Département de formation doctorale en informatique

Université de
La Rochelle

École doctorale de La Rochelle

Recherche par similarité statistique dans une grande base de signatures locales pour l'identification rapide d'extraits vidéo

THÈSE

présentée

pour l'obtention du

Doctorat de l'université de La Rochelle

(spécialité informatique)

par

Alexis JOLY

Directeur de thèse:	Carl Frélicot
Rapporteurs :	Nozha BOUJEMAA Patrick GROS
Examinateurs :	Michel SCHOLL Carl FRELICOT Olivier BUISSON Henri SANSON

Institut National de l'Audiovisuel

Laboratoire Informatique, Image, Interaction

Mis en page avec la classe thloria.

Remerciements

Je tiens tout d'abord à remercier mon directeur de thèse Carl Frélicot pour son encadrement scientifique, ses conseils avisés et sa pleine participation à la rédaction des articles. Je lui suis également reconnaissant pour sa disponibilité et pour avoir pris à sa charge certaines démarches administratives auprès de l'école doctorale.

Je remercie sincèrement les membres du jury de ma thèse pour avoir accepté de rapporter et d'examiner mes travaux : Nozha Boujemaa, Patrick Gros, Michel Scholl et Henri Sanson. Leur temps est précieux et je suis honoré qu'ils m'en consacrent une partie.

Je remercie tout particulièrement Olivier Buisson, mon responsable au sein du département de recherche de l'INA, qui m'a accompagné de très près tout au long de ma thèse. La liste de ses contributions à la réussite de mes travaux est immense : les longues heures passées devant le tableau blanc, l'exposé de ses plus précieuses analyses, les discussions quotidiennes, les recherches bibliographiques, la participation au développement des algorithmes, les relectures de mes codes, de mes articles et de ma thèse, la valorisation de mes travaux et j'en passe. Son implication, son dynamisme et son optimisme sont des atouts précieux.

Je remercie toutes les personnes de l'INA qui ont collaboré de près ou de loin à mes travaux. Je remercie particulièrement Jean-Hugues Chenot, responsable du service qui m'a accueilli à l'INA, pour sa grande disponibilité, ses multiples relectures et sa gentillesse. J'exprime également ma gratitude à Louis Laborelli pour ses nombreuses idées et ses orientations bibliographiques. Merci à Nicolas Rasamimanana qui m'a donné les premières clés du monde Linux.

Je salut amicalement toutes les personnes rencontrées à l'INA avec qui j'ai passé des moments agréables (Alain, Guillaume, Frédéric, Jean-François, Rémi, Marie-Luce, Jean-Etienne et tous les autres).

Un immense merci à ma chère et tendre épouse Alexandra sans qui rien ne serait possible.

Je remercie tendrement mes parents et mon frère qui ont éveillé tout au long de ma vie le plaisir de la connaissance et de la découverte.

Je remercie chaleureusement mes ami(e)s pour tous les moments de détente, pour leur intérêt et leurs encouragements. Je remercie particulièrement Thomas Vaillot pour sa relecture de mon manuscrit de thèse.

Je dédie cette thèse
à mon frère Jérémie.

Table des matières

Table des figures

Liste des tableaux

Table des notations

Notations globales

T	Ensemble des transformations de l'image
t	Une transformation de l'image
Id	Identifiant d'une séquence vidéo
tc	Code temporel d'une image à l'intérieur d'une séquence vidéo
D	Dimension de l'espace des signatures
N	Nombre de signatures dans la base
$\mathbf{S}$	Une signature
$d(\boldsymbol{S_1}, \boldsymbol{S_2})$	Mesure de disimilarité entre deux signatures
D'	Dimension intrinsèque des données
ϵ	Rayon d'une requête à ϵ-près
α	Espérance d'une requête statistique
ϵ_α	Rayon d'une requête à ϵ-près d'espérance α
σ_g	Écart-type du modèle de distorsion gaussien
$\boldsymbol{\Delta S}$	Distorsion
ΔS_j	j^{eme} composante de la distorsion
$p_{\boldsymbol{\Delta S}}(\boldsymbol{X})$	Densité de probabilité de la distorsion
$p_{\Delta S_j}(x)$	Densité de probabilité de la j^{eme} composante de la distorsion
$p_{\|\boldsymbol{\Delta S}\|}(r)$	Densité de probabilité de la norme de la distorsion
$\mathcal{N}(\mu, \sigma)$	Loi normale de moyenne μ et d'écart-type σ
$f_{\mathcal{N}(\mu,\sigma)}(x)$	Densité de probabilité d'une variable suivant la loi normale
$\lceil . \rceil$	Arrondi supérieur
$x_{(\eta)}$	x exprimé en base-(η)

Extraction des signatures et fusion des résultats locaux

σ_h	Echelle d'analyse du détecteur de Harris
N_{cand}	Nombre de signatures candidates dans le *buffer* de décision
$\boldsymbol{P}$	Position spatio-temporelle d'un point d'intérêt
$\boldsymbol{S_i}$	i^{eme} signature candidate
$\boldsymbol{P_i}$	Position spatio-temporelle du point d'intérêt de la i^{eme} signature candidate

K_i	Nombre de signatures similaires à la i^{eme} signature candidate
$\boldsymbol{S_{ik}}$	k^{eme} signature similaire de la i^{eme} signature candidate
tc_{ik}	Code temporel de l'image clé contenant $\boldsymbol{S_{ik}}$
Id_{ik}	Identifiant de la séquence vidéo contenant $\boldsymbol{S_{ik}}$
$\boldsymbol{P_{ik}}$	Position spatio-temporelle du point d'intérêt de $\boldsymbol{S_{ik}}$

Algorithmes d'indexation et de recherche

$H_Q^D(\boldsymbol{S})$	Approximation d'ordre Q de la clé de Hilbert binaire d'une signature $\mathbf{S}$
$H_p(\boldsymbol{S})$	Préfixe binaire de Hilbert de profondeur p
p	Profondeur de la partition de l'espace
q	Ordre de la partition de l'espace $q = \lceil \frac{p}{D} \rceil$
b_i	i^{eme} p-bloc de la partition de l'espace
B_α^{min}	Ensemble minimal de p-blocs satisfaisant une requête statistique d'espérance α
Card(.)	Cardinal d'un ensemble (nombre d'éléments)
$B(\tau)$	Ensemble des p-blocs dont la probabilité est supérieure à τ
$P_{sum}(\tau)$	Probabilité cumulée de $B(\tau)$
τ_{log}	Logarithme du seuil de probabilité $\tau_{log} = log_{10}(\tau)$
N_{sig}	Nombre de signatures cumulées en mode par requêtes groupées
n_{pages}	Nombre de pages en mode par requêtes groupées
θ	Profondeur de pages en mode par requêtes groupées
N_{mem}	Nombre de signatures pouvant être logées en mémoire

Modélisation du temps de recherche

$T(p)$	Temps de la recherche à une profondeur p
$T_f(p)$	Temps de l'étape de filtrage à une profondeur p
$T_r(p)$	Temps de l'étape de raffinement à une profondeur p
p_{min}	Profondeur optimale théorique approximée
T_{min}	Temps de recherche optimal : $T_{min} = T(p_{min})$
$n(p)$	Nombre moyen de p-blocs interceptés par la requête statistique
γ_q	Raison logarithmique de la suite $n(p)$ à l'ordre q
t_s	Débit moyen d'un parcours séquentiel de la base (en sec/signature)
t_a	Coût moyen de l'étape de filtrage par p-bloc
$n_q(p)$	Nombre moyen de p-blocs à une profondeur p d'ordre q
$T_q(p)$	Temps de la recherche à une profondeur p d'ordre q
p_q^{min}	Position du minimum global de $T_q(p)$
Π_{q-1}	Raccourci pour $\prod_{r=1}^{q-1} 2^D \gamma_r$
T_{tot}	Temps moyen total de la recherche par requêtes groupées
t_{page}	Coût des calculs répliqués à chaque page
t_{load}	Débit de lecture du disque en unité de temps par signature
T_{load}	Temps de chargement total de la base ($T_{load} = Nt_{load}$)
A_q, A'_q, A''_q, C_q	Constantes à l'ordre q
κ	Constante de réglage du nombre de signatures cumulées

Expérimentations

r_g	Pourcentage de signatures correctement retrouvées
$\overline{\sigma_g}$	Estimation de σ_g sur des distorsions réelles
t_{rn}	Taux de reconnaissance du système
t_{fa}	Fréquence de fausses alarmes
$n_{HS}(\epsilon)$	Nombre de signatures contenues dans une hypersphère de rayon ϵ

Introduction générale

1 Motivations

L'intérêt culturel, historique et commercial d'un fonds d'archives télévisées est indissociable de sa ré-exploitation. L'essor des nouveaux média numériques (câble, satellite, DVD, Internet) a ainsi largement contribué à la valorisation des archives en facilitant leur accès au public. Cette prolifération s'accompagne d'exigences de plus en plus élevées quant à la traçabilité de leur diffusion, tant d'un point de vue juridique et commercial que d'un point de vue essentiel. L'usage qu'est fait d'un document devient ainsi une information à part entière, tout aussi importante que son contenu même.

La conservation et l'exploitation du patrimoine audiovisuel français a été confié à l'Institut National de l'Audiovisuel (INA), le partenaire industriel du contrat CIFRE dans le cadre duquel s'est déroulée cette thèse. Il s'agit d'un Etablissement Public à caractère Industriel et Commercial (EPIC), créé par la réforme de l'audiovisuel menée en 1974 et mis en place le 6 janvier 1975. l'INA détient, à ce titre, l'un des plus important fonds télévisé au monde avec près de 500 000 heures d'extraits vidéo des années 30 à nos jours, dont plus de 200 000 heures ont d'ores et déjà été numérisées.

D'un point de vue juridique, l'intérêt d'une traçabilité de ce fonds pour l'INA s'explique par son obligation de contrôler l'application du droit de l'image avant la diffusion d'une archive. La possession des supports n'implique en aucun cas celle des droits. Chaque document conservé par l'INA fait l'objet d'un dossier mentionnant autant que possible tous les ayant-droits, chacun ayant la faculté de négocier sa part. Cependant, même lorsque le dossier d'origine a été bien mené, tous les cas d'exploitation n'ont pas toujours été prévus, surtout si le document est antérieur aux années 80. De plus, dresser une liste exhaustive des ayant-droits potentiels n'est pas trivial puisqu'ils peuvent être nombreux et difficilement identifiables : ayant-droits d'auteurs, ayant-droits salariés (réalisateurs, interprètes, musiciens) ou bien encore ayant-droits sur des éléments insérés (extraits issus d'autres fonds, photos, œuvres d'art, etc.).

Un autre intérêt de la traçabilité des archives diffusées concerne l'enrichissement informatif du fonds. Le contexte de l'exploitation d'un document (dates de diffusion, chaînes, émission, format, etc.) est en effet une information pertinente en terme d'indexation, qui peut aider à la recherche de documents dans le fonds ou bien encore aux travaux de recherche en sciences humaines qui se basent sur les archives télévisées. Il est par exemple très intéressant de savoir qu'une séquence détectée dans un programme actuel avait déjà été diffusée de nombreuses fois au cours des dernières décénnies.

2 Problématique de la thèse

L'objectif des travaux de cette thèse était de développer un système de détection de copies de séquences vidéo basée sur le contenu, permettant d'effectuer la surveillance temps-réel d'une chaîne de télévision à partir d'un catalogue de référence de grande taille. La détection de copies basée sur le contenu est une alternative à l'approche dite de tatouage, déjà utilisée par de nombreux professionnels pour des applications de protection des droits. Contrairement au tatouage, consistant à insérer une marque invisible mais identifiable à l'intérieur d'une image ou d'une vidéo, l'approche envisagée consiste à comparer uniquement le contenu de l'objet contrôlé avec celui des objets de référence. Pour des raisons de stockage et de coût de calcul, cette comparaison s'effectue par l'intermédiaire de signatures compactes extraites des objets et non sur les objets eux mêmes. Le principe général se déroule en deux étapes. La première consiste à construire une base de signatures à partir de toutes les séquences vidéo du catalogue de référence. La deuxième étape, la plus critique en ce qui concerne les coûts de calcul, consiste à extraire les signatures des objets candidats et à rechercher les signatures similaires dans la base. Comme ces signatures doivent être suffisamment discriminantes pour différencier les objets entre eux, elles sont parfois également appelées *fingerprints* en anglais, par analogie aux empreintes digitales humaines. L'avantage d'une approche basée sur le contenu par rapport au tatouage est triple :

1. **L'antériorité :** Le tatouage ne permet de détecter que des images ou des vidéos ayant été diffusées postérieurement à la mise en place du système. Dans le cas d'un fonds d'archives anciennes comme celui de l'INA, de nombreux documents sont déjà en circulation depuis plus de 30 ans.
2. **Les détections multiples :** Le tatouage ne permet de détecter qu'une version unique d'un même document puisque chaque document du catalogue de référence possède sa propre marque distincte. La détection par le contenu permet de retrouver simultanément toutes les versions d'un même document présentes dans la base. Ces liens peuvent avoir une importance capitale, tant pour la protection des droits que pour d'autres applications.
3. **La robustesse :** Une signature caractérisant le contenu de l'image est intrinsèquement plus robuste qu'un élément invisible rajouté dans l'image puisqu'il n'est pas possible de la supprimer sans modifier le contenu visuel lui-même.

La détection de copies d'extraits vidéo basée sur le contenu fait partie des applications multimédia émergentes pour lesquelles un effort concerté de plusieurs communautés de chercheurs est nécessaire. Depuis plusieurs dizaines d'années, les chercheurs de la communauté de vision par ordinateur et de traitement du signal ont défini de multiples descripteurs basés sur le contenu et des mesures de similarité entre ces descripteurs. D'un autre côté, les chercheurs de la communauté base de données, ont inventé des structures permettant de rechercher rapidement les plus proches voisins d'une requête dans des ensembles de données très volumineux et de grande dimension. Pour relever le challenge d'une application vidéo temps-réel utilisant un très catalogue de référence de très grande taille, il est indispensable de combiner judicieusement les outils issus de ces deux communautés, voire même, d'en dériver de nouveaux plus performants et spécifiques à l'application visée.

La problématique de la thèse soulève deux enjeux majeurs, en un sens contradictoires :

1. **La robustesse de la détection :** Une copie n'est jamais une réplique exacte du contenu numérique du document d'origine. La détection doit être invariante à l'ensemble des transformations qui peuvent être appliquées au document lors du processus de post-production qui précède sa ré-exploitation. Ces transformations, parfois appelées attaques dans le cadre

de la protection des droits, peuvent être sévères comme l'illustre la figure 1. La robustesse de la détection aux transformations, ne doit en revanche pas se faire au détriment de la discriminance puisque l'identification précise du document diffusé est indispensable.

