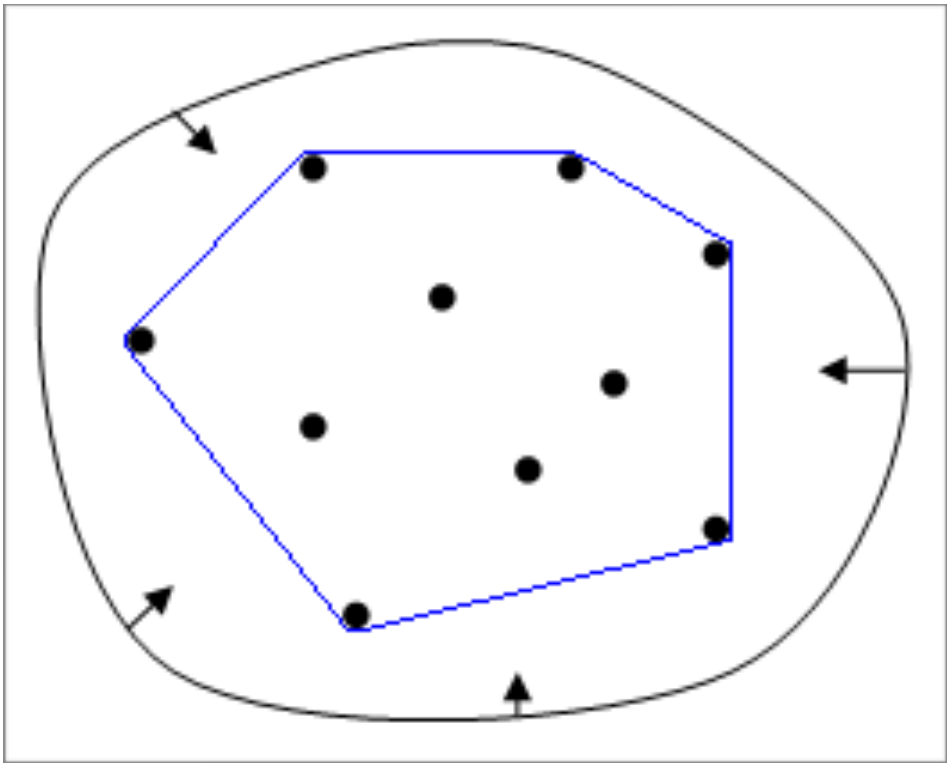
YU Peng

Projet de Mathématiques

Affichage de l'enveloppe convexe



ESIREM DIJON
Ecole d'Ingénieurs
Matériaux InfoTronique

UNIVERSITÉ DE BOURGOGNE

But du projet

Le but de ce projet est l'affichage de l'enveloppe convexe d'un nombre variable de points dans le plan euclidien usuel.

Seulement les points définissant le contour de l'enveloppe convexe seront retenus.

Déroulement du projet

Introduction

- Ensemble convexe
- Enveloppe convexe

Aspects algorithmiques

- Parcours de Graham
- Attaque par force brute
- Diviser pour régner
- Marche de Jarvis
- Algorithme de Melkman

