titik, maka convex hull berupa garis yang menghubungkan 2 titik tersebut. Untuk tiga titik yang terletak pada satu garis, maka convex hull adalah sebuah garis yang menghubungkan dua titik terjauh. Sedangkan convex hull untuk tiga titik yang tidak terletak pada satu garis adalah sebuah segitiga yang menghubungkan ketiga titik tersebut. Untuk titik yang lebih banyak dan tidak terletak pada satu garis, maka convex hull berupa poligon convex dengan sisi berupa garis yang menghubungkan beberapa titik pada S. Contoh convex hull untuk delapan titik dapat dilihat pada gambar 3.



Gambar 3 Convex Hull untuk delapan titik

Pemanfaatan dari convex hull ini cukup banyak. Pada animasi komputer, pemindahan suatu objek akan lebih mudah dengan memindahkan convex hull objek untuk collision detection. Pada bidang statistik, convex hull juga dapat mendeteksi outliers pada kumpulan data. Convex hull juga dapat digunakan dalam persoalan optimasi, karena penentuan titik ekstrimnya dapat membatasi kandidat nilai optimal yang diperiksa.
Convex hull dapat ditemukan dengan algoritma brute Force dan algoritma Divide and Conquer. Pada Tugas Kecil 1 ini digunakan algoritma brute force saja.

Buatlah sebuah aplikasi sederhana dalam Bahasa C/C++/Java (pilih salah satu) untuk menentukan convex hull dari kumpulan titik yang diberikan dalam bidang dua dimensi. Masukan dari program adalah banyaknya titik (yaitu n), dan kemudian titik sebanyak n dibangkitkan secara acak oleh program. Setiap titik dinyatakan dengan korordinat (x, y). Luaran program adalah himpunan titik yang membentuk convex hull, dan waktu yang diperlukan untuk menemukan convex hull (tidak termasuk waktu membaca data dan menuliskan luaran). Tuliskan spesifikasi komputer yang digunakan.

Bonus (10) jika dapat menampilkan semua titik dan convex hull yang terbentuk. Khusus untuk proses penggambaran dapat menggunakan library yang tersedia bebas (freeware).
Contoh untuk n=4, dan titik yang dibangkitkan adalah sebagai berikut:
(12, 32)
(45, 98)
(65, 12)
(10, 30)
Maka himpunan titik yang membentuk convex hull dinyatakan sebagai senarai (list) sebagai berikut: [(10, 30),(45, 98),(65, 12),(10,30)]

BAB II
ALGORITMA PROGRAM

2.1 Analisis

Poligon convex hull mencakup semua titik pada bidang tersebut, maka garis yang membentuk poligon tersebut merupakan garis terluar yang menyentuh 2 titik pada bidang. Garis terluar adalah garis yang mencakup semua titik lainnya di salah satu sisinya saja. Dapat diterapkan algoritma brute force untuk mencari garis yang memenuhi kedua syarat ini.

2.2 Algoritma

1. Untuk semua pasangan titik yang mungkin, lakukan:
2. Pengecekan untuk setiap titik lainnya (n-2 titik lainnya), apakah semua titiknya berada di satu sisi yang sama (atau terletak pada garis tersebut).
3. Bila ya, maka pasangan titik tersebut merupakan bagian dari himpunan titik yang membentuk convex hull.

Pengecekan dapat dilakukan dengan menurunkan rumus persamaan linear garis dengan diketahui dua titik yaitu:

$$\frac{y - y_1}{y_2 - y_1} = \frac{x - x_1}{x_2 - x_1}$$

Ubah menjadi bentuk ax + by + c, dengan **a = y2-y1, b = x1-x2, c = x2*y1-x1*y2.**
Setelah itu, dapat digunakan flag/boolean apakah semua titik lainnya memiliki nilai ax+by+c yang **bertanda sama** (positif /negatif).
Bila ditemukan ada yang memenuhi syarat ini, maka kedua titik dimasukkan dalam himpunan penyelesaian / himpunan convex hull.

2.3 Kompleksitas Waktu

Untuk mencari seluruh pasangan yang mungkin dalam suatu himpunan n buah titik, diperlukan 1+2+3+4+...+n-1 = n*(n-1)/2 kali pemrosesan, masing-masing dilakukan pengecekan terhadap n-2 titik lainnya. Untuk melakukan pengecekan, dilakukan 1x perbandingan, yaitu pengecekan nilai ax+by+c.
Maka kompleksitas totalnya adalah **T(n)** = n*(n-1)*(n-2)/2
Karena **n*(n-1)*(n-2)/2 <= 3n^3**, untuk $n_0 > 2$, maka dalam notasi big Oh, **O(n^3)**.

