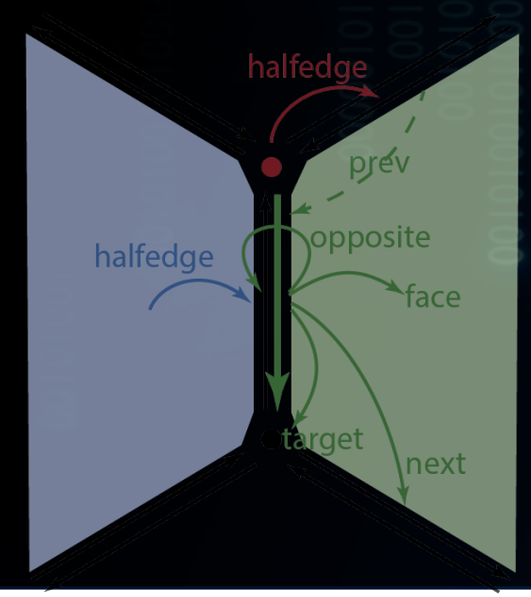
MODELISATION GEOMETRIQUE



Alexandra Bac

POLYTECH 4A INFORMATIQUE REVA

2 (SUITE) – PROPRIÉTÉS DES MAILLAGES

Certaines illustrations sont issues du livre « polygon mesh processing »

« MODÉLISATION GÉOMÉTRIQUE » - MAILLAGES (SUITE)

Chapitre 2

MAILLAGES

Chapitre 4

GÉOMÉTRIE DES SURFACES

Chapitre 1

MODÉLISATION DES SURFACES

Chapitre 3 (+ 5A)

SURFACES PARAMÉTRIQUES

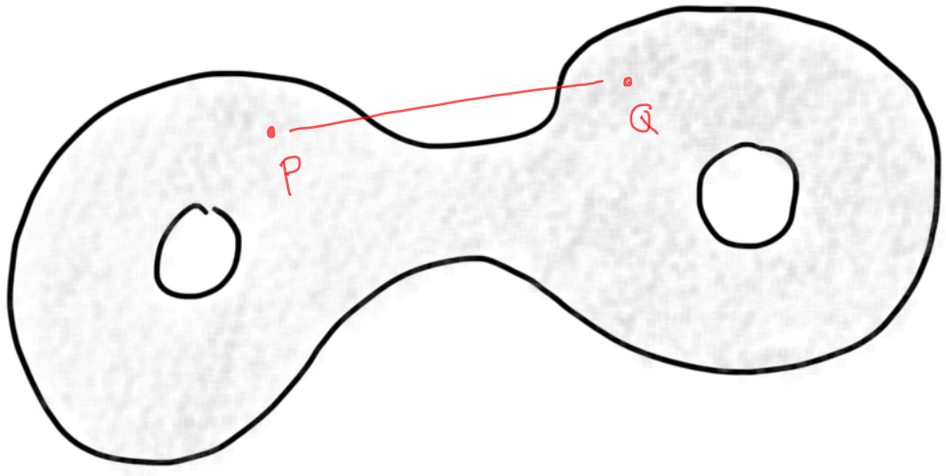
Chapitre 5

SURFACES IMPLICITES

NOTIONS DE GÉOMÉTRIE

ENVELOPPE CONVEXE

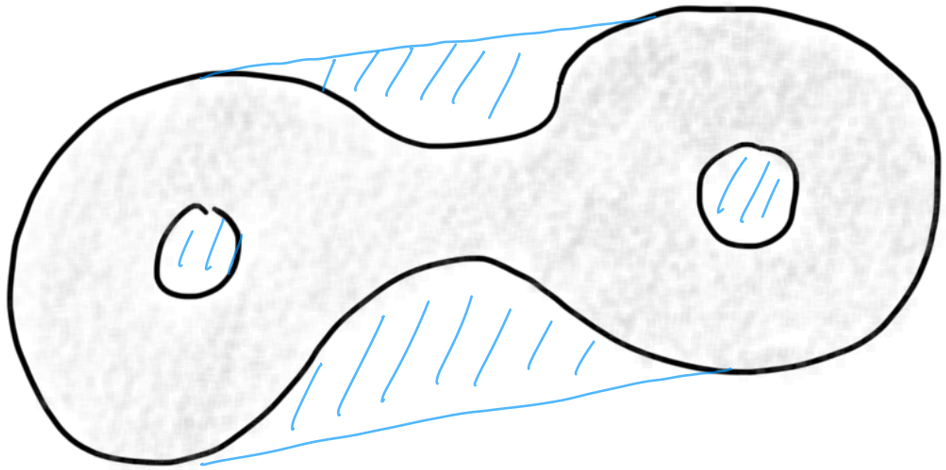
$E \subseteq \mathbb{R}^n$ est convexe sssi
$\forall P, Q \in E \quad [P, Q] \subseteq E$



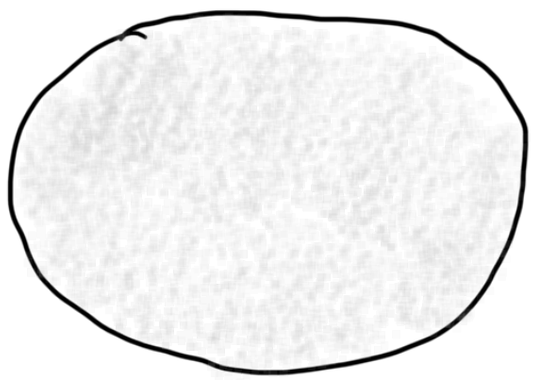
Courbe convexe / concave

ENVELOPPE CONVEXE

$E \subseteq \mathbb{R}^n$ est convexe sssi
$\forall P, Q \in E \quad [P, Q] \subseteq E$



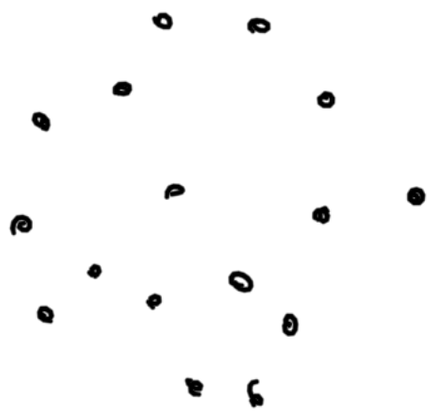
enveloppe convexe



ENVELOPPE CONVEXE

$E \subseteq \mathbb{R}^n$ est convexe sssi
$\forall P, Q \in E \quad [P, Q] \subseteq E$

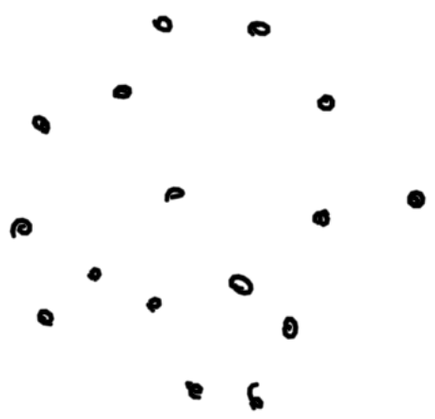
L'enveloppe convexe de $E \subseteq \mathbb{R}^n$ est
le plus petit sous-ensemble convexe contenant E

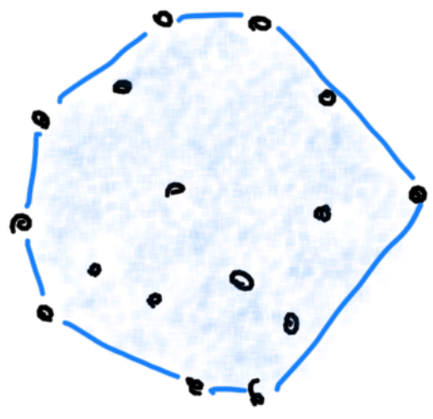


ENVELOPPE CONVEXE

$E \subseteq \mathbb{R}^n$ est convexe sssi
$\forall P, Q \in E \quad [P, Q] \subseteq E$

L'enveloppe convexe de $E \subseteq \mathbb{R}^n$ est
le plus petit sous-ensemble convexe contenant E



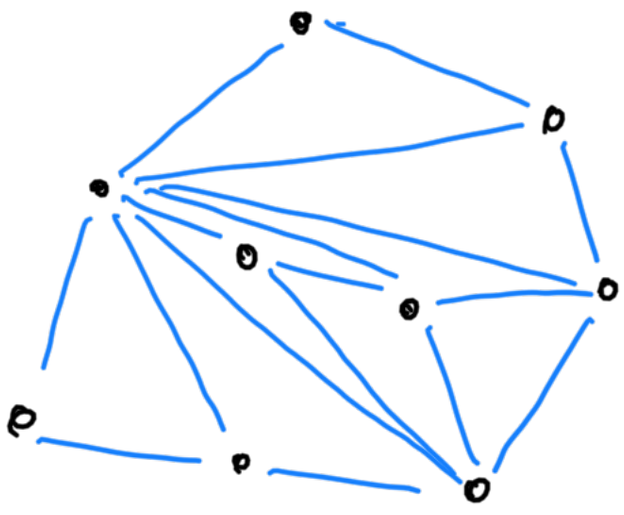


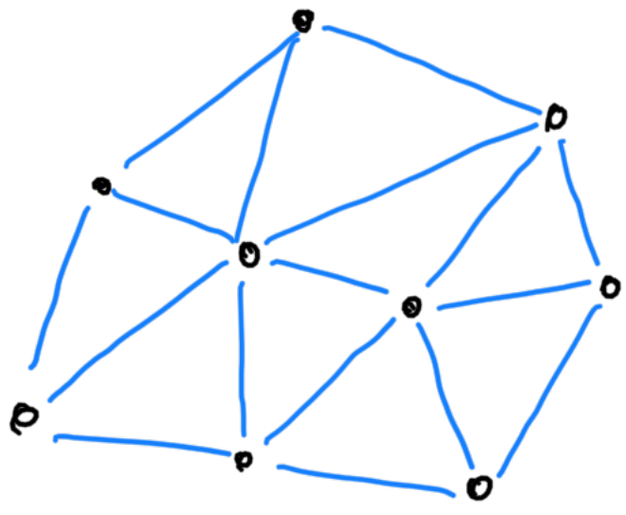
DELAUNAY, VORONOÏ

« MODÉLISATION GÉOMÉTRIQUE » - MAILLAGES (SUITE)

Etant donné un ensemble de points :

→ plusieurs triangulations possibles



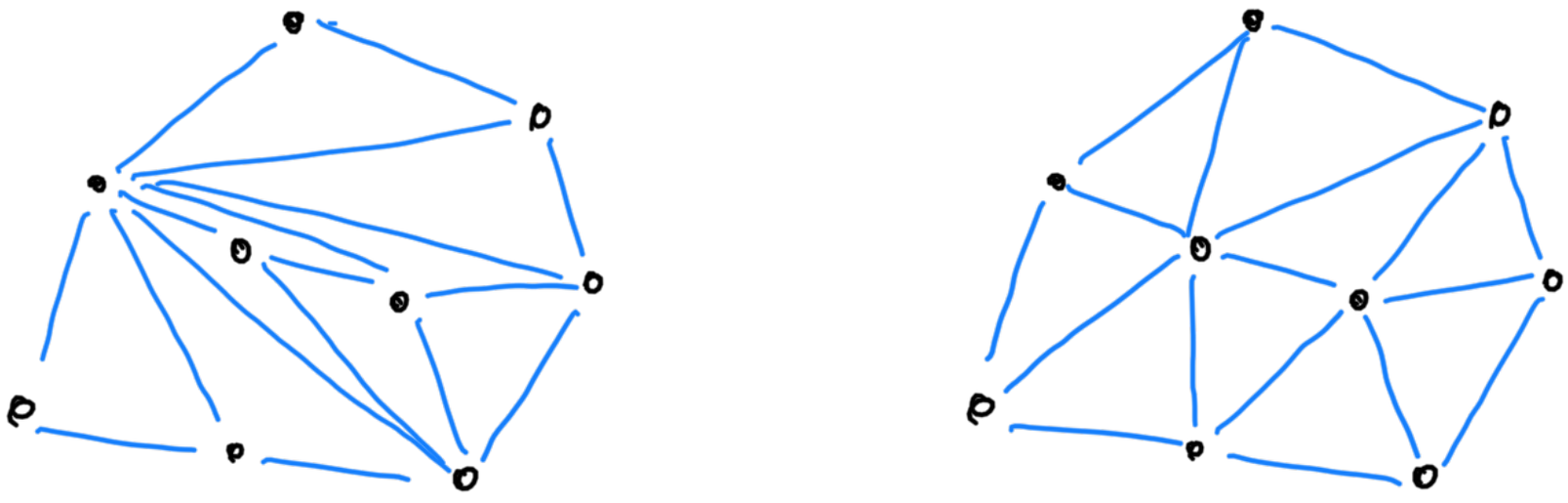


→ quel est le meilleur ?

« MODÉLISATION GÉOMÉTRIQUE » - MAILLAGES (SUITE)

→ triangles les plus réguliers

→ les plus proches de triangles équilatéraux



→ triangulation de Delaunay

CE QUI EST COMMUN . . .

Propriétés combinatoires des maillages

Toute triangulation d'un ensemble de points donné admet :

- Nombre de triangles

$$N_t = 2(N-1) - N_k$$

N nombre de points, N_k nombre de points de l'enveloppe convexe

- Nombre d'arêtes

$$N_t = 3(N-1) - N_k$$

CE QUI EST COMMUN . . .

Propriétés combinatoires des maillages

Toute triangulation d'un ensemble de points donné admet :

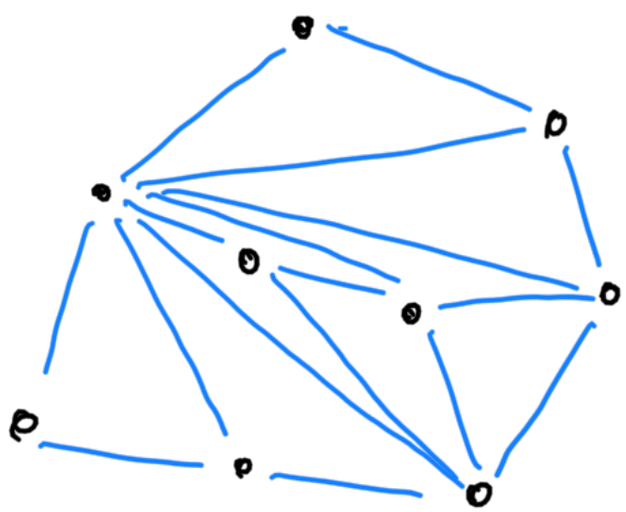
- Nombre de triangles

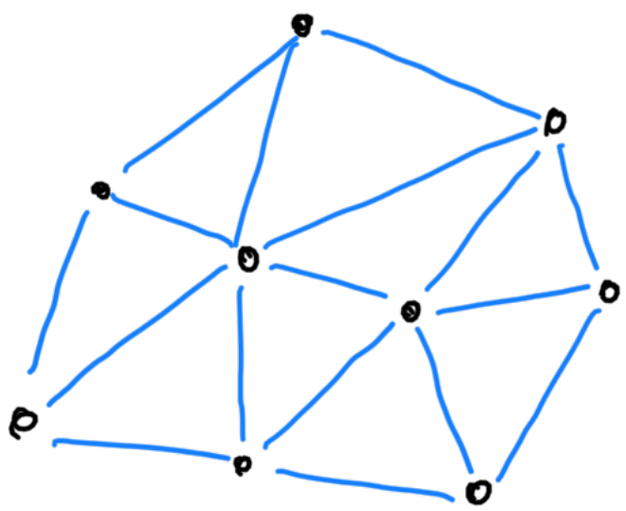
$$N_t = 2(N-1) - N_k$$

N nombre de points, N_k nombre de points de l'enveloppe convexe

- Nombre d'arêtes

$$N_t = 3(N-1) - N_k$$





TRIANGULATION DE DELAUNAY

TRIANGULATION DE DELAUNAY

→ triangulation maximisant le min des angles

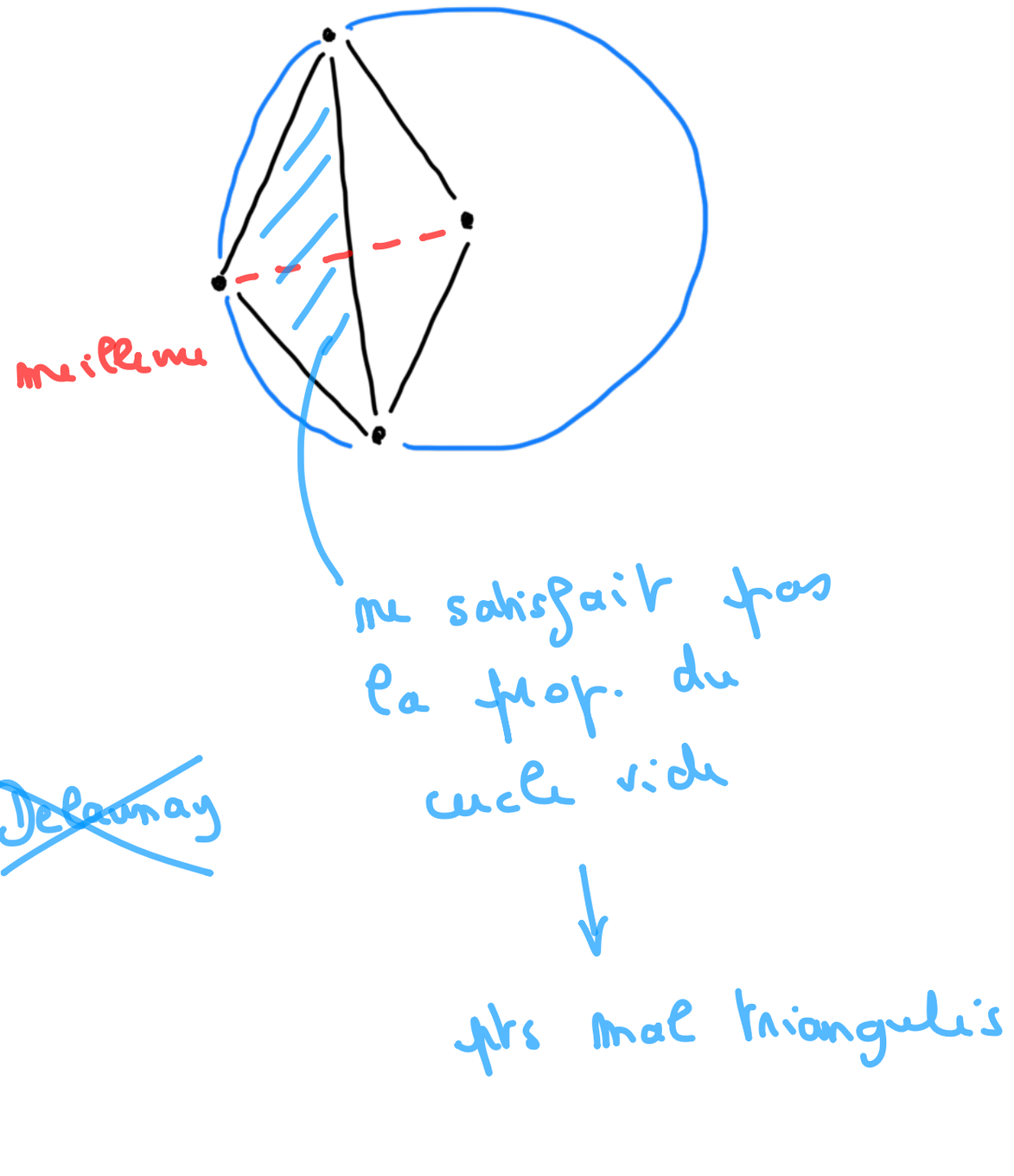
2D

Propriété du cercle vide

Un triangle est dit vide s'il ne contient aucun autre points que les siens dans son cercle circonscrit

Delaunay

Tout triangle a la propriété du cercle vide.