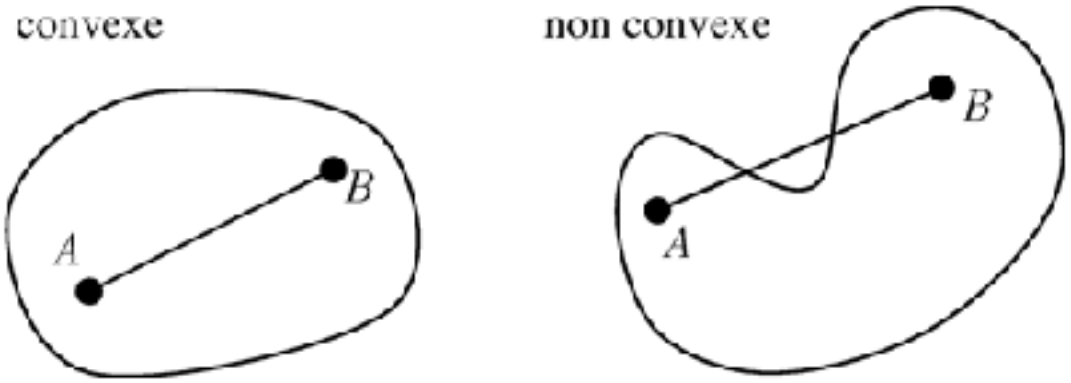
Parcours de Graham

Programmation

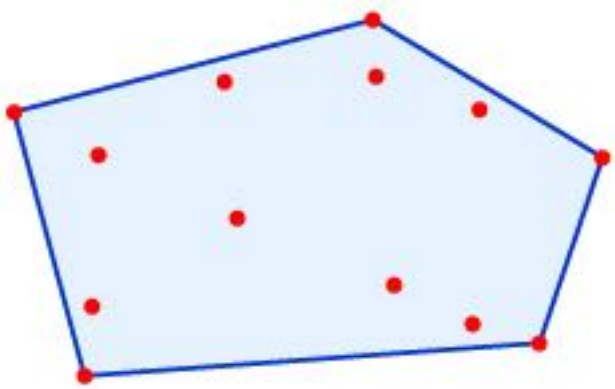
Conclusion

Introduction

1. Ensemble convexe : Un ensemble C est dit convexe lorsque, pour tous x et y de C, le segment [x, y] est tout entier contenu dans C.



2. Enveloppe convexe : Sur le plan, il y a un ensemble de points S. Le nombre de son ensemble convexe est infini. L'intersection de tous ces ensembles convexes s'appelle l'enveloppe convexe d'ensemble S.



Ou on peut dire qu'on veut utiliser les droites pour encercler tous les points sur le plan. Et la surface de polygone qui est composé par des droites est le minimum. On dit alors ce polygone est l'enveloppe convexe de ces points sur le plan.

Aspects algorithmiques

Le calcul de l'enveloppe convexe d'un ensemble de points est un problème classique en géométrie algorithmique, Plusieurs algorithmes ont été inventés pour résoudre ce problème, leur complexité varie :
n - nombre de point
H - nombre de point sur l'enveloppe convexe

1. Parcours de Graham (Durée : $O(n\log n)$)

   La méthode que j'ai utilisé dans ce projet. L'explication précise en dessus.

2. Attaque par force brute (Durée : $O(n^3)$)

   Il permet d'utiliser deux points pour déterminer un segment. Si les autres points sur le plan Euclidien usuel sont toujours sur le même côté du segment. Les deux points qui constituent ce segment sont donc des points sur l'enveloppe convexe.

Les étapes :

1) On va tracer les segments pour laisser tous les points sont relier. On a donc n(n-1)/2 segments.

2) Pour chaque segment, on va chercher si les (n-2) autres points sont sur le même côté de ce segment.

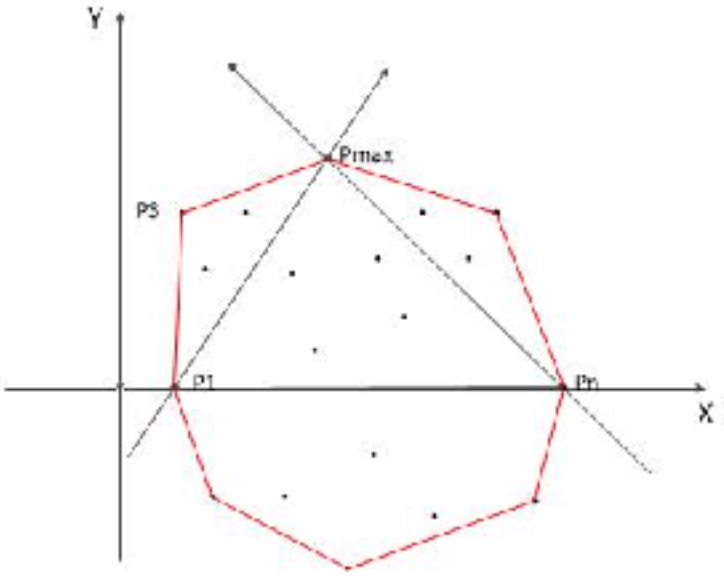
3. Diviser pour régner (Durée : O(nlogn))

On veut séparer cette question par beaucoup de petites questions qui ont les mêmes structures. Si ces questions sont encore compliqués pour nous. On va continuer de séparer ces questions. Finalement, On va utiliser algorithme récursif pour trouver les résultats de chaque question et reconstituer ces questions pour obtenir le résultat de question originale.

