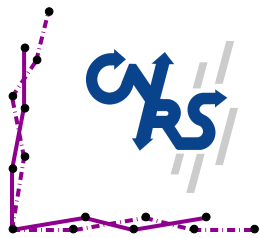


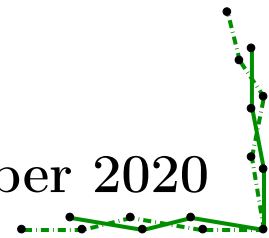
# Algorithmique Avancée

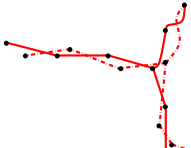
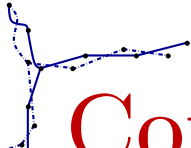
<http://www-apr.lip6.fr/~buixuan/algav2020>

Binh-Minh Bui-Xuan




PARIS, November 2020



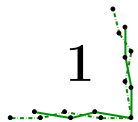
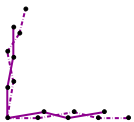


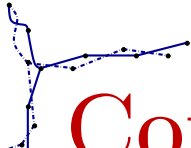
# Cours 6 : enveloppe convexe



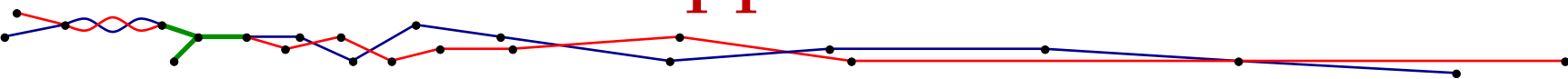
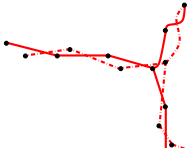
RAPPEL COURS + TME 5 :

- collision : cas le plus simple est la collision des cercles
- problème CERCLEMIN : nuage de points  $\rightarrow$  cercle
- algorithme naïf : complexité  $O(n^4)$
- techniques : incrémental (Ritter), précalcul (Akl-Toussaint)





# Cours 6 : enveloppe convexe

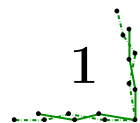
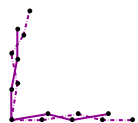


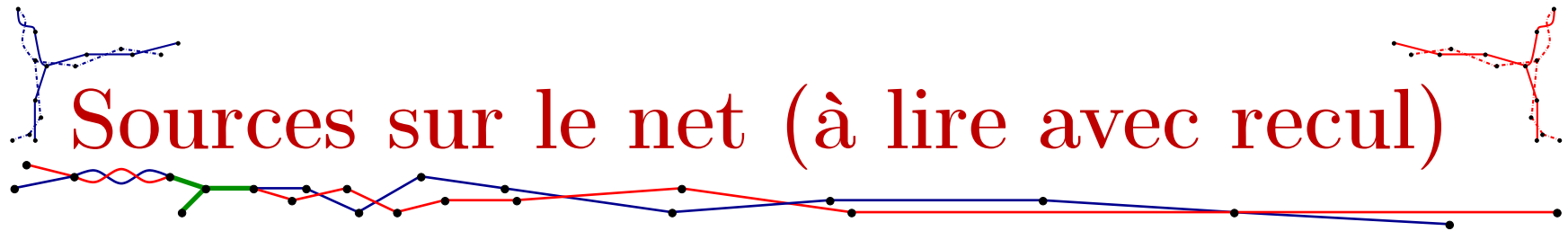
RAPPEL COURS + TME 5 :

- collision : cas le plus simple est la collision des cercles
- problème CERCLEMIN : nuage de points  $\rightarrow$  cercle
- algorithme naïf : complexité  $O(n^4)$
- techniques : incrémental (Ritter), précalcul (Akl-Toussaint)

AUJOURD'HUI :

- collision : cas de polygones convexes (esthétique !)
- problème ENVCONVEXE : nuage de points  $\rightarrow$  polygone convexe
- algorithme naïf : complexité  $O(n^3)$
- techniques : précalcul (pixel, Akl-Toussaint), décomposition
- algorithmes : Jarvis, Graham (+variants), Chan, QuickHull



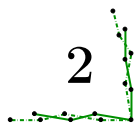
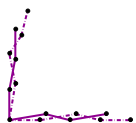