FIG. 1 – Exemple de détection - A gauche : séquence diffusée - A droite : séquence d'origine

2. **Le coût de la détection :** Le processus de détection doit être suffisamment rapide pour traiter continuellement le flux vidéo temps-réel d'une chaîne de télévision. Outre le coût de calcul de l'extraction des signatures, la limite majeure se situe au niveau de la comparaison des signatures candidates avec l'ensemble des signatures de la base. Pour des catalogues de référence très volumineux, pouvant contenir jusqu'à 100 000 heures de vidéo, il n'est pas concevable de comparer séquentiellement chaque signature candidate avec toutes les signatures de la base.

3 Contributions de la thèse

La principale contribution de la thèse est d'avoir proposé et développé une approche globale qui permet de répondre à la fois à la contrainte de robustesse et à celle du coût de la détection, et ce, pour des catalogues de référence très volumineux. Celle-ci est basée sur deux points forts originaux et complémentaires :

1. **signatures locales et cohérence spatio-temporelle :** Les techniques actuelles de détection de copies de séquences vidéo sont basées sur l'utilisation de signatures globales. Un extrait vidéo entier est résumé par une signature unique caractérisant son contenu et la comparaison entre deux séquences s'effectue directement par une mesure de similarité entre les signatures.
   Notre approche est basée sur l'utilisation conjointe de signatures locales et d'une mesure de similarité globale caractérisant la cohérence spatio-temporelle des signatures locales. L'extraction des signatures locales utilise des outils désormais classiques en traitement de l'image et consiste à détecter des points d'intérêt répétables dans le flux vidéo puis à caractériser localement le signal par une signature multidimensionnelle. La comparaison globale entre une séquence candidate et une séquence de référence s'effectue uniquement grâce à la position spatio-temporelle des points d'intérêt. Il s'agit de vérifier que la position relative

des points d'intérêt détectés dans la séquence candidate est cohérente avec celle des points d'intérêt détectés dans la séquence de référence. Cette comparaison n'est pas concevable sur l'ensemble des séquences du catalogue de référence et n'est en fait réalisée que sur les points d'intérêt de la base dont la signature locale est similaire à l'une des signatures locales de la séquence candidate.

2. **recherche approximative des signatures similaires par requête statistique :** La recherche des signatures locales similaires dans la base est l'autre point fort original de notre approche. L'approche classique de recherche de descripteurs similaires dans une base consiste à indexer les descripteurs dans une structure d'indexation multidimensionnelle et à interroger celle-ci par des requêtes géométriques de type K-plus proches voisins ou à ϵ-près. Des travaux récents se sont intéressés au paradigme de recherche de K-plus proches voisins approximative et ont montré qu'une faible perte dans la qualité des résultats de la recherche pouvait permettre des gains de performance importants pour le coût de la recherche.
   Le paradigme de recherche approximative est particulièrement intéressant pour notre schéma global puisque la recherche des signatures locales similaires n'est en quelle que sorte qu'une étape de filtrage précédent la mesure de similarité globale basée sur la cohérence spatio-temporelle des points d'intérêt. Ainsi, la robustesse globale de la détection peut rester inchangée lorsqu'un faible pourcentage de signatures n'ont pas été retrouvées par l'algorithme de recherche approximative.
   En revanche, les requêtes de type K-plus proches voisins ne sont pas adaptées à notre problématique car le nombre de signatures similaires pertinentes peut être très variable d'une signature candidate à une autre, et d'une taille de base à une autre. Notre approche consiste à effectuer la recherche approximative de signatures similaires par un nouveau type de requêtes, dit statistique. Le principe est de retourner les signatures de la base qui sont les plus probables au sens d'un modèle probabiliste caractérisant le vecteur de distorsion entre une signature extraite dans une séquence originale et une signature extraite dans une copie de cette séquence. Ces requêtes statistiques permettent de régler l'espérance du pourcentage de signatures pertinentes qui seront effectivement retrouvées par le système. Elles permettent de s'abstenir de la contrainte géométrique des requêtes classiques et d'obtenir ainsi des gains de performance très significatifs pour le coût de la recherche.

La seconde contribution majeure est d'avoir développé notre propre structure d'indexation pour mettre en œuvre efficacement les requêtes statistiques définies précédemment. Des algorithmes permettant de remplacer les règles de filtrage géométriques habituelles par des règles de filtrage probabilistes ont ainsi été mis en œuvre. Une analyse des travaux existant a également permis de pointer différentes propositions faites par la communauté des bases de données pouvant être mises à profit dans notre structure :

- L'ordonnancement physique et statique des descripteurs de la base permet d'augmenter les performances de recherche d'un système. L'insertion ou la suppression dynamique de signatures dans la base n'étant pas indispensable pour notre application, nous avons appliqué cette proposition en développant une structure basée sur l'ordonnancement physique des signatures selon leur position sur une courbe de Hilbert. Cela permet d'accéder ponctuellement à la base très rapidement par une simple table d'index monodimensionnel.
- Une structure dédiée à un fonctionnement exclusif en mémoire permet des gains de performance importants. Notre structure de recherche fonctionne ainsi en chargeant la base de

signatures en mémoire, soit entièrement, soit par morceaux lorsqu'elle est trop volumineuse. Afin de limiter les sorties des différents niveaux de cache mémoire, un soin particulier a été apporté à la minimisation de la taille des données et des index.
- L'optimisation de la taille des pages de données permet des gains de performance importants. Nous avons développé un modèle de coût du temps moyen d'une requête statistique au sein de notre structure. Celui-ci nous a permis de déduire une valeur optimale théorique pour le principal paramètre de la technique : la profondeur de la partition spatiale sur laquelle se base le filtrage probabiliste. Cette modélisation permet en outre d'avoir une idée du comportement asymptotique de notre système lorsque la taille de la base augmente.

La troisième contribution majeure de la thèse est d'avoir mené des expérimentations de notre système à très grande échelle et dans des conditions réelles, ce qui est rarement le cas dans la littérature. Cela a permis de montrer que l'utilisation d'une décision globale sur un ensemble de signatures locales plutôt qu'une signature globale unique permet d'être très robuste à l'augmentation de la taille de la base. Cela a également montré les interactions existant entre la recherche approximative dans la base et la robustesse globale du système de détection de copies. Enfin, l'expérimentation à grande échelle a pointé des problématiques et des applications nouvelles aux vues de la nature et du nombre de détections obtenues par notre système. Tout cela a contribué à une meilleure connaissance du contenu de la base de l'INA et a mis en évidence la complexité des schémas de ré-exploitation d'un document.

4 Organisation de la thèse

La thèse est organisée en trois parties principales incluant 7 chapitres :

Partie I : La détection de copies à l'aide de signatures locales. Cette première partie traite des aspects plus spécifiques au domaine du traitement de l'image et de la vision par ordinateur. Le premier chapitre est consacré à l'état de l'art de la détection de copies par le contenu et à sa place dans le cadre plus général de la recherche d'images par le contenu et de la reconnaissance des formes. Le deuxième chapitre présente la méthode proposée : la stratégie générale, les signatures locales et la décision globale basée sur la cohérence spatio-temporelle des signatures locales.

Partie II : Recherche des signatures similaires dans la base. La deuxième partie s'intéresse aux aspects plus spécifiques au domaine des bases de données. Le premier chapitre dresse un état de l'art des techniques de recherche de descripteurs similaires dans des bases de grande taille. Le deuxième chapitre présente la méthode proposée pour la recherche de signatures par similarité, en introduisant d'abord le concept de requêtes statistiques puis en décrivant la structure proposée pour leur mise en œuvre. Un troisième chapitre présente les expérimentations de notre technique relatives aux coûts d'une recherche dans la base indépendamment de la robustesse globale du système de détection.

Partie III : Expérimentations et Applications. La dernière partie présente l'évaluation et l'analyse de notre système complet de détection de copies vidéo par le contenu. Le premier chapitre décrit l'ensemble des expérimentations menées pour évaluer séparément et conjointement les différentes parties du système. Le deuxième chapitre présente l'application principale, à savoir le monitorage d'une chaîne de télévision mais également deux autres applications déve-

loppées à partir de notre technique : la détection de rediffusions d'extraits vidéo intra-chaîne et inter-chaîne et la détection de liens à l'intérieur d'un catalogue d'archives.

Premiere partie

La détection de copies à l'aide de signatures locales

Chapitre 1

État de l'art de la détection de copies par le contenu

1.1 Introduction

La détection de copies d'images et de vidéos par le contenu est un cas particulier d'un domaine beaucoup plus vaste qu'est celui de la recherche d'images et de vidéos par le contenu. Ce domaine couvre des travaux très variés allant de l'extraction de caractéristiques bas niveau du signal à la présentation des résultats à un utilisateur en passant par la navigation dans les bases. Si de nombreux thèmes de recherche sont potentiellement applicables à la problématique de la détection de copies par le contenu, il est important de comprendre les spécificités de cette dernière et pourquoi la plupart des systèmes de recherche par le contenu ne sont pas directement exploitables. Ce chapitre a pour objectif de situer la détection de copies par rapport aux autres problématiques du domaine et de dresser un état de l'art des travaux s'y rapportant. Après avoir spécifié quelques définitions pour préciser ce qu'est une copie (section 1.2), nous dresserons un panorama général de la recherche d'images et de vidéos par le contenu (section 1.3), avant de nous intéresser plus spécifiquement à la détection de copies par le contenu et à ses spécificités (section 1.4).

1.2 Similarité, copie et authenticité

La notion de traçabilité d'un document audiovisuel implique de définir certains concepts sous-jacents traduisant ce que l'on considère être deux versions d'un même document audiovisuel. Nous désignerons par la suite ce lien par le terme de *copie* et nous allons tenter ici de le définir ou du moins de le restreindre. Le cas le plus simple est celui où deux documents numérisés possèdent exactement le même contenu numérique, on dira alors qu'il s'agit d'une copie exacte ou que les deux documents sont identiques. Dans le cas plus général, la définition d'une copie est plus problématique puisque les objets audiovisuels subissent toutes sortes de transformations : compression, diffusion, remontage, etc. Il est cependant nécessaire de différencier la notion de copie de la notion de similarité. En effet, de nombreuses applications en traitement de l'image, s'attachent à retrouver des images similaires. Cette similarité peut être de très haut niveau, de nature sémantique (par exemple deux images différentes représentant une voiture), ou de plus bas niveau, plus proche du signal (par exemple deux images ayant des couleurs très proches). La notion de copie est un cas particulier de similarité mais il est clair que celle-ci est plus contraignante. Nous la différencierons donc par la notion de transformations :

Définition Copie - *un objet O_2 est la **copie** d'un objet O_1, si $O_2 = t(O_1), t \in T$ où T est un ensemble de transformations. On dira que O_1 est un original de O_2.*

Par objet, nous entendons un objet audiovisuel sous forme numérique ou analogique. Par transformation, nous entendons tout traitement analogique ou numérique effectivement appliqué et non pas une transformation mathématique théorique. On peut ainsi interpréter cette définition chronologiquement, puisqu'une copie est forcément postérieure à son original. Ainsi deux prises de vue simultanées d'une même scène réelle avec deux caméras différentes ne sont pas des copies l'une de l'autre. En revanche, une vidéo acquise dans une salle de cinéma avec un caméscope est une copie du film diffusé. La définition de l'ensemble des transformations T dépend de l'application et n'est en général pas complètement définissable. Ainsi pour une application de protection des droits, la définition juridique du droit de l'image est très floue et les décisions se prennent au cas par cas. Les figures 1.1 et 1.2 illustrent l'ambiguïté que peuvent soulever certains cas. La définition de l'ensemble T nous ramène donc à une notion de similarité, celle de savoir si l'objet est encore authentifiable par un expert.
La traçabilité d'un catalogue d'objets audiovisuels consiste à détecter la diffusion d'une copie ou d'un original d'un ou plusieurs objets du catalogue. En pratique, il s'agit d'un problème d'invariance ou du moins de robustesse à un ensemble de transformations identifiées comme les plus pertinentes pour l'application visée. La restriction de cet ensemble est très importante car un gain en robustesse se traduit généralement par une perte de discriminance et l'on risque de confondre des objets différents si l'on se rend robuste à des transformations trop sévères.

Il est à noter que les approches par tatouage du contenu, ne permettent de détecter que les copies de l'objet tatoué postérieures au tatouage. L'approche par identification du contenu permet pour sa part de détecter toutes les copies de l'objet signé (postérieures ou antérieures au calcul des signatures) et également d'éventuels originaux de l'objet signé. En revanche, contrairement au tatouage, elle ne permet pas d'authentifier formellement l'objet audiovisuel reconnu, tâche qui doit être réalisée ultérieurement par un expert.

1.3 La recherche d'images et de vidéos par le contenu

1.3.1 Vue d'ensemble

Avec l'essor des réseaux de transmission tel qu'Internet ou la télévision numérique, et la démocratisation de l'imagerie numérique tant au niveau des professionnels que du grand public, le problème de la recherche de documents numériques dans des bases très volumineuses, est devenu un enjeu de recherche majeur au cours des deux dernières décennies. L'approche encore majoritairement utilisée aujourd'hui à l'INA comme dans la plupart des grands centres d'archivage, est l'annotation manuelle des documents par des méta-données. Anciennement stockées sous forme de notice papier et aujourd'hui intégrées dans des bases de données informatisées, ces méta-données contiennent d'une part des informations de contexte telles que le titre, l'auteur ou les conditions d'acquisition et d'autres part des informations sémantiques relatives au contenu du document. Le travail humain colossal que représente l'annotation manuelle d'une base multimédia a rapidement conduit au questionnement de l'automatisation de l'indexation. Ainsi, la recherche d'images par le contenu est apparue au cours des années 90 comme une alternative, ou du moins une approche complémentaire à l'annotation manuelle. Elle consiste à caractériser automatiquement le contenu visuel des images (ou des vidéos) et à effectuer les recherches par similarité de ces caractéristiques. En pratique, des algorithmes s'attachent à extraire des descripteurs de l'image, souvent sous forme de vecteurs multidimensionnels, et la similarité visuelle est

FIG. 1.1 – Illustration de l'ambiguïté du concept de copie - les deux images en haut à droite et les trois images du bas, ont été obtenues à partir de l'image en haut à gauche, par des effets du logiciel Microsoft Photo Editor

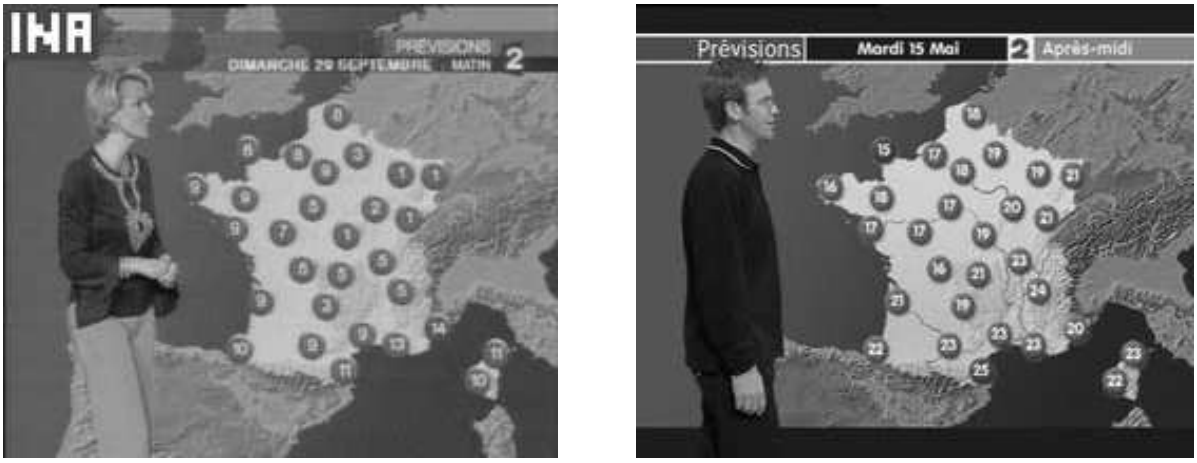


FIG. 1.2 – Ces deux images représentent deux scènes différentes mais ce sont deux copies de la carte d'Europe utilisée en image de fond

ramenée à une mesure de similarité entre ces descripteurs. Il est important de noter que d'une manière générale les annotations manuelles et les descripteurs visuels ne travaillent pas au même *niveau d'indexation*. Les annotations se situent au niveau *sémantique* et s'attachent à décrire ce qui a trait à l'interprétation de l'image ou de la vidéo, tandis que la recherche par contenu visuel se situe au niveau *numérique* faisant référence aux caractéristiques primaires ou bas-niveau telles que la couleur, la forme, le mouvement, etc. Après quelques généralités à propos du contenu audio 1.3.2, le paragraphe 1.3.3 dresse un bref panorama du domaine. Le paragraphe 1.3.4 s'intéresse ensuite aux relations entre la problématique de la recherche par le contenu et le domaine de la reconnaissance d'objets.