Les étapes :

1) On va mettre tous les points dans le système cordonnées de 2D. Le minimum et le maximum d'abscisse de point sont nécessairement sur l'enveloppe convexe. On les appelle P1 et Pn. Le segment P1Pn sépare l'enveloppe convexe par deux parties.

2) Pour la première partie, on va chercher un point. La distance entre ce point et le segment P1Pn est plus longue que les autres. C'est donc Pmax. Il est sur l'enveloppe convexe.



3) On va tracer les segments P1Pmax et PnPmax qui séparent cette partie par trois plus petites parties.

4) Répéter 2) et 3)

5) Pour la deuxième partie, on fait la même chose.

4. Marche de Jarvis (Durée : O(nH))

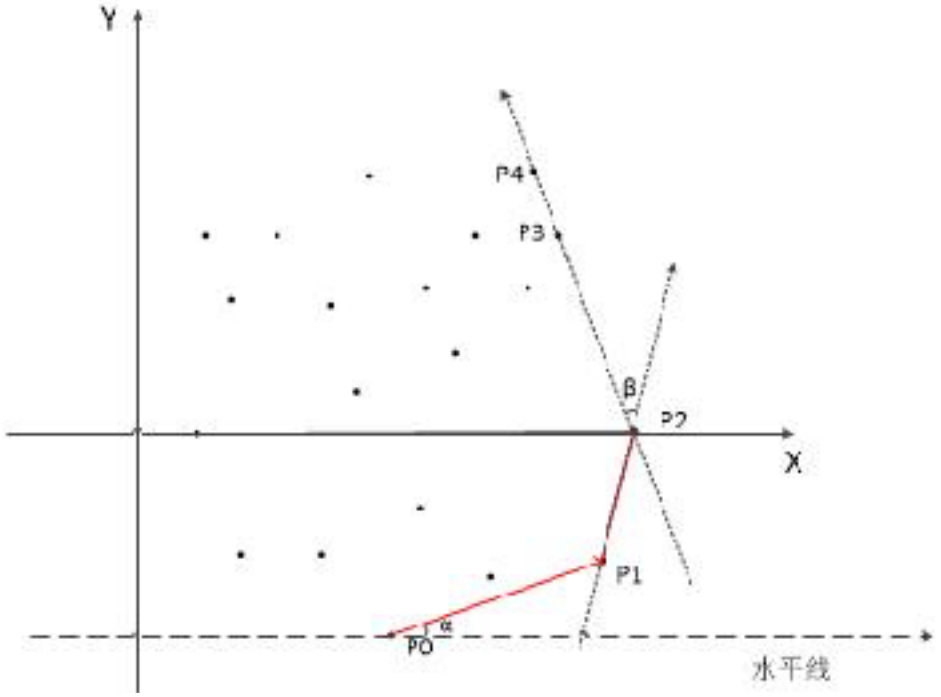
logique de sa pensée :

1) Trouver le point qui a le minimum d'ordonnée. Il est nécessairement sur l'enveloppe convexe. On l'appelle P0.

2) Selon la manière inverse d'horloge, on va commencer par P0 pour trouver les points d'enveloppe convexe un par un.

3) Comment trouver le prochain point? Par l'angle. On suppose qu'on a déjà trouvé les points P1, P2, P0. Les points restes qu'on les appelle Pi qui constituent le vecteur P2Pi avec P2. On a donc l'angle β entre le vecteur P2Pi et le vecteur P1P2. Quand il y a un point qui peut laisser l'angle β plus petit, ce point qu'on l'appelle P3 est sur l'enveloppe convexe.

