Rapport de ALGAV

Caculer rectangle minimum convexe et cercle minimum convexe

Author :
Qiwei XIAN

Professeur :
Prof.Bui-Xuan

21 décembre 2019

Table des matières

1 Préface

1.1 Objectif

Dans ce projet, je vais présenter l'algorithme Toussaint et l'algorithme Ritter qui permet de calculer le rectangle minimum convexe et le cercle minimum convexe, en plus analyser leur performance ainsi que la qualité du rectangle minimum et le cercle minimum.

1.2 Définition, notations, structures de données utilisées

Calcule de vecteur

1. $\overrightarrow{ab}$ est représenté le vecteur depuis a vers b, $|\overrightarrow{ab}|$ est le scalaire de $\overrightarrow{ab}$.
2. $\overrightarrow{ab} \cdot \overrightarrow{cd}$ dénote le produit scalaire entre $\vec{ab}$ et $\vec{cd}$.
3. $\overrightarrow{ab} \times \overrightarrow{cd}$ dénote le produit vectoriel entre $\overrightarrow{ab}$ et $\overrightarrow{cd}$.

On dénote l'ensembles de points par $\mathcal{P}$, le nombre de points de $\mathcal{P}$ par n.
Une enveloppe convexe E de $\mathcal{P}$ est un sous-ensemble E de $\mathcal{P}$ en ordre du sens horaire tel que le polygone composé par tous les points de E peut entourner tous les points de $\mathcal{P}$.
Un rectangle Rec est représenté par une liste de quatre points. $RecMin$ est le rectangle minimum convexe. AR est un ensemble de quatre arêtes de Rec.
Un cercle Cer est représenté par un point du centre de cercle O et le rayon r. $CerMin$ est le cercle minimum convexe.

2 Algorithme Toussaint

2.1 Introduction

Donner un ensemble $\mathcal{P}$ de n points dans $\mathbb{R}$, le rectangle minimum convexe est le rectangle qui entourne tous les points de $\mathcal{P}$ dont l'aire est la plus petite.

Calculer le rectangle minimum convexe est un des problèmes classiques de la géométrie algorithmique. Pour résoudre ce problème, un majorant de $\mathcal{O}(n^2)$ est donné par l'algorithm recherche exhaustive. Il existe aussi les autres meilleurs algorithmes comme **l'algorithme Shamos**, la première fois que Michael Shamos l'a utilisé afin de calculer la diamètre dun polygone convexe en temps $\mathcal{O}(n)$ en 1978. En plus **l'algorithme Toussaint**, Godfried Toussaint a résolu beaucoup de problèmes de la géométrie algorithmique en utilisant la phase "**rotating caliper**". Dans mon projet, je cherche à calculer le rectangle minimum convexe en utilisant l'algorithme Toussaint.

Néanmoins il faut trouver l'enveloppe convexe de $\mathcal{P}$ avant d'utiliser ces algorithmes. Pour faire ce précalcule, il y a plusieurs choix possibles. Par exemple le **parcours de Graham**, il nous permet de calculer une enveloppe convexe en temps $\mathcal{O}(n \log n)$. Un autre algorithme la **marche de Jarvis**, il a aussi une excellente complexité en $\mathcal{O}(nh)$ où h représente le nombre de sommets de l'enveloppe convexe.

Dans ce projet, je choisis la marche de Jarvis afin de précalculer l'enveloppe convexe de $\mathcal{P}$. Dès quobtenir l'enveloppe, je vais utiliser rotating caliper pour chercher les autres points de l'enveloppe qui se trouve aussi dans le rectangle. Après je peux calculer les sommets de rectangle en utilisant ces quatre points. La complexité théorique est $\mathcal{O}(n + r)$ où r est la complexité de la marche de Jarvis.

2.2 Présentation de l'algorithme Toussaint

Afin de calculer les sommets A, B, C, D de rectangle, on a besoin d'abord de trouver α qui est colinéaire avec une arête de rectangle et les points A', B', C', D' de $\mathcal{B}$ qui est sur les arêtes de $RecMin$.