1.3.2 A propos du contenu audio

De nombreux travaux de recherche ont été menés dans le domaine de la recherche de documents audiovisuels pour le son, l'image et la vidéo. Bien que la vidéo soit un document audiovisuel nous ne nous intéressons pas au signal sonore qui sort du cadre de cette thèse. La première raison est que de nombreux extraits vidéo sont réutilisés avec une bande sonore différente, en particulier à la télévision. La deuxième raison est qu'il s'agit de deux domaines distincts très complets qui nécessitent des études à part entière. Une bonne revue des travaux sur la recherche de documents audio par le contenu est fournie dans [54]. Pour une revue concernant plus spécifiquement la détection de copies audio par le contenu, on pourra se référer à [37]. Il est enfin à noter que des approches multimodales ont été proposées ; elles permettent des gains significatifs en terme de qualité de recherche à condition que l'intégrité entre le son et l'image soit conservée. On pourra se référer à [159] pour plus d'informations.

1.3.3 Panorama de la recherche d'images et de vidéos par le contenu

Bien que nos travaux concernent la vidéo, il n'est pas possible d'ignorer les travaux relatifs aux images fixes puisqu'une vidéo peut être vue comme une succession d'images fixes ; de nombreuses approches pour la vidéo sont ainsi des traitements classiques d'images fixes appliquées à certaines images clé de la vidéo. Certaines approches sont en revanche spécifiques au signal vidéo. Il n'est pas question ici de donner une revue complète des travaux sur la recherche d'images et de vidéo par le contenu. Il existe une littérature très abondante depuis le début des années 90 ayant conduit à une grande quantité de systèmes industriels ou expérimentaux. Il existe de plus des états de l'art très complets dont les quatre suivant comptent parmi les plus cités [60, 142, 157, 166], une revue spécifique à la vidéo est réalisée dans [33]. Dans la suite, nous nous efforçons de donner un panorama général du domaine en rappelant le principe et les applications d'un système de recherche d'images par le contenu, ainsi que les alternatives des principales étapes. Cette section 1.3.3 est en partie un résumé des travaux de J. Fauqueur [68] et de J. Fournier [74]. Certaines phrases ayant été extraites de leurs travaux, nous ouvrons doublement les guillemets pour le contenu des pages 13 à 17.

Après avoir décrit le principe général et les domaines applicatifs de la recherche d'images et de vidéo par le contenu, nous aborderons plus précisément trois aspects du problème. Nous nous intéresserons d'abord aux différents paradigmes de recherche proposés à l'utilisateur. Nous focaliserons ensuite sur les principaux attributs utilisés lors de l'extraction de caractéristiques. Nous nous poserons ensuite la question de la structuration de cette information sous forme de descripteurs, c'est à dire la représentation de l'attribut en machine. Enfin, nous étudierons les mesures de similarité employées pour comparer ces descripteurs.

Principe et applications de la recherche d'images et de vidéos par le contenu

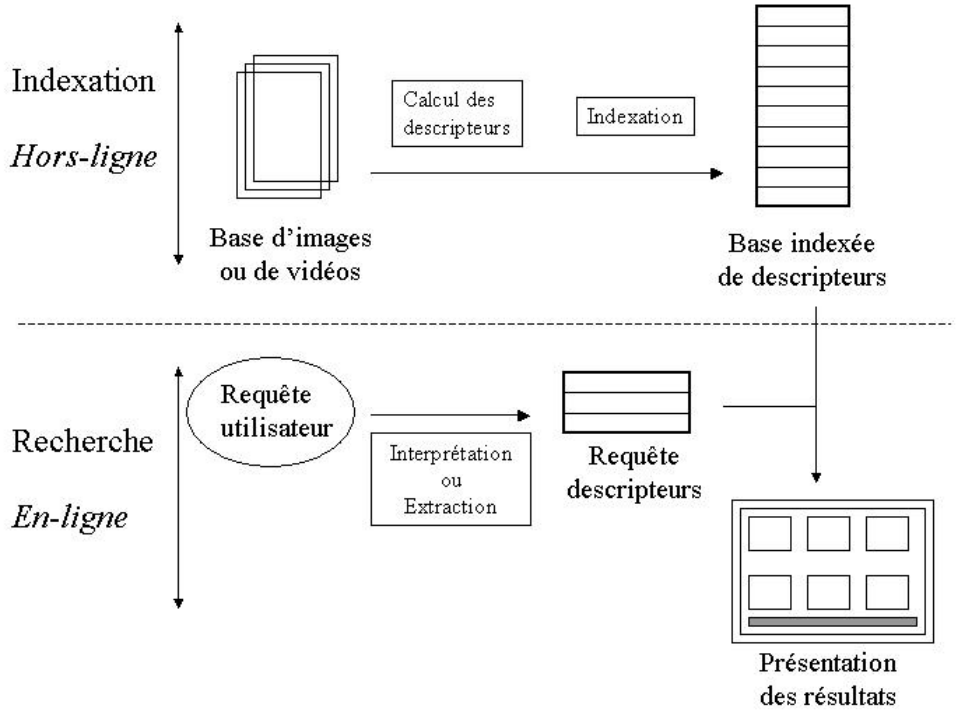


FIG. 1.3 – Principe général d'un système de recherche par le contenu

D'une manière générale, un système de recherche d'images ou de vidéos par le contenu peut être décrit par la figure 1.3. On distingue toujours une phase *hors-ligne* d'indexation et une phase *en-ligne* de recherche d'images ou de vidéo. La phase hors ligne consiste à extraire les caractéristiques des images ou des vidéos du catalogue de référence et à construire la base des descripteurs. La phase *en-ligne* consiste à traduire une requête utilisateur en une recherche dans la base des descripteurs. Pour cette étape, différents paradigmes de recherche sont envisageables et sont discutés dans la suite de cette section.

Les applications de la recherche d'images et de vidéos par le contenu sont très variées. Outre la détection de copies, on peut citer la recherche de documents sur Internet, le commerce électronique (catalogue numérique), l'audiovisuel (recherche de personnages dans les journaux télévisés par exemple), la médecine (aide au diagnostic), la biométrie (empreintes digitales, etc.). On peut classer les applications selon le type de la base de données, qui peut être de nature spécialisée (base de visages, d'empreintes digitales, de tableaux, etc.) ou de nature généraliste (contenu hétérogène).

Les paradigmes de recherche

1. **L'image ou la vidéo exemple :** Le scénario de recherche d'images par le contenu le plus élémentaire, et le premier historiquement, est celui de recherche globale par l'exemple. L'utilisateur choisit une image ou une vidéo exemple et le système sélectionne les éléments de la base de référence dont l'apparence visuelle globale est la plus similaire. Le cœur de l'approche réside sur une description visuelle de chaque image ou de chaque vidéo et sur

une mesure de similarité adéquate pour le descripteur. Le principe de cette approche a été proposé en 1991 par Swain et Ballard [162]. Il s'agit du principe fondateur de nombreux travaux et de systèmes tels que QBIC [132], PhotoBook [135], Ikona [28] pour ne citer qu'eux. Dans le cadre de la recherche globale, de nombreux descripteurs d'images et mesures de similarité ont été proposés afin de caractériser les informations de couleur, texture, forme et mouvement pour la vidéo uniquement.

2. **Requêtes partielles :** La limitation des méthodes de recherche et de navigation exploitant une description visuelle globale est l'hypothèse implicite que l'intégralité du contenu du document est pertinente pour les besoins de l'utilisateur. L'image ou la vidéo est considérée comme une entité visuelle atomique, alors que l'humain perçoit généralement une image ou une vidéo comme une entité composite d'objets. L'intérêt de l'utilisateur peut porter sur une ou plusieurs composantes ou bien il peut souhaiter ignorer le fonds d'une image ou d'une vidéo qu'il ne jugera pas pertinent pour sa requête. Les requêtes partielles permettent à l'utilisateur de désigner explicitement les composantes pertinentes dans les images ou les vidéos et d'effectuer la recherche à partir de celles-ci. Dans la littérature, l'approche la plus développée correspond au paradigme de recherche par région-exemple. L'utilisateur sélectionne une région exemple dans une image ou une vidéo et le système retrouve les images comportant une région visuellement similaire. L'information temporelle des vidéos rajoute une dimension supplémentaire dans la sélection de requêtes partielles. L'approche classique consiste à partitionner temporellement la vidéo en éléments de base. On distingue généralement quatre niveaux de granularité pour le partitionnement temporel d'une vidéo [115] :
   - Les images : chaque image est traitée indépendamment.
   - Les plans : il s'agit d'un ensemble d'images acquises lors d'une même prise de vue. La détection des plans est fondée sur la détection temporelle des changements brusques du signal et non sur une cohérence sémantique des images d'un même plan.
   - Les scènes : une scène est un ensemble de plans successifs ayant une signification sémantique commune.
   - Le programme vidéo : il s'agit de l'objet vidéo lui-même (film, émission, publicité, etc.).
3. **Le bouclage de pertinence :** D'un point de vue de l'interaction utilisateur, le bouclage de pertinence, issu des techniques de recherche de texte, a été appliqué avec succès à la recherche d'images et de vidéos [143]. Il permet à l'utilisateur de raffiner sa recherche en indiquant itérativement si les résultats obtenus sont pertinents ou non-pertinents pour sa recherche.
4. **La navigation :** La navigation dans des bases d'images et de vidéos offre une approche complémentaire au scénario de recherche : lorsque l'utilisateur n'a pas vraiment une idée claire de ce qu'il recherche, les techniques de navigation permettent d'avoir un aperçu global de la base puis de raffiner la visualisation à des sous-ensembles qui lui semblent plus pertinents. Les systèmes de recherche par navigation sont généralement basés sur une structure de graphe calculée a priori pour un ensemble donné d'images. Ce graphe rapprochant les images par similarité sémantique, sa représentation hypermédia est directement exploitable par l'utilisateur final [124, 164].
5. **Les ébauches graphiques :** Notons également que certains paradigmes de recherche ne nécessitent pas d'images-exemple ou de régions-exemple. Les images ou les vidéos de la base sont encore indexées à l'aide de descripteurs du contenu mais les requêtes de l'utilisateur sont directement traduites dans l'espace des descripteurs sans passage par les vidéos, les images ou les régions exemples. On peut employer des représentations graphiques de l'idée

mentale de l'utilisateur via une interface d'ébauche graphique, par exemple. La requête peut également se faire par l'ajustement de paramètres globaux simplifiés de couleur, de forme, de texture ou de mouvement [152, 45].

Les attributs

Le but de l'indexation est de fournir une représentation du contenu de l'image ou de la vidéo permettant des recherches efficaces. Il ne s'agit pas de coder toute l'information de l'image ou de la vidéo comme c'est le cas lorsque l'on veut compresser les images, mais de se concentrer sur l'information qui permet de traduire efficacement une similarité pertinente pour le paradigme de recherche.

L'analyse faite du signal se focalise généralement autour de caractéristiques simples, intuitives et génériques. Nous revenons ici sur les attributs les plus couramment utilisés :

1. **Caractéristiques de couleur :** La couleur, de par son pouvoir discriminant et sa forte valeur sémantique, est un des attributs les plus employés en indexation d'images et de vidéos. Les auteurs se concentrent autour de quelques grands thèmes tels que l'identification de l'espace couleur le plus discriminant [161], l'étude des problèmes d'invariance aux conditions d'illumination et de prise de vue [109] ainsi que la combinaison avec des attributs complémentaires tels que la texture [155].
2. **Caractéristiques de texture :** Il n'existe pas de définition formelle de la texture, elle est cependant généralement modélisée comme une structure spatiale constituée de l'organisation de primitives ayant chacune un aspect aléatoire. Une texture peut avoir un aspect périodique ou bien aléatoire. La prise en compte de ce type de caractéristiques pour représenter globalement ou partiellement une image est courante et discriminante dans de nombreux cas. Parmi les méthodes les plus connues, on trouve la décomposition paramétrique de Wold reprise notamment dans le système Photobook [135]. Les modèles autorégressifs multi-échelle MRSAR[1], les filtres de Gabor ou encore les décompositions en ondelettes sont également très utilisés.
3. **Caractéristiques de forme :** Au même titre que les caractéristiques de texture, les attributs de forme sont complémentaires de la description couleur. Deux catégories de descripteurs de formes se distinguent : les descripteurs de type région et les descripteurs de type frontière. Les premiers font classiquement référence aux moments invariants et sont utilisés pour caractériser l'intégralité de la forme d'une région. Ces attributs sont robustes aux transformations géométriques comme la translation, la rotation et le changement d'échelle [72]. La seconde approche fait classiquement référence aux descripteurs de Fourier [144] et porte sur une caractérisation des contours de la forme.
4. **Caractéristiques de mouvement :** Contrairement aux trois attributs cités précédemment qui peuvent également être utilisés pour des documents vidéo (soit en les appliquant aux images constituant la vidéo, soit en les étendant à un cadre spatio-temporel [175]), l'attribut de mouvement est spécifique à la vidéo. Son analyse consiste généralement à apparier des composantes d'une image à un instant donné avec les composantes des images suivantes ou précédentes. L'écart de position spatiale représente un vecteur mouvement caractérisant la direction et la vitesse du mouvement. On distingue généralement le mouvement global ou dominant (dans de nombreux cas il s'agit du mouvement de la caméra) du mouvement des objets constituant la scène. L'approche classique pour extraire les mouvements des objets, consiste à compenser le mouvement global dans une première étape

[1] MultiResolution Simultaneous AutoRegressive

[62, 8]. Notons également que l'utilisation des formats de compression de type MPEG[2], basés sur l'analyse du mouvement, permet d'accéder directement à une information de mouvement exploitable [127]. Le mouvement peut être utilisé de manière combiné avec d'autres attributs comme par exemple dans [58], où les coefficients continus du signal sont utilisés en plus de l'information de mouvement.