4) Continuer 3)

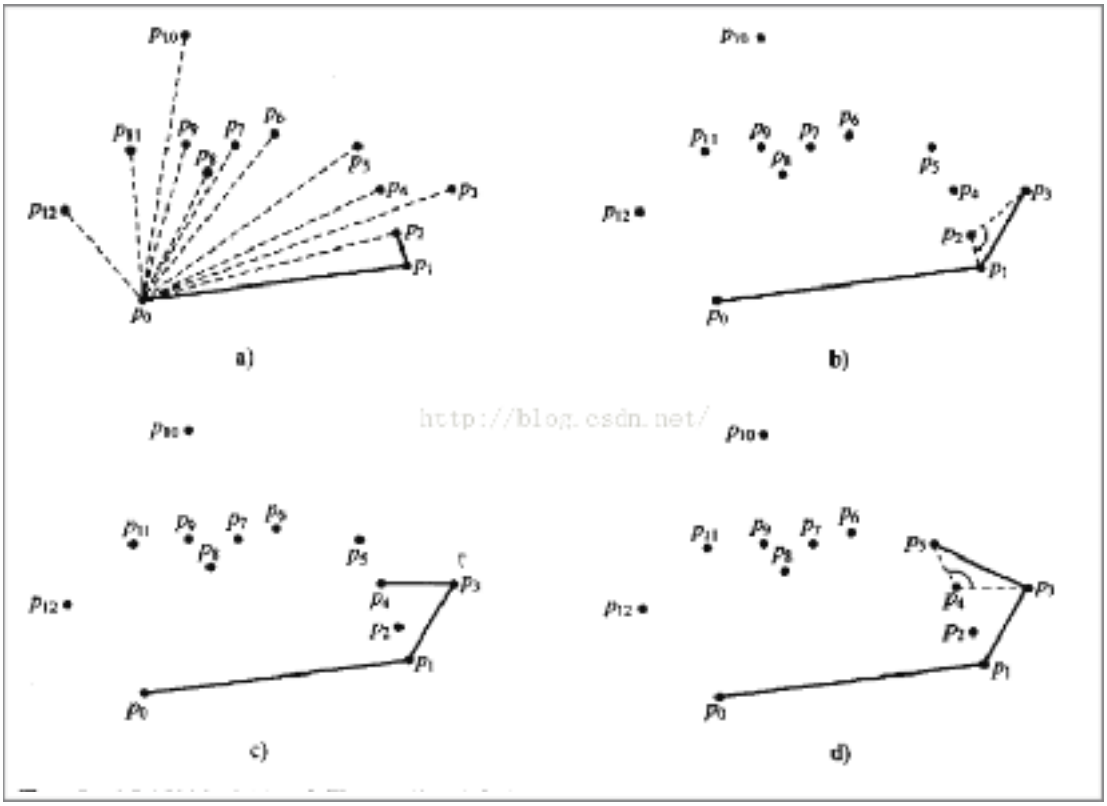


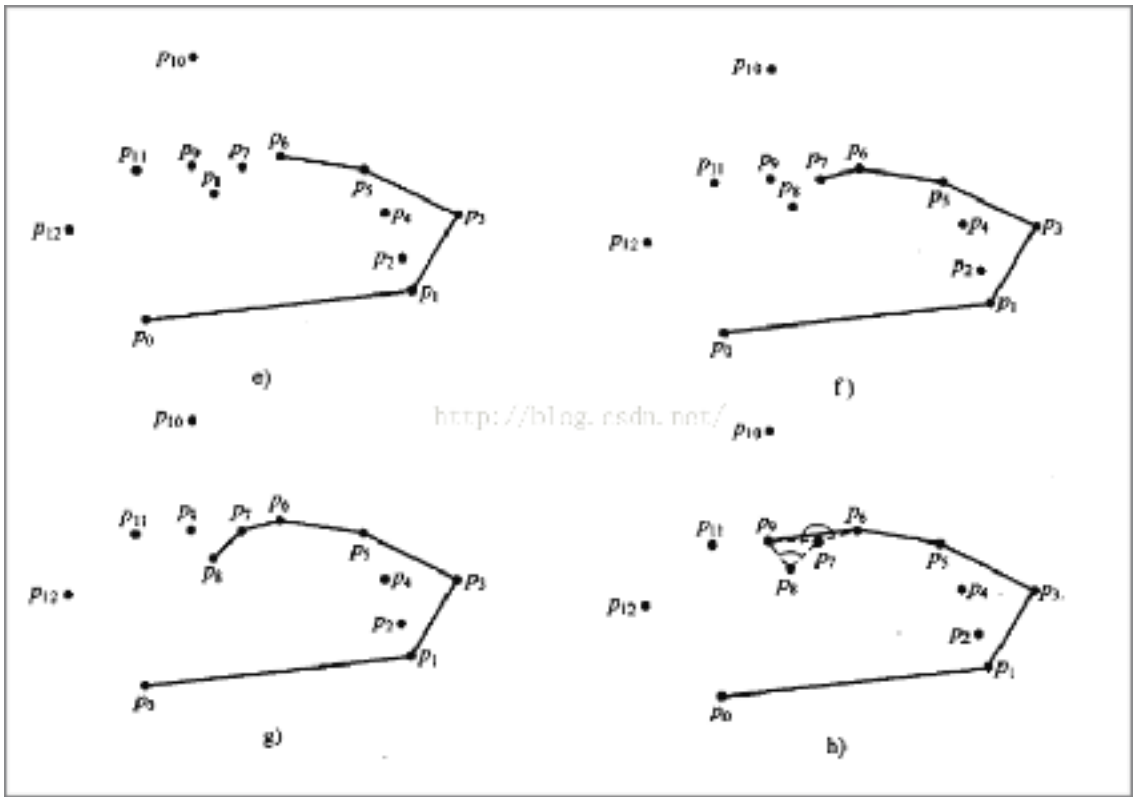
5. Algorithme de Melkman : Le meilleur méthode pour trouver les points sur l'enveloppe convexe. Mais la logique est compliquée. Il n'y a pas beaucoup de documents sur l'internet. Donc, on n'utilise pas cette méthode.