BAB III

KODE PROGRAM

Program terbagi menjadi beberapa class seperti main, button, buttonbox, convexhull, point, dan uicontroller. Pada laporan ini hanya akan dicantumkan class convexhull saja.

class ConvexHull:

attribute

```
ArrayList<Point> ps;
ArrayList<PVector> hull;
```

constructor

```
ConvexHull (int n){
  this.generateNewPoints(n);
}
```

pembangkit titik acak

```
void generateNewPoints(int n){
  this.ps = new ArrayList<Point>();
  for(int i = 0; i < n; ++i){
    this.ps.add(new Point());
  }
  this.getConvexHull();
}
```

wrapper (timer dan debug) program penghitung convex hull

```
void getConvexHull(){
  for(Point p: this.ps){
    p.isHull = false;
  }
  long t1 = System.nanoTime();
  this.getConvexHullUtil();
  long t2 = System.nanoTime();
  if(view.bottomBtns.btns.get("outBtn").toggle){
    print("[");
    for(Point p: this.ps){
      if (p.isHull){
        print("("+str(p.pos.x)+","+str(p.pos.y)+"),");
```

```
      }
    }
    println("]");
    double t = ((double)t2-t1)/1000000;
    println("Time taken: " + String.valueOf(t) + " milliseconds");
  }
}
```

program utama pencari convex hull

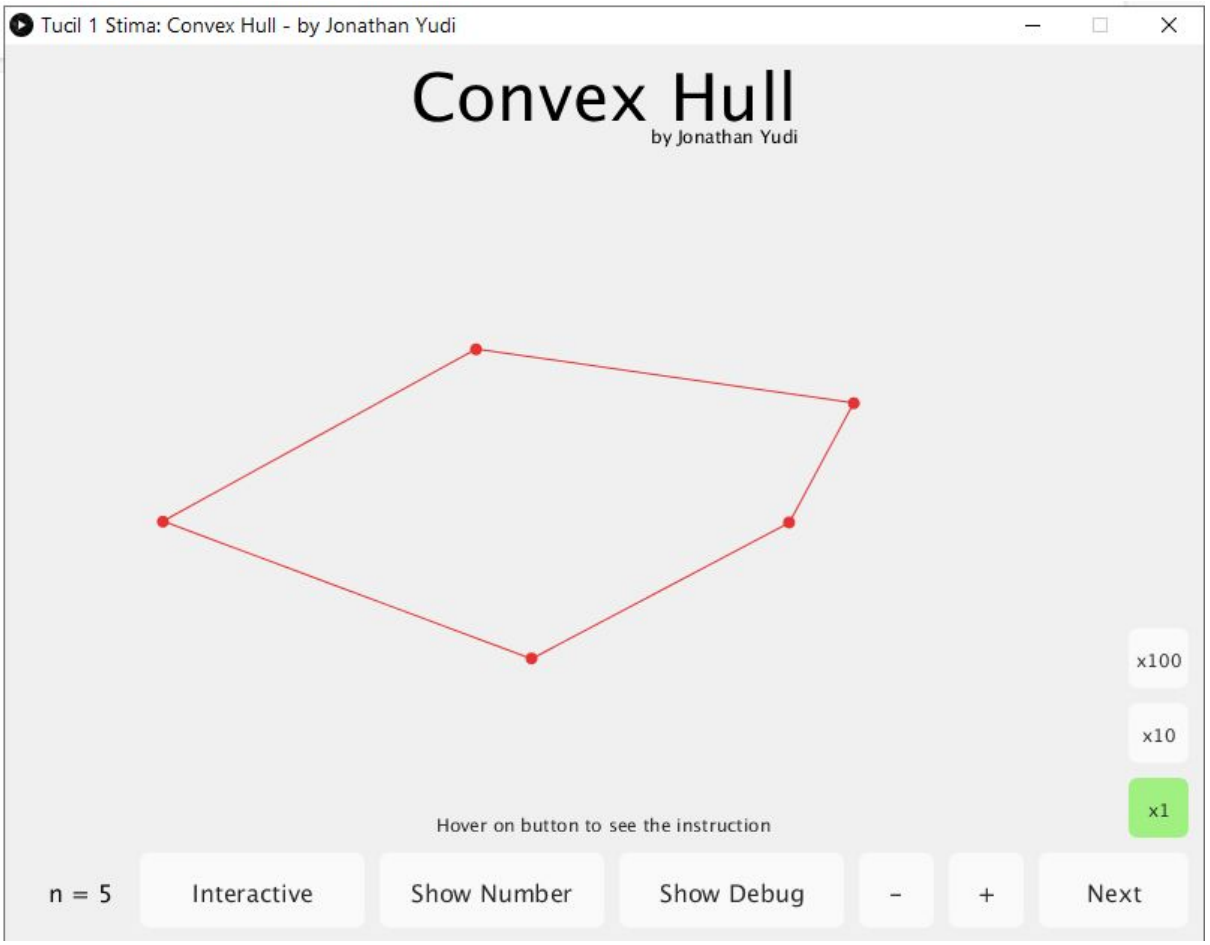
```
void getConvexHullUtil(){
  this.hull = new ArrayList<PVector>();
