Bases de données pour la détection de copies

Laurent Amsaleg

Qui a des documents multimédia ?

- Vous
  - photos et vidéos personnelles
- Les agences de presse
  - AFP, Reuters, …
- L'INA, les radios, les chaînes de télé
- Les sites de partage
  - Flickr, YouTube

Pourquoi détecter des copies ?

- Les agences de photos
  - mettent en ligne des vignettes
  - les clients acquièrent des photos haute définition
  - droits d'auteur = source des revenus

  - prévention difficile ; besoin de détection
    - vérifications manuelles
    - long, pénible

- Peur = pillage des images gratuites
  - pour illustrer un site tiers [et en tirer profit]
  - détourne les clients du site original

Cas réel : pillage de l'AFP

- L'AFP offre un site Web semi-ouvert
  - requiert d'être abonné pour consulter librement les images en basse résolution
  - téléchargements haute résolution facturés
  - un groupe de presse en Asie aspirait les basses résolutions pour alimenter son propre site d'infos en ligne, payant…

Pourquoi détecter les copies ?

- Cas des réseaux pairs à pair :
  - vidéos disponibles illégalement
  - émissions de TV, films, extraits
    - mai 99 : *Star Wars Episode 1*, une semaine après
    - mai 03 : *Hulk*, deux semaines avant

- Vrai pour YouTube
  - camcording
  - mixage de séquences protégées

Taille du problème

- YouTube
  - 65.000 vidéos déposées/j (soit 23.000.000+/an)
- Flickr
  - photo n° 300.000.000 déposée le 18/11
  - photo n° 307.900.221 déposée le 28/11, 15h
- Agences de photos
  - en 2000 : 2,3 milliards de photos gérées (10%)
  - fonds typique : quelques millions

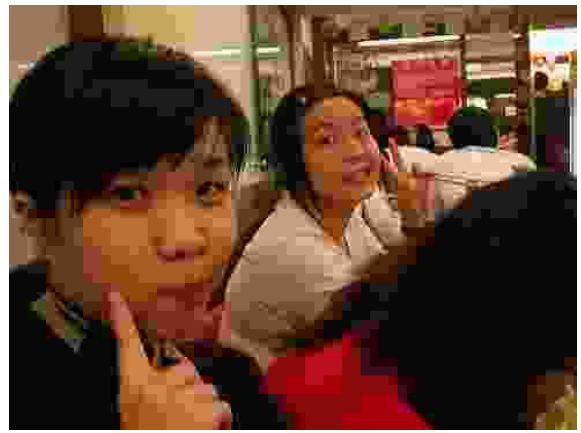
Exemples de difficultés

- ©bubblekiwi, Flickr, 2006:06:16 13:02:13, #173.918.374

Exemples de difficultés

- raisons « éditoriales »

- rotations, recadrages,
- changements couleurs
- compressions
- …



- tout ce que permet Photoshop par exemple

Exemples de difficultés

- raisons « malicieuses »

17 lignes, 5 colonnes

déformation en x et y

Système automatique

- Robuste
  - absorbe les modifications [malicieuses]
- Effective
  - lève peu de fausses alarmes
  - 100% détection pas nécessaire
- Dynamique
  - les fonds croissent
- Efficient
  - petit temps de réponse, fort débit

Tatouage -- Watermarking

- C'est typiquement un problème de tatouage

- Mais
  - aucun schéma de tatouage n'est suffisamment robuste aux retouches
  - on cache assez peu d'informations
    - tant que la marque reste strictement invisible
  - Pour une agence de photos, 2 obstacles majeurs :
    - distribution multiple de la même photo
    - nécessité de tatouer avec de très nombreuses clés

Recherches par le contenu

- Contrer les pirates en utilisant les similitudes visuelles
  - images piratées pour leur contenu
  - celui-ci est donc relativement préservé

- Retrouver l'image originale à partir de la version piratée
  - être robuste aux altérations
  - elles peuvent être très sévères

Conception du système

- 1. Besoin de techniques de description d'images
  - fines, robustes, discriminantes

- 2. Besoin de techniques BD
  - beaucoup d'images à gérer
  - besoin d'être très efficace