Parcours de Graham

L'idée est un peu près comme 'Marche de Jarvis'. Tout d'abord, il faut trouver un point sur l'enveloppe convexe. Et puis, On commencer chercher les autres points par le point qu'on a trouvé, selon le sens inverse des aiguilles d'une montre. Mais cette méthode n'utilise pas l'angle compris.

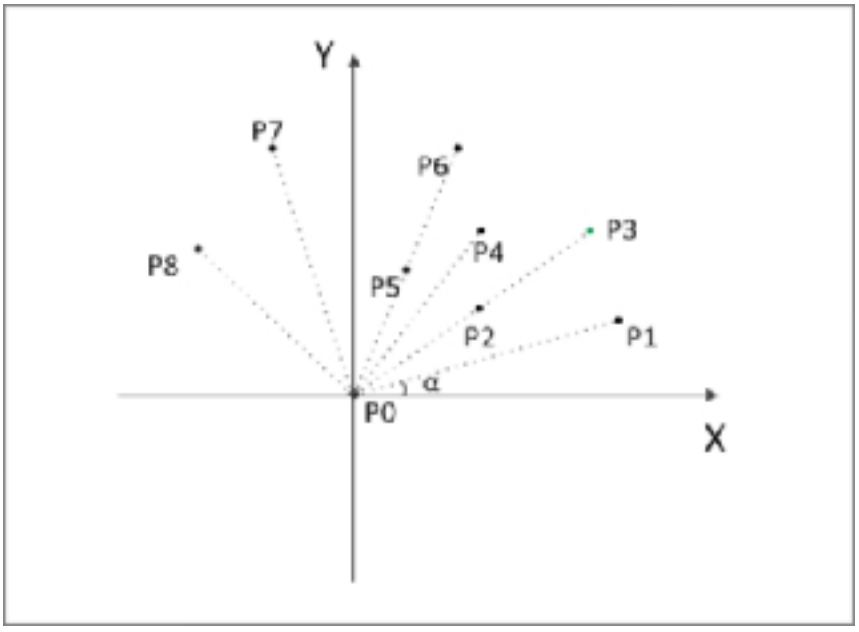
Les étapes :





1) Comme la figure a), on doit trouver le point initial P0. L'ordonnée de P0 est la plus petite par rapport aux autres points. Si on trouve plusieurs points qui satisfont la condition, on va chercher l'abscisse de point qui est le plus petite entre ses points. Les points qui ont la même coordonnée, on les regarde comme un seul point. On va dire que P0 est nécessairement sur l'enveloppe convexe.

2) On doit regarder P0 comme le point d'origine du plan. Comme le figure suivant.



3) On doit relier les restes points qu'on les appellent Pi. On va donc obtenir des angles ∠PiP0X. Il faut comparer les angles et les mise en ordre selon la valeur croissante. On a donc P1, P2 … Pn comme figure a).

Et le point P1 et Pn sont nécessairement sur l'enveloppe convexe.