Lemma 1 if Rec est $RecMin$, alors $\exists \mathcal{B}$, $\mathcal{B} \in$ E tel que $\forall b, b \in \mathcal{B}, \exists c, ar, c \in Rec, ar \in AR$, $\overrightarrow{bc}$ sont colinéaire avec ar.

Lemma 2 il existe une arête α de $RecMin$, α passe une arête du polygone minimum convexe.

2.2.1 Etape 1 calculer A', B', C', D'

Pour A', on peut énumérer chaque point i de E, $A' = E_i$, et $\alpha = \overrightarrow{E_{i+1}E_i}$

Pour B', il est le plus ouest point tel que $|\alpha \cdot \overrightarrow{B'A'}|$ est maximum, il exprime $\angle\alpha\overrightarrow{B'A'}$ est le plus grand. Comme E est en ordre du sens horaire, on énumère chaque point E_r de E, $|\alpha \cdot \overrightarrow{E_rA'}| \leqslant |\alpha \cdot \overrightarrow{E_{r+1}A'}|$, si une fois $|\alpha \cdot \overrightarrow{E_rA'}| > |\alpha \cdot \overrightarrow{E_{r+1}A'}|$, alors $B' = E_r$

Pour C', il est le point antipodal de A', la distance entre A' et C' est plus grande que A' et les autres points, donc $|\alpha \times \overrightarrow{E_uA'}|$ est maximum. Comme E est en ordre du sens horaire, si on énumère chaque point E_u de E, au début $|\alpha \times \overrightarrow{E_uA'}| \leqslant |\alpha \times \overrightarrow{E_{u+1}A'}|$, si une fois $|\alpha \times \overrightarrow{E_uA'}| > |\alpha \times \overrightarrow{E_{u+1}A'}|$, alors $C' = E_u$

Le dernier point D' est le plus est point, $\angle\alpha D'$ est le plus petit. On énumère chaque point E_l de E, comme E est en ordre du sens horaire, au début $|\alpha \cdot \overrightarrow{E_lA'}| \geq |\alpha \cdot \overrightarrow{E_{l+1}A'}|$, si une fois $|\alpha \cdot \overrightarrow{E_lA'}| < |\alpha \cdot \overrightarrow{E_{l+1}A'}|$, alors $D' = E_l$

2.2.2 Etape 2 calculer l'aire $\mathcal{S}$ de rectangle

Pour la hauteur de rectangle h, on suppose que le point C'_α est le projecteur de C' sur α, h = $|\overrightarrow{C'_\alpha C'}|$
$\mathcal{S}_{\text{parallélogramme}} = |\vec{\alpha} \times \overrightarrow{C'A'}|$.
h = $\frac{\mathcal{S}_{\text{parallélogramme}}}{|\vec{\alpha}|}$

Pour la largeur de rectangle w, on suppose que B'_α est le projecteur de B' sur α et D'_α est le projecteur de D' sur α, w = $|\overrightarrow{B'_\alpha D'_\alpha}|$
$\overrightarrow{B'_\alpha A'} = \frac{\alpha \cdot \overrightarrow{B'A'}}{|\alpha|}$
$\overrightarrow{D'_\alpha A'} = \frac{\alpha \cdot \overrightarrow{D'A'}}{|\alpha|}$
$|\overrightarrow{B'_\alpha D'_\alpha}| = |\overrightarrow{D'_\alpha A'} - \overrightarrow{B'_\alpha A'}|$
Donc, $\mathcal{S} = h \times w$. Si $\mathcal{S} < \mathcal{S}_{RecMin}$, on va continuer à l'étape 3 afin de construire le nouveau $RecMin$ sinon on essaie le A' suivant.