Localisation de zones d'intérêt

Les approches faisant appel à la localisation de zones d'intérêt sont principalement développées dans le cadre du paradigme de requêtes partielles mais peuvent également constituer une étape intermédiaire dans le cadre d'une requête globale. L'idée est encore une fois de ne pas coder toute l'information portée par l'image ou la vidéo mais de sélectionner des composantes visuelles susceptibles de constituer des supports de requête pertinents. Elles peuvent être de natures diverses : zones saillantes, régions uniformes, détails plus ou moins précis, objets spatio-temporels. Outre le détourage manuel de régions utilisé dans certains systèmes, nous distinguons les zones d'intérêt suivantes :

1. **Les blocs :** La subdivision systématique d'images ou de vidéos en blocs ou en intervalles temporels réguliers présente l'avantage d'être simple et rapide [133, 122, 129] mais elle est très approximative. Les descripteurs sont en effet calculés sur les cellules d'un quadrillage constant (spatial, temporel ou spatio-temporel). Un des principaux inconvénients est que la localisation est indépendante du contenu et que des problèmes de non invariance aux phénomènes de translation apparaissent systématiquement. De plus, cette subdivision est tributaire de l'échelle de la grille choisie et ne s'adapte pas au contenu informatif local. Les cellules sont par exemple inutilement trop nombreuses dans les régions uniformes et pas assez nombreuses dans les régions à fort contenu informatif.
2. **Les points d'intérêt :** Initialement développés dans un contexte de stéréovision, les points d'intérêt dans une image correspondent aux lieux de hautes fréquences photométriques dans plusieurs directions. Ils ont été employés pour des problèmes de mise en correspondance précise de lieux entre deux images correspondant à des vues différentes d'une même scène. Les propriétés de stabilité et de répétabilité de leur détection ont motivé leur application à la recherche d'images par le contenu. Ils ont été initialement appliqués à la recherche globale d'images en niveaux de gris par Schmid et al. [150]. Les points d'intérêt couleur [130] ont ensuite été utilisés pour une approche de recherche par parties d'images [81]. La caractérisation de composantes d'images par points est efficace pour retrouver avec précision des détails fins de l'image. Dans le contexte général de la recherche d'images par le contenu, ils présentent l'inconvénient de ne pas caractériser les zones lisses ou uniformes et d'engendrer une grande quantité de points dans les régions texturées. Nous verrons par la suite que dans le contexte plus spécifique de la détection de copies leurs propriétés de répétabilité et de localité sont en revanche des atouts majeurs.
   La détection d'images clé dans une vidéo, peut également être considérée comme une détection de points d'intérêt dans l'espace temporel unidimensionnel. Une image clé correspond à une propriété particulière de l'évolution temporelle de la vidéo comme un maximum d'activité, un changement brusque du mouvement de la caméra ou encore un changement de plan. Récemment le concept de points d'intérêt a été étendu aux points d'intérêts spatio-temporels [111]. Il s'agit de points remarquables à la fois dans l'espace et dans le temps, comme par exemple l'instant où le talon de la chaussure d'un marcheur rencontre le sol. La détection

[2] Motion Pictures Expert Group

d'images clé et la détection de points d'intérêt spatio-temporels possèdent les mêmes propriétés que les points d'intérêt spatiaux, à savoir la répétabilité fine de la localisation et la localité.

3. **Les régions :** L'approche par segmentation de régions est la plus utilisée pour le paradigme de requêtes partielles. La segmentation des images est une question centrale en traitement de l'image. De nombreux auteurs se sont focalisés sur cet aspect et des algorithmes de segmentation originaux ont été proposés [173, 134] ainsi que des techniques complètes pour l'indexation et la recherche par le contenu [39]. Le principe est de délimiter des régions dans lesquelles il y a homogénéité de certaines caractéristiques (couleur, texture, mouvement, etc.). L'avantage pour la recherche par le contenu est que le partitionnement en régions se rapproche plus d'une détection d'objets de nature sémantique. La détection de régions spatio-temporelles dans la vidéo permet par exemple d'isoler un personnage du décor. Notons que le partitionnement temporel de la vidéo aux différents niveaux de granularité, dont nous avons parlé précédemment, constitue également une détection de régions d'intérêt dans le domaine temporel.

Les descripteurs

Dans le cas d'une description globale sans étape de détection de zones d'intérêt, la forme de descripteur historiquement la plus utilisée en indexation est l'histogramme [68, 74], notamment parce qu'il convient bien à la structuration des attributs de couleur, les plus utilisés, comme nous l'avons déjà dit. A un facteur près, l'histogramme représente une estimation de la distribution de l'attribut dans l'image ou la vidéo. Il constitue une description globale qui a l'avantage d'être simple à utiliser et à calculer tout en conférant une relative robustesse à de multiples transformations comme le changement d'échelle, la rotation, les occlusions. L'inconvénient majeur est la perte de l'information spatiale, caractéristique essentielle pour la discriminance. L'exemple d'images très différentes ayant le même histogramme pour certaines caractéristiques est fréquemment mis en avant pour illustrer cette perte d'information. Les histogrammes sont cependant encore souvent utilisés comme par exemple dans [44], où Cheung propose une méthode de recherche rapide de vidéos, fondée sur un histogramme couleur dans chacun des 4 quadrants de chaque image (une signature de dimension 168 par image).
La liste des autres descripteurs utilisés dans la littérature est très variée et très abondante et nous laissons le lecteur se référer aux états de l'art précédemment cités pour plus de détails [60, 142, 157, 166]. Il peut s'agir des coefficients d'une projection de l'image dans d'autres espaces (transformée de Fourier, de Radon, de Kerhumen-Loëve, etc.), ou encore des coefficients d'une décomposition en ondelettes, ou bien des moments spatiaux, des moments statistiques, etc. Dans le cas, où la description est précédée d'une détection préalable de zones d'intérêt (points d'intérêt, régions, contours), il est possible d'utiliser l'information de position de chacune des primitives et d'en dériver des descripteurs basés sur des éléments géométriques (angles, surfaces, distances, etc.).
La méthode que nous proposons pour la traçabilité d'un catalogue de séquences vidéo fait appel à des signatures locales extraites autour de points d'intérêt ; nous reviendrons donc plus en détail sur les différents descripteurs locaux les plus utilisés par la suite (cf. 2.2.3).

Les mesures de similarité

Le choix de la mesure de similarité (ou de disimilarité[3]) des caractéristiques dépend des descripteurs utilisés et de la similarité recherchée. Les relations entre similarité sémantique, similarité visuelle et mesures de similarité ont été largement étudiées ([163, 119]) et nous ne nous y attarderons pas ici. Il est cependant important de comprendre que la nature de la mesure de similarité (vote, distance, norme, etc.) et le type de descripteurs utilisés (vecteurs, histogrammes, liste d'objets, etc.) sont fondamentaux vis-à-vis des performances du système d'indexation, en particulier les temps de réponse de l'algorithme de recherche dans la base. Les systèmes permettant d'indexer un espace métrique dans le cas général sont ainsi moins performants que les systèmes permettant d'indexer uniquement des espaces vectoriels [49, 30]. Un certain nombre de systèmes d'indexation ne peuvent même indexer que des espaces vectoriels avec une distance de type L_p [47, 78]. Nous rappelons donc ici quelques définitions et précisions concernant les mesures de similarité avant de donner un bref aperçu des mesures les plus couramment utilisées. Tout d'abord, toutes les mesures de similarité ne sont pas des distances.

Définition Distance - *Une* ***distance*** *ou* ***métrique*** *sur un ensemble E, est une application $d : E \times E \rightarrow [0, +\infty[$, vérifiant $\forall \boldsymbol{X_i}, \boldsymbol{X_j}, \boldsymbol{X_k} \in E$:*

1. $d(\boldsymbol{X_i}, \boldsymbol{X_j}) = d(\boldsymbol{X_j}, \boldsymbol{X_i})$*(symétrie)*
2. $d(\boldsymbol{X_i}, \boldsymbol{X_j}) = 0 \Leftrightarrow \boldsymbol{X_i} = \boldsymbol{X_j}$*(identité)*
3. $d(\boldsymbol{X_i}, \boldsymbol{X_j}) \leq d(\boldsymbol{X_i}, \boldsymbol{X_k}) + d(\boldsymbol{X_k}, \boldsymbol{X_j})$*(inégalité triangulaire)*

On dit alors que (E, d) est un ***espace métrique****.*

Notons que la définition d'un espace métrique n'implique pas que les descripteurs soient exprimables sous forme de vecteurs. Ainsi, la distance peut être une application comparant directement deux images comme par exemple dans [145], où une distance entre images fondée sur les arbres quaternaires est décrite. La seule donnée d'une table de disimilarité, fournissant pour tous les couples d'objets possibles une mesure de disimilarité répondant à la définition d'une distance est suffisante pour définir un espace métrique.

Lorsque les éléments de l'ensemble E, sont des vecteurs, (E, d) est alors un espace vectoriel. Sans rentrer dans le détail algébrique de la définition d'un espace vectoriel, il est important de noter que dans ce cas, tous les descripteurs doivent être exprimables sous une même forme vectorielle et avoir la même dimension, l'application d doit encore être une distance. Contrairement au cas plus général de l'espace métrique, à chaque descripteur doit correspondre une position dans l'espace vectoriel. Les métriques de Minkowski (ou normes L_p) sont les distances les plus répandues pour les espaces vectoriels. Elles sont définies de la manière suivante :

$$L_p(\boldsymbol{X_1}, \boldsymbol{X_2}) = \left[\sum_{j=1}^{D} \left| x_j^1 - x_j^2 \right|^p \right]^{\frac{1}{p}}$$

$$L_\infty(\boldsymbol{X_1}, \boldsymbol{X_2}) = \max_{j \in D} \left| x_j^1 - x_j^2 \right|$$

où x_j^1 est la j^{eme} composante de $\boldsymbol{X_1}$ et x_j^2 est la j^{eme} composante de $\boldsymbol{X_2}$.
L_2 n'est autre que la distance euclidienne (non pondérée) ; L_1 est également appelée distance de

[3] Nous parlerons de mesure de similarité dans le cas général et plus spécifiquement de mesure de disimilarité lorsque la mesure est nulle pour deux objets identiques et croissante lorsque les objets sont de moins en moins similaires

Manhattan.
On peut ainsi hiérarchiser les espaces de descripteurs selon leur besoin de généralité pour leur indexation. Lorsque la similarité est mesurée par une distance L_2 ou L_1 dans un espace vectoriel, presque tous les systèmes d'indexation multidimensionnelle développés durant les dix dernières années peuvent être utilisés. Lorsqu'il s'agit d'espaces métriques non vectoriels, les méthodes d'accès à la base ont été développés plus récemment et sont moins nombreuses. D'une manière générale, plus le besoin en généralité est important et moins le système d'indexation sera performant pour le temps de recherche. Il existe par exemple certaines mesures de similarité non métriques pour lesquelles aucune méthode d'accès n'a été proposée ; on est alors obligé de calculer séquentiellement la mesure de similarité avec tous les éléments de la base. En revanche, d'un point de vue de la qualité de la recherche, des mesures plus complexes permettent de gagner sensiblement par rapport à une simple distance L_1, notamment en ce qui concerne la robustesse. Le compromis entre qualité et efficacité dépend encore une fois de l'application envisagée.

Nous allons maintenant faire un inventaire de quelques mesures de similarité plus complexes que les distances L_p et ayant des besoins d'indexation différents :

1. **Distances élaborées dans des espaces vectoriels :** Pour gagner en robustesse, de nombreuses alternatives ont été proposées aux distances de Minkowski (L_p). Une des plus connues est la distance de Mahalanobis permettant de tenir compte des incertitudes sur les vecteurs ainsi que de la corrélation éventuelle de leurs composantes.
   L'intersection d'histogrammes a été longuement étudiée dans le cadre de l'indexation de la couleur, comme par exemple la distance proposée par Smith [158] : soient deux histogrammes $\boldsymbol{H_1}$ et $\boldsymbol{H_2}$ de dimension D alors
   $$d_{smith}(\boldsymbol{H_1}, \boldsymbol{H_2}) = 1 - \frac{\sum_{j=1}^{D} min(h_j^1, h_j^2)}{min\left[\sum_{j=1}^{D}(h_j^1), \sum_{j=1}^{D}(h_j^2)\right]}$$
   où h_j^1 est la j^{eme} composante de $\boldsymbol{H_1}$ et h_j^2 est la j^{eme} composante de $\boldsymbol{H_2}$.
   Pour gagner encore en robustesse, des distances plus élaborées mais plus difficilement accessibles par les méthodes d'indexation classiques ont été proposées comme les distances d'histogramme sous forme quadratique généralisée, les distances entre distributions cumulées [161], l'*earth mover's distance* [141] ou encore les distances tangentes [156] issues de la reconnaissance des formes.
2. **Distances dans des espaces métriques non-vectoriels :** Un des avantages majeur qu'il y a de s'abstenir de la contrainte vectorielle est l'utilisation de descripteurs de dimensions différentes pour les deux objets comparés ou l'utilisation d'un nombre variable de descripteurs par objet (comme c'est souvent le cas après une détection de zones d'intérêt). La distance *edit*, par exemple, est issue du domaine de la reconnaissance des formes. Elle permet de comparer deux chaînes de caractères par le nombre d'itérations d'édition nécessaires pour passer de l'une à l'autre (changements, ajouts ou suppressions d'un caractère). Elle a notamment été adaptée à la mesure de similarité entre vidéos, les caractères représentant les classes d'un ou plusieurs attributs [2].
3. **Mesures de disimilarité non métriques :** Certains travaux ont montré que la contrainte de l'inégalité triangulaire n'avait pas systématiquement un sens en terme de similarité visuelle et des mesures non métriques ne respectant pas l'inégalité triangulaire ont vues le jour comme la fonction dynamique partielle (*DPF*, [78]), définie par :
   $$L_p(\boldsymbol{X_1}, \boldsymbol{X_2}) = \left[\sum_{j \in \Delta_m} \left|x_j^1 - x_j^2\right|^p\right]^{\frac{1}{p}}$$

où Δ_m représente l'ensemble des m plus petites différences $\left|x_j^1 - x_j^2\right|$. Elle permet de ne pas prendre en compte dans la mesure les exceptions qui se produisent parfois sur certaine composantes.
La mesure de Hausdorff, appelée à tort distance de Hausdorff, n'est également pas une distance car elle ne respecte pas la propriété de symétrie :

$$d_{haus}(A, B) = \max_{a \in A} \min_{b \in B} \|a - b\| \neq d_{haus}(B, A)$$

A et B représentant deux ensembles de points dans un espace vectoriel et $\|.\|$ une métrique quelconque. Elle est par exemple utilisée par Indyk et al. [98], pour comparer deux ensembles de positions temporelles dans une vidéo.