1) Aspect image

- Utilisation de descripteurs numériques
- Traduisent des caractéristiques physiques des images *via* l'analyse du signal
  - textures, couleurs, formes
  - descriptions globales, descriptions locales
- Un descripteur définit un point dans un espace multidimensionnel

1) Exemple


quantité de vert
le visage de Shrek
la campagne un jour de beau temps
la mer
quantité de bleu

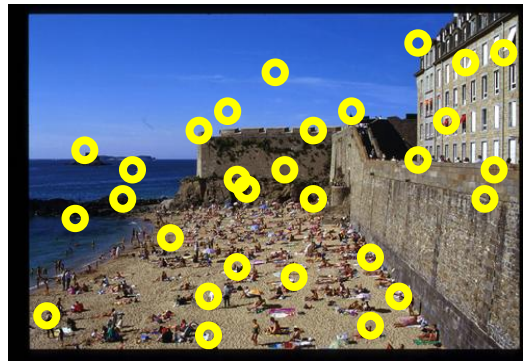
1) Décrire une image fixe

- La similarité entre documents est proportionnelle à la similarité des descripteurs de ces documents

- 2 documents se ressemblent si leurs descripteurs sont proches dans l'espace multidimensionnel

1) Fonctionnement

- Besoin de signatures locales



- Détails pour SIFT
  - vecteurs numériques de dimension 128
  - histogrammes d'orientation
  - ± 700 par image
  - recherches de plus proches voisins
  - métrique = distance euclidienne
  - 1 recherche = ± 700 requêtes, résultat consolidé

2) Aspect bases de données

- Pourquoi utiliser un SGBD ?
  - gestion transparente des disques
  - recherches rapides
  - accès concurrents, partages faciles
  - optimisation, fiabilité, reprise sur pannes, distribution
  - évolutivité, connexions avec le Web

- Oracle, Microsoft SQL Server, …

2) Que stocker dans la BD ?

- Les descripteurs
  - pré-calculés sur le stock d'images
  - dans la BD car utilisés lors de la recherche

- Les images
  - utilisées pour le calcul des descripteurs
  - utilisées pour montrer le résultat
  - éventuellement dans la BD

2) Recherches + SGBD = rapidité

- Vrai pour les données en tables (relations)
  - données typique des applis de gestion
  - recherches ponctuelles
  - recherches par intervalle
  - recherches avec «joker»

  - comparaisons élémentaires

  - tables de hachage, arbres-B, arbres-B+

- Maître mot = indexation

2) Pourquoi indexer ?

- Pour organiser les données
- Pour limiter la quantité de données examinée durant une recherche
- Pour confiner la recherche au maximum
- Permettre un accès rapide aux données
  - super pour la gestion

2) Indexer des descripteurs numériques

- Différences fondamentales :
  - données multidimensionnelles
  - recherches approximatives (plus proches voisins)
  - fonction de distance et non comparaison
  - pas d'ordre total possible

- Conséquences :
  - index traditionnels impossibles à utiliser
  - nécessité d'index multidimensionnels
  - certains SGBD proposent le R-Tree

2) Indexer des descripteurs numériques

- Il n 'existe aucune technique d'indexation multidimensionnelle efficace en grande dimension

- La meilleure approche reste la recherche séquentielle

- Ou bien, il faut viser des résultats approximatifs plutôt que des résultats exacts

2) Recherches approximatives ?

- Répondre plus vite avec une réponse approximative
  - ignorer des données
  - quel gain ?
  - bonne ou mauvaise approximation ?

Les chercheurs agissent là

NV-Tree

- Travaux en collaboration avec

Herwig Lejsek

Friðrik Heiðar Ásmundsson

Björn Þór Jónsson

(Université de Reykjavík, Islande)

NV-Tree

- élections : une liste de N candidats
- chacun ordonne les candidats selon sa préférence
- question : comment trouver le vainqueur ?