2.2.3 Etape 3 calculer A, B, C, D

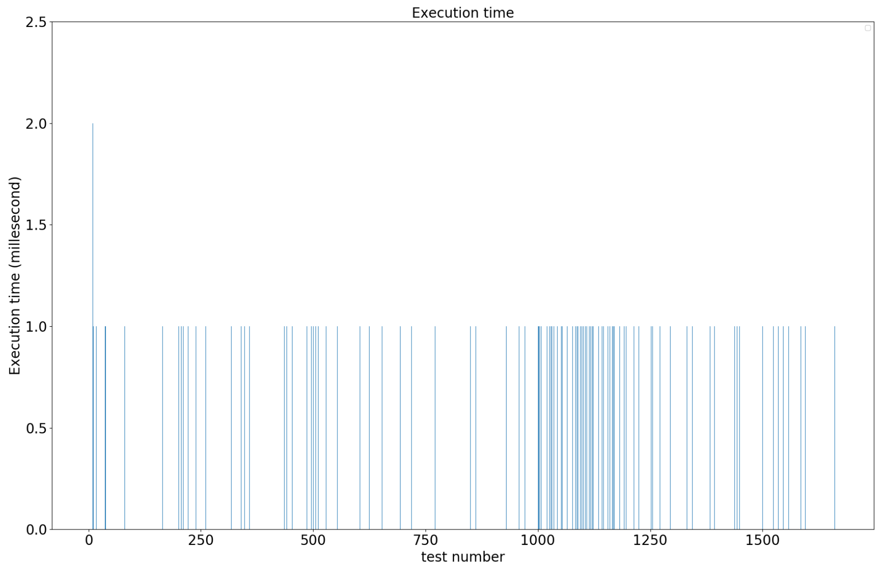
Dès quon obtient A', B', C', D', α, $\overrightarrow{B'_\alpha A'}$, $\overrightarrow{B'_\alpha D'_\alpha}$, $\overrightarrow{C'_\alpha C'}$, on peut facilement calculer A, B, C, D.
A est le point B'_α, $B'_\alpha = A' - \alpha \cdot \frac{\overrightarrow{B'_\alpha A'}}{|\alpha|}$
B est le point suivant de A au sens de horaire, $B = B'_\alpha + \overrightarrow{B'_\alpha B'} \cdot \frac{\overrightarrow{C'_\alpha C'}}{|\overrightarrow{B'_\alpha B'}|}$
C est le point suivant de B au sens de horaire, $C = B + \overrightarrow{B'_\alpha A'} \cdot \frac{\overrightarrow{B'_\alpha D'_\alpha}}{|B'_\alpha A\prime|}$
Le dernier point $D = A - B = C + \overrightarrow{BB'_\alpha}$

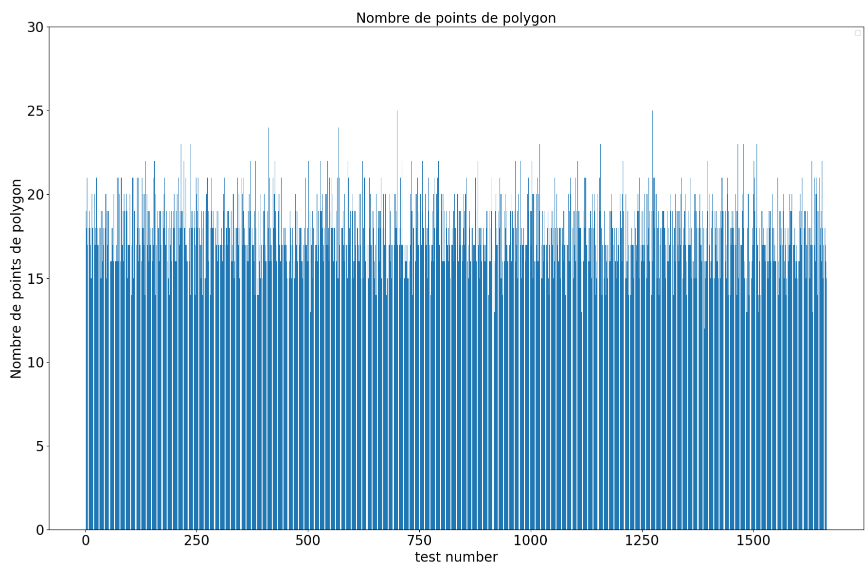
Algorithm 1 : Pseudocode de l'algorithme Toussaint