4. **Mesures de similarité quelconques :** La similarité entre deux objets n'est pas forcément quantifiée par une mesure de disimilarité. Un vote consistant à compter un certain type d'appariements locaux entre deux objets est par exemple croissant lorsque les objets sont de plus en plus similaires.

Exemples de quelques systèmes de recherche d'images par le contenu

Un bref descriptif de trois systèmes de recherche d'images fixes par le contenu est donné en annexe A. Il s'agit des systèmes QBIC [132], Ikona [28] et BlobWorld [39].

1.3.4 La reconnaissance d'objets

Certaines approches de la recherche d'images par le contenu se rapprochent des problématiques de reconnaissance d'objets ou de formes. Depuis les années 90, beaucoup d'études ont été menées au sujet de la reconnaissance d'objets dans des images ou des vidéos. Il peut s'agir de retrouver des formes, des motifs, des logos, des visages [176], des mouvements particuliers [57], etc. Le principe général est d'utiliser l'information d'une image (ou d'une vidéo) pour prendre une décision parmi un ensemble d'hypothèses. Les hypothèses peuvent être uniquement conceptuelles (Quelle lettre représente telle image ? Quelle espèce de poisson ? Représente-t-elle une boule ou un cube ?) contrairement à la recherche par le contenu où les hypothèses sont des images (ou des vidéos). La reconnaissance d'objets se ramène souvent à un problème d'apprentissage supervisé sur un ensemble d'images ou de vidéos témoins représentant les hypothèses. La plupart des techniques employées sont alors soit issues de l'intelligence artificielle, principalement des réseaux de neurones [128] soit des approches statistiques faisant appel à la théorie de la décision [102].

Une des différences importantes entre la recherche par le contenu et la reconnaissance d'objets est la spécificité des images utilisées en reconnaissance. Pensons par exemple la reconnaissance de caractères où toutes les images sont a priori des caractères tracés sur un fond uniforme. On comprend alors que les descripteurs utilisés dans de telles applications soient eux aussi souvent très spécifiques aux images utilisées, aux objets recherchés et au type d'invariance ou de généralisation recherchée. On est alors très éloigné des bases d'images ou de vidéos généralistes utilisées dans un contexte de recherche par le contenu, pour lesquelles les descripteurs doivent caractériser les images dans toute leur diversité.

Cependant, l'abondance des travaux en reconnaissance d'objets fournit une très large quantité d'outils applicables dans notre contexte de détection de copies, notamment en ce qui concerne l'extraction de descripteurs invariants à un ensemble donné de transformations [156, 130].

1.4 La détection de copies d'images et de vidéo par le contenu

Dans cette partie, nous nous intéressons plus spécifiquement à la détection de copies d'images ou de vidéos par le contenu. L'objectif est de présenter le principe général (section 1.4.1) et de préciser les spécificités et les enjeux de la détection de copies par le contenu, notamment en terme de taille du catalogue de référence (section 1.4.2) et de discriminance de la signature (section 1.4.3). Nous focaliserons ensuite sur les transformations qui peuvent advenir entre une copie et son original avant de dresser un état de l'art des méthodes de détection de copies par le contenu. L'objectif applicatif des travaux de cette thèse étant la traçabilité d'un catalogue d'archives télévisées, nous préciserons le cas échéant les contraintes liées à la vidéo ou à la nature des archives télévisées.

1.4.1 Principe général

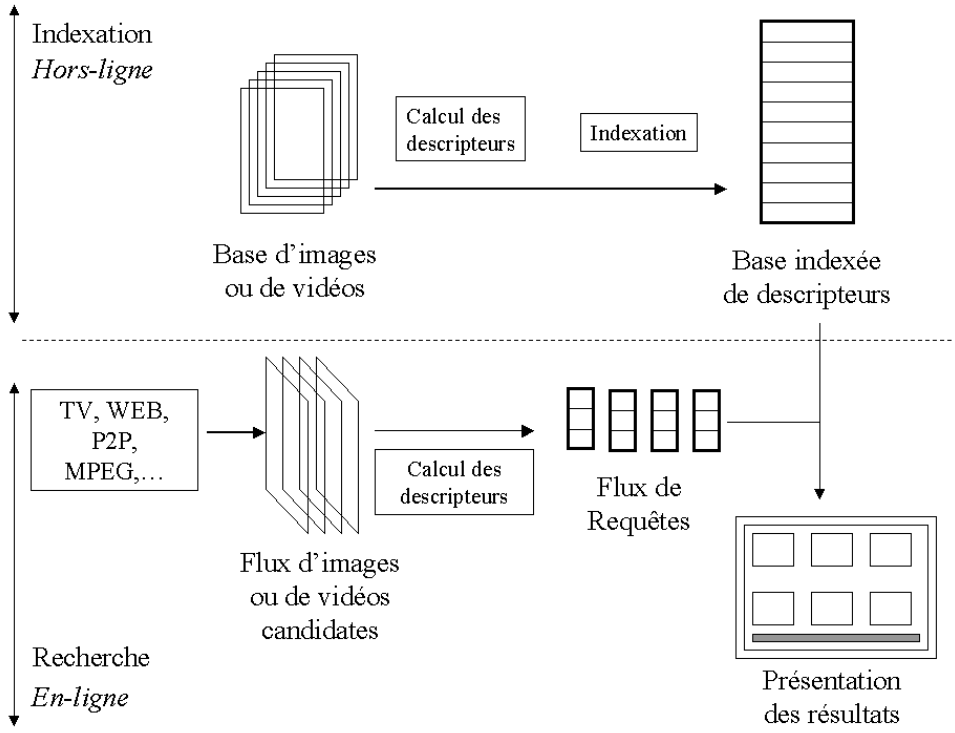


FIG. 1.4 – Principe général d'un système de détection de copies par le contenu

Au niveau du principe général de fonctionnement (voir figure 1.4), un système de détection de copies d'images ou de vidéos par le contenu, se distingue d'un système de recherche par le contenu principalement dans sa phase *en-ligne*, c'est à dire lors de la recherche proprement dite. Il s'agit d'un système dans lequel le paradigme de recherche est toujours de type *image-exemple* sans qu'il y ait nécessairement un utilisateur à cette étape. Les requêtes sont souvent constituées d'un flux d'images (ou de vidéos) transmises automatiquement au système de recherche, par exemple par un aspirateur automatique d'images sur le Web, ou bien par une carte d'acquisition TV, ou encore par des lectures sur disque. L'objectif est de surveiller en permanence un médium particulier afin d'avoir la traçabilité la plus complète possible. Des objets indépendants peuvent toutefois toujours être soumis au système. Lorsqu'un utilisateur a un doute sur la provenance

d'un document qu'il souhaite exploiter, il peut par exemple vérifier qu'il ne fait pas partie d'un catalogue protégé.
En ce qui concerne, la phase *hors-ligne*, le principe est, comme dans le cas général, de construire une base de descripteurs en les calculant sur la totalité du catalogue d'images ou de vidéos de référence.
Dans le cas de la détection de copies, l'objectif des descripteurs étant d'identifier des objets et non pas de caractériser des objets similaires, nous emploierons le terme de **signature**[4] pour les désigner.
En sortie du système de recherche, le résultat brut est une liste d'appariements entre des objets candidats et des objets référencés. Le système n'étant cependant pas à l'abri des fausses alarmes et les résultats bruts n'étant pas directement exploitables pour certaines applications, l'intervention finale d'un utilisateur expert est souvent nécessaire. Il convient alors de présenter les résultats sous la forme d'une interface graphique permettant le contrôle visuel des résultats.

1.4.2 Taille du catalogue de référence et flux diffusé

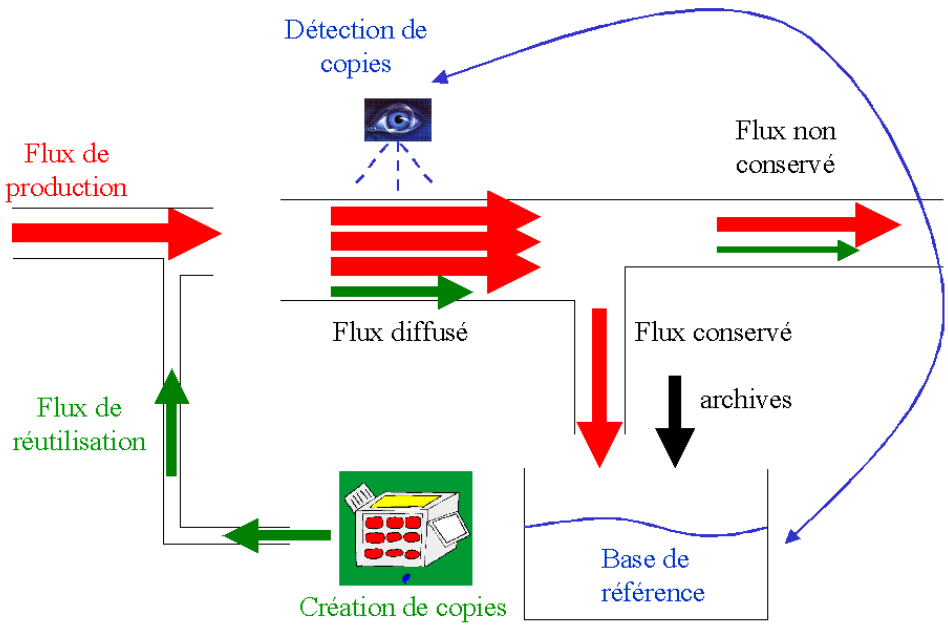


FIG. 1.5 – Chaîne de diffusion de documents audiovisuels

Si on s'intéresse à une chaîne complète de diffusion d'un document audiovisuel de manière globale, on peut l'illustrer par la figure 1.5. L'objectif d'un système de détection de copies est de détecter dans le flux diffusé, la partie correspondant à des copies d'objets du catalogue de référence, c'est à dire le flux de réutilisation. Dans le cas d'Internet par exemple, le flux diffusé est constitué des nouvelles images ou vidéos mises sur le web ; dans le cas de la télévision, le flux

[4] Le terme *fingerprint* est parfois également employé en anglais (surtout en audio), par analogie au pouvoir discriminant des empreintes digitales humaines

diffusé est constitué d'une ou plusieurs chaînes. Ce flux n'est pas égal à la somme du flux réutilisé et du flux produit (nouvelles images, nouveaux programmes, etc.) car le flux produit est très souvent lui même dupliqué sur plusieurs canaux de transmission ou retransmis plusieurs fois sur le même canal. La part du flux réutilisé dans le flux diffusé, est en général assez faible. Ce constat est important dans un contexte temps-réel puisqu'il permet d'envisager des post-traitements plus lourds lorsque des copies sont détectées, comme par exemple une deuxième comparaison image à image plus élaborée. Potentiellement, en plus d'archives anciennes, la totalité du cumul dans le temps du flux produit, est susceptible d'alimenter le catalogue de référence. En réalité, seule une partie du flux produit sera conservée dans le catalogue de référence, mais on peut considérer que la croissance du catalogue sera proportionnelle au flux produit. Si on note M, la taille du catalogue de référence et $\Phi_{prod}(t)$ le flux de production, on a $\frac{dM}{dt} = k\Phi_{prod}(t)$. Si le flux de production reste constant le catalogue de référence aura une croissance linéaire ; en revanche, si à l'avenir, le flux de production continuait à croître de manière linéaire, la croissance de la base de référence serait quadratique. Quoi qu'il en soit, cela illustre deux contraintes majeures pour le fonctionnement d'un système de détection de copies à long terme :

- La taille du catalogue de référence est sans cesse croissante et peut atteindre des volumes très importants. Dans le cas de la base d'archives de l'INA, le catalogue actuellement numérisé dépasse les 200 000 heures de vidéo et croit de plus en plus rapidement (40 000 heures en 2003, 60 000 heures en 2004). Nous verrons dans la partie II de ce mémoire, que cette contrainte doit être prise en compte dans l'algorithme de recherche des signatures dans la base afin que le temps de réponse soit sous-linéaire en fonction de la taille de la base. Cette contrainte est rarement prioritaire dans les systèmes de recherche classiques, qui se sont plutôt focalisés sur l'impact de l'augmentation de la dimension de la signature, afin d'améliorer la robustesse [5].
- Les flux de diffusion, qui doivent être contrôlés, sont importants et contraignent fortement les temps de calcul et de recherche dans la base, durant la phase *en-ligne*. Entre un système de recherche d'images fixes, où le temps disponible pour traiter les requêtes indépendantes de plusieurs utilisateurs est proche de la seconde et les contraintes temps-réel d'un système de monitorage de plusieurs chaînes de télévision, il y a un gouffre. Les critères d'efficacité sont donc tout aussi importants que les critères qualitatifs de la recherche puisqu'il n'est pas question de prendre du retard par rapport au flux de diffusion. Ainsi, parmi les nombreux descripteurs proposés dans la littérature dans les approches de recherche par le contenu classiques, beaucoup sont trop coûteux en temps de calcul et l'extraction des signatures est déjà trop coûteuse à elle seule. Des mesures de similarité coûteuses ont également une influence radicale sur les temps de recherche. Car même si les méthodes d'indexation permettent de limiter le nombre de distances à calculer, les temps de recherche restent fortement dépendants du temps de calcul d'une seule distance.

1.4.3 Discriminance et robustesse de la signature

L'autre principale spécificité de la détection de copies tient à la nature même de la similarité recherchée et à l'impératif de **discriminance** entre les différents objets. Intuitivement, on souhaiterait atteindre la propriété d'unicité de la signature. Le cas idéal serait d'avoir une signature unique pour chaque objet du catalogue de référence tout en conservant une invariance parfaite à la totalité des transformations de l'ensemble T, définissant ce que l'on entend par copie (cf. 1.2). Ce cas idéal est bien entendu irréaliste, et en premier lieu à cause du fait que l'ensemble des transformations n'est pas complètement définissable. Dans le cas réel, les transformations advenues entre un original et une copie conduiront à une altération de la signature. L'objectif

d'invariance consiste à limiter cette altération, tandis que l'objectif de discriminance consiste à augmenter la dispersion inter-signatures à l'intérieur de la base.