TRIANGULATION DE DELAUNAY

→ triangulation maximisant le min des angles

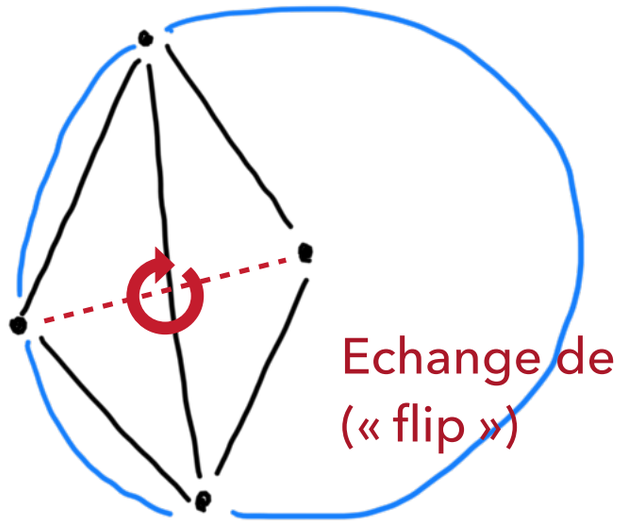
2D

Propriété du cercle vide

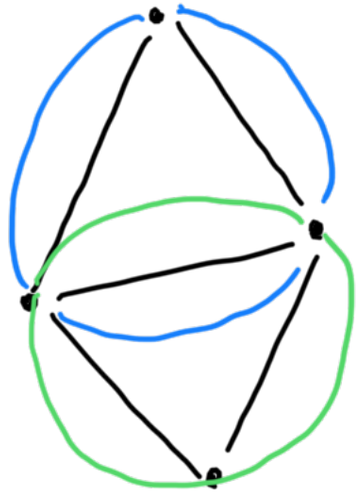
Un triangle est dit vide s'il ne contient aucun autre points que les siens dans son cercle circonscrit

Delaunay

Tout triangle a la propriété du cercle vide.



Echange de l'arête (« flip »)



Les 2 triangles satisfont la propriété du cercle vide

Delaunay ✓

TRIANGULATION DE DELAUNAY

→ triangulation maximisant le min des angles

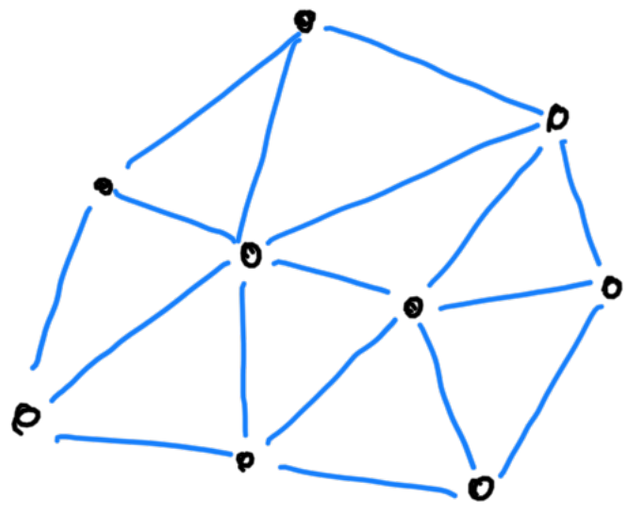
2D

Propriété du cercle vide

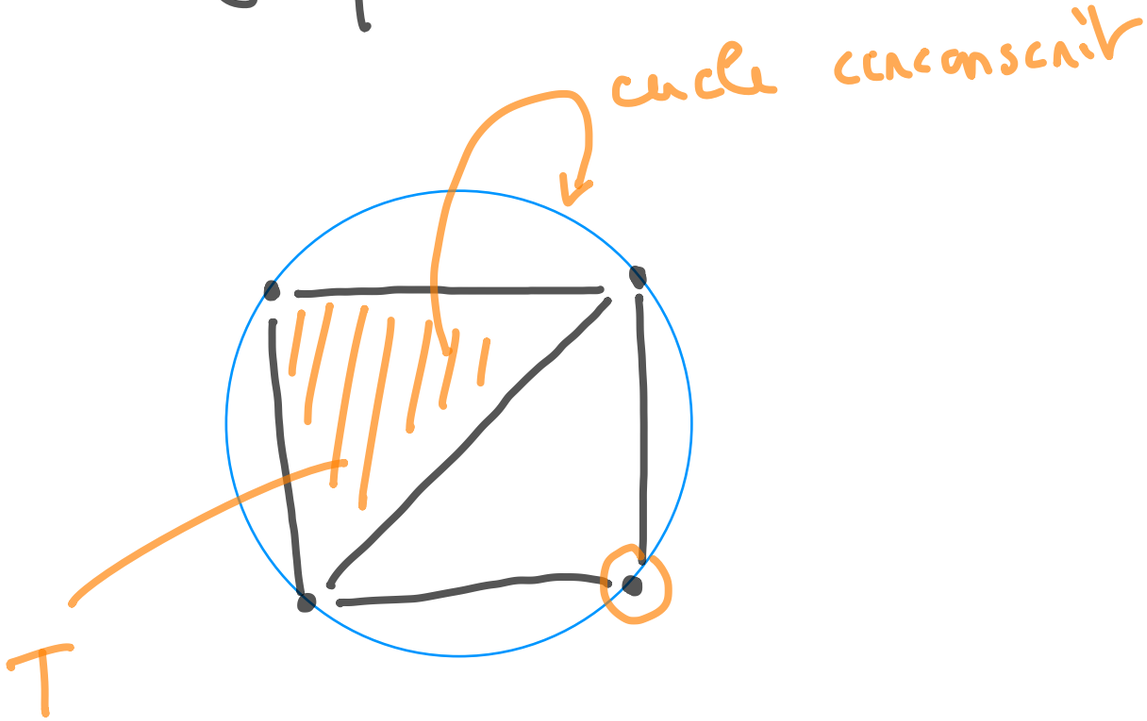
Un triangle est dit vide s'il ne contient aucun autre points que les siens dans son cercle circonscrit

Delaunay

Tout triangle a la propriété du cercle vide.



Triang. Delaunay unique sauf s'il y a des pts cocycliques



ss prop. cercle vide de T

pas satisfait

TRIANGULATION DE DELAUNAY

→ triangulation maximisant le min des angles

2D

3D

Maille l'enveloppe convexe

Propriété du cercle vide

Un triangle est dit de cercle vide s'il n'existe aucun autre points que les siens dans son cercle circonscrit

Propriété de la sphère vide

Un tétraèdre est dit de sphère vide s'il n'existe aucun autre points que les siens dans sa sphère circonscrite

Delaunay

Tout triangle a la propriété du cercle vide.

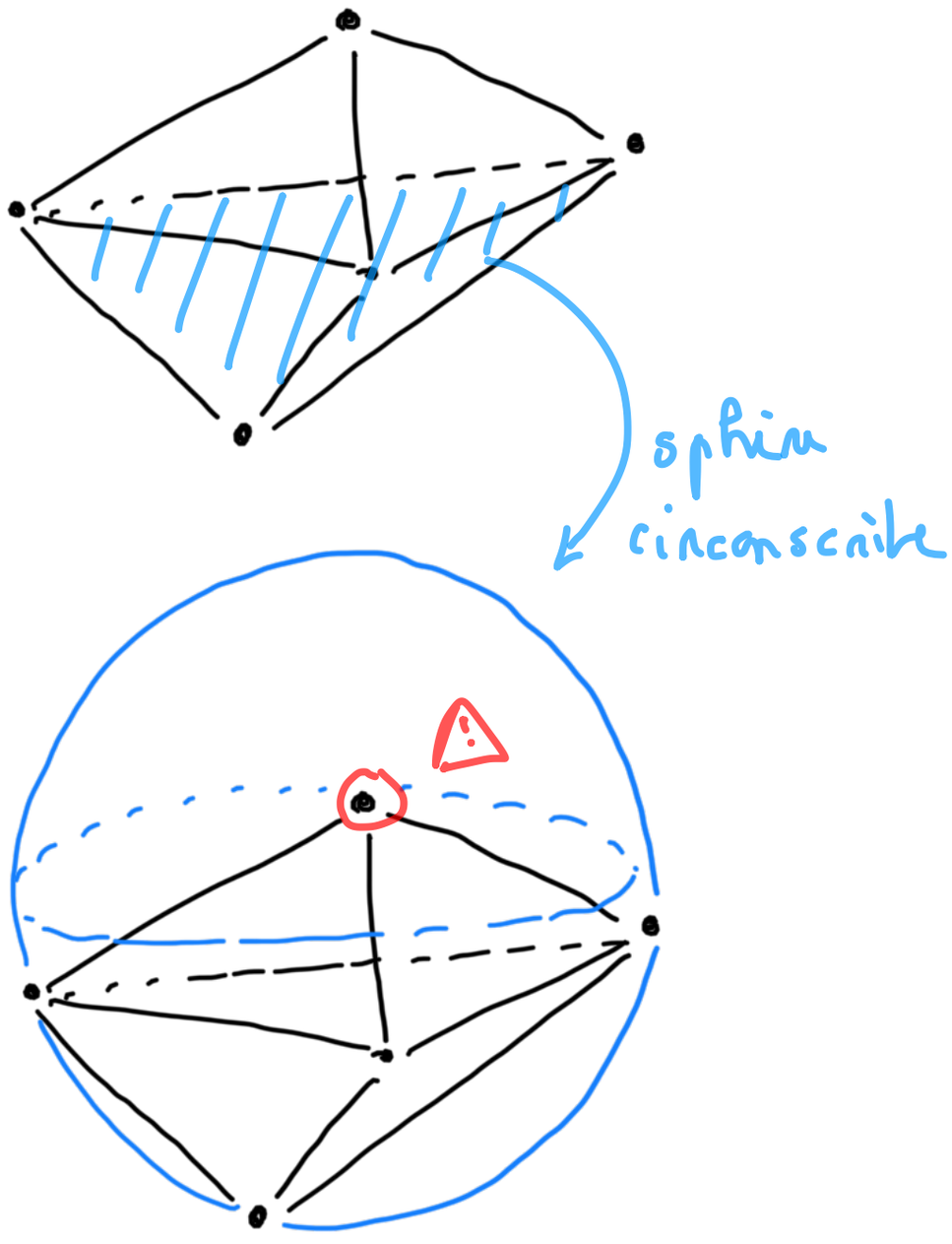
Delaunay

Tout tétraèdre a la propriété de la sphère vide.

- Unicité de la triangulation de Delaunay sauf ...

TRIANGULATION DE DELAUNAY

→ triangulation maximisant le min des angles



3D

Propriété de la sphère vide

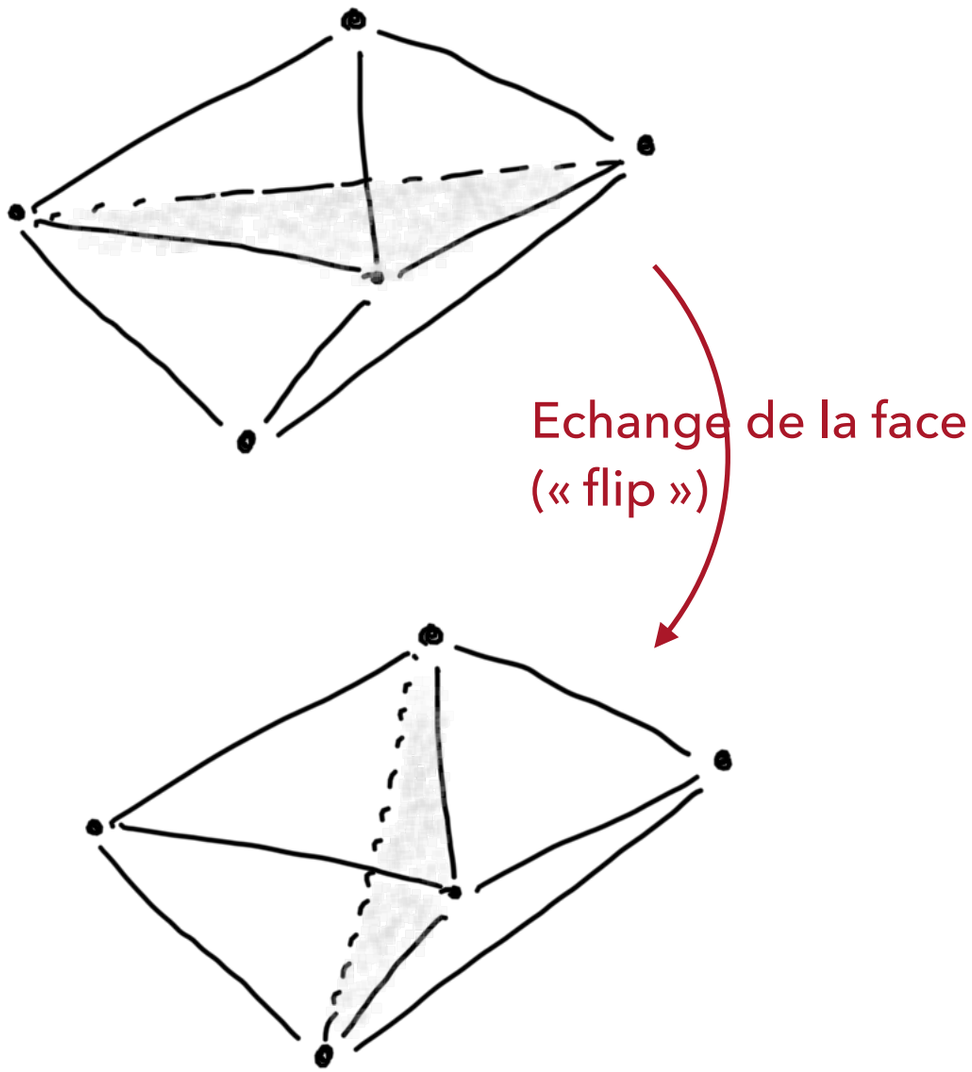
Un tétraèdre est dit de sphère vide s'il n'existe aucun autre points que les siens dans sa sphère circonscrite

Delaunay

Tout tétraèdre a la propriété de la sphère vide.

TRIANGULATION DE DELAUNAY

→ triangulation maximisant le min des angles



Répéter le « flip » est une algorithme de maillage …

3D

Propriété de la sphère vide

Un tétraèdre est dit de sphère vide s'il n'existe aucun autre points que les siens dans sa sphère circonscrite

Delaunay

Tout tétraèdre a la propriété de la sphère vide.

TRIANGULATION DE DELAUNAY

→ triangulation maximisant le min des angles

2D

3D

2,5 D

?

Propriété du cercle

Un triangle est dit de [illegible] s'il n'existe aucun autre po[illegible] siens dans son cercle circo[illegible]

Delaunay

Tout triangle a la propriété du cercle vide.

Propriété de la sphère vide

Un tétraèdre est dit de sphère vide s'il n'existe aucun autre points que les siens dans sa sphère circonscrite

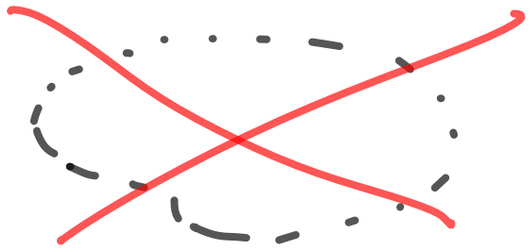
Delaunay

Tout tétraèdre a la propriété de la sphère vide.

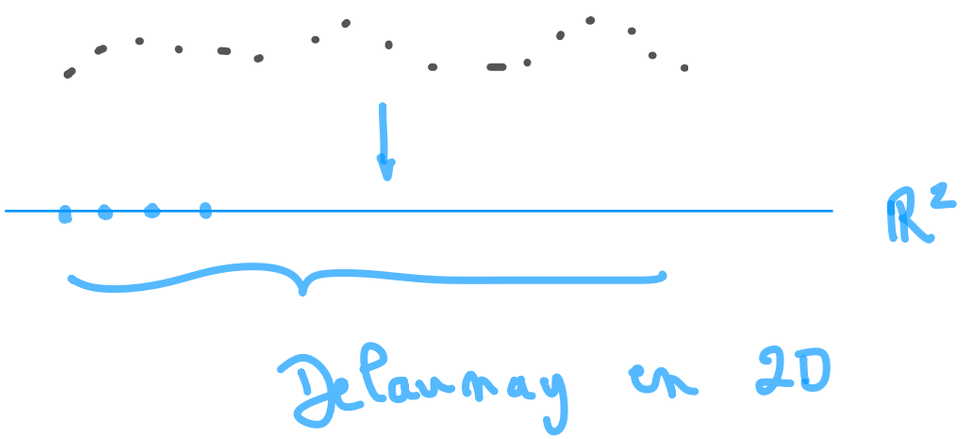
Delaunay n'existe pas ...

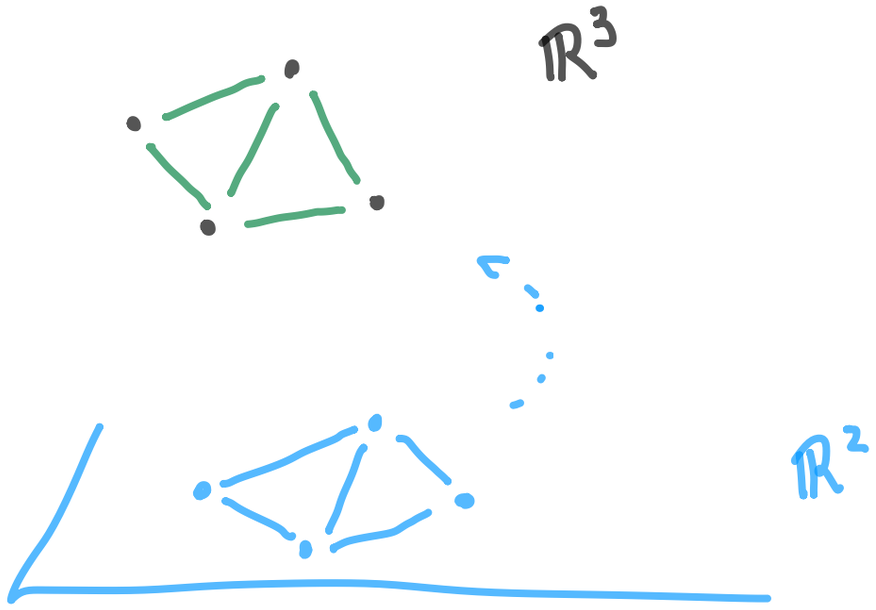
Algorithmes de maillage

→ si les données sont "en élévation" / plan (modèle de terrain...)