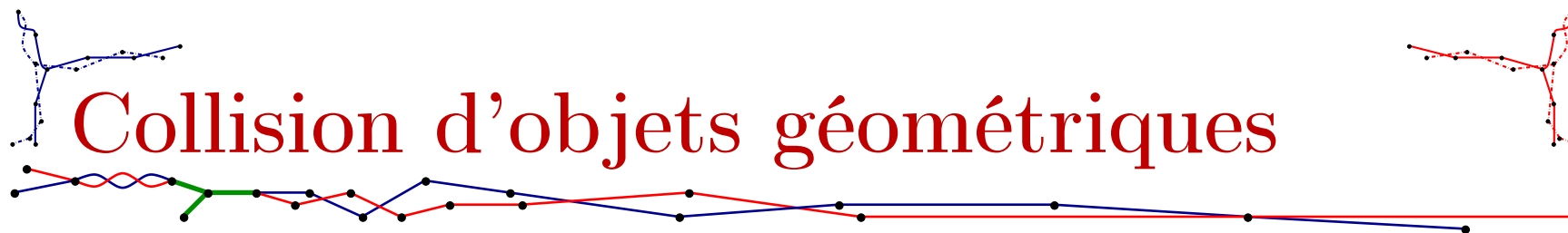
## GAME DEVELOPMENT :

– ici: <https://gamedevelopment.tutsplus.com/tutorials/collision-detection-using-the-separating-axis-theorem--gamedev-169>

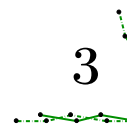
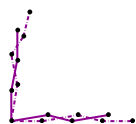
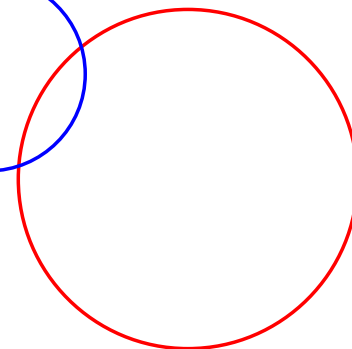
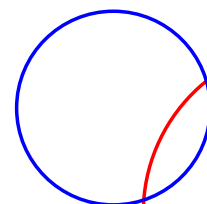
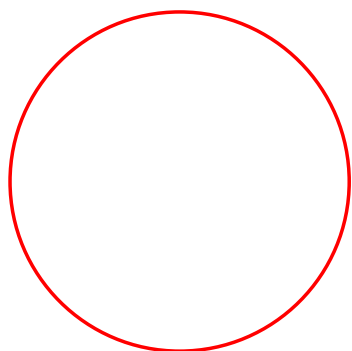
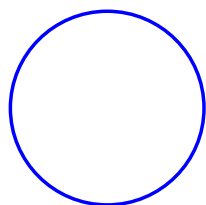
## WIKIPEDIA :

– [http://en.wikipedia.org/wiki/Convex\\_hull\\_algorithms](http://en.wikipedia.org/wiki/Convex_hull_algorithms)





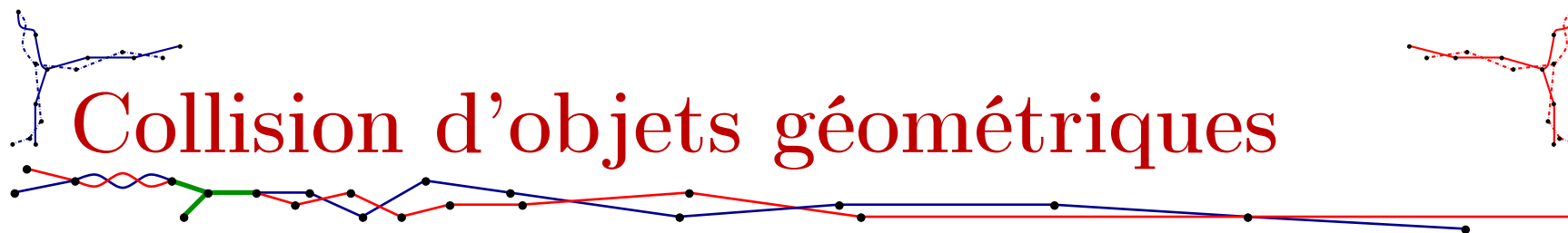
QUESTION : touché ?



# Collision d'objets géométriques

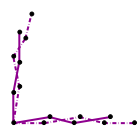
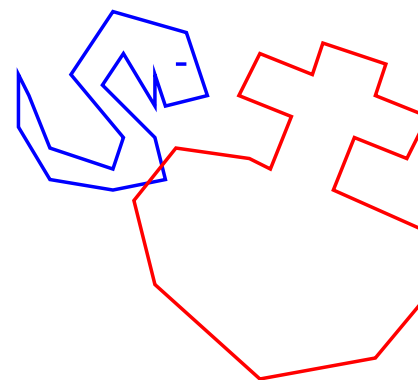
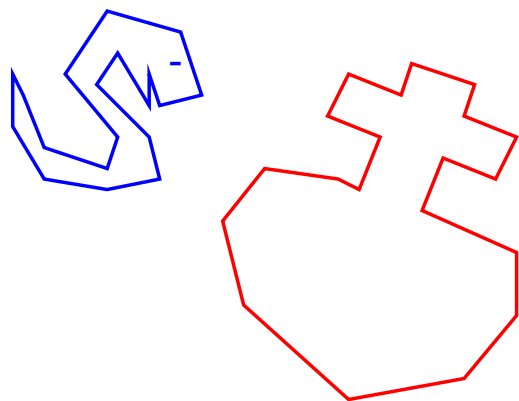
QUESTION : touché ?