  for(int i = 0; i < this.ps.size(); ++i){
    Point p1 = this.ps.get(i);
    float x1,y1;
    x1 = p1.pos.x;
    y1 = p1.pos.y;
    for(int j = i+1; j < this.ps.size(); ++j){
      Point p2 = this.ps.get(j);
      float x2,y2;
      x2 = p2.pos.x;
      y2 = p2.pos.y;

      boolean allPos, allNeg;
      allPos = true;
      allNeg = true;
      for(int k = 0; k < this.ps.size(); ++k){
        if(i == k || j == k){
          continue;
        }
        Point p3 = this.ps.get(k);
        float x,y;
        x = p3.pos.x;
        y = p3.pos.y;
        // linear eq: (y2-y1)*x + (x1-x2)*y - x1y2 + x2y1
        if(((y2-y1)*x + (x1-x2)*y - x1*y2 + x2*y1) > 0){
          allNeg = false;
          if(!allPos){
            break;
          }
```

```
        } else if(((y2-y1)*x + (x1-x2)*y - x1*y2 + x2*y1) < 0){
          allPos = false;
          if(!allNeg){
            break;
          }
        } else {
          // segaris, ambil 2 titik terjauh
          // karena titik yang diambil random, maka tidak perlu
diimplementasikan
        }
      }
      if(allPos || allNeg){
        p1.isHull = true;
        p2.isHull = true;
        this.hull.add(new PVector(i, j));
      }
    }
  }
}
```