1) projeter en 2D
2) Delaunay sur les projetés
3) on en déduit un maillage des pts

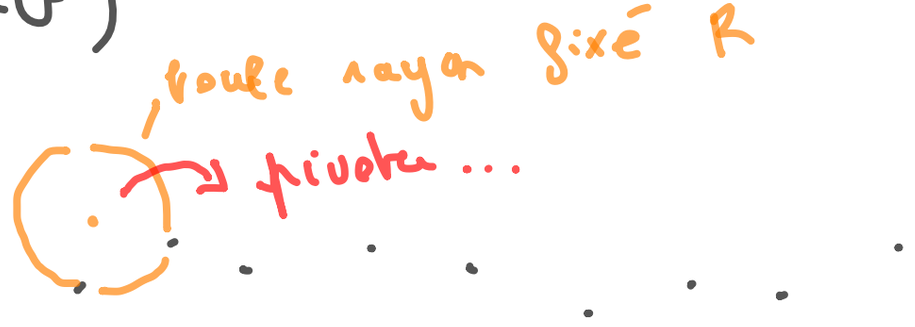




→ algo. pivoting ball (MeshLab)

↓

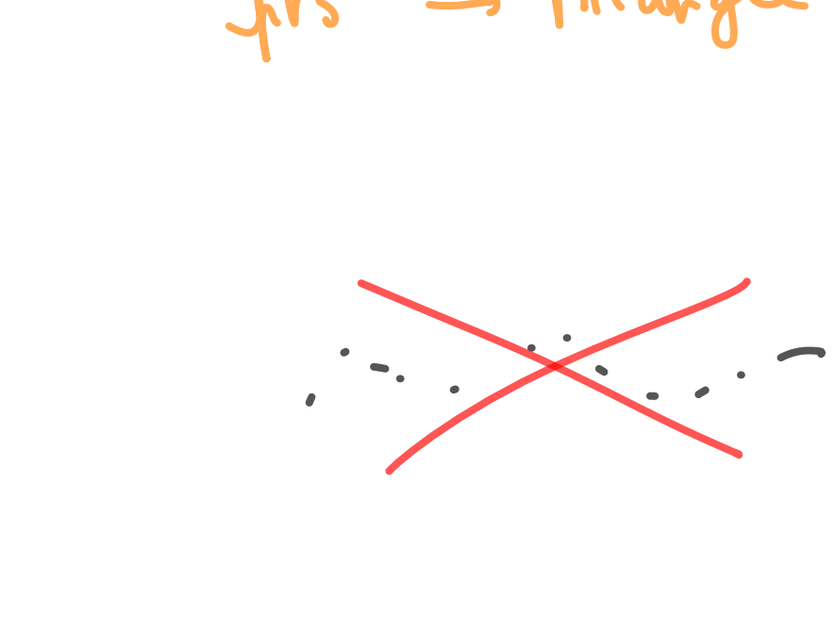
α-shape



→ maillage 3D (tétraèdres)

↓

garde seulement son bord



forme fermée

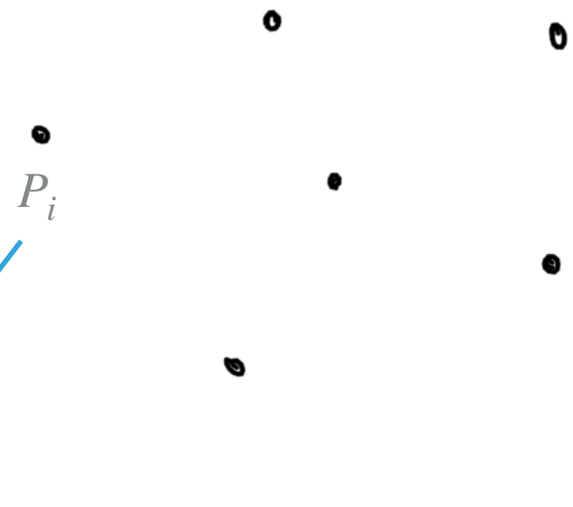
boule se pose sur 3 pts → triangle

DIAGRAMME DE VORONOÏ

DIAGRAMME DE VORONOÏ

Ensemble de points

$\{P_i \in \mathbb{R}^n \,;\, i = 1 \ldots N\}$



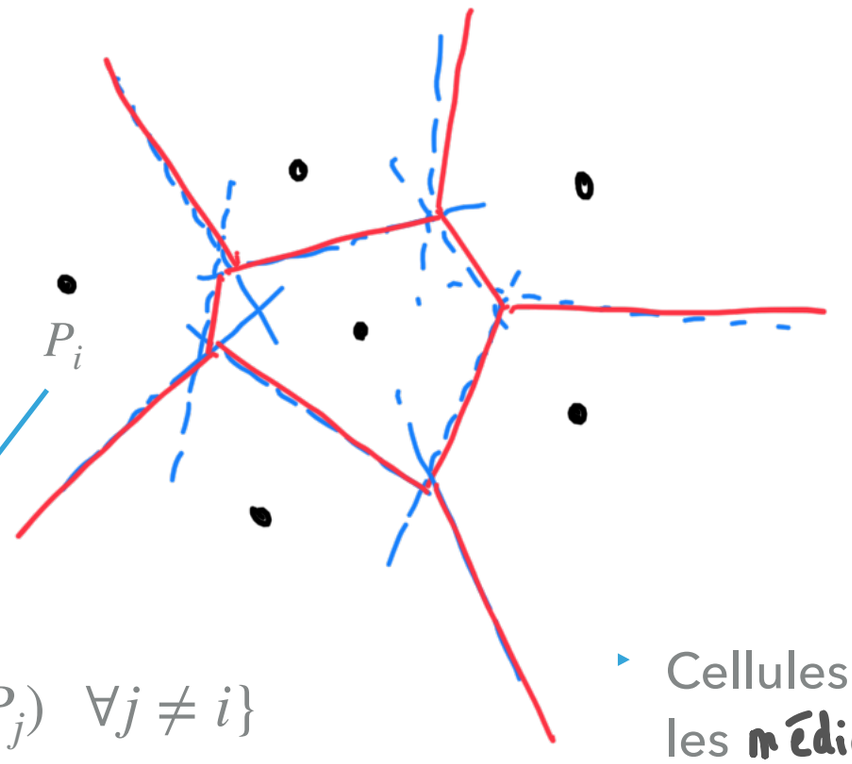
$$V_i = \{X \in \mathbb{R}^n \,;\, d(X, P_i) \leqslant d(X, P_j) \quad \forall j \neq i\}$$

Cellule de Voronoï associée à P_i

DIAGRAMME DE VORONOÏ

Ensemble de points
(germes ou centres)

$\{P_i \in \mathbb{R}^n \,;\, i = 1 \ldots N\}$



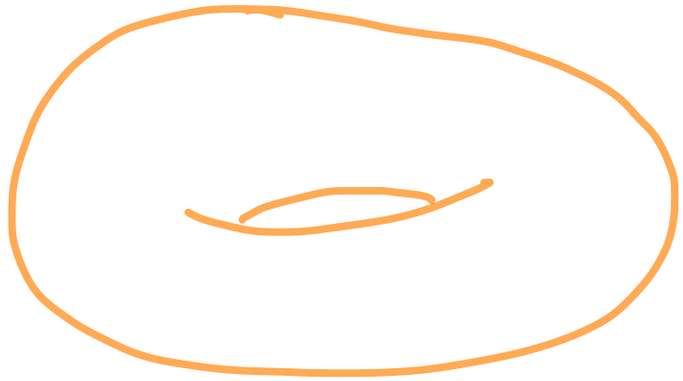
$$V_i = \{X \in \mathbb{R}^n \,;\, d(X, P_i) \leqslant d(X, P_j) \quad \forall j \neq i\}$$

Cellule de Voronoï associée à P_i

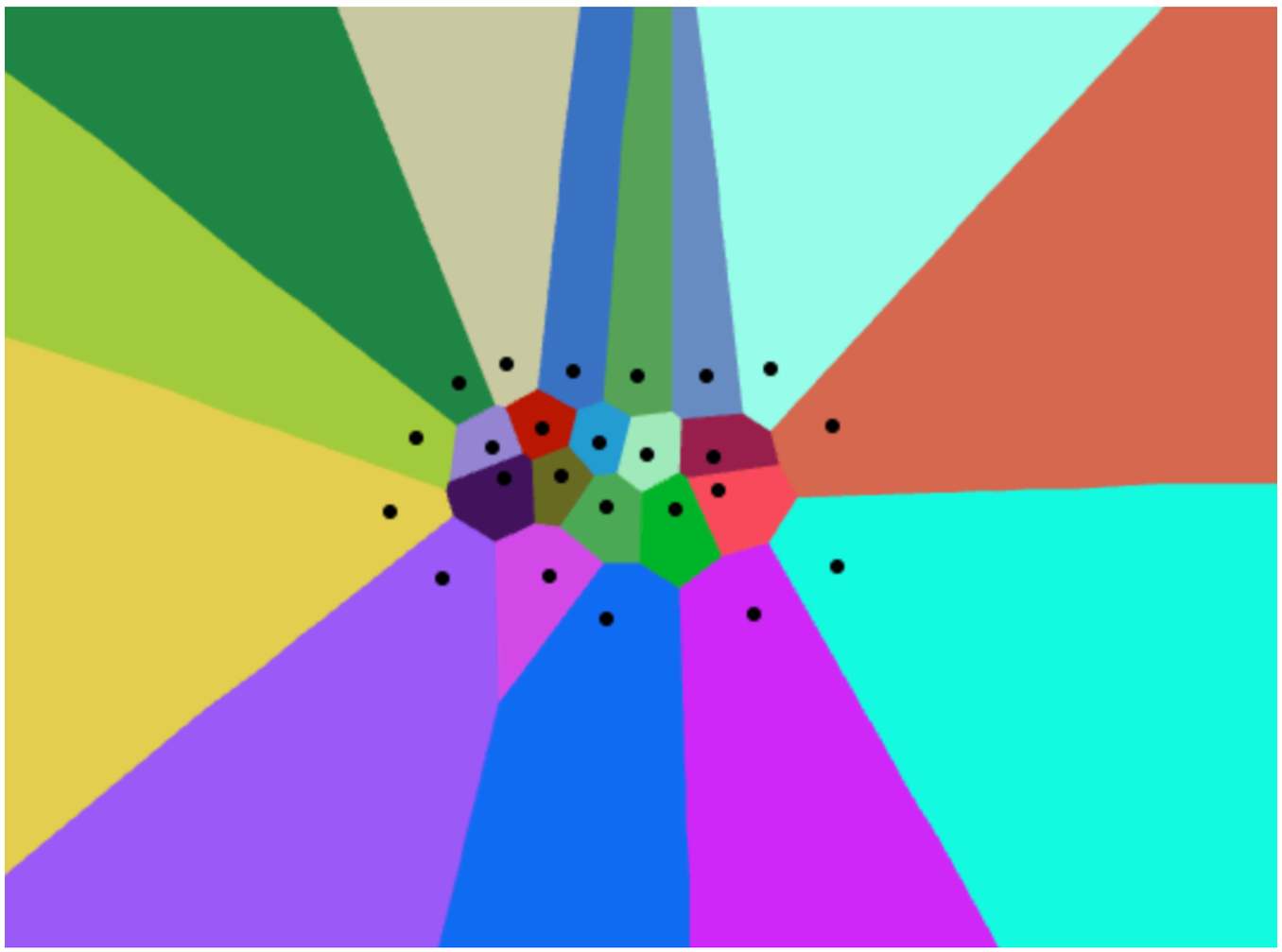
- Cellules délimitées par les médiatrices joignant les points
- Unicité du diagramme de Voronoï

Diagramme de Voronoï (maillage polygonal / polyédrique)

DIAGRAMME DE VORONOÏ



[Exemple online]

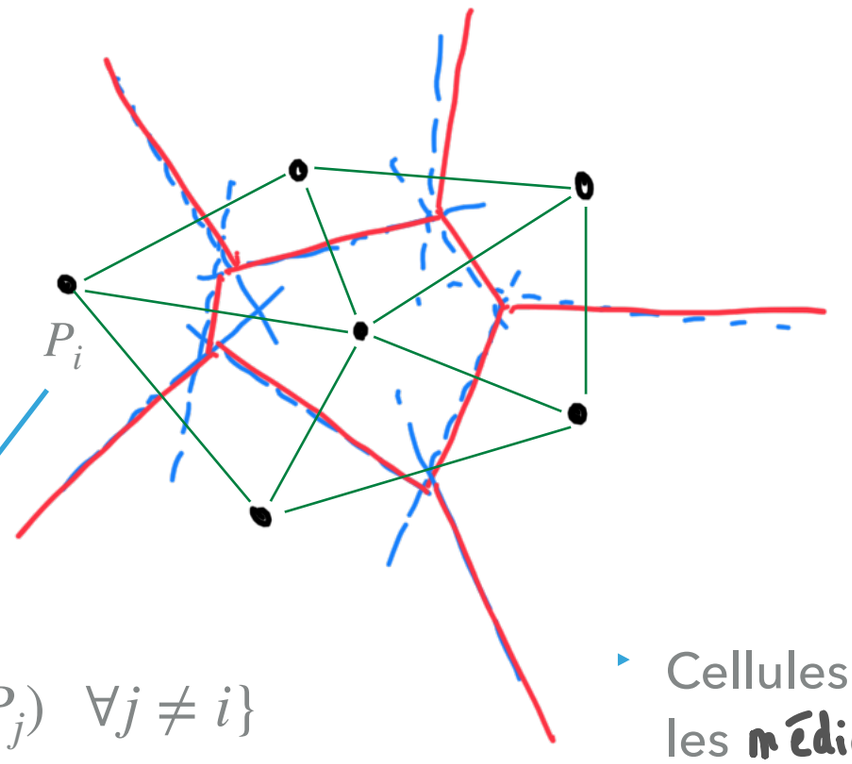


Voronoi
↕
squelette

DIAGRAMME DE VORONOÏ

Ensemble de points
(germes ou centres)

$\{P_i \in \mathbb{R}^n \,;\, i = 1 \ldots N\}$



$$V_i = \{X \in \mathbb{R}^n \,;\, d(X, P_i) \leqslant d(X, P_j) \quad \forall j \neq i\}$$

Cellule de Voronoï associée à P_i

- Cellules délimitées par les médiatrices joignant les points
- Unicité du diagramme de Voronoï

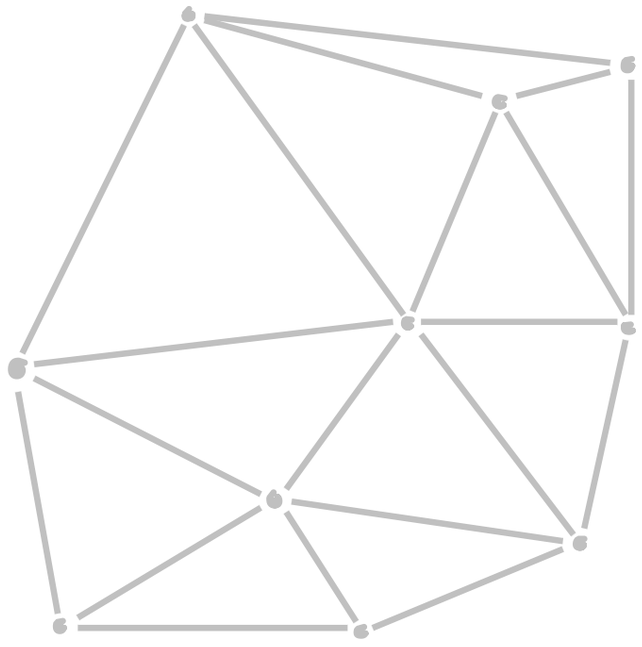
Diagramme de Voronoï (maillage polygonal / polyédrique)

DUALITÉ DELAUNAY/ VORONOÏ

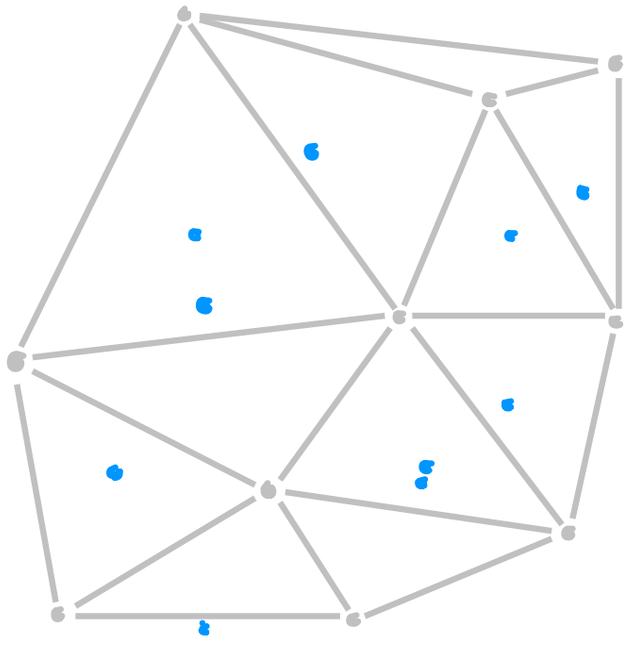
DUALITÉ DELAUNAY / VORONOÏ

Les deux maillages sont duaux :