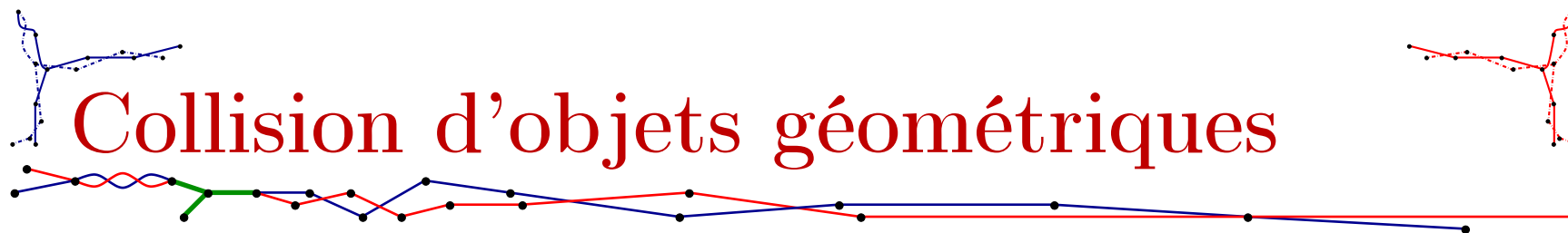




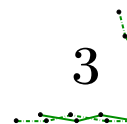
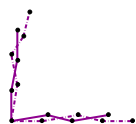
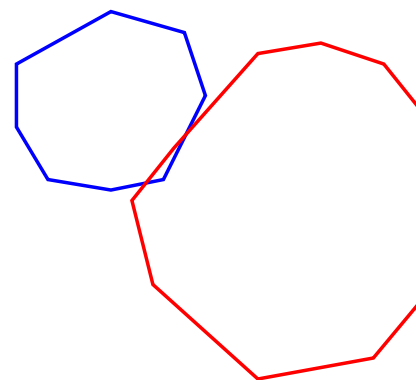
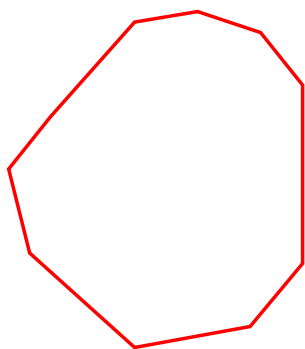
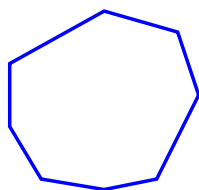
# Collision d'objets géométriques

QUESTION : touché ?

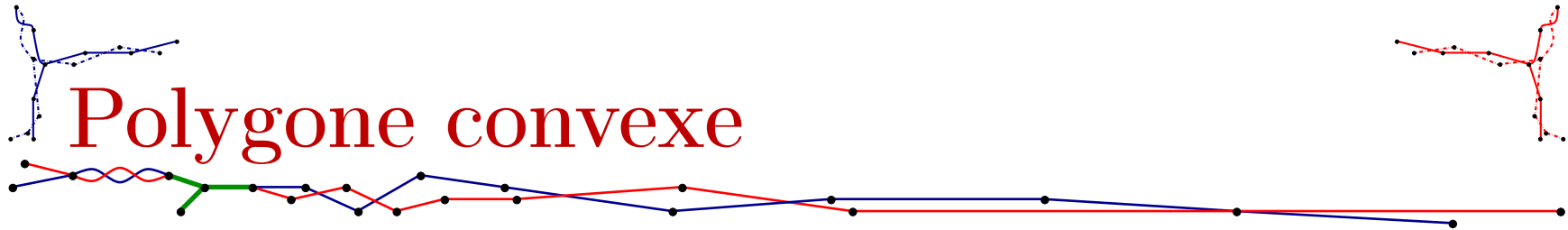




QUESTION : touché ?

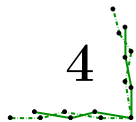
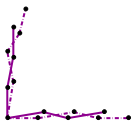