4) Il faut créer un dépôt pour enregistrer tous les points sur l'enveloppe convexe. Maintenant, on sait déjà les deux premiers points sur l'enveloppe convexe P0 et P1. On pose P0 et P1 dans le dépôt. On utilise P2 comme le nouveau point initiale. Et on pose aussi P2 dans le dépôt. On relie P0P1 et P1P2 comme figure a).

5) On regarde le prochain point comme le point initial (ici, c’est le P3).

6) On relie le point de haut du dépôt et le point initial (ici, on relie P2 et P3). On va donc obtenir un droite L. Et le droite dernier, on l’appelle M (ici, c’est P1P2). Maintenant, on veut chercher comment tourner le droite M pour laisser lui coïncider avec le droite L. Si on doit le tourner à droite, exécute l’étape 7). Si on doit le tourner à gauche ou M et L sont déjà coïncidents, exécute l’étape 8).

7) Si on doit tourner M à droite, ça veut dire que le le point de haut du dépôt n’est pas le point sur l’enveloppe convexe et on l’annule. Recommence exécuter l’étape 6).

8) Si on doit le tourner à gauche ou M et L sont déjà coïncidents, ça veut dire que le point initial peut être le point sur l’enveloppe convexe et on le pose dans le dépôt. Et on va exécuter l’étape 9).

9) Est ce que le point initiale est Pn. Si oui, On a fini, Bravo! Sinon, retourner à l’étape 5).

Programmation

Le code ne sera pas écrit explicitement dans cette partie, je vais simplement étudier la logique de programmation que j’ai utiliser et détailler les quelques fonctionne que j’ai créées.

Logique :

- Trouver le premier point pour commencer l'algorithme Graham.

Pseudo-code :

```
Pour tous les points sur le plan
        Si 'y' d'un point < 'y' de minimum
            minimum = ce point là
        Ou si 'y' d'un point = 'y' de minimum et 'x' du point < 'x' de minimum
            minimum = ce point là
```

- Regarder le point qui a le minimum d'ordonné comme le premier point.

J'ai écrit la fonction 'Swap' pour échange le minimum que j'ai trouvé et le premier point.

```
tmp <— premier point
    premier point <— minimum
    minimum <— tmp
```

- Relier les autres points avec le premier point par les segments et calculer les angles entre ces segments et l'axe de X.

J’ai écrit la fonction ‘Angle’ pour calculer l’angle. L’idée est qu’on utilise le cordonnée du point pour trouver le valeur de cosinus et on a ‘acos’ pour trouver l’angle par le valeur de cosinus.

- Selon des angles du plus petit vers le plus grand, on va trier les points.

J’ai écrit la fonction ‘QSort’ pour trier l’angle. J’utilise l’algorithme Tri rapide (Quick Sort en anglais). Il est fondé sur la méthode de conception diviser pour régner. Il est généralement utilisé sur des tableaux, mais peut aussi être adapté aux listes.

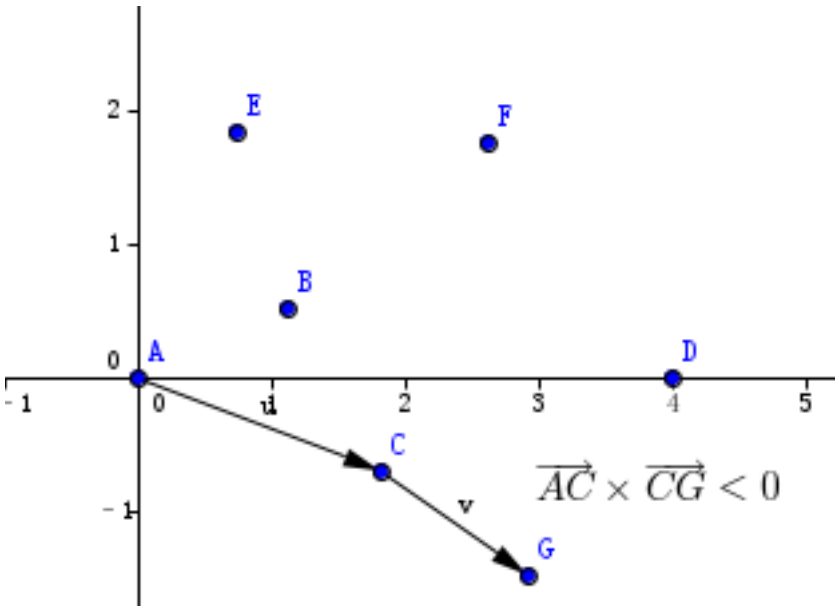
L’idée est :