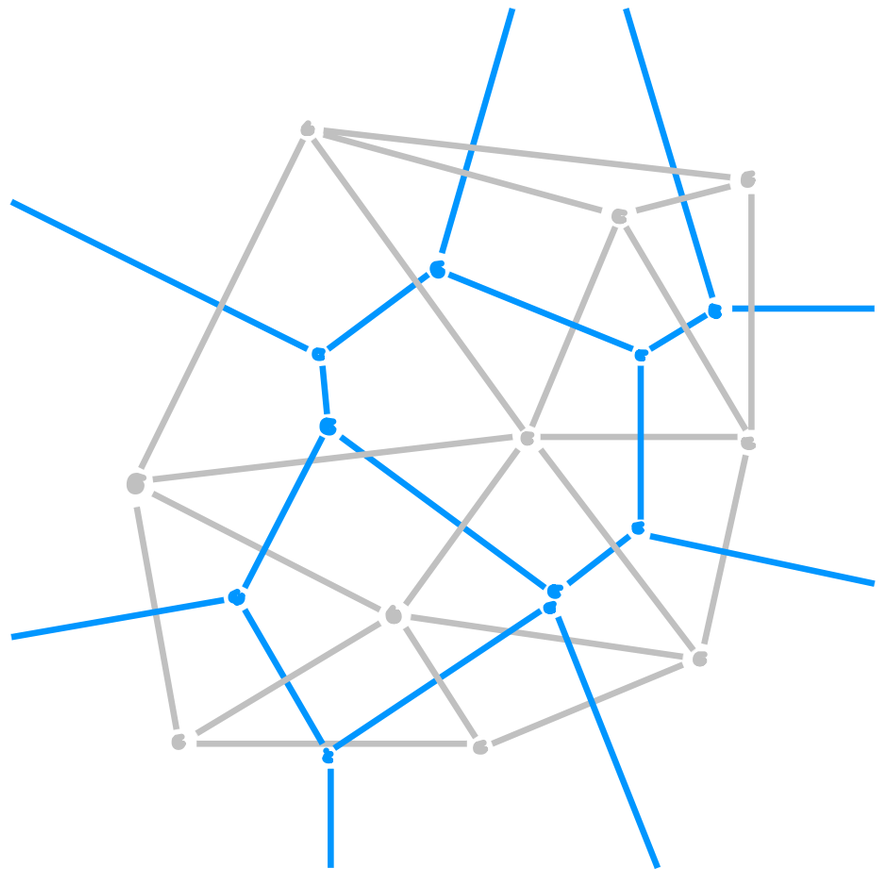
DELAUNAY → **VORONOÏ**



Sommets : centres des cercles circonscrits



Arête : adjacente des faces



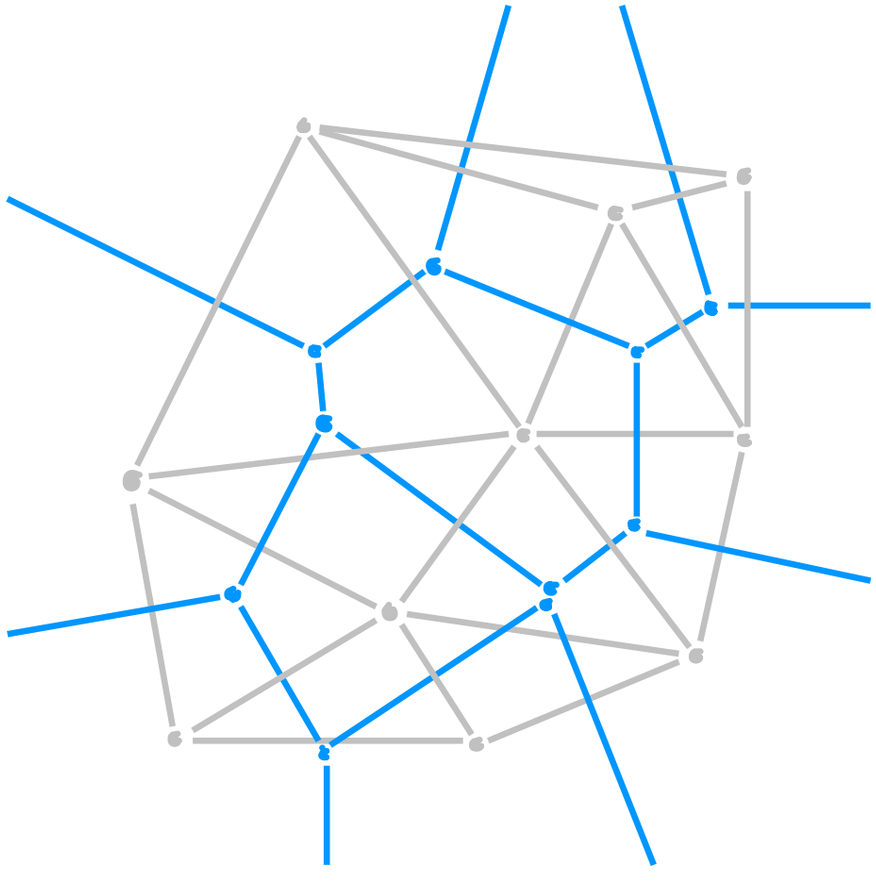
DUALITÉ DELAUNAY / VORONOÏ

Les deux maillages sont duaux :

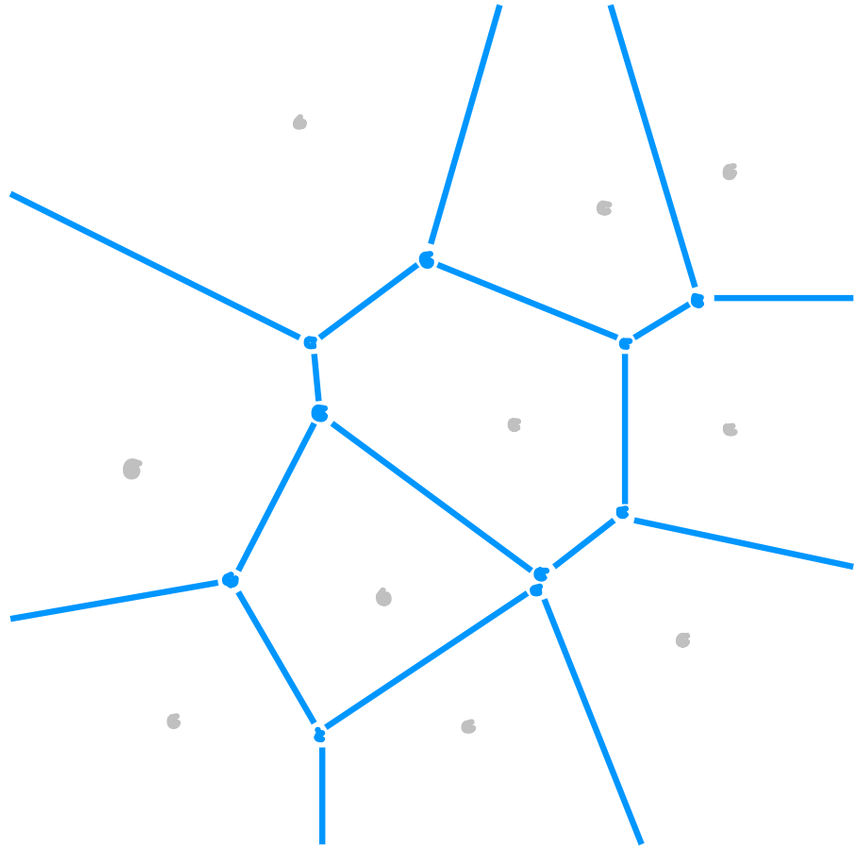
DELAUNAY ⟵ **VORONOÏ**

Arête : adjacente des faces

Sommets : germes des cellules de Voronoï



Voronoï non borné



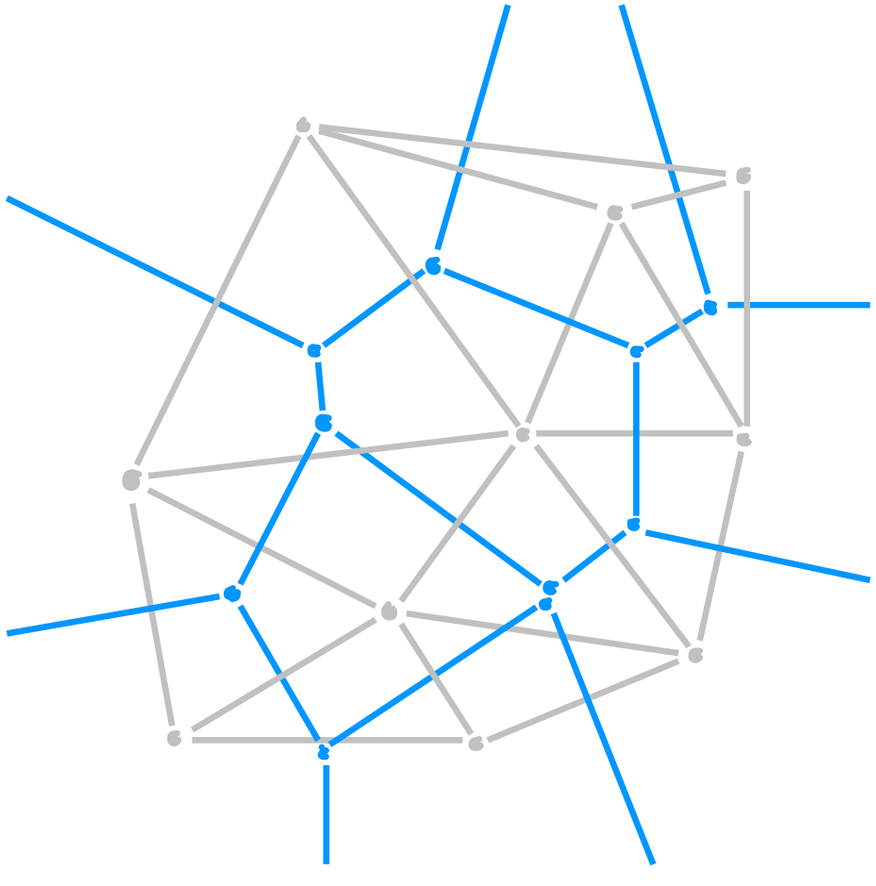
DUALITÉ DELAUNAY / VORONOÏ

Les deux maillages sont duaux :

DELAUNAY

Algorithmes

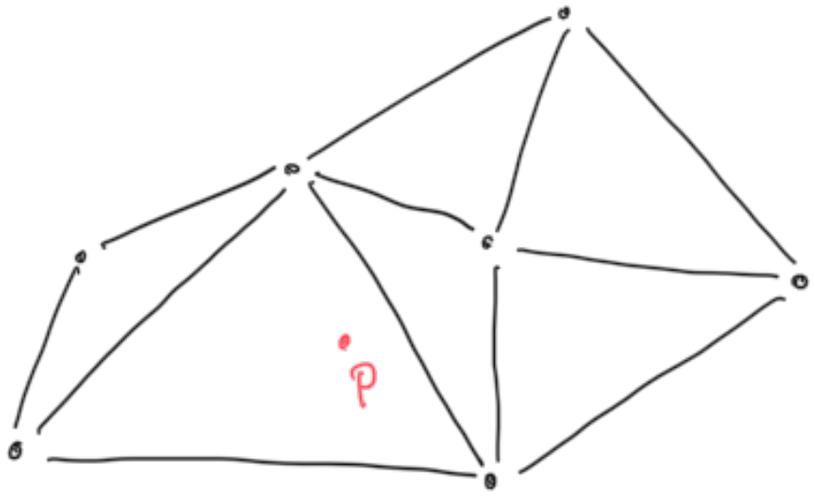
VORONOÏ



Algorithmes de maillage (Delaunay/Voronoï)

ALGORITHME BOWYER-WATSON (INSERTION)

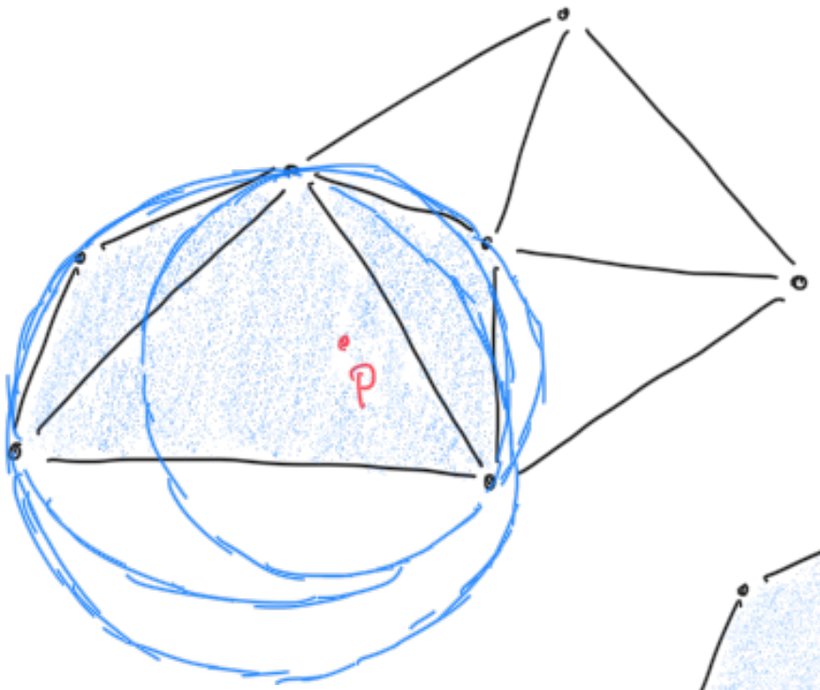
ALGORITHME DE BOWYER / WATSON – PRINCIPE D'INSERTION

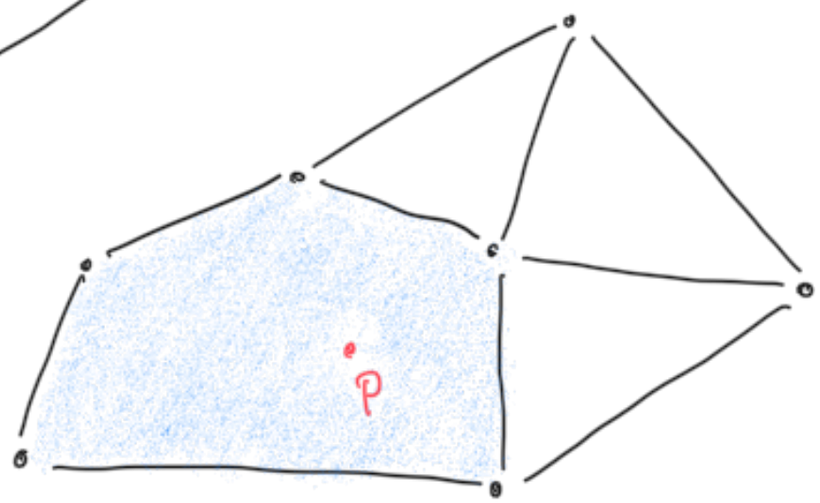


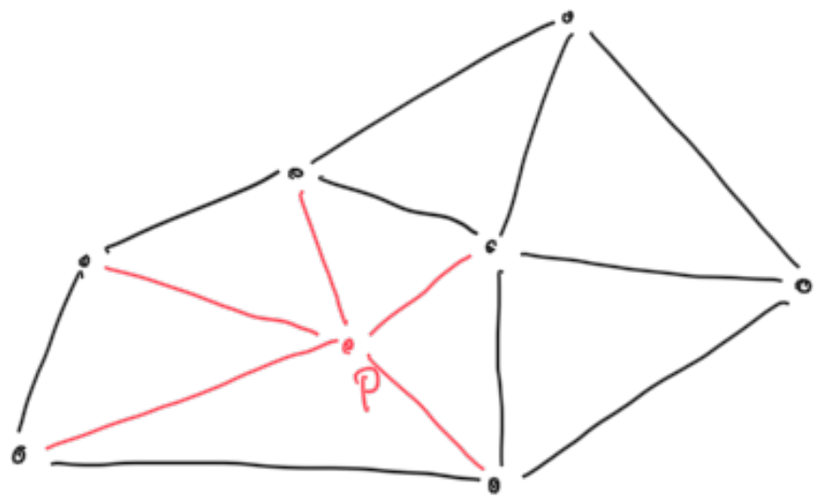
Principe

Etant donnée une triangulation de Delaunay, on sait efficacement insérer un point P :

- Trouver tous les triangles ne respectant plus la propriété du cercle vide pour P
- Les supprimer → polygone d'insertion de P
- Trianguler à partir de P





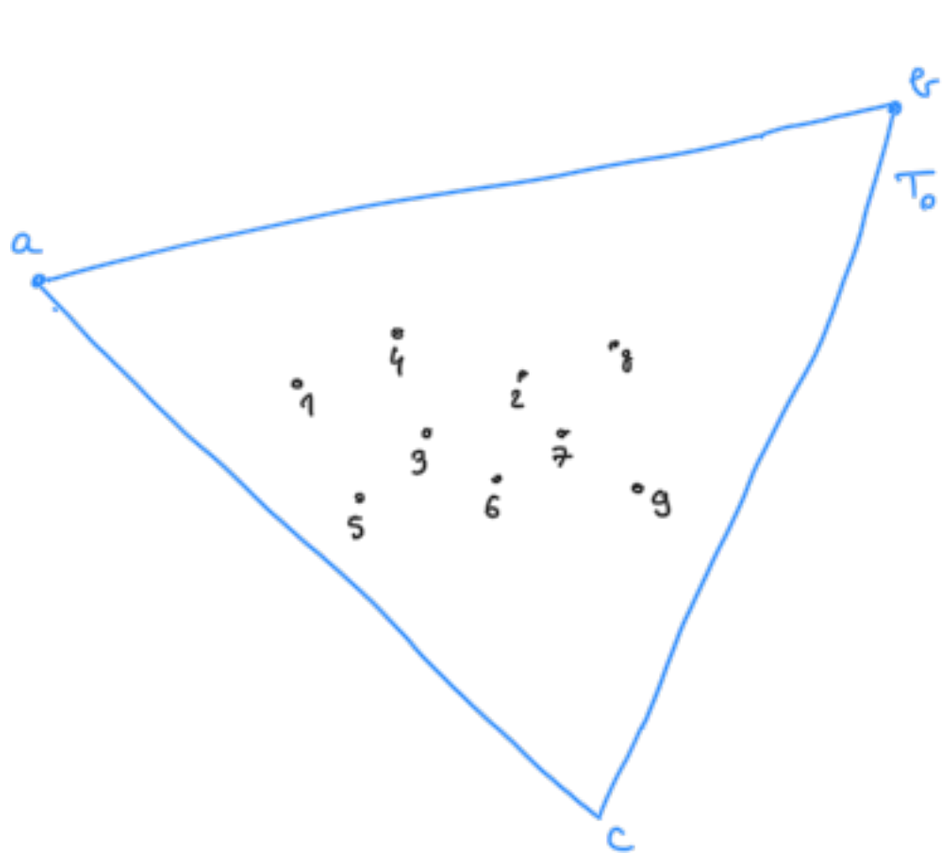


ALGORITHME DE BOWYER / WATSON – INSERTION

Algorithme

$\{P_1, \ldots, P_n\}$ ensemble de points

- Soit T triangle initial contenant tous les points (ajout de 3 points virtuels)
- Pour i de 1 à n
  - Insérer le point P_i dans la triangulation
- Supprimer les triangles connexes aux points virtuels



ALGORITHME DE BOWYER / WATSON – INSERTION

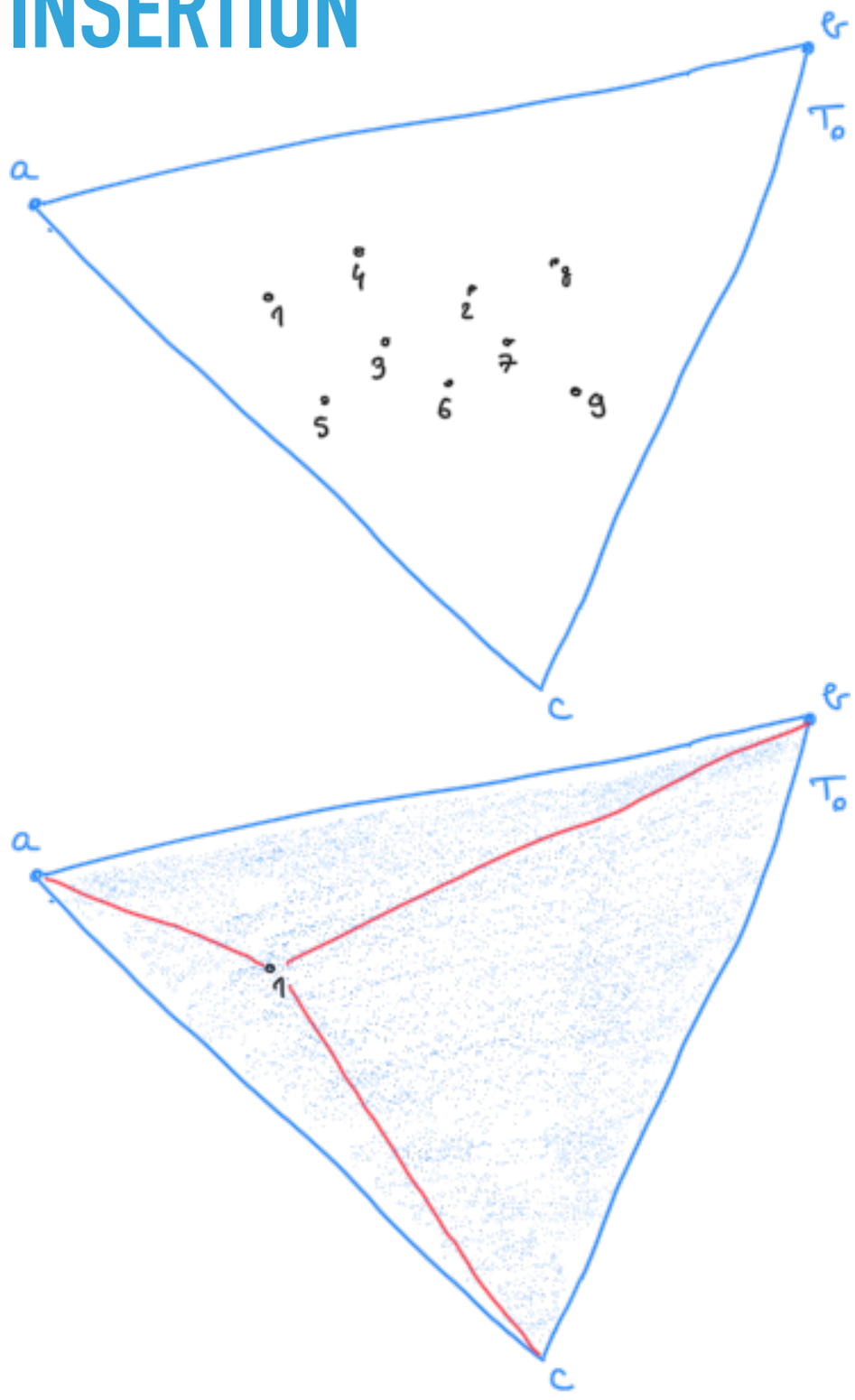
Algorithme

$\{P_1, \ldots, P_n\}$ ensemble de points

- Soit T triangle initial contenant tous les points (ajout de 3 points virtuels)