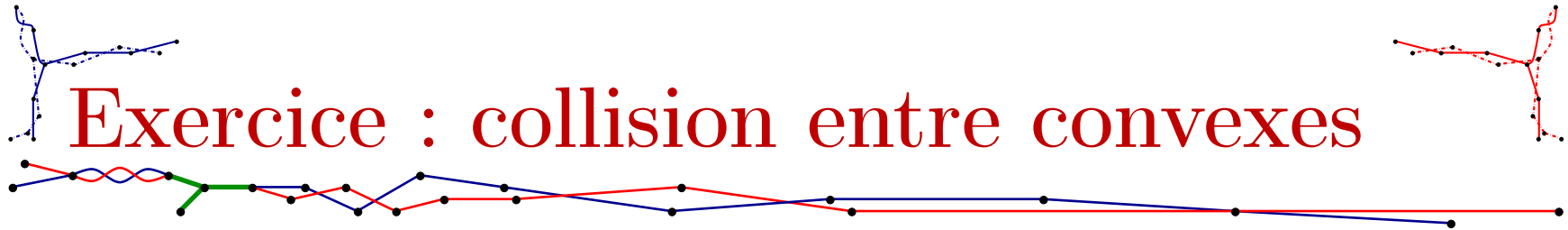




## DÉFINITIONS :

- *polygone convexe* : polygone tel que tout segment joignant deux points du polygone appartient au polygone
- *côté* d'un polygone convexe : segment maximal du polygone tel que tout autre point du polygone appartient au même demi-plan défini par le segment
- *coin* d'un polygone convexe : extrémité d'un côté
- *contour positif* d'un polygone convexe : l'ordre cyclique de ses coins selon le sens trigonométrique



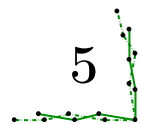
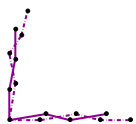


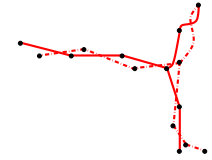
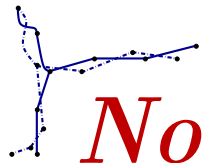
## Exercice : collision entre convexes

EXERCICE : Soient deux polygones convexes  $p_1$  et  $p_2$ . Montrer qu'il n'y a pas de collision entre  $p_1$  et  $p_2$  si et seulement si une des deux conditions suivantes est vérifiée :

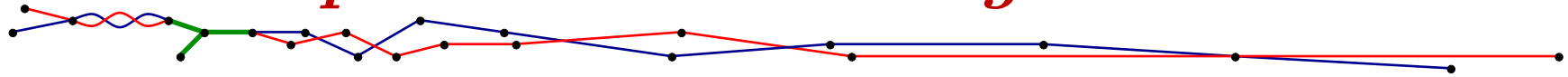
- il existe un côté de  $p_1$  tel que tout coin de  $p_2$  est “de l'autre côté” des coins de  $p_1$  dans les demi-plans définis par ce côté
- il existe un côté de  $p_2$  tel que tout coin de  $p_1$  est “de l'autre côté” des coins de  $p_2$  dans les demi-plans définis par ce côté

QUESTION : implantation ?

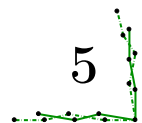
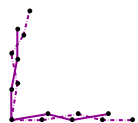


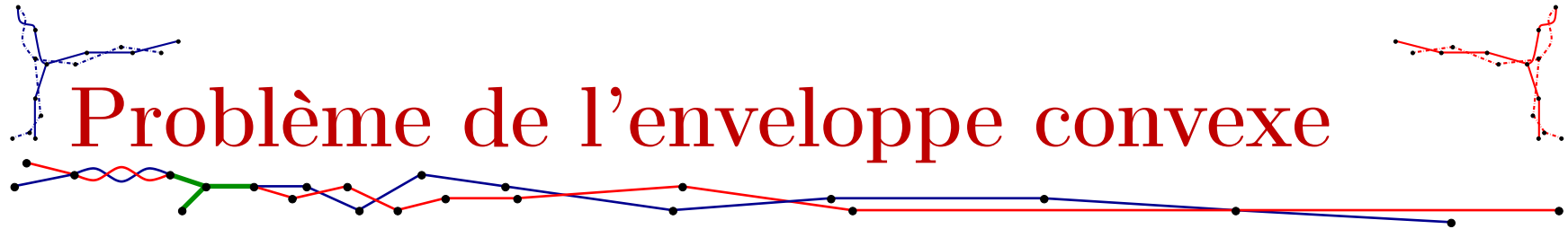


# *Notes pour révision : algorithme*



1. pour tout côté  $pq$  de  $p_1$
2.    $r \leftarrow$  un coin de  $p_1$  distinct de  $p$  et de  $q$
3.   côtéSéparant  $\leftarrow$  true
4.   pour tout coin  $s$  de  $p_2$
5.     si  $s$  est du même côté que  $r$  par rapport à  $(pq)$  alors
6.       côtéSéparant  $\leftarrow$  false
7.   si côtéSéparant = true alors retourner pas\_de\_collision
8. pour tout côté  $pq$  de  $p_2$
9.    $r \leftarrow$  un coin de  $p_2$  distinct de  $p$  et de  $q$
10. côtéSéparant  $\leftarrow$  true
11. pour tout coin  $s$  de  $p_1$
12.   si  $s$  est du même côté que  $r$  par rapport à  $(pq)$  alors
13.     côtéSéparant  $\leftarrow$  false
14. si côtéSéparant = true alors retourner pas\_de\_collision
15. retourner collision

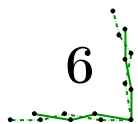
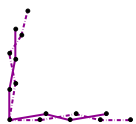