1) Choisir un valeur comme pivot.
2) Tous les valeurs qui sont plus petites que pivot, on les déplace au gauche de pivot. Et tous les valeurs qui sont plus grandes que pivot, on les déplace au droite de pivot. Après cette opération, la position de pivot est sa dernière position dans le tableau.
3) Pour les deux ensembles d’enfant qui sont situés dans la gauche et la droite de pivot, on va continuer répéter sans cesse les étape 1) et 2) jusqu’à ce que tous les ensembles d’enfant ont seulement une valeur.

- Tester le point qui est situé sur l’enveloppe convexe ou pas. Si oui, on le pose dans le dépôt.

Si le point initial et les deux points qui sont situé dans le plus haut de dépôt constituent un angle rentrant(angle supérieur), on va annuler le plus haut point de dépôt et recommencer faire la même chose jusqu'à ce qu'il y a plus de points dans le dépôt.

Pour tester angle supérieur ou inférieur, j'ai créé une fonction 'Multi'.



Selon les connaissances de produit vecteur de deux dimension, si le produit vecteur de AC et CG (|***AC***|| ***CG***|sin<***AC***,***CG***>) inférieur à 0, on va dire que l'angle ACG est supérieur à 180° (On tourne AC vers CG dans le sens inverse de la rotation). Cet angle est un angle rentrant. si le produit vecteur de AC et CG supérieur à 0, ça veut dire que cet angle est un angle saillie.

Pseudo-code de fondino Multi

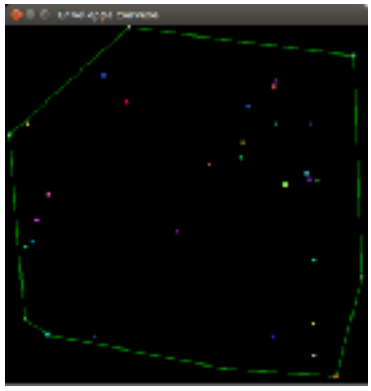
```
Pour les point P1, P2 et P3
k = (x1 - x0)(y2 - y1) - (x2 - x1)(y1 - y0)
Si k < 0
      retourner 1
retourner 0
```

Pseudo-code pour trouver les points sur l'enveloppe convexe

```
//Pour les premier trois point(P1, P2 et P3)
Pour (i=0; i<= 2; i++)
      stack[i] = i; // on va mettre ces points dans le
                        dépôt
End
t = 2 //t est le nombre de point dans le dépôt

Pour(i=3; i<le nombre de point sur le plan; i++)
Tant que Multi(P1, P2 et P3==1 et t>1
      t= t-1;
End
t=t+1;
stack[++t] = I;
End
```

Finalement, j'obtiens la résultat comme ce figure.



Conclusion

La plus grande difficulté de ce projet était de tester si le point est sur l'enveloppe convexe ou pas. Il faut bien comprendre les connaissances de produit vecteur de deux dimension pour déterminer si l'angle est rentrant ou saillie.

Et puis, il faut aussi comprendre l'utilisation de dépôt (Quand on doit mettre les points dans le dépôt et quand on doit annuler le point dans le dépôt).

Pour la partie informatique, la difficulté majeure était de « s'approprier » le programme et de pouvoir l'utiliser au mieux afin d'y rajouter notre propre code. Il est toujours difficile de modifier et de compléter un code déjà écrit.

Au final, ce projet me a permis de comprendre et de pouvoir désarmais utiliser correctement l'enveloppe convexe ainsi que les méthode de mathématiques comme la méthode de conception diviser pour régner etc.

Les libraires OpenGL sont complètes et permettent de modéliser graphiquement et de manière simple en deux et trois dimensions. Certains logiciels tel que Blender l'utilisent pour gérer leurs interfaces.

Sources

1. https://www.youtube.com/watch?v=QYrpHE8iDGg&t=259s
2. https://fr.wikipedia.org/wiki/Enveloppe_convexe
3. http://blog.csdn.net