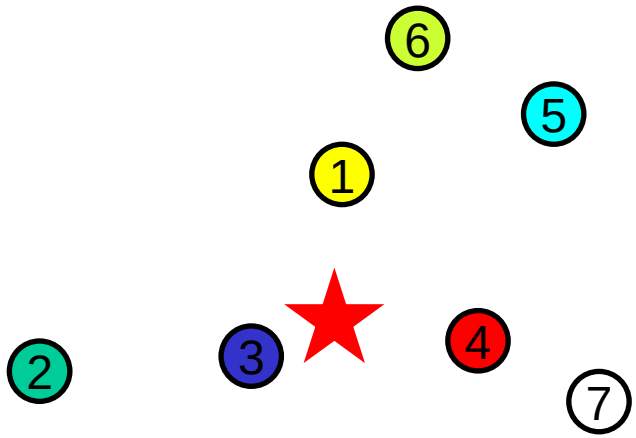
- méthode de Borda : rang moyen des candidats
- méthode de Condorcet :
  - trier toutes les listes par rang croissant
  - parcourir toutes les listes, un rang à chaque fois
  - mémoriser le nom des candidats vus
  - vainqueur : 1er nom vu dans plus de la moitié des listes
  - on continue pour trouver le deuxième vainqueur, …

NV-Tree

- Analogie
  - ordonnancer les descripteurs par proximité au descripteur requête
  - projection des descripteurs sur L lignes aléatoires
    - transformation mono dimensionnelle
    - ordonne les descripteur sur chaque ligne
  - projection du descripteur requête sur ces lignes
    - fixe un point de départ sur chaque ligne
  - application de la méthode de Condorcet
    - 1er descripteur rencontré dans plus de la moitié des listes est le ppv de la requête,
    - on continue pour les autres