- Pour i de 1 à n
  - Insérer le point P_i dans la triangulation

- Supprimer les triangles connexes aux points virtuels



ALGORITHME DE BOWYER / WATSON – INSERTION

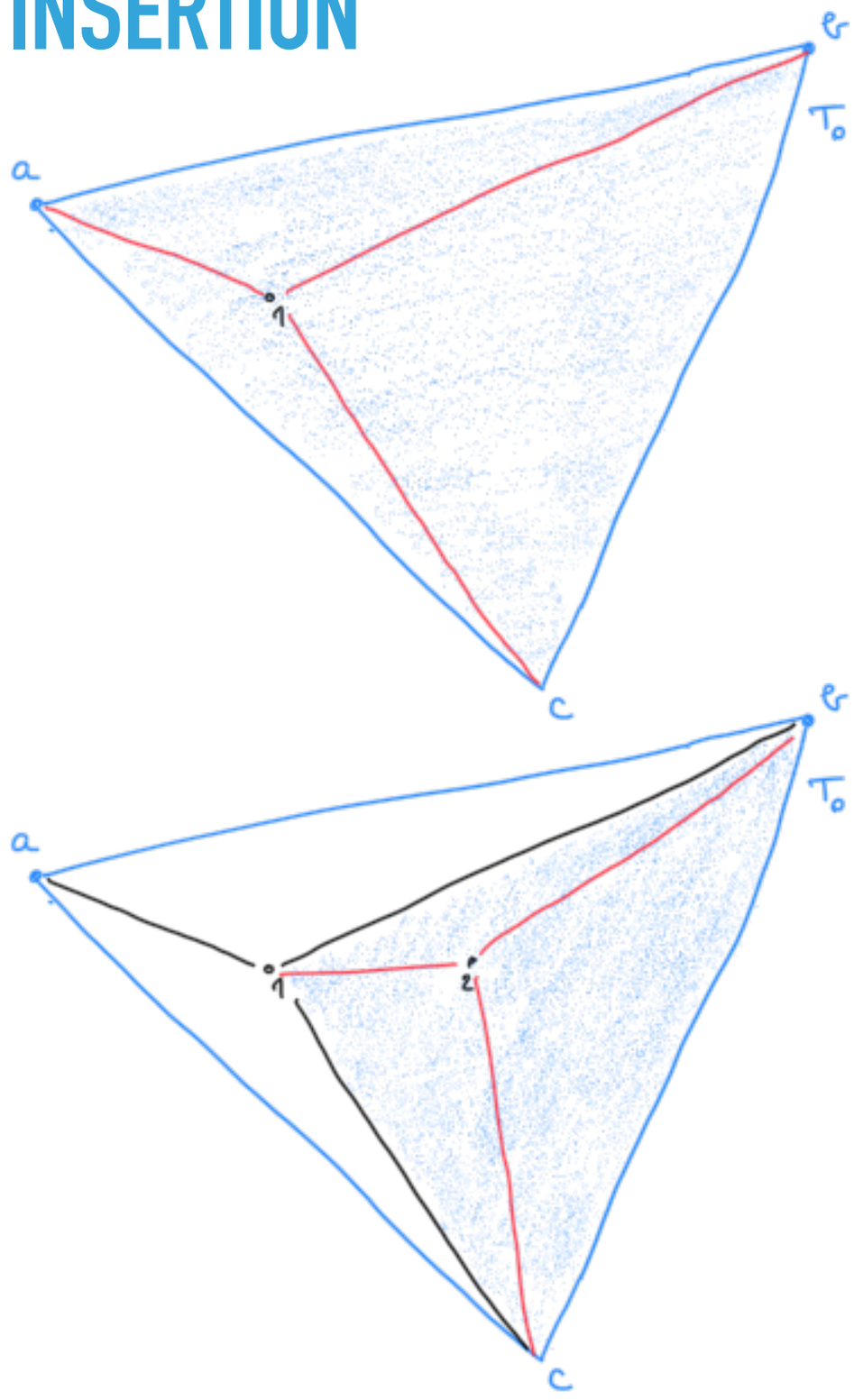
Algorithme

$\{P_1, \ldots, P_n\}$ ensemble de points

- Soit T triangle initial contenant tous les points (ajout de 3 points virtuels)

- Pour i de 1 à n
  - Insérer le point P_i dans la triangulation

- Supprimer les triangles connexes aux points virtuels



ALGORITHME DE BOWYER / WATSON – INSERTION

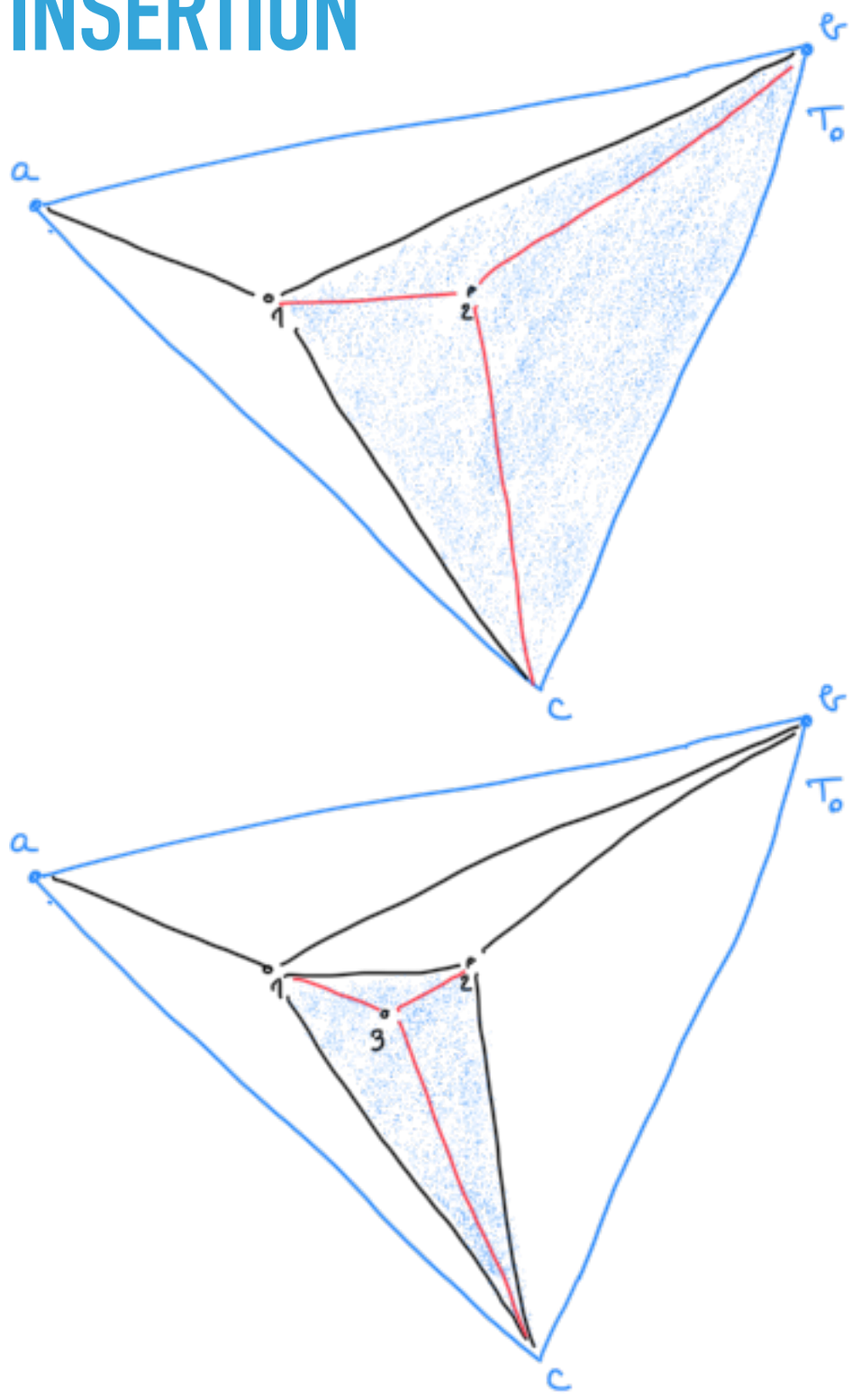
Algorithme

$\{P_1, \ldots, P_n\}$ ensemble de points

- Soit T triangle initial contenant tous les points (ajout de 3 points virtuels)

- Pour i de 1 à n
  - Insérer le point P_i dans la triangulation

- Supprimer les triangles connexes aux points virtuels

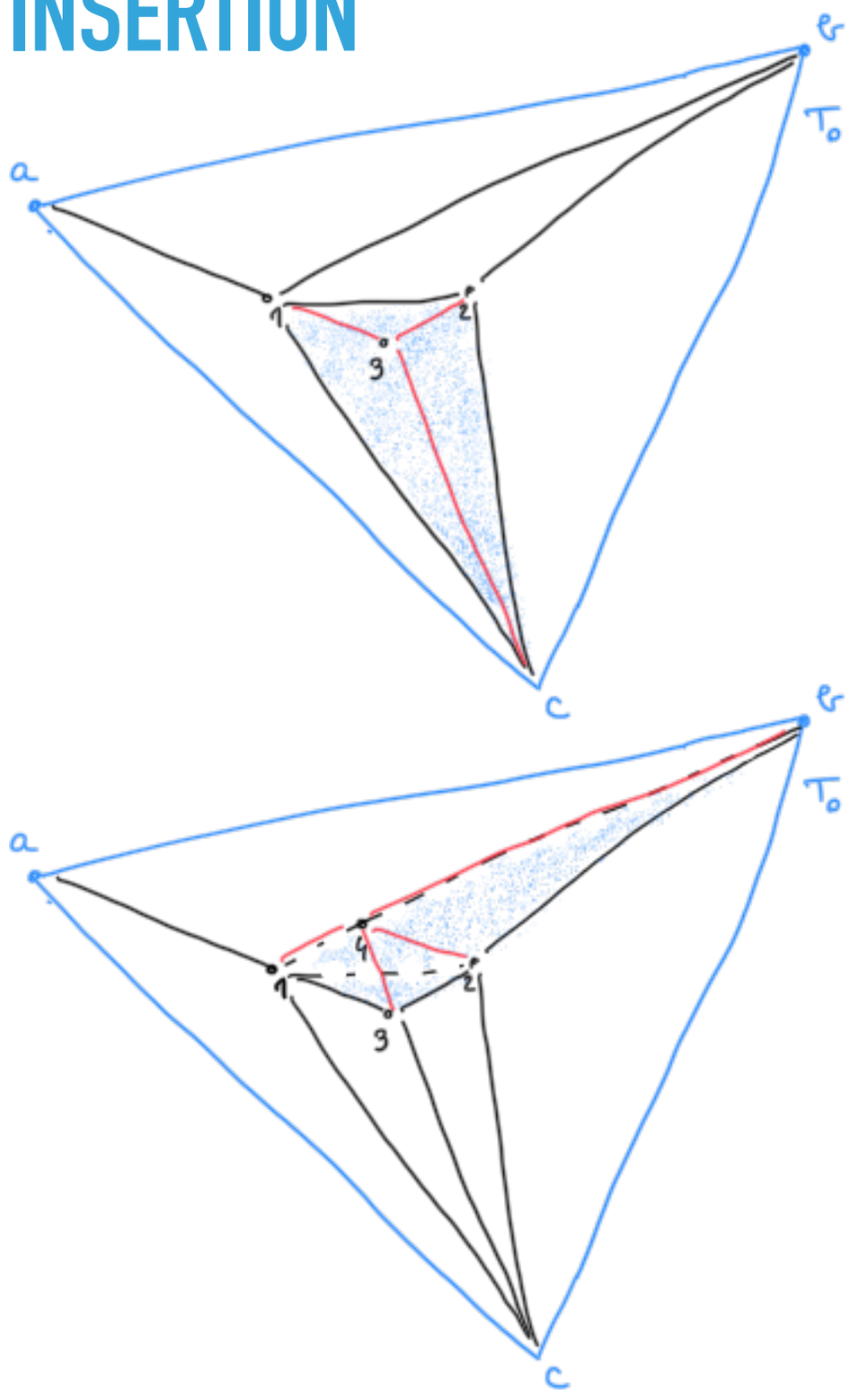


ALGORITHME DE BOWYER / WATSON – INSERTION

Algorithme

$\{P_1, \ldots, P_n\}$ ensemble de points

- Soit T triangle initial contenant tous les points (ajout de 3 points virtuels)
- Pour i de 1 à n
  - Insérer le point P_i dans la triangulation
- Supprimer les triangles connexes aux points virtuels

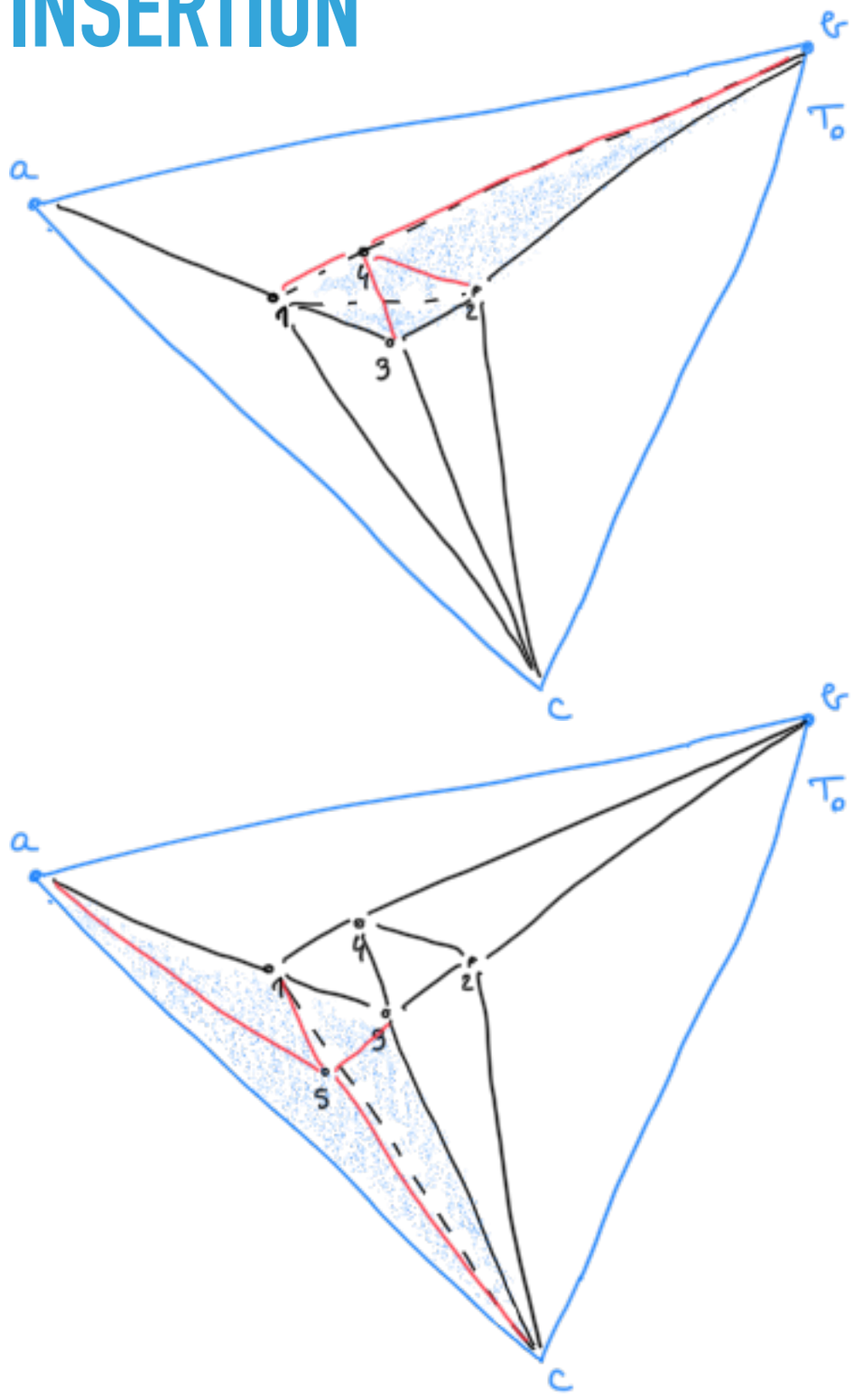


ALGORITHME DE BOWYER / WATSON – INSERTION

Algorithme

$\{P_1, \ldots, P_n\}$ ensemble de points

- Soit T triangle initial contenant tous les points (ajout de 3 points virtuels)
- Pour i de 1 à n
  - Insérer le point P_i dans la triangulation
- Supprimer les triangles connexes aux points virtuels



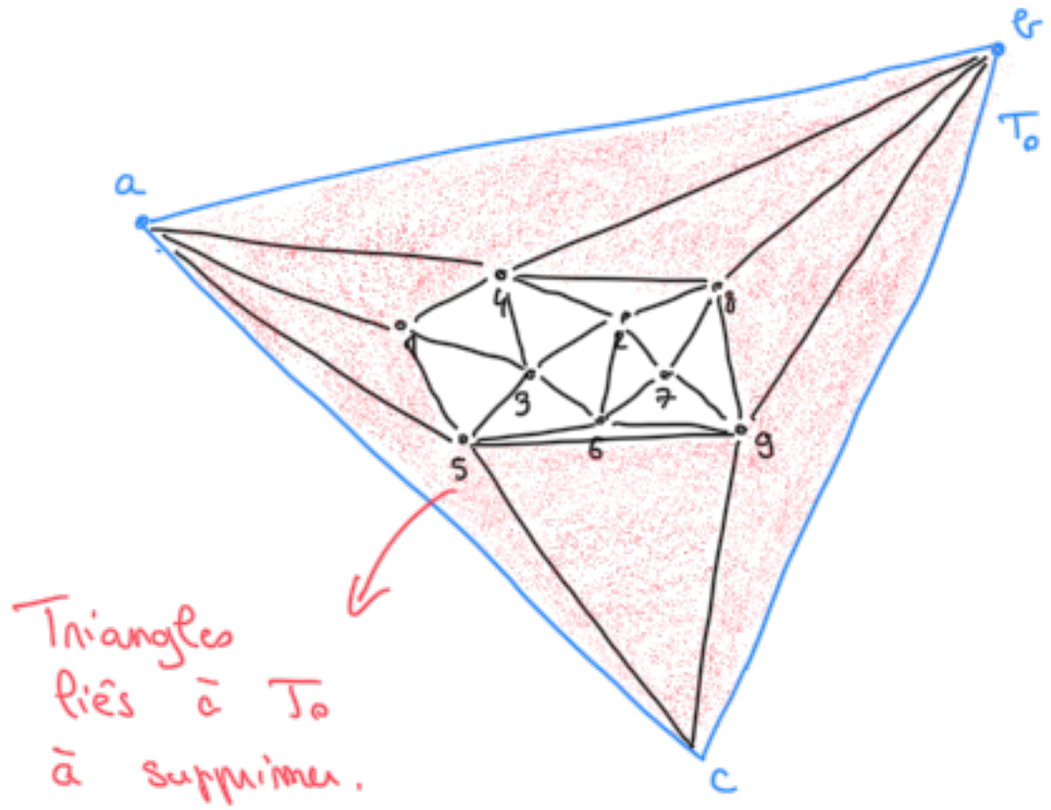
ALGORITHME DE BOWYER / WATSON – INSERTION

Algorithme

$\{P_1, \ldots, P_n\}$ ensemble de points

etc. ...