$\mathcal{P} \leftarrow$ Ensemble de points ;
E $\leftarrow$ Jarvis($\mathcal{P}$) ;
$u \leftarrow 1$;
$l \leftarrow 1$;
$r \leftarrow 1$;
$minAire \leftarrow \infty$;
for ***each*** *arête* $\overrightarrow{E_{i+1}E_i}$, $0 \leqslant i < |E|$ **do**
  $A' \leftarrow E_i$;
  $\alpha \leftarrow \overrightarrow{E_{i+1}E_i}$;
  while $|\alpha \times \overrightarrow{E_u E_i}| \leqslant |\alpha \times \overrightarrow{E_{u+1}E_i}|$ **do**
    $u \leftarrow (u+1) \bmod |E|$;
  end while
  $C' \leftarrow E_u$;
  while $|\alpha \cdot \overrightarrow{E_r E_i}| \leqslant |\alpha \cdot \overrightarrow{E_{r+1}E_i}|$ **do**
    $r \leftarrow (r+1) \bmod |E|$;
  end while
  $B' \leftarrow E_r$;
  if $i == 0$ **then**
    $l = r$;
  end if
  while $|\alpha \cdot \overrightarrow{E_l E_i}| \leqslant |\alpha \cdot \overrightarrow{E_{l+1}E_i}|$ **do**
    $l \leftarrow (l+1) \bmod |E|$;
  end while
  $D' \leftarrow E_l$;
  $\overrightarrow{B'_\alpha A'} \leftarrow \frac{\alpha \cdot \overrightarrow{B'A'}}{|\alpha|}$;
  $\overrightarrow{D'_\alpha} \leftarrow \frac{\alpha \cdot \overrightarrow{D'A'}}{|\alpha|}$;
  $height \leftarrow |C'_\alpha C'| \leftarrow \frac{\alpha \times \overrightarrow{C'A'}}{|\alpha|}$;
  $width \leftarrow \overrightarrow{B'_\alpha A'} - \overrightarrow{D'_\alpha A'}$;
  $aire \leftarrow width \times height$;
  if $aire < minAire$ **then**
    $minAire \leftarrow aire$;
    $A \leftarrow A' - \alpha \cdot \frac{\overrightarrow{B'_\alpha A'}}{|\alpha|}$;
    $B \leftarrow B'_\alpha + \overrightarrow{B'_\alpha B'} \cdot \frac{\overrightarrow{C'_\alpha C'}}{|\overrightarrow{B'_\alpha B'}|}$;
    $C \leftarrow B + \overrightarrow{B'_\alpha A'} \cdot \frac{\overrightarrow{B'_\alpha D'_\alpha}}{|B'_\alpha A\prime|}$;
    $D \leftarrow C + (A - B)$;
  end if
end for
return A, B, C, D ;

2.3 Performance de l'algorithme

Le graphe dessous est le temps d'exécution pour chaque teste de VAROUMAS.