Si on assimile les signatures à des points dans un espace vectoriel de dimension D, comme cela est souvent la cas, on peut formuler une interprétation graphique du problème de discriminance et d'invariance. L'altération d'une signature due à une transformation de l'image (ou de la vidéo), se traduit par un déplacement de la signature dans cet espace. Ainsi, l'ensemble T des transformations tolérées, peut être assimilé à un hyper-volume connexe autour de la signature. Plus les recouvrements entre les différents hyper-volumes augmentent, plus les risques de confusion augmentent. L'objectif est donc de limiter l'étendue de ces hyper-volumes (invariance) tout en augmentant la dispersion des signatures (discriminance). Notons, qu'un gain en invariance, par une opération de normalisation d'une signature par exemple, aura des conséquences sur la répartition des signatures dans l'espace et que la perte de discriminance résultante peut engendrer un compromis global encore plus médiocre. L'**inter-variabilité** des signatures est donc indissociable de l'**intra-variabilité** d'une signature pour qualifier un descripteur. On voit également clairement le problème de l'augmentation du nombre de signatures dans la base : celle-ci se traduit par une augmentation de la densité des signatures sans que les hyper-volumes changent, augmentant ainsi les risques de confusion. **L'augmentation de la taille du catalogue de référence diminue donc la discriminance de la signature**. L'influence d'une telle augmentation sur l'efficacité globale des systèmes de recherche par le contenu est pourtant rarement étudiée.

La recherche par le contenu dans le cas général partage également les objectifs d'invariance et de discriminance. En revanche, la recherche de généralisation qui est un autre objectif n'est pas sollicitée dans le cas de la détection de copie. Deux objets différents (images ou vidéos) ayant des attributs communs ou des propriétés sémantiques communes doivent en effet être discriminés au même titre que les autres objets, et non pas avoir des signatures proches. C'est pour cette raison que beaucoup de descripteurs utilisés en recherche par le contenu, comme les histogrammes couleur, par exemple, ne nous paraissent pas adaptés à la détection de copies.
De plus, en ce qui concerne la discriminance et l'invariance, le compromis ne se situe pas au même niveau. L'ensemble des transformations auxquelles les systèmes de recherche par le contenu tentent de se rendre invariants est beaucoup plus vaste et inclut, par exemple, l'invariance aux changements de point de vue d'un même objet, ou de manière générale l'invariance à toutes les transformations 3D des scènes réelles n'entrant pas dans notre définition d'une copie. Ce type d'invariance conduit systématiquement à une perte de discriminance ; deux scènes différentes d'un même film, prises dans un même lieu, ne seront par exemple plus discriminées. La figure 1.6 illustre un cas de ce type. Ainsi, c'est souvent la conception même de la signature qui est inadaptée à la détection de copies. Autrement dit, **lorsqu'une transformation ne fait pas partie de l'ensemble T, il est non seulement inutile de s'y rendre invariant mais c'est en plus contre-efficace.**

La problématique de la détection de copies peut être vue à la fois comme un problème de recherche par le contenu dans le sens où la définition d'une copie inclut une notion de similarité visuelle mais également comme un problème de reconnaissance d'objets dans lequel le catalogue de référence représente un ensemble d'hypothèses. Les outils issus de la reconnaissance d'objets ne sont cependant pas nécessairement exploitables pour la détection de copies. Les techniques d'apprentissage ne sont par exemple pas vraiment concevables : pour ce genre d'approche, il faut plusieurs exemples d'appariement par hypothèse, hors dans notre cas le nombre d'objets

FIG. 1.6 – **Deux scènes d'une même vidéo :** La transformation affine entre les deux images, est typiquement le genre d'invariance recherchée dans un système de recherche d'images par le contenu. Du point de vue de la détection de copies, il s'agit de deux objets différents

dans le catalogue, c'est à dire le nombre d'hypothèses, peut être très grand. La large gamme d'opérateurs invariants utilisés en reconnaissance d'objets n'est également pas vraiment adaptée. Ils sont généralement spécifiques à un ensemble très restreint de transformations et à la nature particulière des hypothèses.

1.4.4 Ensemble des transformations

Nous avons vu dans la section précédente que le pouvoir discriminant d'une signature dépend largement de sa sensibilité à l'ensemble des transformations pouvant advenir entre un original et une copie. Lorsque l'on compare les performances de différents systèmes de détection de copies, il ne faut donc pas oublier que l'efficacité de la détection dépend pour une large part de la complexité des transformations prises en compte. L'objectif de cette section est de dresser un inventaire des principales transformations rencontrées dans les différents contextes applicatifs et de discuter leur complexité. La nature de l'ensemble des transformations dépend de plusieurs critères dont le type de médium de diffusion (Web, TV, etc.) mais aussi, de la nature des objets de la base de référence.

Transformations non visuelles

Il s'agit de l'ensemble des transformations appliquées aux images et aux vidéos, non pas pour modifier volontairement le contenu visuel de l'image, mais uniquement dans le but de diffuser, stocker ou protéger les images. En voici une liste non exhaustive :

- Compression / décompression
- Changement de résolution (horizontale et verticale)
- Changement d'espace colorimétrique
- Changement de support
- Conversion analogique / numérique
- Dégradation / Restauration

– Tatouage
– Débruitage

Et plus spécifiquement pour la vidéo :

– Changement de standard (PAL, NTSC, SECAM)
– Changement du taux d'images par seconde (25 fps, 30 fps)
– Diffusion analogique
– Changement de support (VHS, SDTV, DVD, HDTV)

Lors d'une utilisation normale, ces transformations sont les plus bénignes car elles ne modifient pas profondément le contenu visuel. Par la suite, on les qualifiera de **légères**. Il existe bien sûr des exceptions où ce type de transformations peut sérieusement altérer le contenu visuel, comme par exemple un changement de résolution radical pour obtenir des vignettes.

Transformations visuelles

Il s'agit des transformations ayant pour objectif de modifier le contenu visuel. Cet ensemble étant a priori infini, nous nous contentons ici de dresser une liste non exhaustive des principales catégories et des transformations les plus fréquentes :

– Incrustations (logos, textes, cadres, personnages, etc.)
– Transformations géométriques (redimensionnement, rotation, décalage, inversion, *fish eye*, etc.)
– Corrections colorimétriques (changement de gamma, changement de contraste, inversion de couleurs, saturation, etc.)
– Remontage temporel des vidéos (redécoupage, ralenti, lecture à l'envers, etc.)

La robustesse à ce type de transformations est en général beaucoup plus complexe à obtenir et dépend de la sévérité avec laquelle elles sont appliquées. Il paraît en tout cas irréaliste de concevoir une signature invariante à toutes ces transformations tellement certaines nécessitent une attention toute particulière (par exemple le fish eye ou la lecture à l'envers). Il importe donc pour une application donnée d'identifier l'ensemble le plus pertinent.

Transformations spécifiques aux archives télévisées

L'application principale des travaux de cette thèse étant la traçabilité d'un catalogue d'archives télévisées, nous précisons ici quelques spécificités de l'ensemble des transformations.

Un point fondamental dans la problématique de la détection de séquences vidéos est la **granularité temporelle**, c'est à dire la longueur minimale des extraits que l'on souhaite détecter. Lorsque l'on recherche des versions intégrales de longs métrages, le problème n'est pas le même que lorsque l'on souhaite détecter la diffusion d'une seule image extraite d'une vidéo. Les flux vidéo rediffusés à la télévision ont eux même des granularités très différentes (film, publicité, émission, reportage de journal télévisé, etc.). Dans le cas de la ré-exploitation d'archives qui a un prix élevé, les diffusions sont de courte durée et les redécoupages temporels sont très fréquents. La granularité est donc plutôt de l'ordre de quelques secondes.
Une autre difficulté liée à la diffusion télévisuelle, est qu'il n'y a pas systématiquement adéquation entre *l'image de l'écran* (celle que l'on traite) et *l'image diffusée à l'écran*. Ainsi, lors de la diffusion d'un film, l'ajout de bandes noires ne se fait pas toujours au même endroit et les bandes ne sont pas toujours de la même largeur. Beaucoup d'archives sont également diffusées entourées d'un cadre, et l'image à l'intérieur du cadre est généralement redimensionnée et décalée. Les habillages de chaîne, éléments omniprésents de la diffusion télévisuelle, conduisent à de nombreux ajouts de bandes bordant l'écran, de textes ou encore de logos qui contribuent aussi

au décalage et au redimensionnement des images, en plus des problèmes d'occlusion engendrés par ces incrustations (cf. annexe G).
Notons enfin que pour la cohérence de la diffusion, la quasi-totalité des images diffusées à la télévision subissent des transformations colométriques qui peuvent être sévères, en particulier les archives dont la qualité visuelle initiale n'est pas toujours bonne. La couleur n'étant apparue à la télévision française qu'en 1967, de nombreuses archives sont également en noir et blanc.
Parmi les différentes catégories d'objets rediffusés (films, publicités, reportages, etc.) les archives les plus anciennes sont souvent celles qui subissent les transformations les plus sévères, comme l'illustre la figure 1.7.

FIG. 1.7 – **Exemple de détection d'une archive ancienne remontée** - Les transformations les plus importantes sont : un décalage, un zoom avant, un changement colorimétrique et une incrustation importante

1.4.5 État de l'art de la détection de copies d'images ou de vidéos par le contenu

Nous avons évoqué tout au long de ce chapitre les différences entre les systèmes de recherche par le contenu dans le cas général et la détection de copie par le contenu. La limite entre les deux est cependant floue et dans cet état de l'art nous nous sommes efforcés de regrouper les références ayant un intérêt direct pour la détection de copies même si la problématique n'est pas exactement la même.
Ainsi, Jaimes et al. [101] ont introduit la problématique de la détection de *doublons* dans une collection personnelle de photographies. Les laboratoires ont en effet constaté que leurs clients faisaient souvent plusieurs photos de la même scène et que ces *doublons* représentaient 19 % des images. Ce genre de similarité n'entre pas dans la définition que nous avons donné d'une copie mais, au niveau de la modélisation, les transformations sont très proches. En revanche, la taille du catalogue de référence utilisé reste limitée et l'approche envisagée à trois niveaux de similarité est peu envisageable pour des bases volumineuses.
Certains auteurs se sont intéressés à la détection de copies de *documents image*, c'est à dire de documents textuels sous forme d'images, en général scannées. Doermann [59] propose par exemple une signature basée sur la forme des caractères sans passer par une phase de reconnaissance de

caractères. Le descripteur utilisé reste cependant très spécifique à cette application.

Le premier système complet proposé dans la littérature, réellement dédié à la détection de copies d'images est le système RIME (*Replicated IMage dEtector*) [41]. La signature est basée sur les coefficients d'une décomposition en ondelettes de Daubechies. La base comprend 30 000 images mais les transformations restent bénignes.

Une des approches de la détection de copies par le contenu est également appelée *perceptual hashing* ou *visual hashing* et se rapproche plus des techniques de cryptographie. La signature est extraite par une fonction de hachage et est généralement constituée d'un vecteur binaire, la distance utilisée étant alors la distance de Hamming. En plus de l'identification, ces approches se posent la question de l'authentification des contenus. Ainsi, le compromis entre robustesse et discriminance est traité en termes de compromis entre robustesse et sécurité [137]. Cela signifie que le risque de collision entre les vecteurs binaires doit être nul, ce qui revient à imposer une discriminance encore plus sévère sur la signature. Le principe consiste à rendre le calcul de la signature aléatoire et à conserver une clé secrète pour chaque image afin de réitérer le même calcul lors de la phase de détection ; l'objectif de cette phase aléatoire est de rendre les signatures des différentes images statistiquement indépendantes. Ainsi, dans [167], l'image est découpée de façon aléatoire avant de calculer des statistiques sur le signal à l'intérieur de chaque bloc. L'inconvénient majeur est que, pour une image candidate donnée, on doit calculer une signature différente pour chaque image de la base (avec chaque clé secrète) ce qui rend ces approches inexploitables sur des grandes bases.

Plus spécifiquement à la détection de copies vidéo, Lienhart et al. [116], proposent l'utilisation d'un vecteur de cohérence couleur calculé sur des images clé mais n'étudient que les artefacts liés à la diffusion et à la numérisation. Sanchez et al. [147], utilisent une décomposition en composantes principales de l'histogramme couleur de chaque plan. Les approches basées sur les plans sont dangereux pour notre application, car le redécoupage temporel des archives est très fréquent. Hampapur et al. [86] proposent l'utilisation de moments spatiaux de contours à l'intérieur d'une grille fixe. Cette approche par bloc (cf. 1.3.3) n'est pas adaptée aux nombreuses translations qui peuvent advenir dans le cas de notre application. Ces trois approches sont dédiées à la détection de spots publicitaires et le catalogue de référence ne dépasse jamais 224 spots d'environ 30 secondes, ce qui limite la portée des résultats. L'objectif de la surveillance d'un catalogue de spots publicitaires est en général de permettre aux publicitaires de contrôler la diffusion intégrale de la version qu'ils ont fournie au diffuseur, et les transformations susceptibles d'advenir restent assez légères.
La signature proposée par Indyk et al. [98], est basée sur la position relative des changements de plan. Elles est très robuste à un grand nombre de transformations mais présente l'inconvénient d'avoir une granularité assez élevée (environ 30 secondes de diffusion minimale), ceci afin d'avoir suffisamment de changements de plan. D'autres approches de ce type, basées sur l'alignement local des changements de plans ont été proposées par la suite [94, 93]. L'originalité est que ce genre d'approches repose sur des mesures de similarité évoluées (distance *edit*, distance de Hausdorff, etc.), qui confèrent une très bonne robustesse. Malheureusement, la complexité de recherche est beaucoup plus importante du fait qu'elles ne peuvent pas être indexées dans des espaces vectoriels. Les catalogues de référence ne dépassent ainsi jamais quelques heures de vidéo. En outre, le redécoupage temporel, très fréquent dans notre cas, reste problématique.
Dans [87], Hampapur et al. ont comparé trois signatures de nature différente en fonction de leur robustesse à des changements de résolution MPEG1. Ils ont montré qu'une signature de type