- Soit T triangle initial contenant tous les points (ajout de 3 points virtuels)
- Pour i de 1 à n
  - Insérer le point P_i dans la triangulation
- Supprimer les triangles connexes aux points virtuels



ALGORITHME DE BOWYER / WATSON – COMPLEXITÉ

Algorithme

$\{P_1, \ldots, P_n\}$ ensemble de points

- Soit T triangle initial contenant tous les points (ajout de 3 points virtuels)
- Pour i de 1 à n
  - Insérer le point P_i dans la triangulation
- Supprimer les triangles connexes aux points virtuels

Complexité min : $\mathcal{O}(n.\log n)$

Complexité moyenne : $\mathcal{O}(n^2)$

Nécessite une implémentation efficace :

- SDD efficace pour le maillage *(1/2-arêtes)*
- Détection rapide des triangles du polygone d'insertion *(structure accélératrice - cf. chapitre 4)*

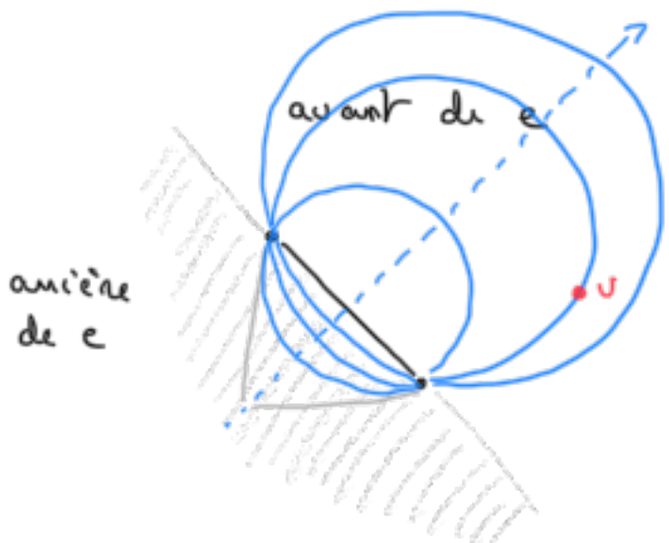
MÉTHODE DU PAQUET-CADEAU (AVANCÉE DE FRONT)

ALGORITHME DU « PAQUET CADEAU »

Principe

On suppose qu'on a déjà maillé une partie des points avant un « front » donné par une arête e :

- On cherche le point v donnant le cercle circonscrit ne contenant aucun autre point
- Ajouter le triangle (e, v)

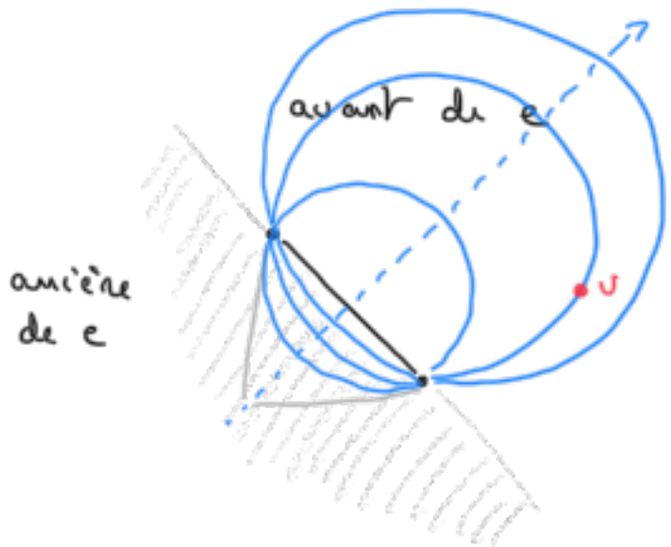


ALGORITHME DU « PAQUET CADEAU »

Algorithme

Soit e arête de front :

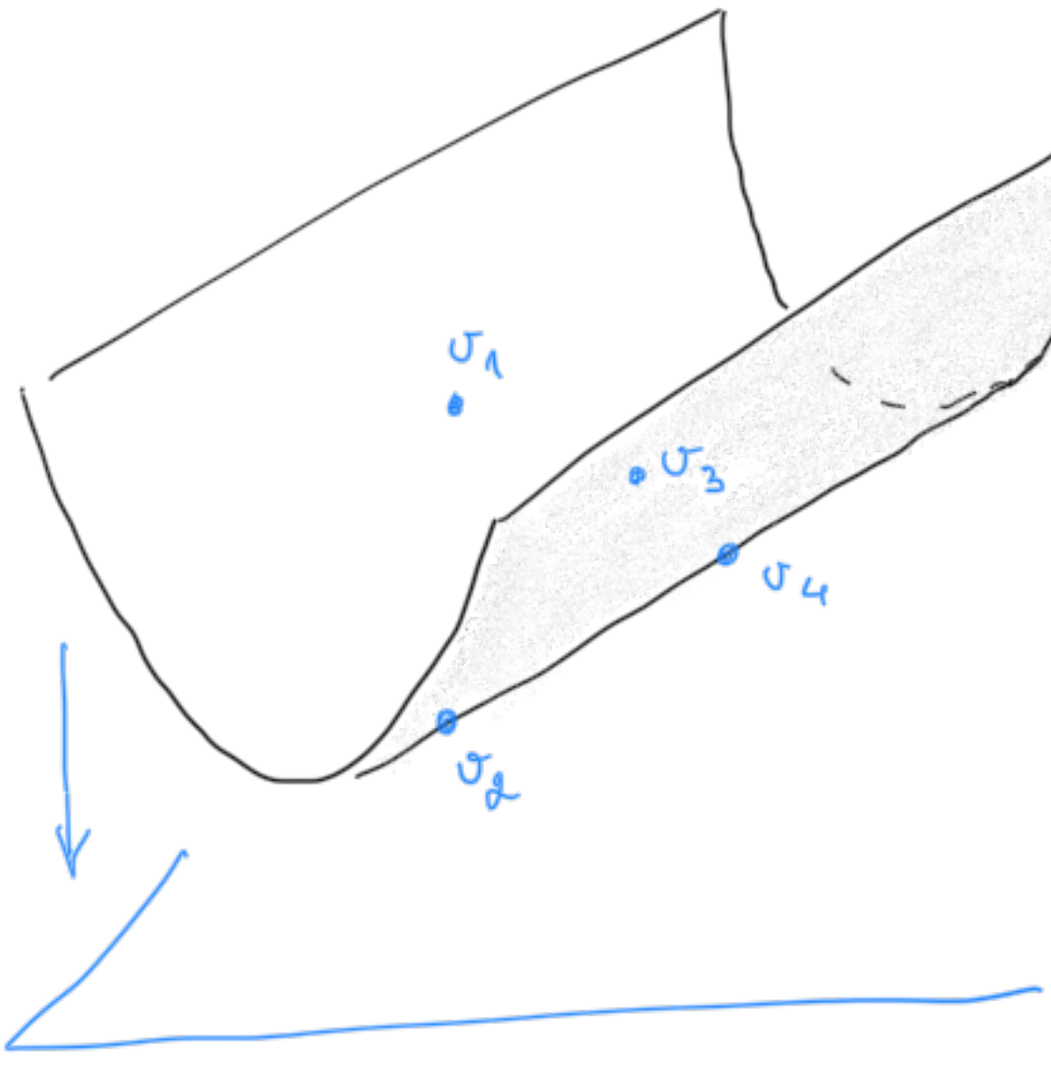
- $\tau \leftarrow \emptyset$ // Triangle « en cours »
- Pour chaque sommet v non maillé
  - Si (v à l'avant de e) ET (($\tau = \emptyset$) OU ($v \in \text{circonscrit}(\tau)$))
    - $\tau \leftarrow \text{triangle}(\tau, e)$
- Puis :
  - $\tau = \emptyset$ si e arête de bord
  - Sinon ajouter τ au maillage et choisir une arête de bord comme nouveau front



MAILLAGE 2,5D

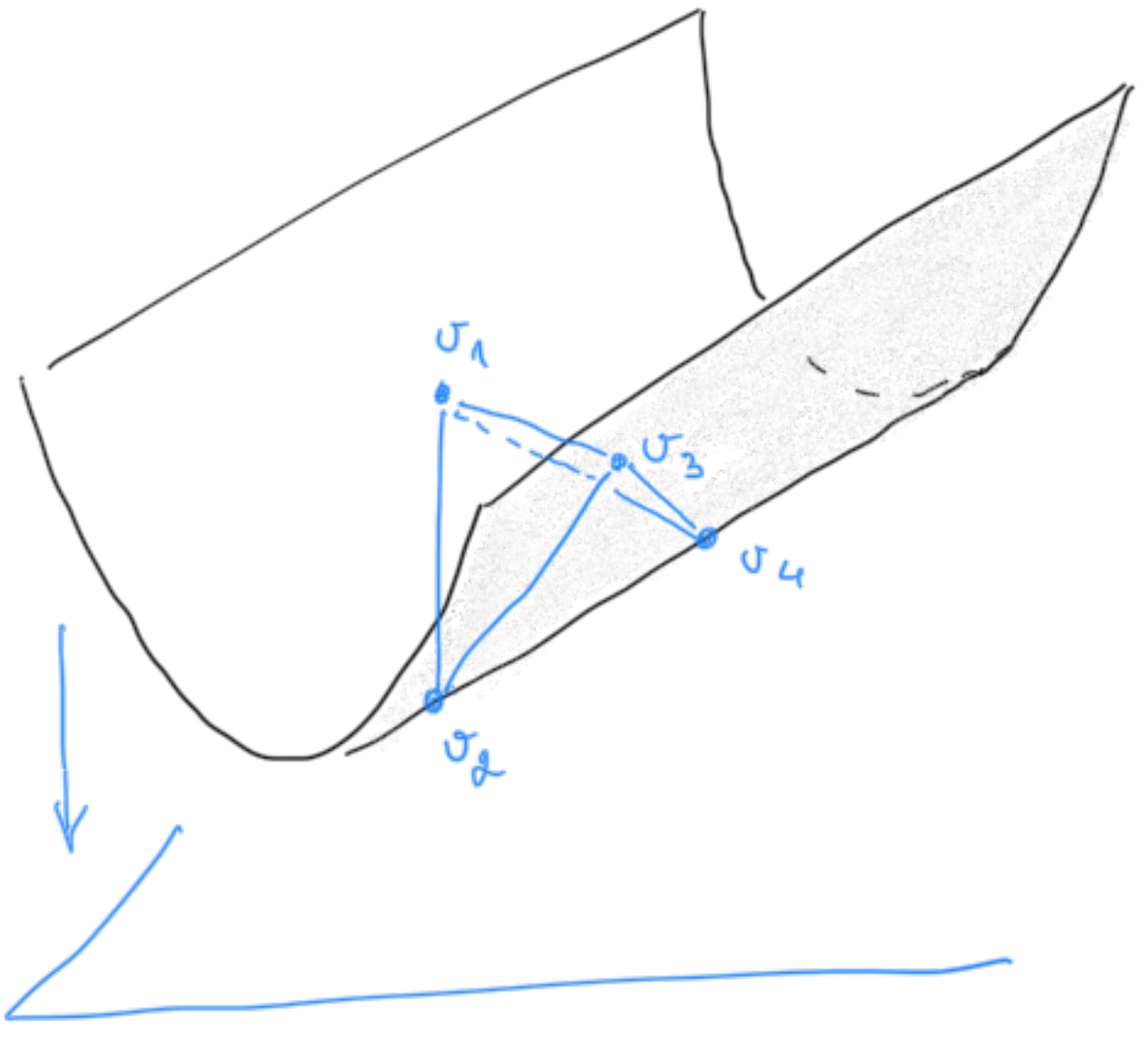
MAILLAGES 2,5D

- La question n'est pas simple …



MAILLAGES 2,5D

- La question n'est pas simple …



STRUCTURES ACCÉLÉRATRICES

POURQUOI DES STRUCTURES ACCÉLÉRATRICES ?

Donner une structure à des données non structurées

Ensemble « brut » de données

Ensemble de points
Ensemble de triangles
Ensemble de …

Voisinage ?

Adjacence ?

Parcours ?

- Quel est le plus proche voisin de P ?
- Quels sont les k plus proches voisins de P ?
- Quels sont les voisins de P à une distance inférieure à d ?

POURQUOI DES STRUCTURES ACCÉLÉRATRICES ?

Donner une structure à des données non structurées

Structure de donnée permettant d'organiser les données

- grille régulière
- kD-tree
- quadtree / octree

Ensemble « brut » de données

Ensemble de points
Ensemble de triangles
Ensemble de …

Voisinage ?

Adjacence ?

Parcours ?

POURQUOI DES STRUCTURES ACCÉLÉRATRICES ?

Si l'ensemble est stocké dans un simple tableau - complexités :

- Quel est le plus proche voisin de P ? → $\mathcal{O}(n)$
- Quels sont les k plus proches voisins de P ? → $\mathcal{O}(k \times n)$
- Quels sont les voisins de P à une distance inférieure à d ? → $\mathcal{O}(n)$

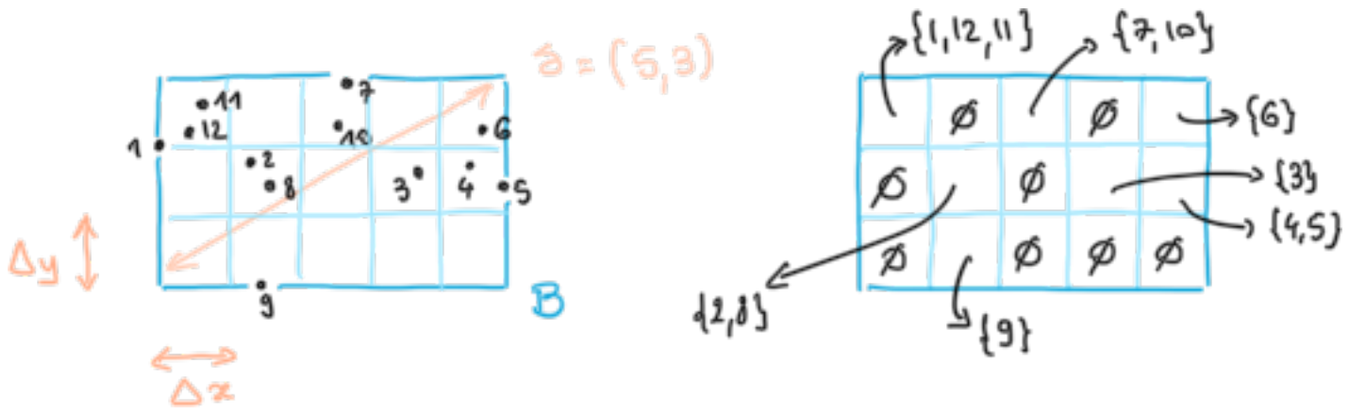
Exorbitant !

GRILLES RÉGULIÈRES

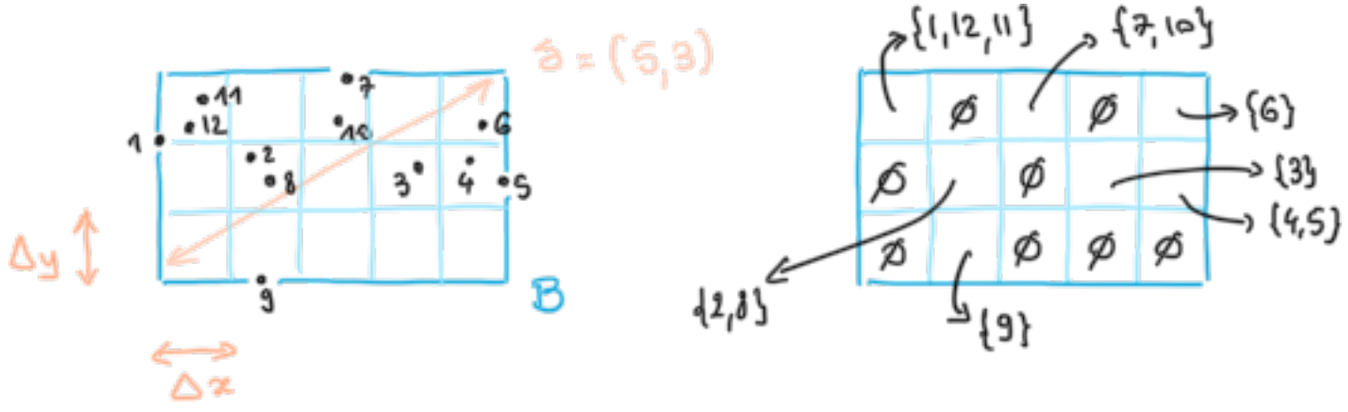
STRUCTURE LA PLUS BASIQUE : GRILLE RÉGULIÈRE

Soit $\{P_i \ : \ i = 1 \ldots N\}$ un ensemble de points

- Soit B la boîte englobante des points
- $(\Delta x, \Delta y, \Delta z)$ des pas de discrétisation
- On subdivise B en cellules rectangulaires de taille $(\Delta x, \Delta y, \Delta z)$: pixel en 2D / voxels en 3D
  - Cellule de la grille → liste des indices des sommets qu'elle contient



STRUCTURE LA PLUS BASIQUE : GRILLE RÉGULIÈRE



- Quel est le plus proche voisin de P ?

$\mathcal{O}(8 \times \max_n)$

Peut-on faire mieux ?