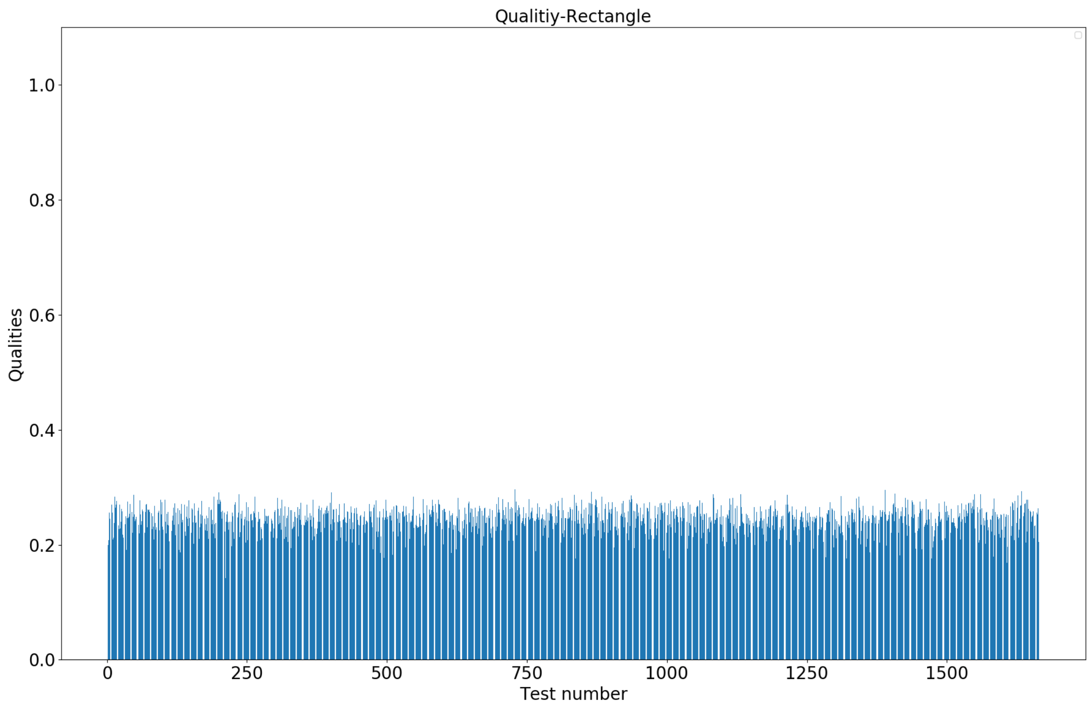




Pour chaque cas de test, il y a 255 points aléatoire, après calculer l'enveloppe convexe en utilisant Jarvis, il reste seulement h points de l'enveloppe convexe $12 \leqslant h \leqslant 25$ dans tous les tests. En fonction de la complexité théorique de l'algorithme $\mathcal{C} = \mathcal{O}(nh) + \mathcal{O}(h)$. Donc le temps d'exécution n'est pas très évident. Pour la plupart de tests, le temps d'exécution t, $0 < t <= 1ms$. Le test-10 prend le plus temps 2 ms.

2.4 Résultat du test

Le graphe dessous est le résultat de qualité associé à chaque teste de VAROUMAS.



$\text{Qualité}_{rectangle} = \frac{\text{aire rectangle}}{\text{aire polygone}} - 100\%$

L'abscisse est représentée le numéro de test et l'ordonnée est la qualité de rectangle minimum convexe. On peut voir que l'aire du rectangle minimum convexe est toujours plus grand que l'aire de polygone convexe.

3 Algorithme Ritter

3.1 Introduction

Donner un ensemble $\mathcal{P}$ de n points dans $\mathbb{R}$, le cercle minimum convexe est le cercle qui entourne tous les points de $\mathcal{P}$ à la fois son aire est plus petit possible.
Pour calculer le cercle minimum convexe, il existe une majoration de $\mathcal{O}(n^4)$ en utilisant la recherche exhaustive. On énumère trois points pour déterminer un cercle, et vérifier s'il peut entourner tous les autres points. Dans ce projet, j'utilise l'algorithme Ritter et cela me permet de résoudre ce problème en $\mathcal{O}(n)$.

3.2 Présentation de l'algorithme Ritter

Algorithme Ritter est proposé en 1990 par Jack Ritter afin de calculer efficacement la sphère minimum convexe. Cet algorithme peut évidemment calculer le cercle.