BAB IV

EKSPERIMEN

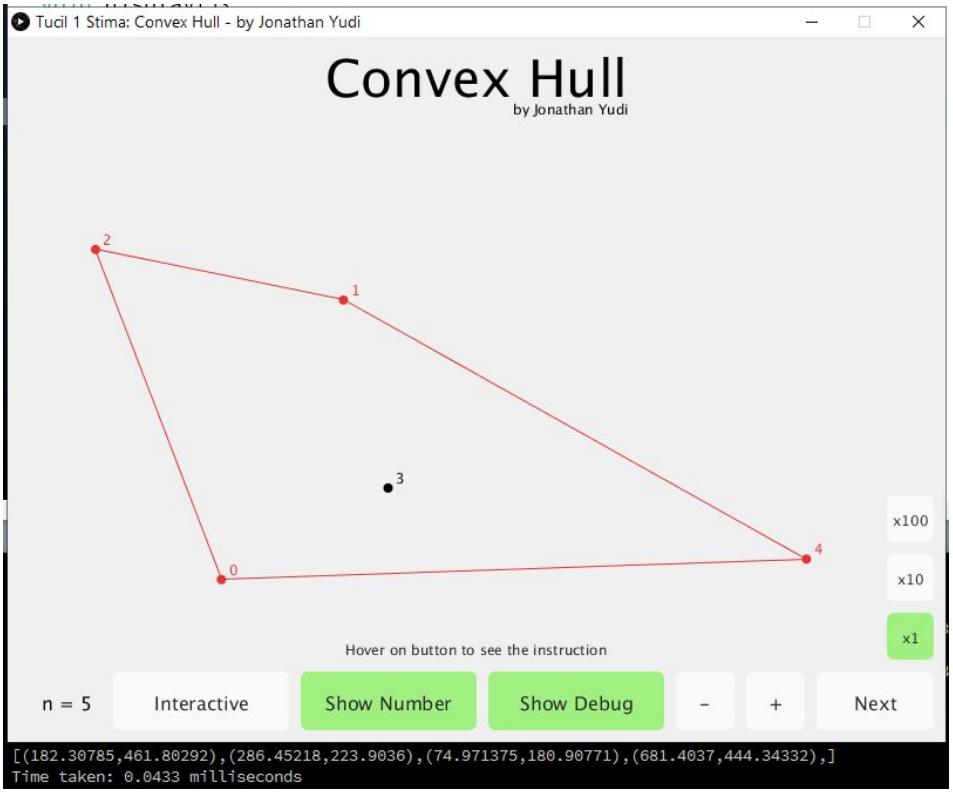
4.1 Tampilan GUI



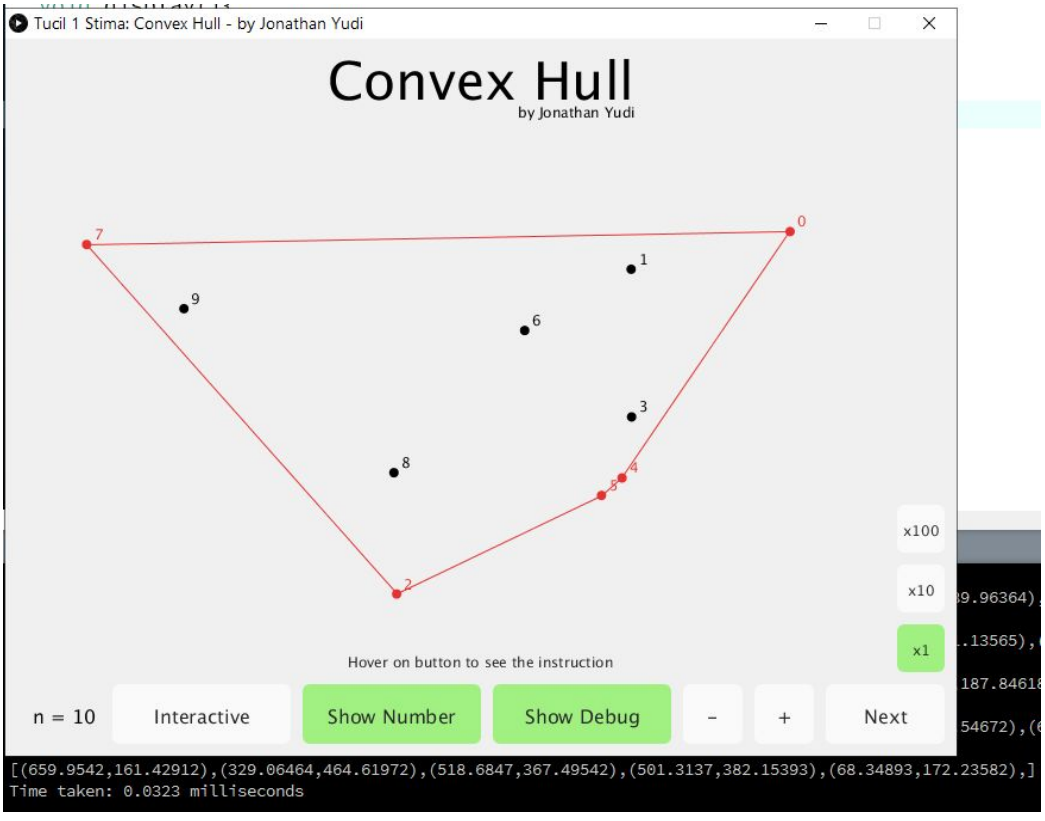
Keterangan:

- Interactive : Titik dapat digeser dengan menarik tetikus
- Show Number : Menampilkan penomoran titik
- Show Debug : Menampilkan senarai himpunan titik convex hull dan waktu yang diperlukan pada terminal
- + / - : Menambah atau mengurangi titik pada bidang
- Next : Membangkitkan n titik baru pada bidang
- x1, x10, x100 : Pengganda penambahan titik

