# La méthode K-d tree

Le kd-Tree, abréviation pour « k-dimensional tree »,est un partitionnement spatial de l’espace à k dimensions permettant de structurer les données en fonction de leur répartition dans l’espace.

Un arbre k-d tree ( k-dimensional tree) est une [structure de données](http://fr.wikipedia.org/wiki/Structure_de_donn%C3%A9es) de partition de l'espace permettant de stocker des [points](http://fr.wikipedia.org/wiki/Point) . Les arbres k-d sont des [arbres binaires](http://fr.wikipedia.org/wiki/Arbre_binaire), dans lesquels chaque nœud contient un point en dimension *k*. Chaque nœud non terminal divise l'espace en deux demi-espaces. Les points situés dans chacun des deux demi-espaces sont stockés dans les branches gauche et droite du nœud courant.

Construction de l’arbre K-d tree :

1-Principe : Il y a plusieurs possibilités de construction d'arbres *k*-d. La construction standard se fait en choisissant la direction de l’hyperplan en fonction de la hauteur du point dans l’arbre ; pour un arbre *3*-d en dimension (k=3), le plan de la racine sera par exemple normal au vecteur (1,0,0), le plan des deux enfants sera normal au vecteur (0,1,0), celui des petits-enfants sera normal au vecteur (0,0,1), puis à nouveau normal au vecteur (1,0,0), et ainsi de suite , le point inséré dans l'arbre à chaque étape est souvent celui qui a la coordonnée [médiane](http://fr.wikipedia.org/wiki/M%C3%A9diane) .

Exemple :

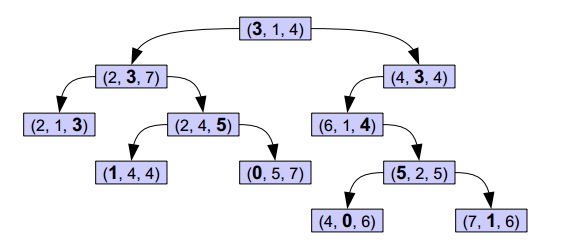


Fig 1 : Illustration de la méthode kd-tree

Dans chaque niveau du kd-tree, un composant de chaque noeud est on gras .pour chaque niveau n de l'arbre, le (n % 3) représente le rang du composant du noeud qui est en gras.La raison que ces valeurs sont en gras est parce que chaque noeud agit d’une manière binaire.  
Par exemple, le premier composant de chaque noeud dans le sous-arbre à gauche est inférieur que le premier composant de la racine de l'arbre, tandis que le premier composantde chaque noeud dans le sous-arbre droite est supérieur ou égal au premier composant du noeud de la racine.  
De même considérez la sous-arbre gauche du kd-tree La racine de cet arbre a la valeur (2, 3, 7), avec trois en gras. Si vous regardez tous les noeuds dans son sous-arbre gauche, vous remarquerez que le deuxième composant a une valeur strictement inférieur que trois. De même dans le sous-arbre droite le deuxième composant de chaque noeud est au moins égal à trois. Cette tendance continue partout dans l'arbre.

2-Structure de l’arbre K-d tree :

La k-d tree est définit par : la racine ,voxel ,des nœuds et des feuilles.

* La racine : la boite englobante toute la scène 3D
* Voxel  :la branche courante de l’arbre binaire à partir de laquelle on cherche à créer deux branches filles dites sous-voxels , au début du traitement le voxel egale la racine ,
* Un noeud : un plan séparateur + deux ﬁls correspondant aux deux sous-voxels
* Une feuille : la liste des objets contenus dans le voxel correspondant

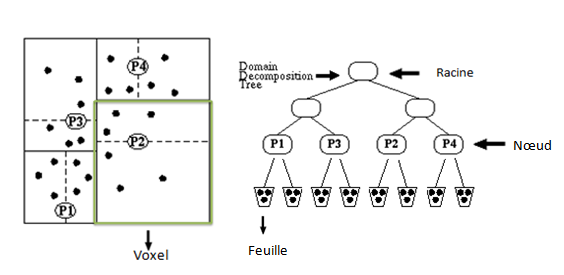


Fig2 : Structure de l’arbre k-d tree

3-Algorithme récursif :

Fonction recConst(voxel V , objets O)