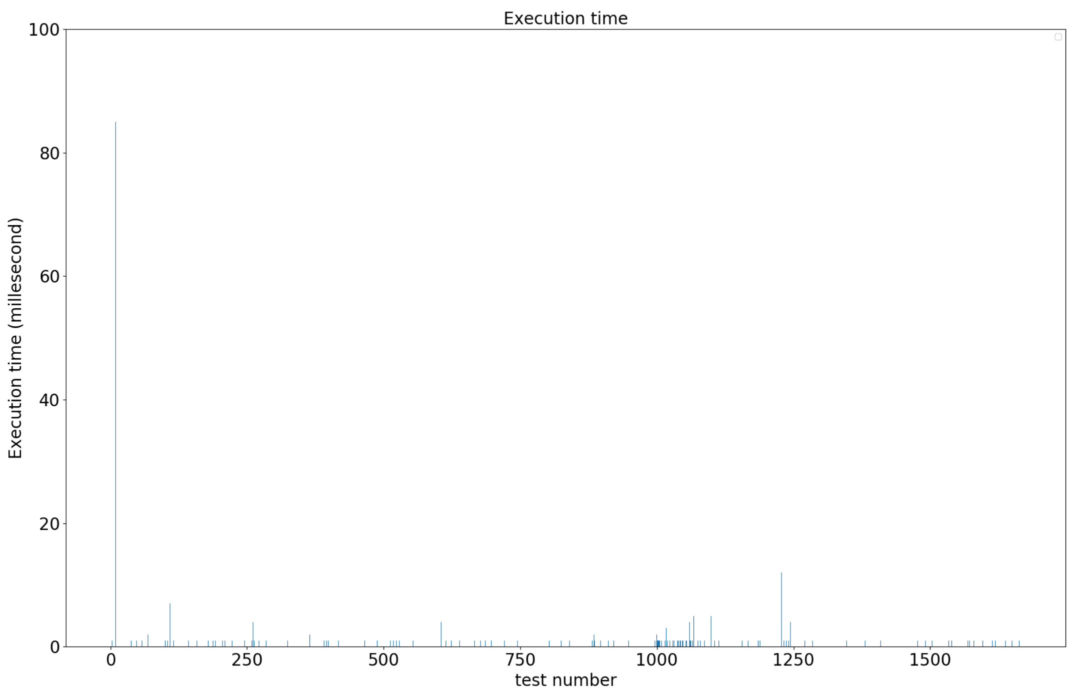
Algorithm 2 : Pseudocode de l'algorithme Ritter

```
𝒫 ← Ensemble de points ;
d ← point aléatoire ;
p ← le point plus éloigné de d ;
q ← le point plus éloigné de p ;
c ← (p+q)/2 Cer ← Cercle(c, |cp|) ;
𝒫 ← 𝒫 \ {p, q} ;
while P ≠ ∅ do
    for each Point s, s ∈ 𝒫 do
        if s ∈ Cer then
            𝒫 ← 𝒫 \ {s} ;
        end if
        else
            |st| ← |cp| + |cs| ;
            |sc'| ← |st|/2 ;
            c' = c + cs→ · (|sc| − |sc'|)/(|sc| + |cp|) ;
            Cer ← new Cercle(c', |sc'|) ;
        end if
    end for
end while
return Cer ;
```

1. Choisir un point d aléatoirement, $d \in \mathcal{P}$, chercher un point p tel que la distance $|pd|$ est maximum.
2. Chercher le point q, $q \in \mathcal{P}$ tel que la distance $|pq|$ est maximum. Construire un cercle Cer par le point milieu c entre p et q et le rayon $|cp|$.