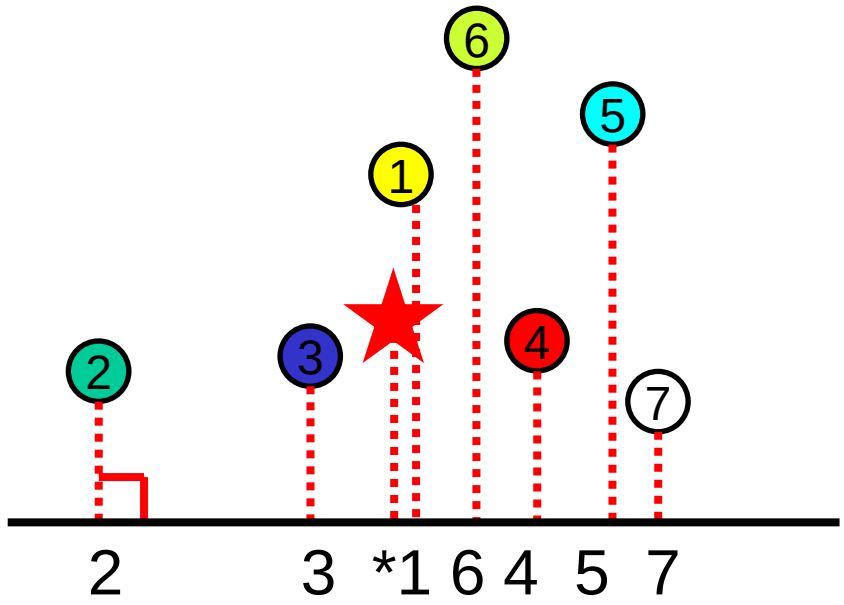
NV-Tree

un ensemble de descripteurs

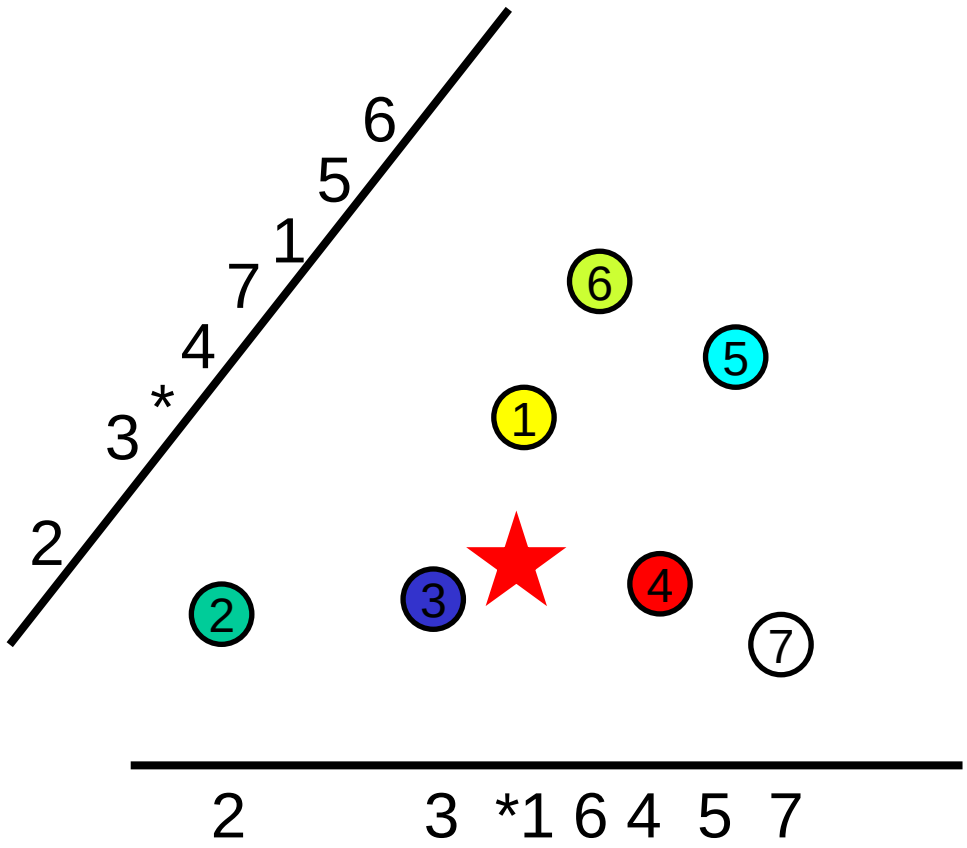


NV-Tree

projection sur une 1ere droite

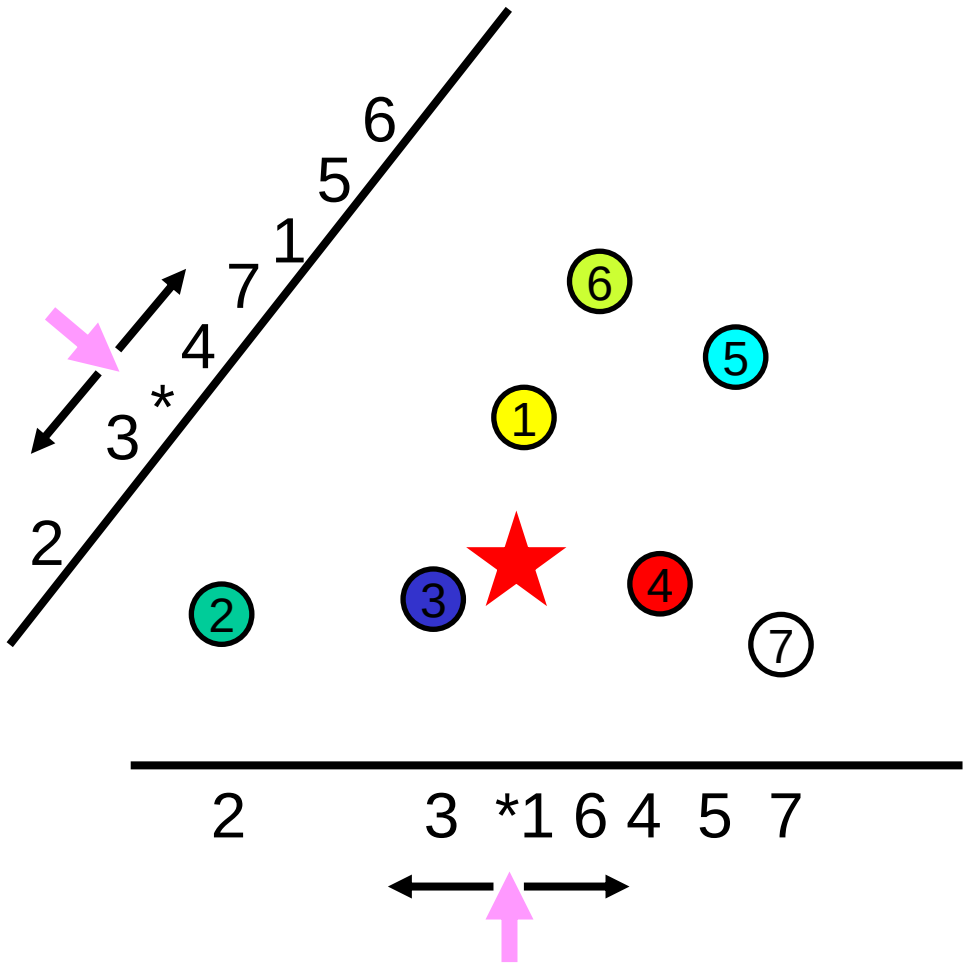


NV-Tree



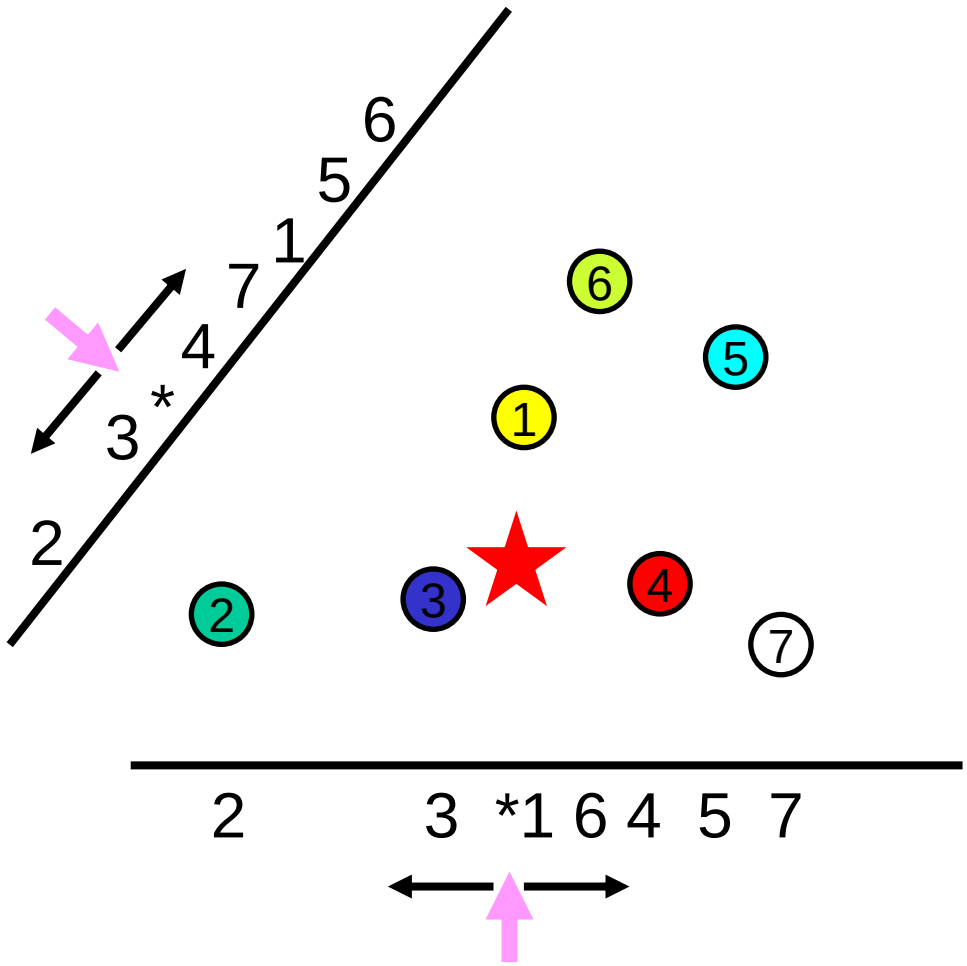
projection sur une 2e droite

NV-Tree



analyse des listes

NV-Tree



analyse des listes

*	*
4	1
3	3
7	6
2	2
1	4

3
2
4

NV-Tree

- Rech. de plus proches voisins très efficace
  - projection sur des droites aléatoires
  - forte réduction de la dimension
  - pas de calculs de distance
  - agrégation de multiples résultats
  - utilise des B+-Tree

  - attention si très grande échelle

Pourquoi faire attention ?