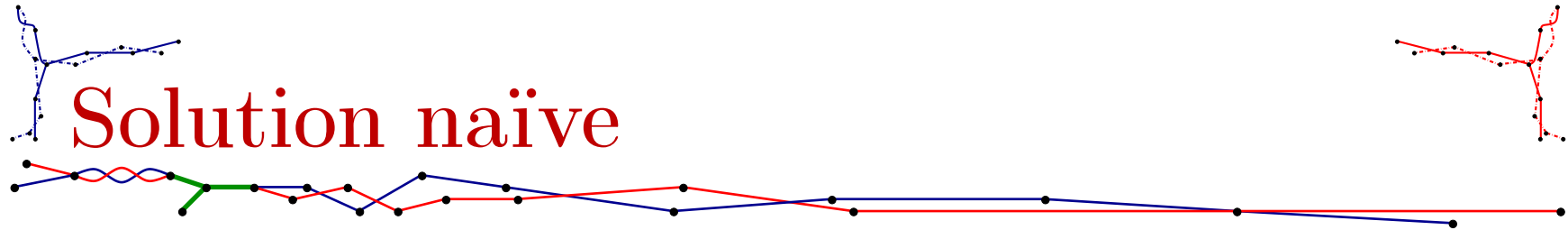


IN : Points, une liste de coordonnées de points en 2D

OUT : Enveloppe, une liste d'éléments de Points représentant le contour positif de l'enveloppe convexe de Points

DÉFINITION : l'enveloppe convexe d'un ensemble de points est le plus petit polygone convexe contenant ces points

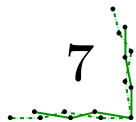
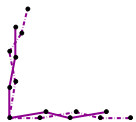




PRINCIPE : pour toute paire de points dans Points, vérifier s'elle forme un côté de l'enveloppe convexe de Points en testant si tout autre point de Points sont “du même côté” des demi-plans définis par le segment passant par la paire de points

QUESTION : complexité ? implantation ?

*Notes pour révision : voir TME6.*



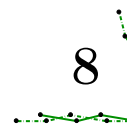
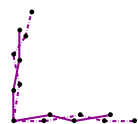


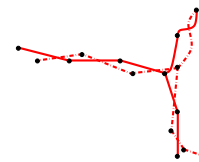
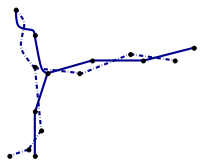
# Exercice : estimation du temps



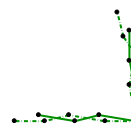
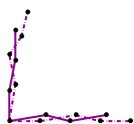
QUESTION : un ordinateur de l'ordre du Giga-Hertz exécutant un algorithme en  $O(n^3)$ , avec  $n = 10000$ , quel est le temps d'exécution attendu (en ordre de grandeur) ?

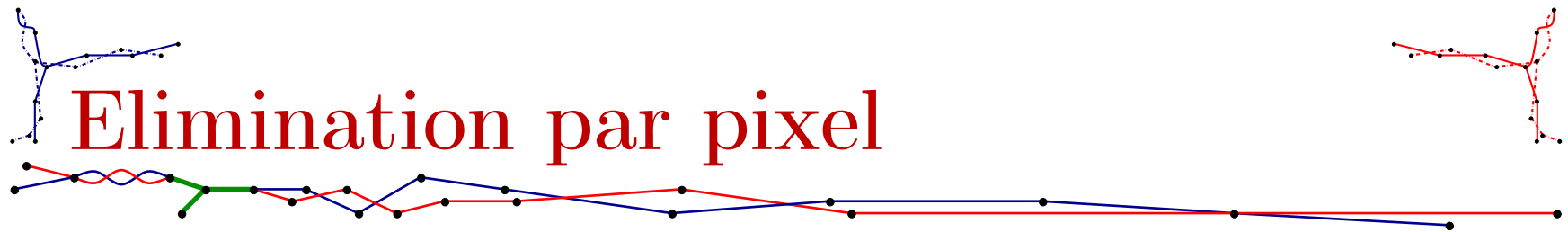
- algorithme en  $O(n \log n)$  ?
- algorithme en  $O(n \log h)$ , avec  $h \ll n$  ?
- algorithme en  $O(n + n' \log h)$ , avec  $n' \ll n$  et  $h \ll n$  ?