ordinale donnait de meilleurs résultats que deux autres signatures reposant sur le mouvement et la couleur. La signature ordinale est basée sur le découpage de chaque image en blocs suivi de l'ordonnancement de ces blocs en fonction de leur intensité moyenne et paraît peu prometteuse pour d'autres types de transformations comme les incrustations ou les translations. Le catalogue de référence n'est qu'un long métrage de 2h12.
Oostveen et al. [133], proposent simultanément une signature et une technique d'indexation adaptée qui rappelle les approches de type *perceptual hashing*, puisque la signature est un vecteur binaire. La technique est issue d'un système de détection de copies sonores qui a déjà fait ses preuves [85]. La signature est basée sur la quantification binaire de statistiques calculées sur une grille fixe et pose encore le problème de la translation. La qualité de la détection n'a pas été évaluée en termes de robustesse.
L'extraction de signatures dans le domaine compressé est une autre approche de la détection de copies par le contenu. L'idée est de calculer les descripteurs directement sur les flux compressés tant au niveau de la construction de la base de référence que de la phase *en-ligne* de détection. L'avantage est double : la plupart des données multimédia stockées et diffusées étant déjà souvent sous forme compressée, l'économie de la décompression permet d'accélérer les deux phases *en-ligne* et *hors-ligne* du système. D'autre part, les processus de compression s'attachent déjà eux-même à ne conserver que l'information visuelle pertinente sous une forme compacte et peuvent constituer une étape fort utile dans l'extraction de signatures qu'il n'est pas nécessaire de recalculer. Ainsi, Coudray et al. [52], proposent une signature basée sur les vecteurs mouvement issus du flux MPEG. Après une première étape de compensation du mouvement global, peu discriminant, diverses statistiques et moments spatiaux sont calculés sur les vecteurs mouvement restants. La robustesse de la signature n'a malheureusement pas été étudiée dans un processus complet de détection de copies, pas plus que l'influence des différences de codage entre les séquences candidates et les séquences référencées. La taille des GOP[5], le taux de compression, le choix de l'image initiale sont autant de paramètres du standard qui peuvent avoir une influence importante sur la signature.
Dans [43], Cheung et al. proposent une étude intéressante consistant à estimer la multiplicité des vidéos sur le web à différents niveaux de similarité (identité, copies, similarités visuelles de plus haut niveau). Ils proposent une mesure de similarité entre deux vidéos basées sur le nombre d'images similaires que deux vidéos ont en commun. Une stratégie originale permet d'adapter la technique à tout type de distances entre images fixes sans avoir à calculer la distance sur toutes les paires d'images. Cette réduction de la complexité leur permet de calculer toutes les similarités d'environ 50 000 clips vidéos représentant 1 800 heures de vidéo en un temps acceptable. Ils ont montré que chaque clip parmi les 50 000 avait en moyenne 1,53 copies.
Dans [55], DeMenthon et al. proposent des descripteurs spatio-temporels basés sur des régions spatio-temporelles grossières (formes parallélépipédiques assez larges et courtes dans le temps), limitant la complexité algorithmique de ce genre d'approche. Des descripteurs de couleur, de mouvement et de position (dimension $D = 7$) sont tout d'abord extraits pour tous les pixels de la vidéo (réduite de manière conséquente) et sont ensuite classifiés par intervalle d'une demi-seconde afin de segmenter les régions spatio-temporelles les plus pertinentes. Les descripteurs correspondant au centre de chaque agrégat sont ensuite conservés comme signatures. Il est dommage que l'utilisation d'un découpage temporel fixe nuise à l'approche locale obtenue grâce aux régions spatio-temporelles. L'utilisation de la position et de la couleur dans la signature sont également problématiques pour notre application (translation, recolorimétrie, images noir&blanc). La base de référence utilisée dans les expérimentations présentées ne dépasse pas quelques heures.

[5]Group Of Images

L'utilisation de signatures locales autour de points d'intérêt pour la détection de copies, a été proposée, pour les images fixes par Berrani et al. [20], et pour les vidéos dans notre première contribution [104]. Nous reviendrons sur nos propres travaux dans la suite de ce mémoire. La méthode de Berrani, pour sa part, repose sur l'extraction d'environ 300 signatures de dimension 24 pour chaque image, sur la recherche des K-plus-proches signatures dans la base et sur un vote qui permet de fusionner les résultats de toutes les signatures d'une même image. Il obtient de bons résultats sur une base d'environ 10 000 images pour des transformations de type colorimétrique et géométrique assez sévères avec cependant des taux de fausses alarmes importants.

Notons enfin qu'il existe des systèmes industriels de détection de copies, qui sont pour l'instant plutôt dédiés à la surveillance d'un catalogue de spots publicitaires comme le système *K-track* de Kinomaï ou le système *Image-seeker* de LTU technologies. Les transformations tolérées sont cependant très limitées et la taille du catalogue ne peut dépasser 1 000 heures de vidéo. Leur application principale est le contrôle de la diffusion de spots publicitaires.

1.5 Synthèse

Dans ce chapitre, nous avons situé la problématique de la détection de copies par le contenu dans le cadre plus général de la recherche d'images et de vidéos par le contenu. Nous avons dans un premier temps dressé un panorama de ce domaine, puis précisé les spécificités de la détection de copies. Nous avons ainsi noté que les problématiques n'étaient pas vraiment les mêmes : l'une recherche la généralisation et la robustesse à un niveau sémantique, tandis que l'autre recherche la discriminance et l'invariance. Ainsi, les descripteurs utilisés ne sont pas directement applicables, d'autant plus que leur complexité de calcul et l'utilisation de métriques élaborées rendent leur usage inadapté au contexte temps réel de la traçabilité d'un catalogue de séquences vidéo. Nous avons également soulevé le problème de la croissance du catalogue des objets de référence et nous avons vu que très peu d'approches s'y intéressaient, laissant ainsi planer un doute sur l'évolution future des performances. L'état de l'art des méthodes dédiées à la détection de copies a de son côté montré que les approches existantes ne permettaient pas de répondre actuellement à notre problématique. La majorité ne se pose pas la question de la taille importante du catalogue de référence et beaucoup sont inadaptées aux problèmes de redécoupage temporel, de décalage des images et d'incrustations, qui sont très fréquents dans le cas des images diffusées à la télévision. Enfin, le problème des copies multiples dans le catalogue de référence est très rarement abordé.

Chapitre 2

Méthode proposée pour la détection de copies vidéo par le contenu

2.1 Stratégie générale

Une des lacunes des systèmes de détection de copies vidéo existant est qu'ils ne résolvent pas vraiment les problèmes cruciaux de la diffusion télévisée à savoir : le décalage des images à l'écran, les incrustations et le redécoupage temporel. Une des raisons est qu'ils considèrent l'extraction de signatures comme un problème global pour chaque objet. Intuitivement, disposant d'un clip vidéo, ils tentent de compresser l'information au maximum pour obtenir une signature unique représentant un condensé de toute l'information du clip. Partant de la constatation que toute l'information présente à l'écran n'appartient pas forcément à l'objet recherché, nous pensons que l'utilisation de signatures locales est plus adaptée à la détection de copies. Plutôt que de résumer toute l'information, le principe est de ne coder que certains *signes distinctifs* des objets du catalogue de référence. On peut faire une analogie avec le problème d'identification d'une personne. Si on identifie une personne par des caractéristiques visuelles globales comme la couleur de ses cheveux, de ses yeux, sa taille et son poids, un bon déguisement conduira facilement à une mesure de disimilarité importante. Alors que l'utilisation seule des empreintes digitales est beaucoup plus discriminante.
L'autre originalité de la méthode proposée est de dissocier la recherche des signatures dans la base, de la mesure de similarité globale finalement utilisée pour prendre la décision définitive :

- Dans un premier temps, toutes les signatures locales d'un même objet sont recherchées individuellement dans la base via la méthode de recherche par similarité statistique décrite dans la deuxième partie de ce mémoire (cf. partie II). Le critère de recherche de cette étape repose uniquement sur la signature indexée. L'objectif est de retrouver, pour chaque signature candidate, l'ensemble des signatures provenant potentiellement d'une copie ou d'un original.

- L'objectif de la deuxième étape est de fusionner les résultats d'un ensemble de signatures locales cumulées dans le temps et dans l'espace. Celle-ci n'utilise pas les signatures elles mêmes mais uniquement les données associées aux résultats de la première étape (identifiant, code temporel, position dans l'image, etc.). Pour chaque objet du catalogue de référence ayant au moins une signature dans l'ensemble des résultats cumulés, une estimation de paramètres suivie d'une mesure de similarité non métrique permet alors de décider

s'il s'agit ou non d'une copie.

Ainsi, la première étape peut être considérée comme un filtre très efficace basé sur une indexation de type vectoriel permettant la mise en œuvre, lors de la deuxième étape, d'une mesure de similarité plus élaborée, à un coût devenu abordable.

Dans la suite de ce chapitre, nous discutons d'abord de l'extraction des signatures locales : les propriétés recherchées, les outils existant et la mise en œuvre dans notre contexte vidéo (section 2.2). Nous abordons ensuite le sujet de la fusion des résultats issus de la recherche dans la base et de la décision finale (section 2.4).

2.2 Les signatures locales

2.2.1 Intérêts et enjeux des signatures locales

L'extraction de signatures locales comporte deux étapes : une étape de détection qui consiste à localiser précisément des primitives bas niveau et une étape de caractérisation qui consiste à calculer un descripteur autour de ces primitives, constituant la signature proprement dite. Les principaux intérêts et enjeux d'une signature locale sont sa **localité**, sa **localisabilité** et son **contenu informatif**. Ces trois propriétés répondent parfaitement aux contraintes principales de notre application :

- La **localité** signifie que le descripteur ne caractérise qu'une petite région de l'image. L'intérêt majeur est que la plupart des signatures locales d'une image (ou d'une vidéo) ne seront pas du tout affectées par la suppression d'une partie de l'image ou par l'incrustation d'un objet dans l'image.

- La **localisabilité** est la propriété des primitives bas-niveau à être localisables de manière précise. L'enjeu est que les régions signées par le descripteur local soient les mêmes pour les objets du catalogue de référence que pour une copie de ceux-ci. La localisabilité inclut ainsi la notion de répétabilité, qui caractérise la propriété des primitives visuelles à être détectées même après transformation de l'image ou de la vidéo. Cette propriété est particulièrement intéressante pour le problème du décalage des images sur l'écran de télévision.

- Le **contenu informatif** que l'on peut rapprocher de la notion d'entropie, qualifie la propriété des signatures à être différentes les unes des autres. Un contenu informatif important a donc une influence directe sur la discriminance des signatures.

Plusieurs types de primitives visuelles peuvent servir de support à une signature locale : des points, des tâches, des contours, des régions ou encore des formes particulières ou des mouvements particuliers. Elles peuvent être spatiales ou spatio-temporelles. Les points d'intérêt sont cependant les plus séduisants car ils offrent de meilleures perspectives pour les trois propriétés recherchées. Les contours sont par exemple moins bien localisables car ils possèdent une contrainte géométrique dans une seule direction. Les régions sont pour leur part bien moins locales.

Il faut enfin parler d'un dernier enjeu majeur des signatures locales : la fréquence d'apparition des primitives (spatiale et temporelle). L'empreinte digitale est pratique pour l'identification humaine car tout le monde en possède une. On peut trouver des primitives répondant parfaitement

aux trois critères précédents mais qui ne sont pas assez fréquentes pour signer tous les objets du catalogue de référence (par exemple : les intersections de 5 régions distinctes). A l'inverse, une primitive trop fréquente peut nuire en terme de discriminance et de coûts d'indexation (stockage, temps de recherche dans la base).
Notons, pour finir, que d'un point de vue applicatif le temps de calcul correspondant à l'extraction des signatures locales est aussi un facteur déterminant.

2.2.2 La détection de points d'intérêt

Un point d'intérêt est une primitive bas niveau ayant une contrainte spatiale bidimensionnelle contrairement à un contour, par exemple, qui n'impose une contrainte spatiale que suivant une direction. Ils sont des points de calcul localisables de manière fiable et ont, pour cela, trouvé leur première utilisation pour des problèmes d'appariement d'images et de stéréovision : l'appariement éparse d'images par exemple, consiste à utiliser les points d'intérêt d'images réelles pour faire de la réalité virtuelle. Étant données des images 2D qui couvrent tous les points de vue d'une scène réelle, l'appariement éparse de deux images voisines, c'est-à-dire la mise en correspondance de points significatifs, permet ensuite par interpolation de se situer suivant n'importe quel point de vue dans cette scène. Dans ce contexte, plusieurs détecteurs ont été proposés dans la littérature dans les années 90, comme le détecteur de Harris [88] ou encore les détecteurs de Horaud [95], Heitger [89] ou Förstner [73].
L'utilisation de points d'intérêt dans un contexte de recherche d'images par le contenu a été introduite par Schmid et Mohr [150]. La nécessité de développer des détecteurs de points adaptés à cette problématique a alors engendré de nombreux travaux. Inspirés de la théorie des espace-échelle de Lindeberg [118], des détecteurs invariants à l'échelle ont ainsi été proposés. Le principe de l'espace-échelle est de construire un espace à trois dimension représentant une image. Les deux premières dimensions sont les coordonnées spatiales habituelles et la troisième dimension correspond à l'échelle d'analyse de l'image. En pratique, les différentes échelles sont obtenues en filtrant l'image par une gaussienne dont l'écart type définit l'échelle d'analyse. La propriété théorique intéressante est que la représentation d'une image dans cet espace est la même quelle que soit l'échelle réelle de l'image d'origine. En construisant, dans cet espace, la réponse d'opérateurs différentiels comme le laplacien par exemple, il est possible de trouver certains points de l'image qui possèdent un maximum dans l'espace-échelle, qui est alors invariant à un changement d'échelle réel de l'image. Ce type d'approche est utilisé dans les détecteurs invariants à l'échelle développés par Mikolajczyk et al. [125] et Lowe [121].
Dans un souci d'adaptation de la détection des points d'intérêt à l'information visuelle réellement perçue par l'homme, des auteurs ont proposé des détecteurs basés sur le contraste [32], les ondelettes [120] ou encore la nature symétrique des objets présents dans une image [138].
Très récemment, Laptev et Lindeberg [111] ont étudié un détecteur de points d'intérêt spatio-temporels pour la vidéo. Il s'agit en fait de l'extension du détecteur de Harris à la dimension temporelle couplée à une approche espace-échelle qui le rend invariant au changement d'échelle.