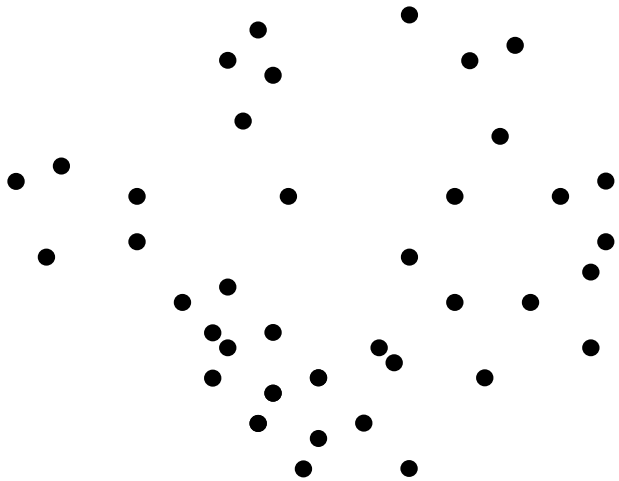
- Les projections écrasent tout :
  - les descripteurs proches dans l'espace ont des projections voisines
  - des descripteurs éloignés dans l'espace peuvent aussi avoir des projections voisines

- Plus il y a de descripteurs, plus souvent cela se passe
  - les « mauvaises » projections éloignent les « bonnes »

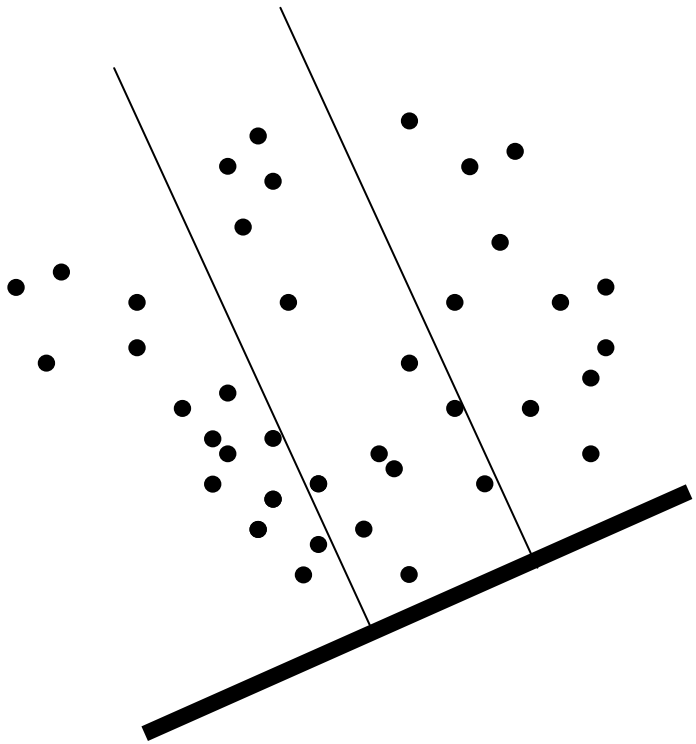
NV-Tree : idée maîtresse

- Ne pas se contenter de lignes aléatoires
- Segmenter les valeurs projetées
  - les valeurs éloignées restent séparées
  - re-projeter les valeurs proches sur une nouvelle ligne aléatoire
    - tente de séparer ce qui semble proche sur une ligne mais est loin en réalité
  - faire cela plein de fois
- Multiplication des points de vue