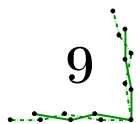
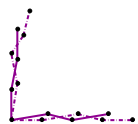
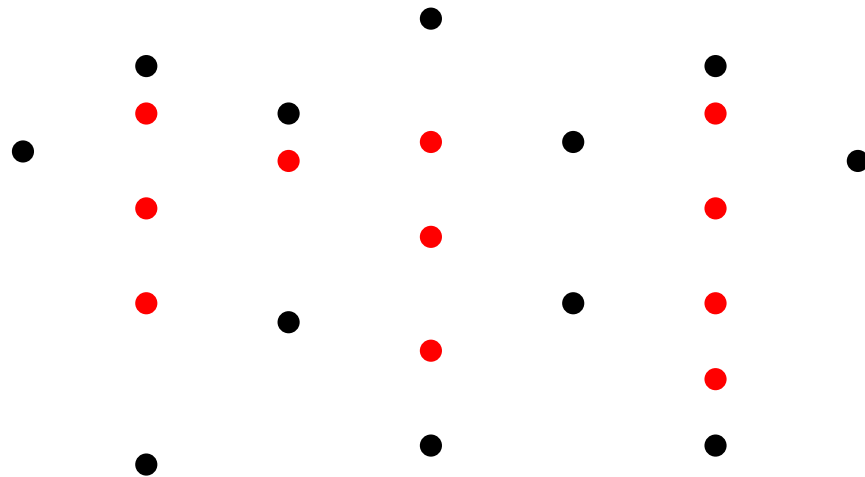


# Enveloppe convexe : précalculs

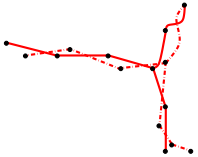
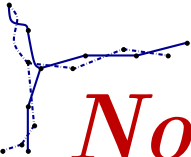




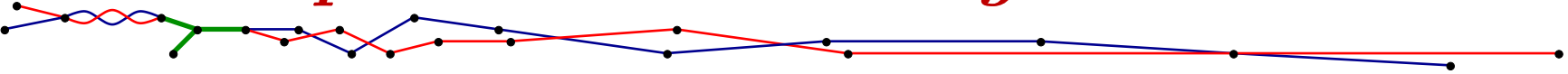
IDÉE : si trois points de Points ont le même abscisse, alors au plus deux de ces points appartiennent à l'enveloppe convexe de Points



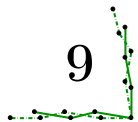
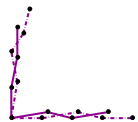




# *Notes pour révision : algorithme*



1.  $ymin \leftarrow$  tableau de points de taille assez grande
2. pour tout  $p$  dans Points
3.     si  $ymin[p.x] = \text{NULL}$  ou si  $ymin[p.x].y < p.y$
4.          $ymin[p.x] \leftarrow p$
5. similairement avec ymax
6. parcourir ymin et ymax et retourner les éléments non NULL

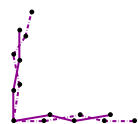
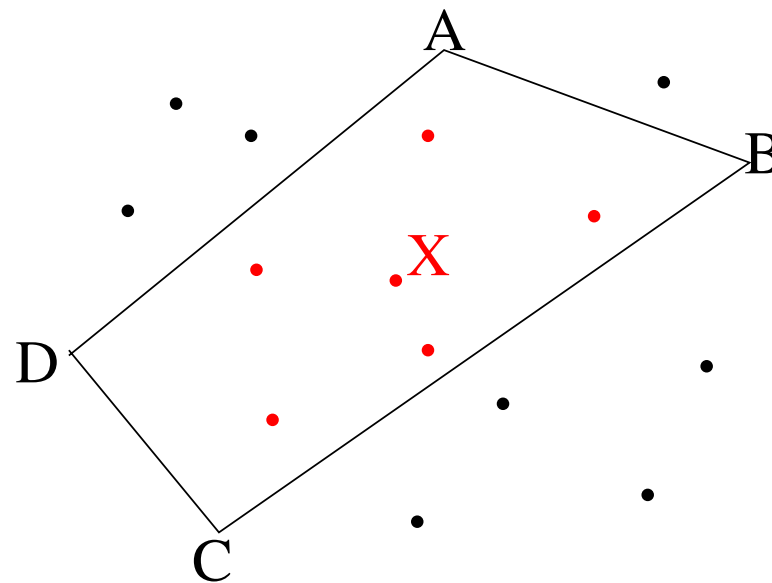


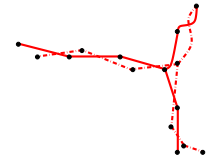
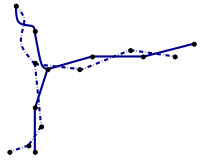


# Filtrage Akl-Toussaint



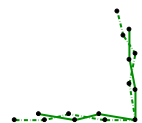
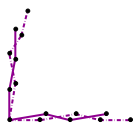
IDÉE : soient A, B, C, D les points les plus au Nord, Est, Sud, Ouest parmi Points, alors aucun point appartenant strictement au quadrilatère ABCD est sur l'enveloppe convexe de Points