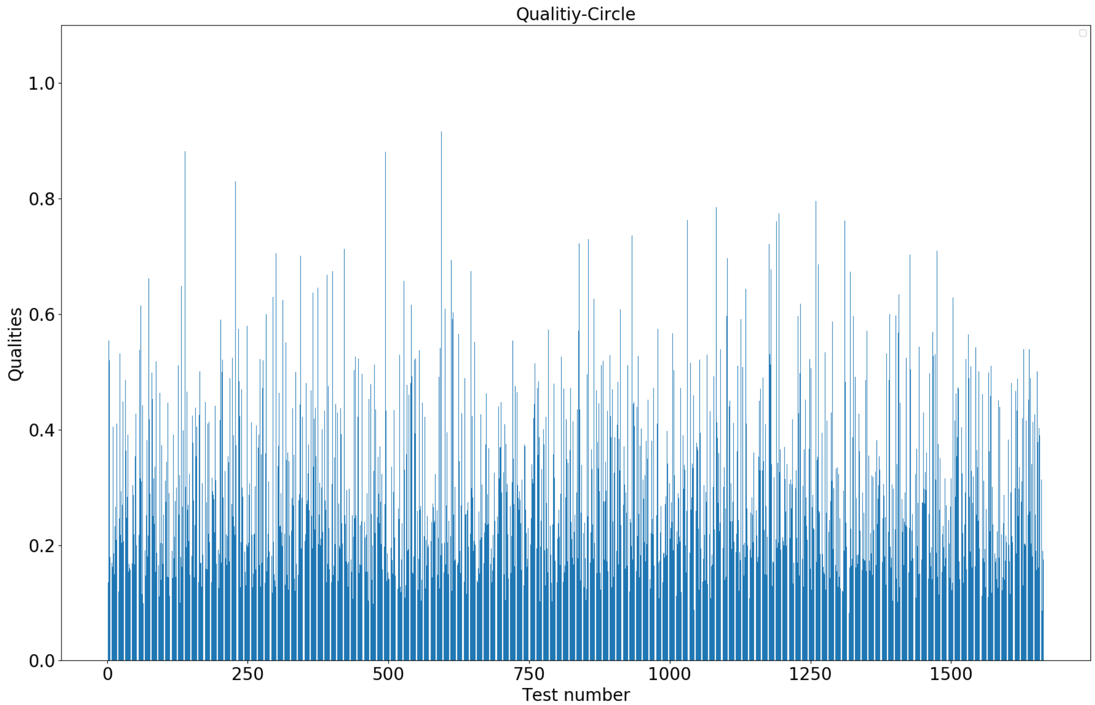
3. Enumérer tous les points s, $s \in \mathcal{P}$. Si s est couvert par Cer, enlève s directement.
4. Sinon, on chercher un point t sur Cer tel que t est le plus éloigné de s, $|st|$ passe $|sc|$. En plus calculer le point c' qui est le point milieu entre s et c. Construire un nouveau cercle dont le rayon est $|c's|$ et le centre est c'.
   Tips :
   $c' = c + \overrightarrow{cs} \cdot \frac{|cc'|}{|st|} = c + \overrightarrow{cs} \cdot \frac{|sc|-|sc'|}{|sc|+|cp|} = c + \overrightarrow{cs} \cdot \frac{|sc|-|st|\times\frac{1}{2}}{|sc|+|cp|}$
5. Remplacer Cer par ce nouveau cercle centré en c'.
6. Tant que $\mathcal{P} \neq \varnothing$, retourne l'étape 3.

3.3 Performance de l'algorithme



La complexité de l'algorithme Ritter est linéaire. Il y a n points dans $\mathcal{P}$, au plus il fait seulement n fois boucle pour enlever tous les points. Donc c'est linéaire. Néanmoins, cet algorithme utilise la fonction probabiliste afin d'optimiser sa performance, donc il nous permet seulement d'obtenir un résultat dapproximation au lieu d'une réponse parfaite. Dans ces 1664 cas des tests, le test-10 coûte toujours le plus de temps 83 ms, mais les autres prennent seulement quelques millisecondes.

3.4 Résultat du test



Le résultat obtenu par l'algorithme Ritter est approximatif, donc l'aire de cercle change toujours souvent, la qualité de cercle convexe n'est pas définitive.