→ subdivision adaptative

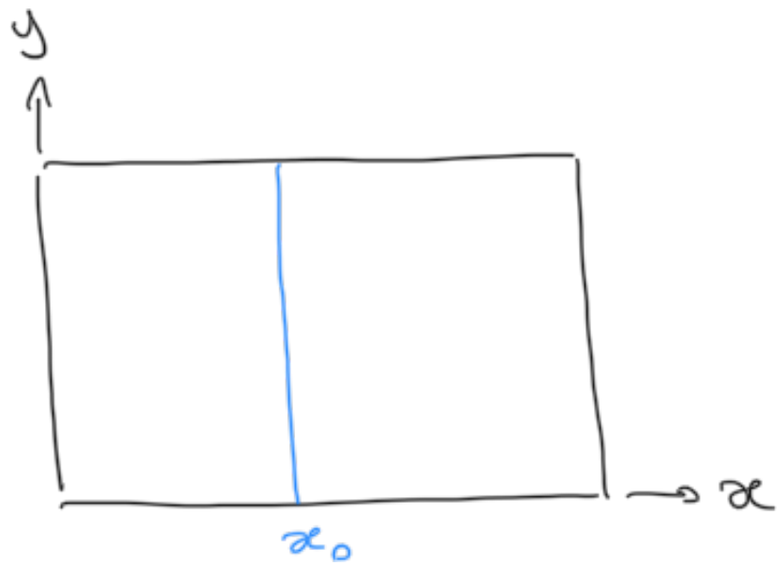
KD-TREE

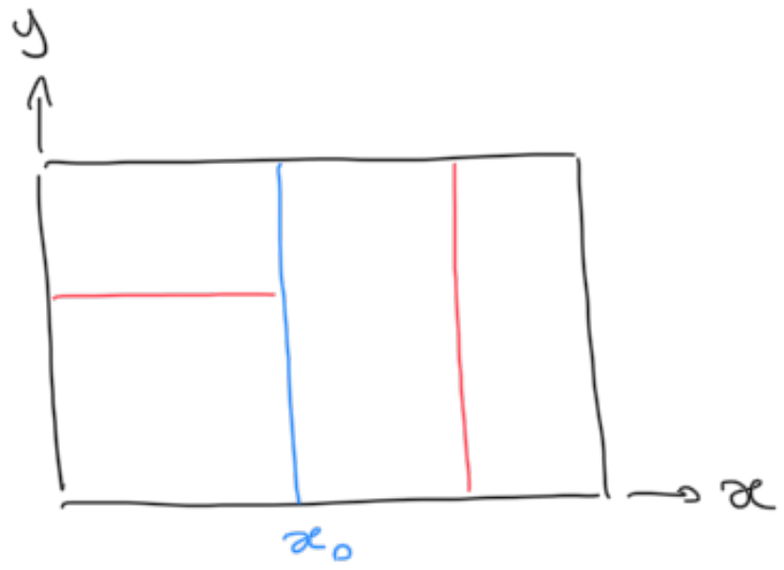
KD-TREE

Structure arborescente

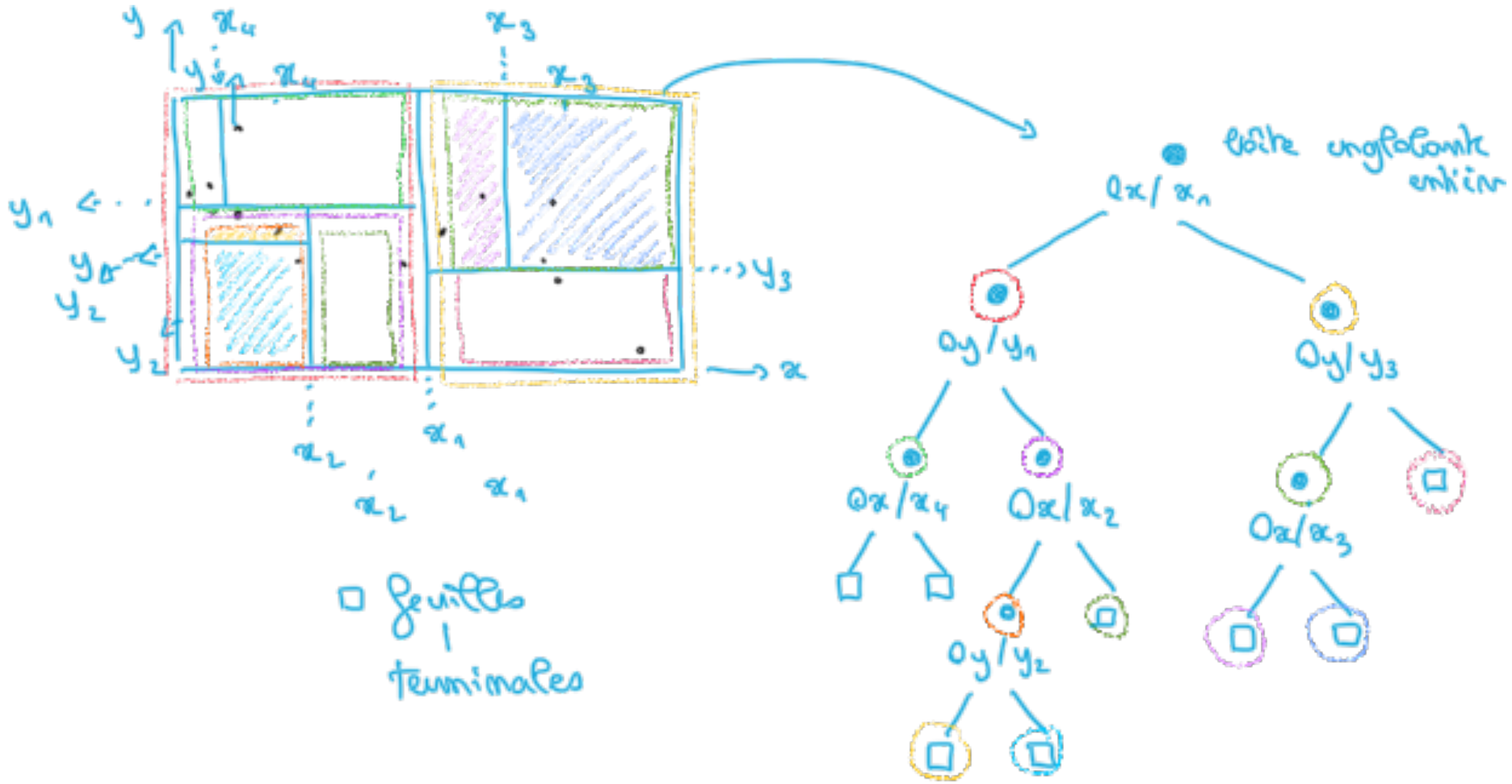
A chaque étape, l'espace est subdivisé en 2 :

- Selon un axe
- En un point choisi





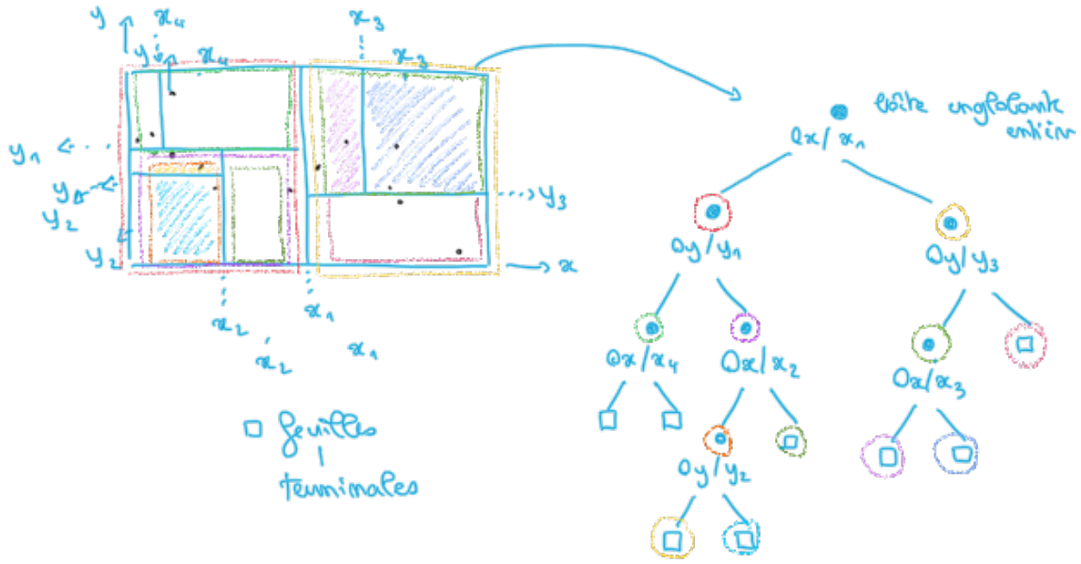
KD-TREE



Arrêt de la subdivision :

- Nombre de point par feuille $< \alpha$
- Profondeur $= \delta$

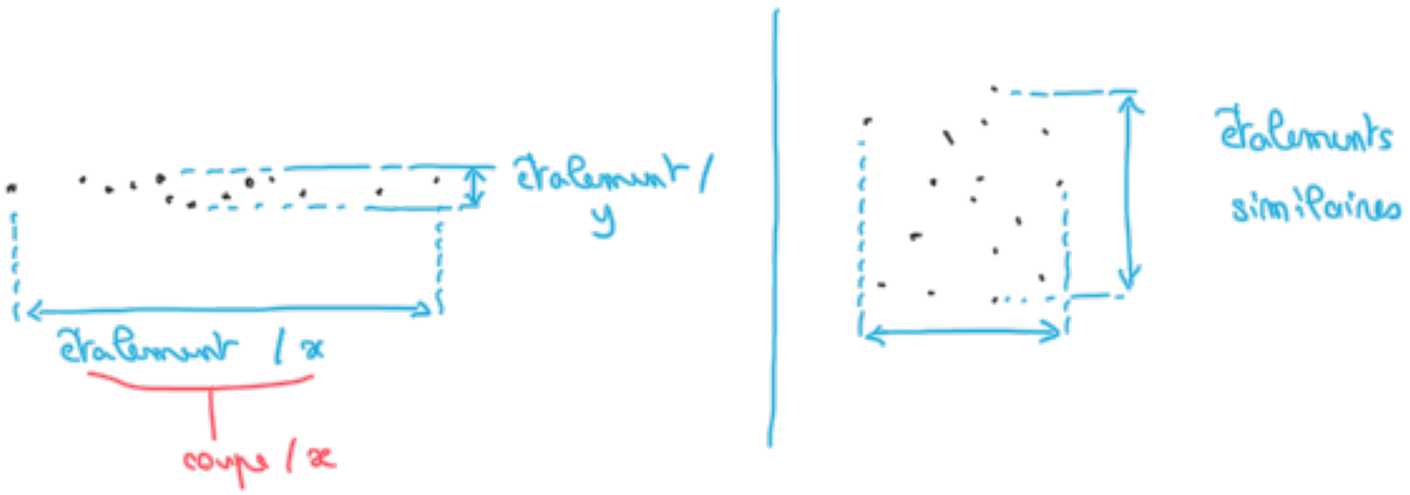
KD-TREE



- Choix de l'axe / position de la coupe :
- Différentes stratégies en fonction des besoins
- Les plus standard :
  - Axe :
    - Alternance des axes de coupe $(x, y, z, x \ldots)$
    - **Axe de plus grande extension** des coordonnées
  - Position :
    - Médiane des points
    - Le « sliding midpoint » / point milieu glissant

KD-TREE / CHOIX DE L'AXE

- Axe :
  - Alternance des axes de coupe ($x, y, z, x \ldots$)
  - **Axe de plus grande extension** des coordonnées



KD-TREE / POSITION DE COUPE

- Position :
  - **Médiane des points**
  - Le « sliding midpoint » / point milieu glissant

→ Choix naturel (surtout si l'axe est celui d'extension maximale)

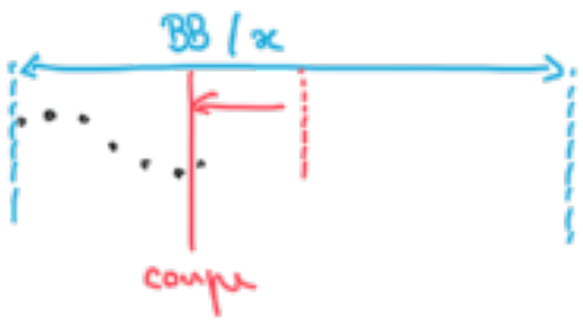
→ Le plus adaptatif

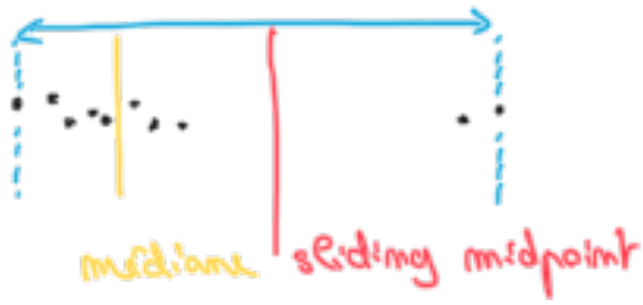
→ Nécessite un tri ($\mathcal{O}(n \log n)$)

→ Potentiellement loin d'une subdivision binaire

KD-TREE / POSITION DE COUPE

- Position :
  - Médiane des points
  - Le **« sliding midpoint »**
    - On subdivise en deux
    - Si l'un des deux sous-espaces est vide, on fait glisser la coupe jusqu'à ajouter un point à cet espace





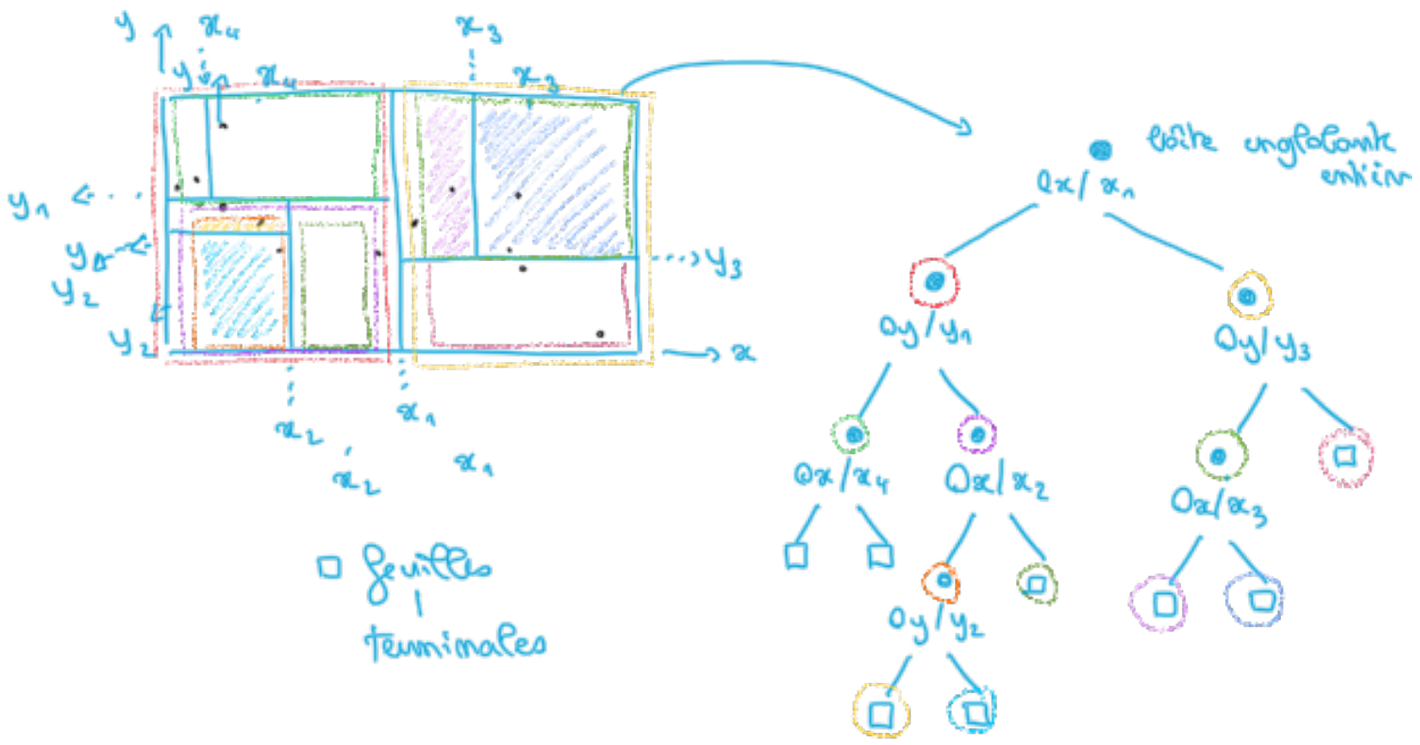
→ Moins adaptatif

→ Plus proche d'une subdivision binaire

KD-TREE / POINT IMPLÉMENTATION

- Stockage dans les noeuds :
  - Dimension de coupe
  - Position de coupe
- Stockage dans les feuilles :
  - Liste des indices des points contenus

Boîte englobante des points non stockée en général

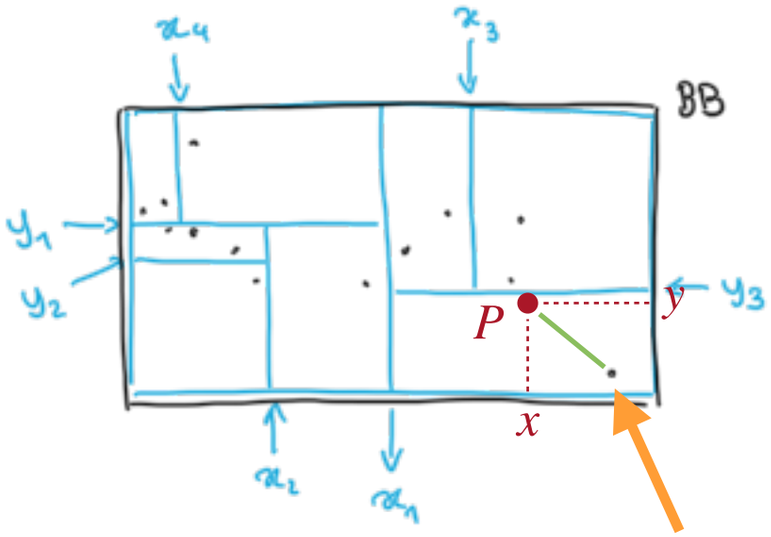


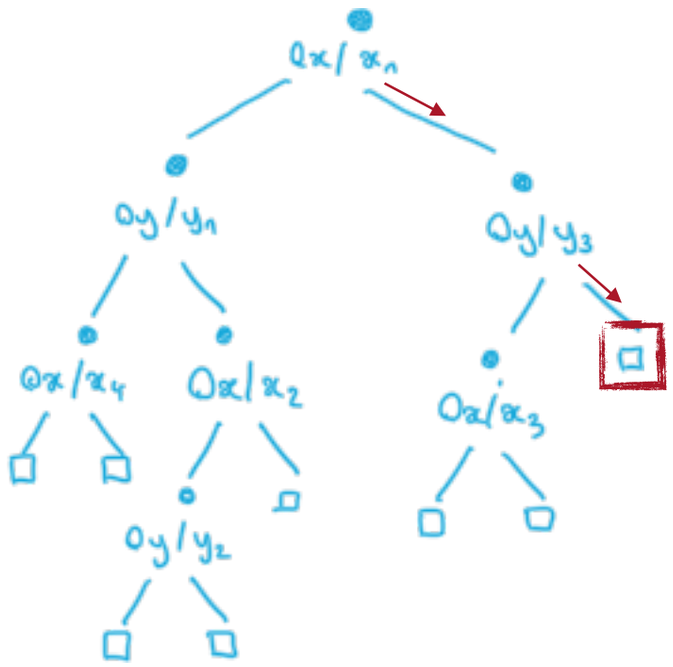
KD-TREE

- Quel est le plus proche voisin de P ?