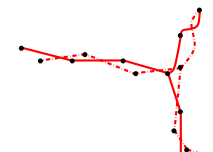
Enveloppe convexe : algorithmes

*Notes pour révision : voir TME6*

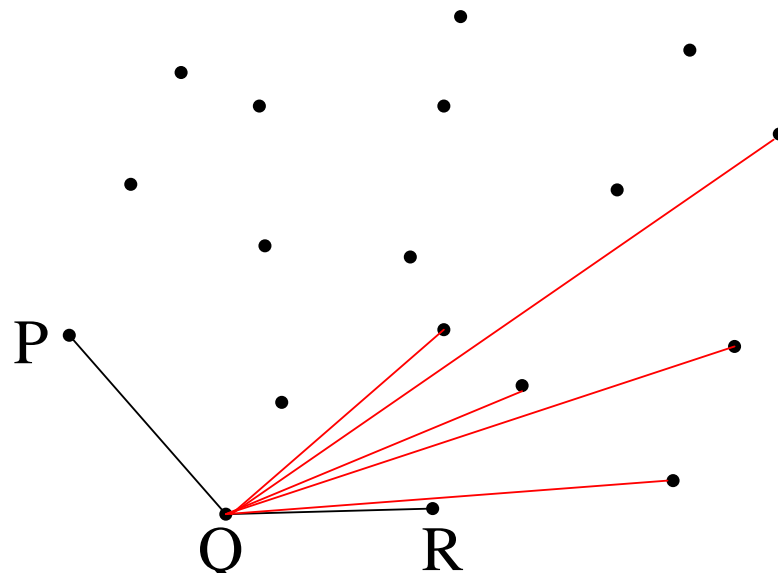




# Algorithme Jarvis



IDÉE : si PQ est un côté de l'enveloppe convexe de Points, et R le point de Points tel que l'angle  $(\overrightarrow{PQ}, \overrightarrow{QR})$  est minimum, alors QR est un côté de l'enveloppe convexe de Points

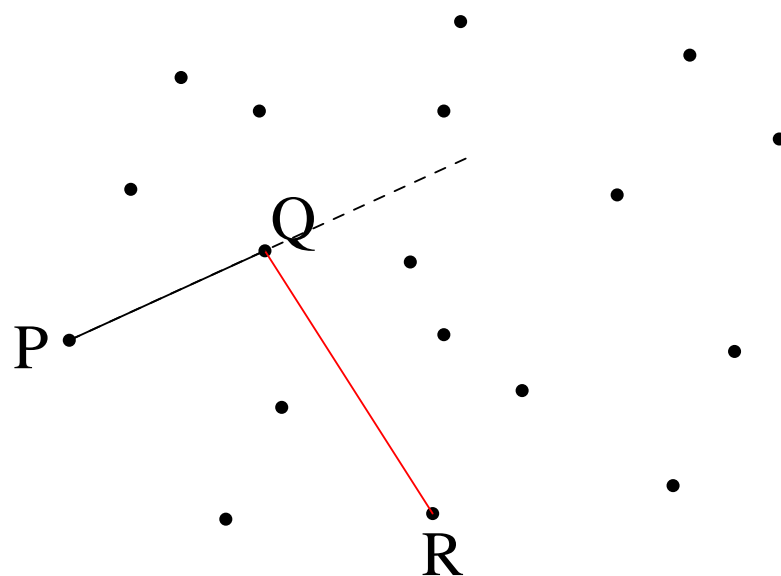





# Algorithme Graham

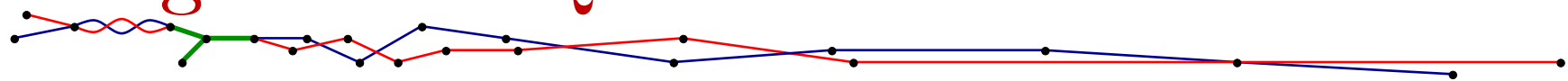
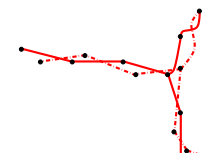


IDÉE : si  $P$ ,  $Q$  et  $R$  sont trois points de Points d'abscisse croissant, et  $R$  appartient au “mauvais côté” des demi-plans définis par  $PQ$ , alors  $Q$  n'est pas sur l'enveloppe convexe de Points

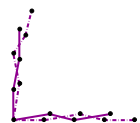
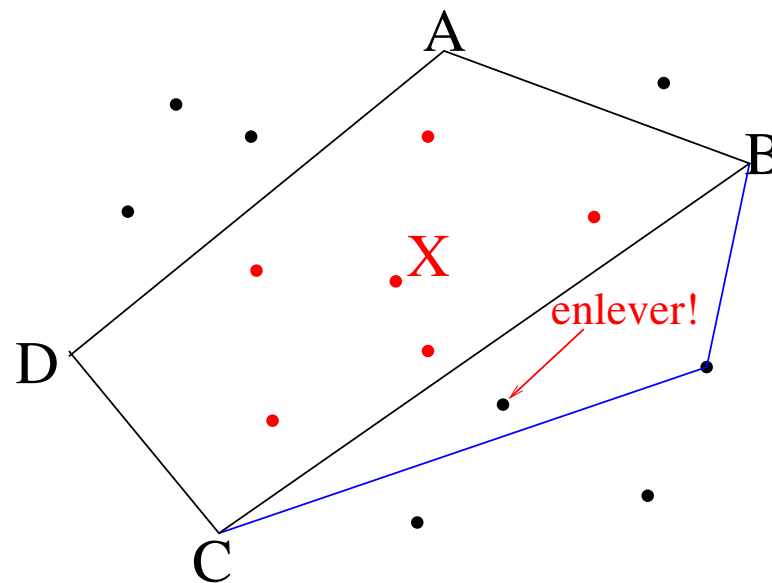