Exemple



Exemple pour un index



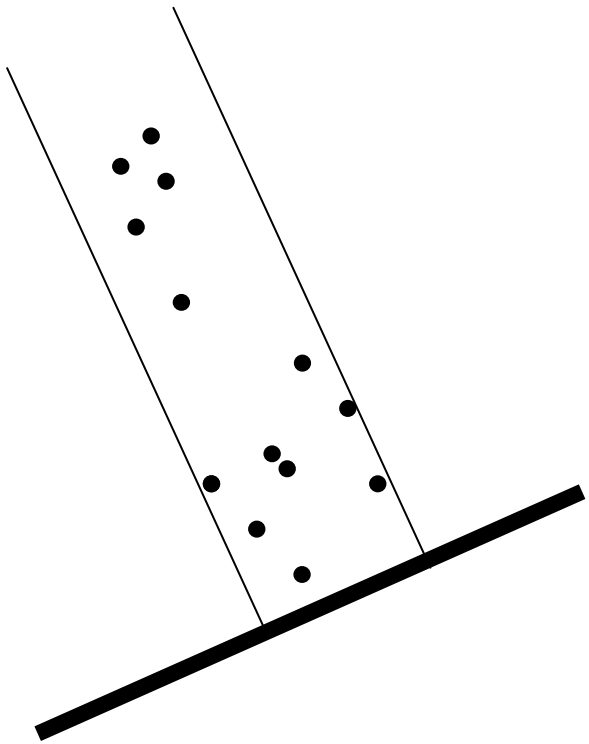
On crée 3 segments ici :

- segments différents ≈ points **plutôt** éloignés dans l'espace d'origine

- même segment ≈ points **peut-être** éloignés

- les points situés près des frontières de segment subissent un traitement spécial…

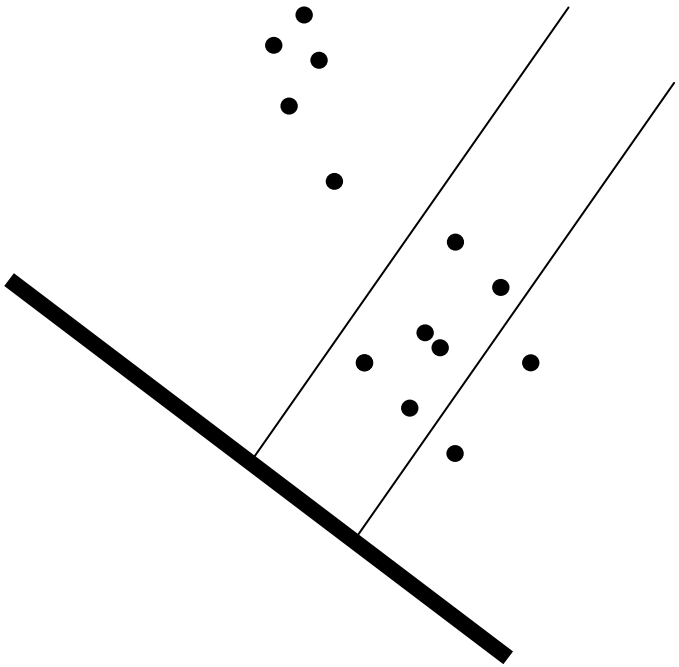
On se focalise sur 1 unique segment



les segments sont considérés comme indépendants

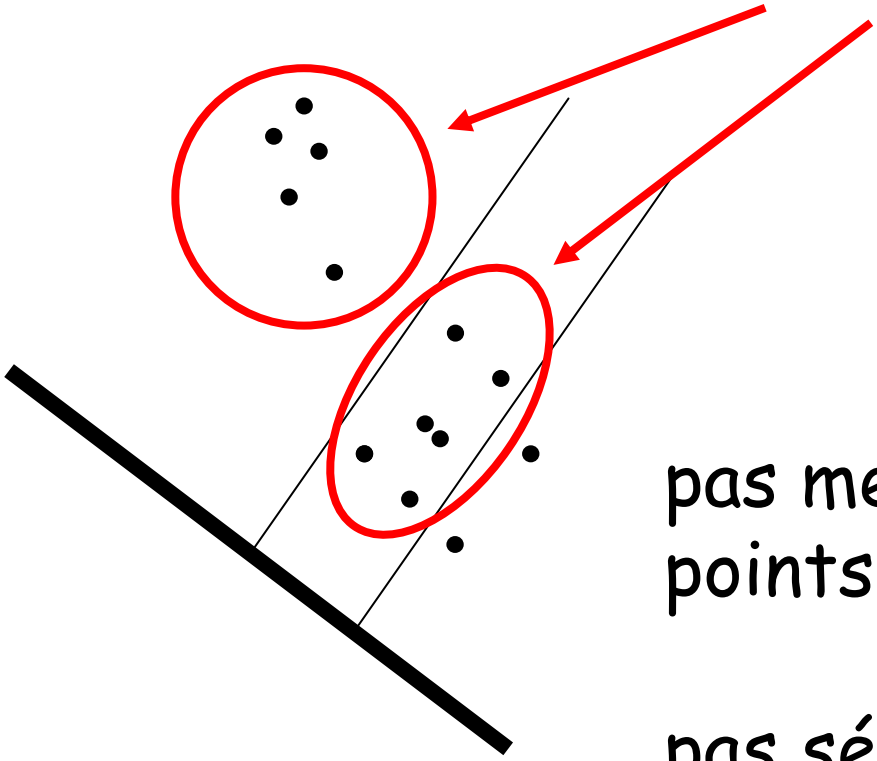
on va re-projeter les points sur une nouvelle ligne