    1. Déterminer la feuille de P

    2. Trouver le plus proche voisin de P dans cette feuille → distance d

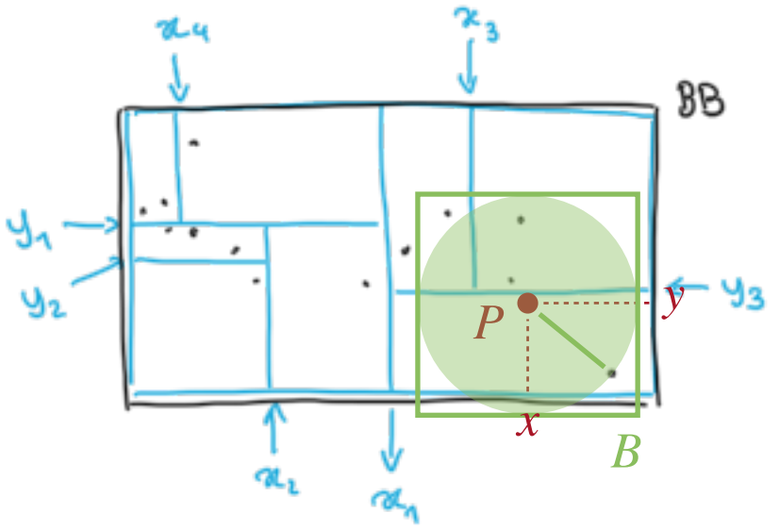


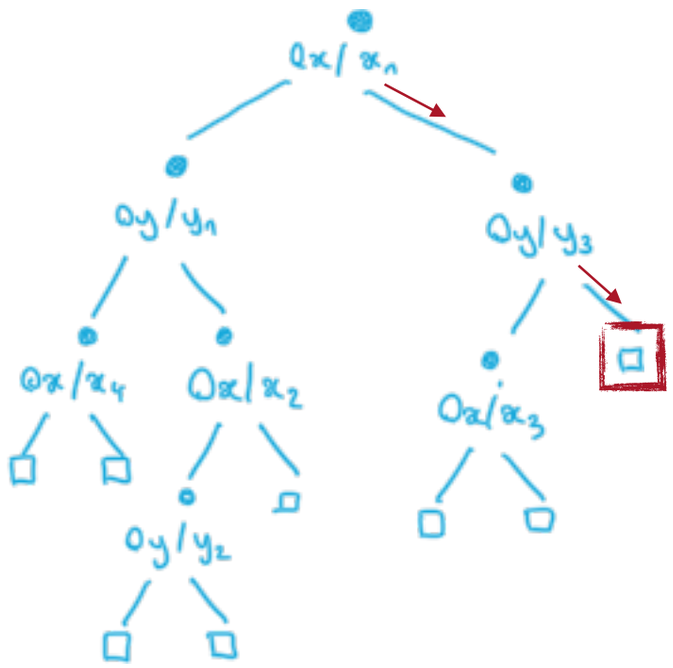


KD-TREE

- Quel est le plus proche voisin de P ?

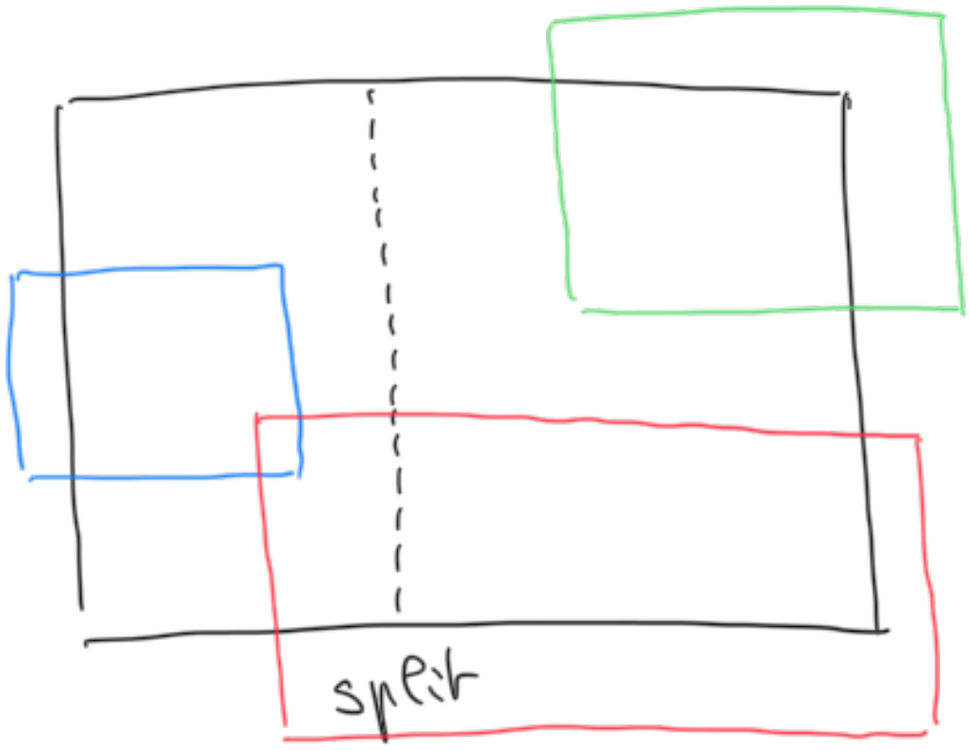
    1. Déterminer la feuille de P
    2. Trouver le plus proche voisin de P dans cette feuille → distance d
    3. Déterminer le plus proche voisin de P dans la boîte $B = [P - (d, d, d)^t, P + (d, d, d)^t]$

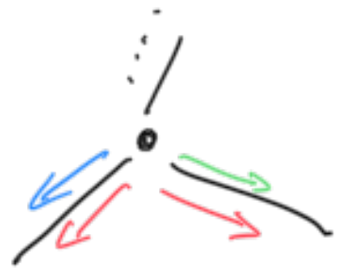




KD-TREE

- Déterminer les feuilles rencontrant une boîte B donnée ?
  - Parcours récursif …

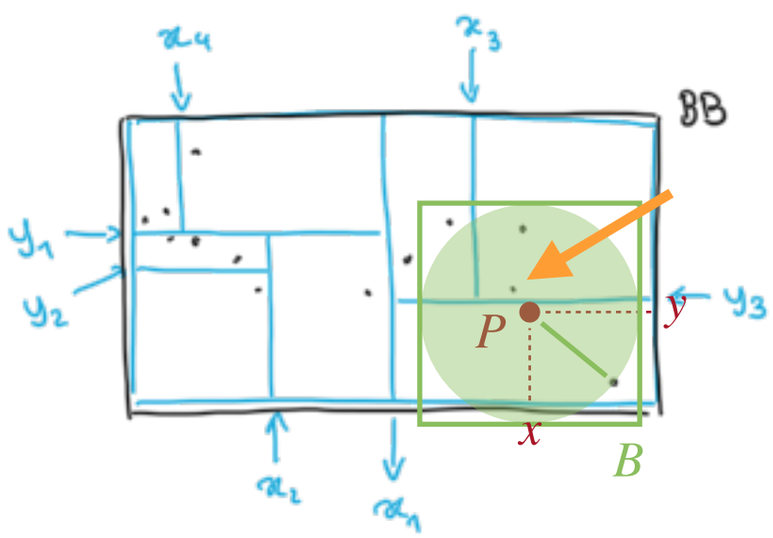


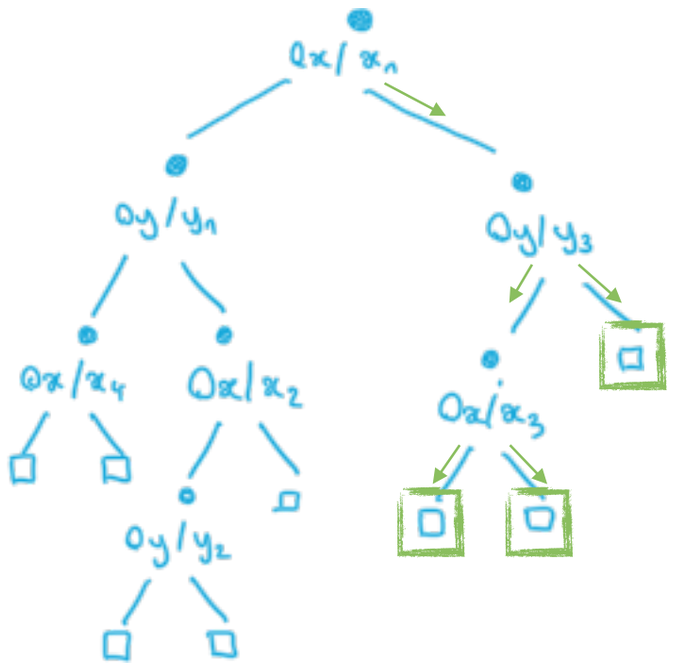


KD-TREE

- Déterminer les feuilles rencontrant une boîte B donnée ?

    1. Déterminer la feuille de P
    2. Trouver le plus proche voisin de P dans cette feuille → distance d
    3. Déterminer le plus proche voisin de P dans la boîte $B = [P - (d, d, d)^t, P + (d, d, d)^t]$





$\mathcal{O}(\text{hauteur} + \text{nbre points par feuille}) \rightarrow \mathcal{O}(\delta + \alpha)$

→ Equilibre δ/α important …

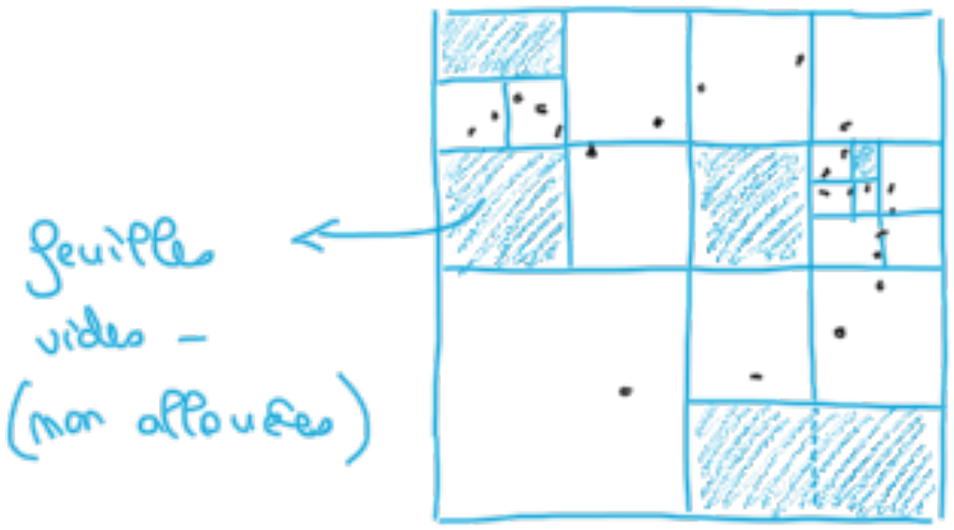
QUADTREE/ OCTREE

QUADTREE / OCTREE

Structure arborescente (**quadtree** en 2D / **octree** en 3D)

- Partant d'une boîte englobante carrée
- Chaque noeud est subdivisé en 2^d cellules
  - Tant qu'il contient plus de α points
  - Ou que la profondeur $\leqslant \beta$
- [Seuls les fils non vides sont alloués]

Exemple pour $\alpha = 2$

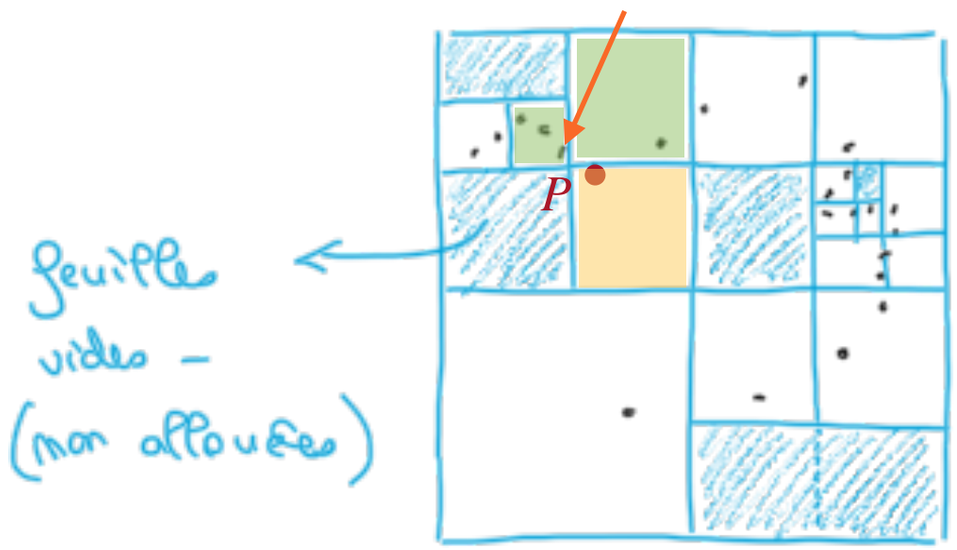


QUADTREE / OCTREE

- Quel est le plus proche voisin de P ? ⟶ Parcours de l'arbre comme pour les kD-tree : $\mathcal{O}(\alpha + \beta)$
  - La structure d'octree est régulière : **indexation entière efficace**
    1. Déterminer l'indice I de la feuille de P (via ses coordonnées) ⟵
    2. Déterminer les feuilles voisines ⟵
    3. Calculer le plus proche voisin ⟵

Exemple pour $\alpha = 2$

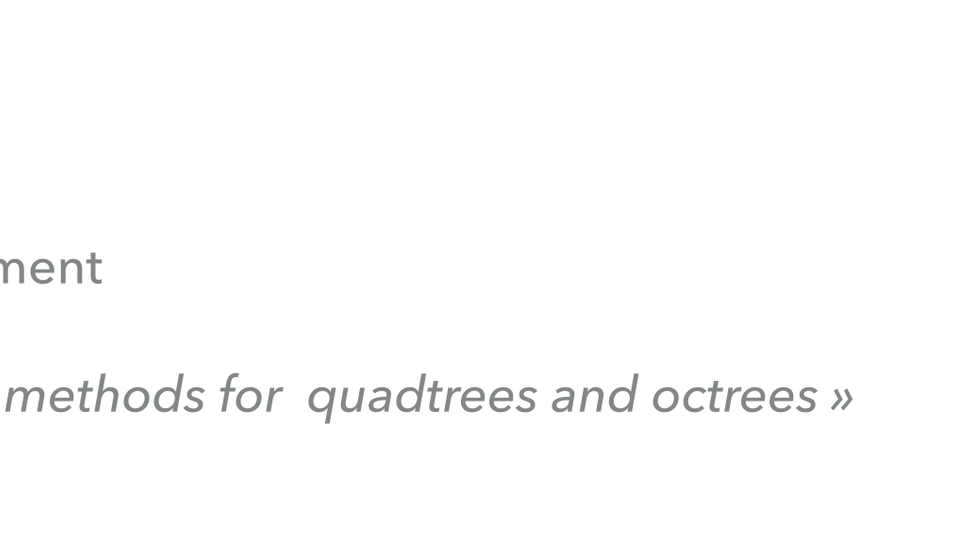
$\mathcal{O}(\text{nbre points par feuille}) \rightarrow \mathcal{O}(\alpha)$



QUADTREE / OCTREE

- Plus technique à implémenter efficacement
  - *Voir « Simple and efficient traversal methods for quadtrees and octrees » S. Frisken - R. Perry*
- On le trouve dans toutes les bonnes bibliothèques :
  - PCL
  - CGAL
  - Matlab, Python …

Exemple pour $\alpha = 2$



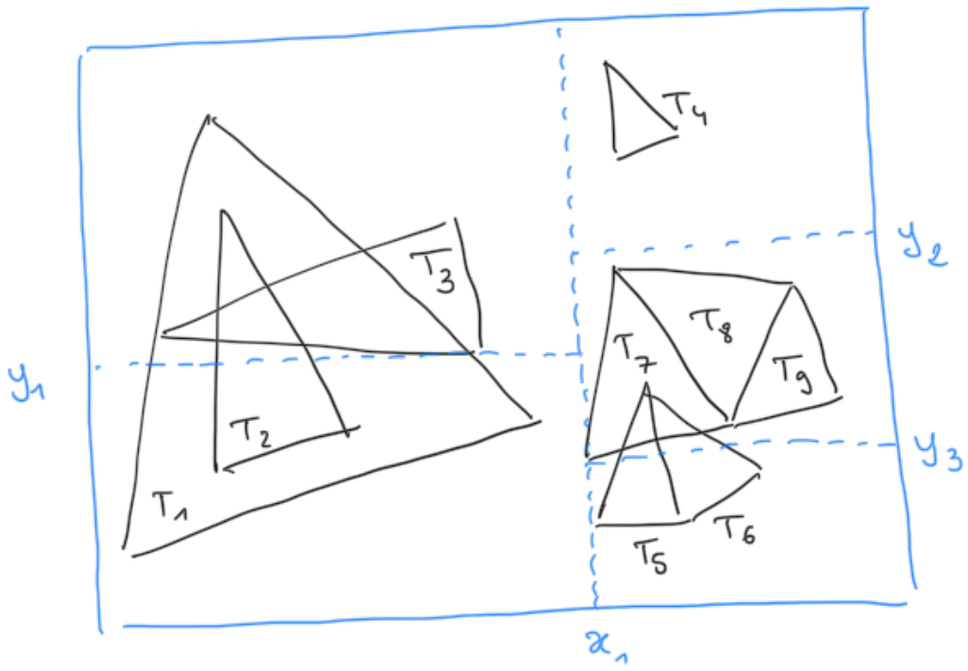
STRUCTURES ACCÉLÉRATRICES: DES POINTS MAIS PAS QUE . . .

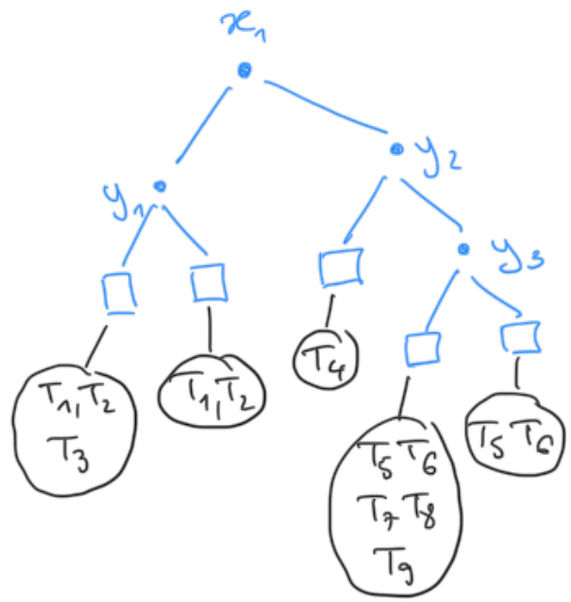
USAGE(S) LARGES DES STRUCTURES ACCÉLÉRATRICES

Exemple :

Etant donné un ensemble de triangles quelconques, déterminer les triangles contenant P ?

- kd-tree ou octree
- Un triangle est inscrit dans toutes les feuilles que sa boîte englobante coupe





→ Delaunay / Voronoi