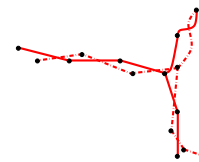
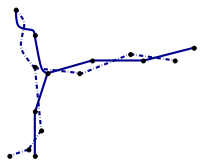


# Algorithme QuickHull

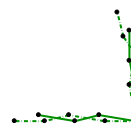
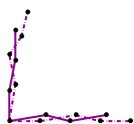



IDÉE : Akl-Toussaint est génial, comment l'étendre ?



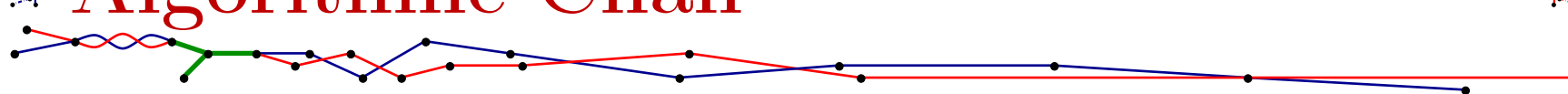
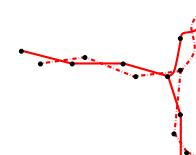


Peut on faire mieux ?

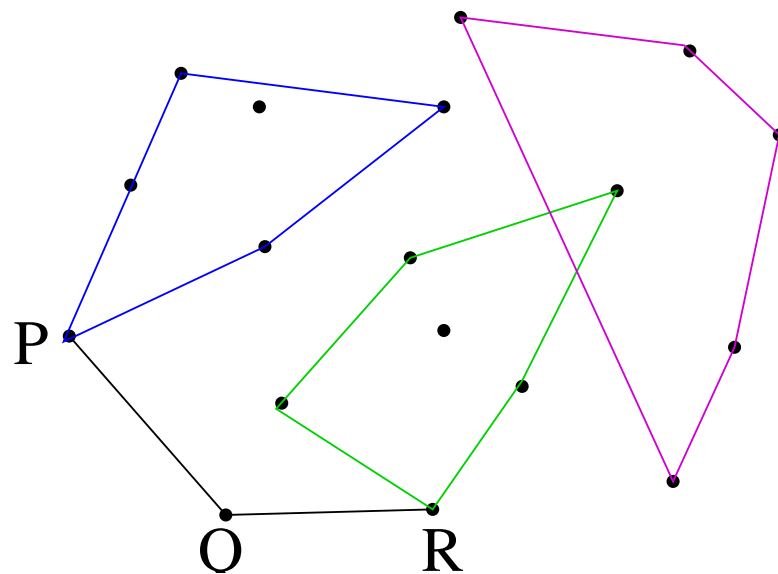




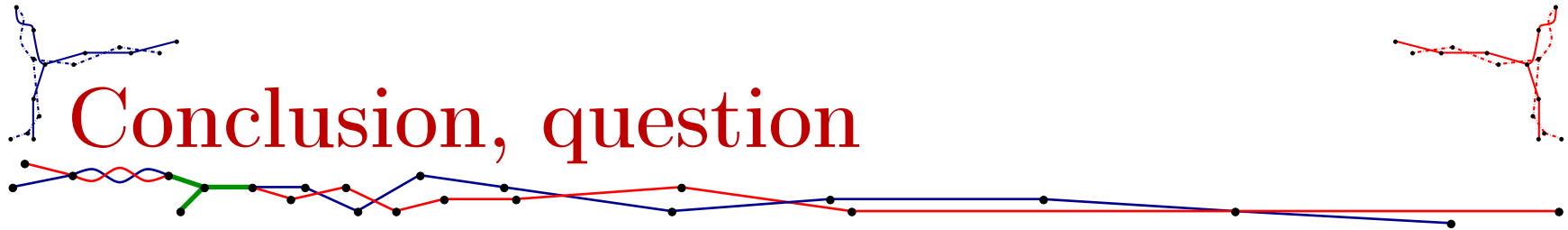
# Algorithme Chan



IDÉE : Jarvis + décomposition







# Conclusion, question

## CONCLUSION :

- algorithme naïf en  $O(n^3)$
- technique incrémental (Jarvis)
- technique de précalcul (pixel, Akl-Toussaint, QuickHull)
- technique d'élagage (Graham)
- technique ultime : décomposition (Chan)

## QUESTION :

- implantation ? (voir TME)