Reprojection des points d'un segment



- On construit une seconde ligne aléatoire
- re-projection des points d'origine
- re-segmentation des valeurs projetées

Reprojection des points d'un segment

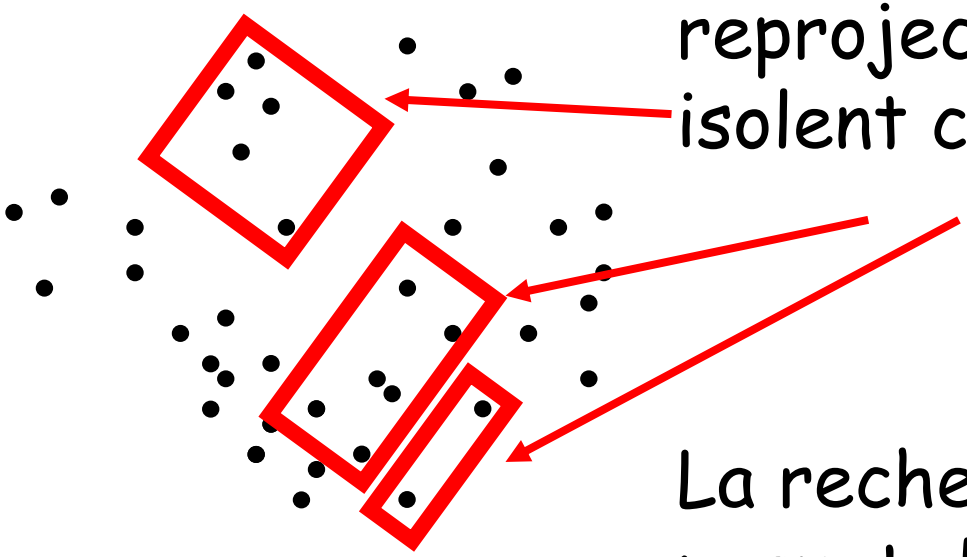


2 ensembles ici séparés

pas mélangés avec les autres points des autres segments

pas séparés si on se limite à une simple projection

Globalement



Segmentations et reprojections multiples isolent ces paquets

La recherche se limite au paquet dans lequel le descripteur requête tombe

NV-Tree : objectif temps de réponse

- hiérarchie de projections / segmentations
- la recherche charge un unique segment par index
- résultats robustes avec 3 index
- temps de réponse = 3 entrées sorties

- temps de réponse fixe
  - 30 descripteurs / sec (±30 milli sec chaque)
  - quelle que soit la taille de la base !

Conclusion (1)

- Échelle :
  - tests réussis avec 300.000 images (200M desc)
    - très bonne reconnaissance,
    - temps de réponses inchangés
  - tests en cours 2M d'images, presque 1,9 milliards de descripteurs de dimension 72 ou 128

- Objectif corollaire : le débit
  - important pour la détection de copies

Conclusion (2)

- Schémas d'indexation multidimensionnelle très performants :
  - NV-Tree, p-Stable LSH, OMEDRANK
  - encore dans le monde de la recherche
  - pas encore dans les SGBD