1 -Tester s’il faut diviser le voxel courant V :

* Si non : Rendre la Feuille(O) contenant la liste des objets O
* Si oui : Continuer

2-Trouver le bon plan séparateur p

3-Couper le voxel V avec p pour obtenir VG et VD

4-Repartir les objets O dans VG et VD ⇒ OG et OD

5-Rendre le Noeud( p, recConst(VG,OG), recConst(VD,OD))

fonction constKdTree(Objet O) retourner Racine

V = boite\_Englober(O)

retourner recConst (T,V )

)

Dans l’algorithme précédent, on peut distinguer deux paramètres diffèrents qui influencent directement les performances du kd-Tree :

* la façon de choisir le plan séparateur
* la façon de terminer l’algorithme

3-Le choix du plan séparateur :

Le plan de séparation est perpendiculaire à chacun des axes du repère à tour de rôle, il peut avoir deux positions possibles :

* **Couper le voxel au milieu par le plan médian (Fig2) :** Tout d’abord, pour choisir l’orientation du plan séparateur, la solution la plus simple est de choisir comme axe normal au plan chacun des axes du système de l’espace `a tour de role. Puis, la méthode plus basique pour choisir la position du plan séparateur est de choisir le plan médian qui divise le voxel courant en deux sous-voxels de taille identique
* **Couper le voxel au niveau de l’objet « médian »(Fig3)** : Comme précédemment, l’axe normal au plan séparateur sera choisi `a tour de role en fonction de la profondeur courante dans l’arbre. Par contre, au lieu de couper les voxels au milieu, cette méthode consiste à couper les voxels en fonction de la répartition des objets`a l’intérieur de ces derniers

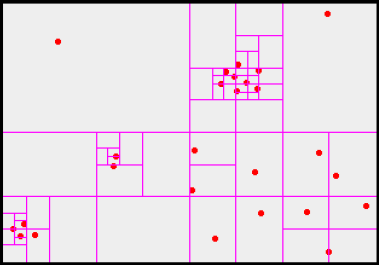


Fig. 2 – Exemple de kd-Tree construit en utilisant des plans médians de l’espace.

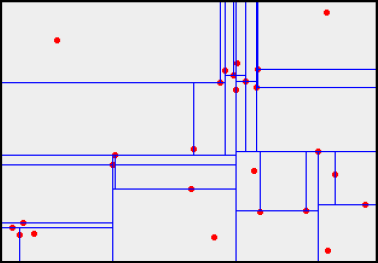


Fig. 3 – Exemple de kd-Tree construit en utilisant des plans positionnés sur l’objet « médian » de chaque voxel.

4-Terminer l’algorithme

Pour terminer simplement l’algorithme, il existe deux solutions qui sont généralement combinées. La subdivision s’arrête lorsque :

– le nombre des objets présents dans le voxel est inférieur à un certain seuil prédéfini KminTri.

– le nombre de subdivision attend un certain seuil prédéfini KmaxProf . Ce nombre correspond aussi au nombre d’itérations maximum de l’algorithme, ainsi qu’`a la profondeur maximum de l’arbre binaire.

L’algorithme de la Fonction «  terminer » qui décrit la fin de l’algorithme :

Terminer(O, V ) ={Nombre des objets (O)< KminTri U Profondeur de la subdivision > KmaxProf }

Contrainte d’utilisation de la K-d tree :

La contrainte sur la sélection du plan médian n'est pas une obligation, mais permet de s'assurer que l'arbre sera équilibré. Le tri des points à chaque étape a un coût en temps, ce qui peut amener à un temps de création de l'arbre assez long.

Applications :

Faire des recherches : [recherche par plage](http://fr.wikipedia.org/wiki/Recherche_par_plage), [plus proche voisin](http://fr.wikipedia.org/wiki/Recherche_des_plus_proches_voisins), etc.

BIBLIOGRAPHIE : Site internet .

Definition de la méthode kd-tree : Wikipédia

Algorithme : <http://www.irisa.fr/prive/kadi/DIIC/SDI-2-annee/Expose_KdTree.pdf> (partie: Construction : algorithme général)

Explication détaillées de la méthode : http://cedric.fleu.free.fr/media/rapportCTR\_cfleury.pdf