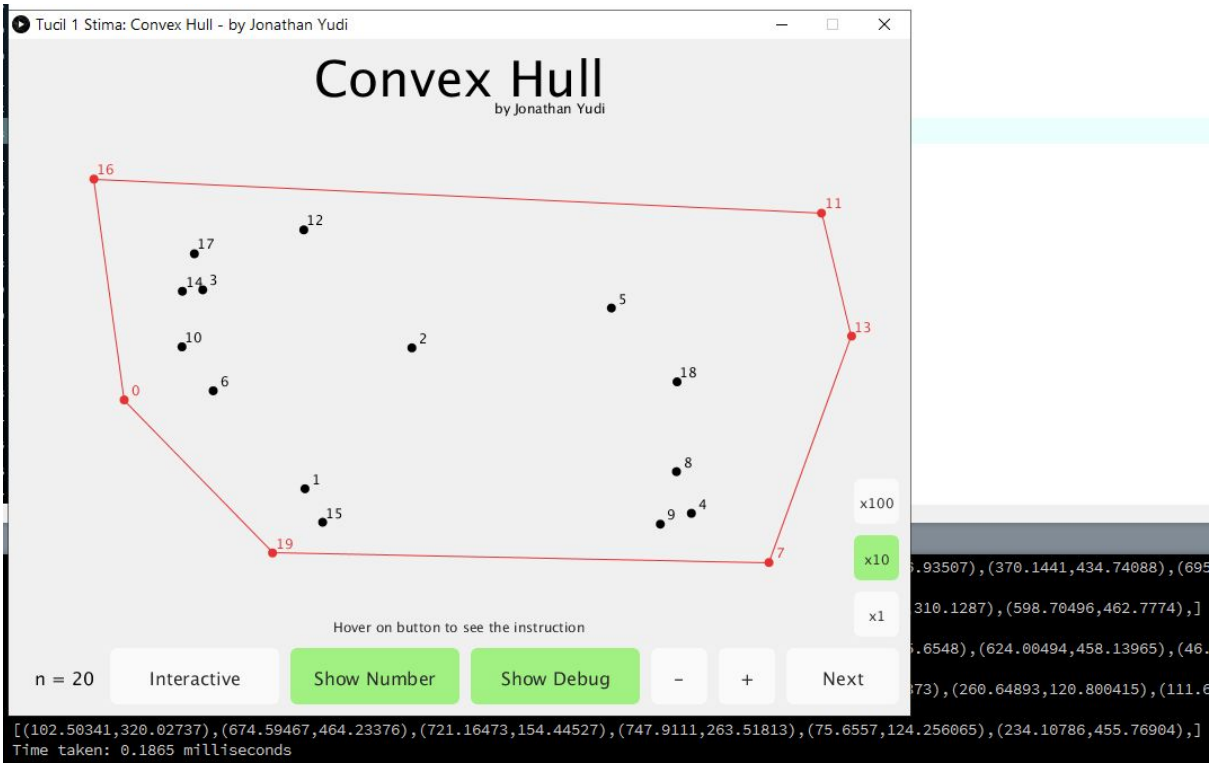
4.2 Tangkapan Layar untuk n = 5



4.3 Tangkapan Layar untuk n = 10



4.4 Tangkapan Layar untuk n = 20



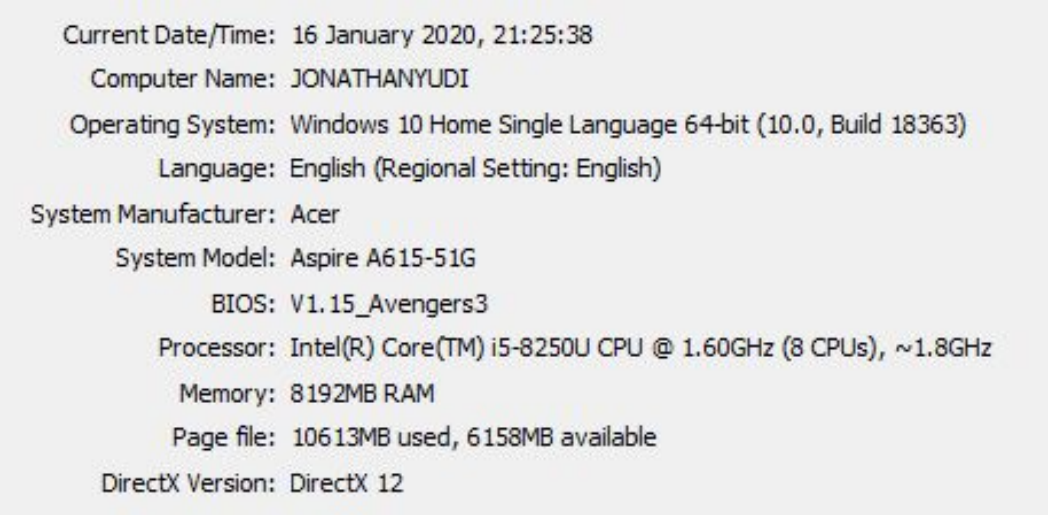
4.5 Tambahan

Berikut tabel checklist implementasi program saya. Perlu diketahui program menerima input melalui device *mouse,* dengan cara menekan tombol yang sudah dijelaskan pada bagian 4.1.

Poin	Ya	Tidak
1. Program berhasil dikompilasi	v	
2. Program berhasil running	v	
3. Program dapat menerima input dan menuliskan output.	v	
4. Luaran sudah benar untuk semua n	v	

Tabel Checklist Implementasi Program

Berikut adalah spesifikasi laptop yang saya gunakan untuk menjalankan program di atas.



Gambar Spesifikasi Laptop yang Digunakan untuk Menjalankan Program

---

DAFTAR REFERENSI

https://processing.org/

https://natureofcode.com/

https://www.youtube.com/user/shiffman