2.2.3 La caractérisation locale

Avant toute chose, puisque nous allons être amenés à parler de la notion d'invariants, nous en donnons ici la signification. D'une manière générale, un invariant est une propriété qui est constante pour un ensemble de fonctions. On peut en donner la définition formelle de Hilbert :

Définition Invariant- *Étant donnés deux ensembles E et F, un ensemble T de transformations de E dans F et I une fonction dont l'ensemble de départ est F, alors I est invariante par T si*

et seulement si :

$$\forall e \epsilon E \; \forall t, t' \epsilon T \; I(t(e)) = I(t'(e))$$

De nombreuses techniques ont été développées pour décrire les régions locales d'une image. Le descripteur le plus simple est le vecteur des pixels de la région. Sa dimension est cependant très élevée et ses composantes très redondantes. L'objectif des autres descripteurs consiste à approximer le signal local de la manière la plus compacte possible, tout en essayant de tendre vers l'indépendance statistique des composantes du vecteur.
Une fonction peut être approximée localement par ses dérivées. Plus on augmente l'ordre du développement limité et plus l'approximation est proche du signal. Un vecteur stockant les dérivées en un point jusqu'à un certain ordre est appelé jet local. Il est possible d'en dériver des descripteurs différentiels invariants à certaines transformations, comme la rotation, les transformations inversibles de la luminosité, ou encore les transformations affines [71]. Cela nécessite de combiner les composantes du jet local pouvant aller jusqu'à l'ordre 3. Le jet local peut également être utilisé de manière orientée afin de caractériser les dérivées dans d'autres directions que les deux principales. L'orientation peut s'obtenir en utilisant des filtres orientés [75] ou tout simplement en appliquant une rotation à la région caractérisée.
On peut également caractériser une fonction par une description fréquentielle. La phase de la transformée de Fourier est ainsi indépendante de la luminosité des images ainsi que de leur contraste, et constitue un bon descripteur fréquentiel. Il s'agit cependant d'un descripteur global qui ne permet pas une localisation spatiale (on ne sait pas quelle fréquence appartient à quel point). La transformée de Gabor permet d'obtenir une bonne précision à la fois en fréquence et en espace par l'utilisation d'un fenêtrage gaussien sur les fonctions sinusoïdales. Les filtres de Gabor ont ainsi été utilisés comme descripteurs locaux dans plusieurs travaux sur la recherche d'images par le contenu, comme par exemple dans [120], où ils caractérisent le voisinage de points extraits par un détecteur basé sur les ondelettes. Les filtres de Gabor sont souvent interprétés comme des descripteurs de l'attribut visuel de texture. Ils peuvent également être utilisés de manière orientée par l'utilisation de filtres orientables.
Les moments spatiaux permettent également de caractériser un signal. Un moment d'ordre $(p+q)$ est défini par

$$M_{pq} = \sum \sum x^p y^q I(x, y)$$

Ils caractérisent plutôt l'attribut visuel de forme. Des combinaisons de ces moments peuvent également conduire à la construction d'invariants en translation, en rotation et en changement d'échelle. Van Gool et al. ont ainsi introduit les moments invariants généralisés pour décrire la nature multi-spectrale des données [80].
Les filtres complexes dérivent de la famille $K(x, y, \theta) = f(x, y)e^{i\theta}$ où θ définit l'orientation et $f(x, y)$ peut être une dérivée gaussienne ou encore un polynôme. Ils ont, par exemple, été utilisés récemment par Schaffalitzky et Zisserman [148], qui proposent un filtre complexe invariant à la rotation.
Dans [121], Lowe propose le descripteur SIFT. Un ensemble de 8 images est calculé à partir du gradient de l'image à caractériser, chacune ne contenant que les gradients orientés dans une certaine direction (les plans d'orientation). Pour chaque point d'intérêt, un vecteur de dimension 160 est calculé à partir du voisinage de ce point dans les 8 plans d'orientation.
Dans [38], Carneiro et al. proposent un descripteur orientable basé sur la phase du signal. La mesure de la phase est basée sur l'utilisation d'une paire de filtres en quadrature. Le second filtre H_2 est obtenu à partir du premier H_1 par la transformée de Hilbert où H_1 est la dérivée seconde

gaussienne du signal. La combinaison de H_1 et H_2 donne une réponse complexe de la forme :

$$h(x, y) = h_1(x, y) + ih_2(x, y)$$

où $h_1(x, y)$ et $h_2(x, y)$ sont les réponses des filtres H_1 et H_2 au pixel (x, y). Cette paire de filtres est de plus utilisée de manière orientée dans plusieurs directions et également à plusieurs échelles d'analyse. La phase de la réponse complexe est utilisée telle quelle dans le descripteur tandis que l'amplitude est saturée pour se rendre robuste à certains changements légers de luminosité.

Dans le cas de la vidéo, les descripteurs peuvent être spatio-temporels. En combinaison de leur détecteur de points d'intérêt spatio-temporels, Laptev et al. [111] utilisent comme descripteur local, le jet local étendu à la coordonnée temporelle jusqu'au troisième ordre de dérivation.

2.2.4 Évaluation et comparaison des signatures locales

Certains travaux [151, 153, 82] ont été menés pour comparer les détecteurs de points d'intérêt présents dans le littérature. La comparaison est faite sur deux critères, la répétabilité et le contenu informatif.
La répétabilité mesure la propriété des points d'intérêt et être localisables après transformation de l'image. Formellement, elle est définie par la nature projective des images :

Définition Répétabilité - *Étant donné un point $\boldsymbol{P}$ dans une scène réelle 3D et deux matrices de projection $\boldsymbol{M_1}$ et $\boldsymbol{M_2}$ de la scène réelle sur deux images $\boldsymbol{I_1}$ et $\boldsymbol{I_2}$, on dit que les points images $\boldsymbol{p_1} \in \boldsymbol{I_1}$ et $\boldsymbol{p_2} \in \boldsymbol{I_2}$ sont **répétables** si et seulement si $\boldsymbol{p_1} = \boldsymbol{M_1 P}$ et $\boldsymbol{p_2} = \boldsymbol{M_2 P}$.*
Dans le cas où les transformations entre images ne sont pas affines mais uniquement planaires cette relation se ramène à $\boldsymbol{p_2} = \boldsymbol{H_{21} p_1}$ où $\boldsymbol{H_{21}}$ est une homographie représentant la transformation entre les deux images.

Le contenu informatif est généralement mesuré par l'entropie d'un descripteur particulier autour des points d'intérêt, par le biais d'une partition de l'espace des descripteurs. En cela, elle n'est pas une caractéristique du détecteur seul mais d'une signature locale complète (détecteur + descripteur) et nous pensons qu'elle n'a pas de valeur comparative rigoureuse entre deux détecteurs même s'il s'agit du même descripteur utilisé. L'information présente autour d'un point d'intérêt peut en effet être de nature très variée (texture, forme, couleur, mouvement) et **le descripteur doit être adapté à l'information détectée**. Nous estimons donc qu'un détecteur de points d'intérêt n'a pas de valeur informative en soi et que seule des signatures locales complètes peuvent être comparées par cette mesure. De la même manière, un descripteur seul n'a pas de contenu informatif et est indissociable de l'endroit où il est calculé.

Dans [151], Schmid et al. comparent cinq détecteurs de points d'intérêt, dont leur version stabilisée du détecteur de Harris qui montre une meilleure répétabilité à toutes les transformations testées qu'elles soient géométriques (rotation, changement d'échelle, changement de point de vue) ou non (variation uniforme ou complexe des conditions d'illumination, bruits de caméra). Ces même détecteurs couplés à un descripteur de type invariants différentiels montrent que la signature locale constituée avec le détecteur de Harris stabilisé est la plus informative.

Dans [153], Sebe et al. font un comparatif entre le détecteur de Harris stabilisé et leur propre détecteur basé sur les ondelettes. Ils montrent que ce dernier possède une meilleure répétabilité à la rotation et au changement d'échelle, mais ne parle pas des transformations non géométriques. Il montrent également qu'une signature locale basée sur leur détecteur et un descripteur de type invariants différentiels est plus informative que le même descripteur utilisé avec le détecteur de Harris.

Dans [82], Gouet et al. montrent, avec leurs deux détecteurs basés sur la couleur, dont une adaptation du détecteur de Harris, que l'ajout de la couleur permet de gagner sensiblement en répétabilité pour de nombreuses transformations.

D'autres travaux [38, 126], se sont directement intéressés à l'évaluation des signatures locales en prenant en compte les courbes ROC[6] résultant d'une recherche de signatures candidates dans une base de signatures de références. Dans ces approches, deux points d'intérêt sont considérés comme similaires si la distance entre leur signature est inférieure à un certain seuil d_s. Les courbes ROC sont alors obtenues en faisant varier la valeur de ce seuil. Le taux de bonnes détections $p_{correct}$ est évalué en comparant les signatures des images avec une version transformée de celles-ci. Il est alors défini par le rapport entre le nombre de points correctement appariés et le nombre d'appariements possibles :

$$p_{correct} = \frac{\#appariements\ corrects}{\#appariements\ possibles}$$

Un appariement est correct lorsque l'erreur de positions relatives des deux points est inférieure à quelques pixels (3 en général) et que les signatures sont similaires (distance inférieure à d_s). Le taux de faux positifs p_{faux} est la probabilité de faire un mauvais appariement lors de la recherche dans la base. Il est estimé en recherchant dans la base les signatures d'un ensemble d'images candidates et est défini comme le nombre d'appariements faux (distance inférieure à d_s) divisé par le produit du nombre de signatures dans la base et du nombre de signatures candidates :

$$p_{faux} = \frac{\#appariements\ faux}{(\#signatures\ dans\ la\ base)(\#requetes)}$$

Ce critère est utilisé dans [38] où Carneiro et al. comparent leur descripteur basé sur la phase avec un descripteur différentiel en utilisant le détecteur de Harris dans les deux cas. Ils ont montré que leur descripteur était meilleur dans le cas de changements de la luminosité mais moins bon dans le cas d'un changement d'échelle.

Toujours avec le même critère, Mikolajczyk et al. [126] comparent plusieurs types de descripteurs, dont des filtres différentiels orientables, des invariants différentiels, des moments invariants et des descripteurs SIFT. Ils utilisent également plusieurs détecteurs de points d'intérêt récents, adaptés aux transformations testées. Ainsi, dans le cas d'un changement d'échelle ils utilisent deux détecteurs basés sur des maxima dans l'espace-échelle et théoriquement invariants à cette transformation ([125] et [121]). Dans le cas d'un changement de point de vue, ils utilisent un détecteur invariant aux transformations affines. La conclusion de leur étude est que les descripteurs SIFT sont meilleurs pour toutes les transformations de type géométrique (rotation, échelle et transformations affines) mais pas pour les changements d'illumination. Les filtres différentiels orientables donnent également de bons résultats pour toutes les transformations et réalisent un bon compromis étant donné leur faible dimension.

[6] Receiver Operating Characteristics

2.3 Méthode proposée pour l'extraction de signatures locales dans les vidéos

2.3.1 Détection d'images clé

L'approche brutale consistant à calculer des signatures locales spatiales sur chaque image du flux vidéo n'est pas envisageable en terme de coûts de calcul, de stockage et de recherche dans la base, sans compter la redondance résultante qui risquerait de nuire à la discriminance. Une solution consiste à introduire une localité temporelle en déterminant des instants caractéristiques de la vidéo. La localité temporelle présente les mêmes intérêts que la localité spatiale, à savoir la robustesse aux décalages et aux incrustations, c'est-à-dire dans le cas temporel, aux manipulations de remontage comme le redécoupage temporel. Les détections de régions d'intérêt temporelles classiques comme les plans, ne présentent pas une granularité temporelle assez fine car les plans sont souvent eux-mêmes redécoupés. Nous avons donc mis en œuvre une détection d'images clé, correspondant à des instants caractéristiques de l'activité moyenne de la vidéo. Celle-ci est basée sur l'intensité de mouvement, déjà utilisée par exemple pour la détection de changements de plan [61] et définie par :

$$a(t) = \sum_{x,y \epsilon W(I)} |I_{t+1}(x,y) - I_t(x,y)|$$

où $I_t(x,y)$ représente l'intensité du pixel de coordonnées (x,y) à l'instant t et W(I) représente un fenêtrage de l'image permettant de ne pas tenir compte des bords de l'image où l'activité n'est en général pas directement en rapport avec la scène principale de la vidéo (textes défilant par exemple). En filtrant $a(t)$ par un filtre gaussien de paramètre $\sigma_{activity}$ pour éliminer les petites variations trop sensibles au bruit, on obtient un signal $a_\sigma(t)$ dont les maxima locaux correspondent aux instants caractéristiques choisis. En plus de leur propriété de localité et de répétabilité, ils ont l'avantage, comme les points d'intérêts spatiaux, de correspondre à un contenu informatif temporel important. Le signal en lui-même n'est pas invariant aux changements de luminosité, en revanche la position des extrema est très robuste à de nombreuses transformations. On peut augmenter le taux d'images clé par seconde, c'est à dire le taux de compression, en augmentant la valeur de $\sigma_{activity}$ auquel cas on affine la granularité de la détection. Pour éviter de n'obtenir aucune signature lorsque l'activité est très faible et que le signal $a_\sigma(t)$ n'atteint aucun maximum, un échantillonnage arbitraire est effectué lorsque la fréquence d'apparition des maxima devient trop faible. On peut en effet considérer dans ce cas que le contenu de la vidéo sera statique et donc invariant à des petits écarts temporels.

2.3.2 Adaptation du détecteur de Harris stabilisé

Parmi les récents détecteurs de points d'intérêt efficaces récemment proposés dans la littérature (cf. 2.2.2) mais n'ayant pas encore fait l'objet d'études comparatives (comme par exemple les détecteurs invariants à l'échelle), aucun n'est pour l'instant raisonnablement applicable à un contexte vidéo temps réel. La plupart nécessitent des calculs coûteux à différentes échelles d'analyse pour chaque image et les performances des calculateurs sont encore loin d'être suffisantes. Nous avons donc utilisé le détecteur de Harris qui reste le plus utilisé et qui a largement fait ses preuves en terme de répétabilité aux transformations colorimétriques, géométriques et au bruit comme nous l'avons vu à la section 2.2.4. La principale différence avec des détecteurs plus évolués est qu'il est n'est théoriquement pas invariant à des changements d'échelle sévères ou des transformations affines. Les transformations affines ne font pas partie de l'ensemble des

transformations les plus fréquentes dans le cas de la détection de copies et le détecteur de Harris a tout de même une bonne robustesse aux changements d'échelle d'un facteur compris entre $0,75$ et $1,5$, ce qui couvre la grande majorité des cas observés. Nous avons essayé d'adapter au maximum le détecteur de Harris à notre contexte de détection de copies vidéo que ce soit pour le réglage des paramètres ou pour l'implémentation.

Le détecteur de Harris

Le détecteur de Harris [88], a été initialement proposé pour la détection de jonctions en L ou en T. La réponse du détecteur $h(x,y)$ est calculée pour chaque pixel (x,y) à partir de la matrice suivante qui prend en compte les valeurs des dérivées premières du signal de niveaux de gris :

$$\boldsymbol{H} = \begin{bmatrix} I_{xx} & I_{xy} \\ I_{xy} & I_{yy} \end{bmatrix}$$

avec :

$$\begin{aligned} I_{xx} &= \sum_{k,l \epsilon R(x,y)} g(k,l) \cdot \left(\frac{\partial I}{\partial x}(k,l) \right)^2 \\ I_{yy} &= \sum_{k,l \epsilon R(x,y)} g(k,l) \cdot \left(\frac{\partial I}{\partial y}(k,l) \right)^2 \\ I_{xy} &= \sum_{k,l \epsilon R(x,y)} g(k,l) \cdot \left(\frac{\partial I}{\partial x}(k,l) \right) \cdot \left(\frac{\partial I}{\partial y}(k,l) \right) \end{aligned}$$

- $g(k,l)$ est une fonction de pondération gaussienne bidimensionnelle de paramètre σ_{Ha} suivant les deux dimensions.
- R est une fenêtre carrée autour du point (x,y), dont la taille est dépendante de σ_{Ha}, de manière à conserver un pourcentage suffisant de l'énergie totale de la fonction de pondération $g(k,l)$.
- $\frac{\partial I}{\partial x}$ et $\frac{\partial I}{\partial y}$ sont les dérivées premières du signal de l'image.

Les valeurs propres de la matrice $\boldsymbol{H}$ correspondent aux courbures principales de la fonction d'autocorrélation. Si ces deux courbures sont grandes, cela indique la présence d'un point d'intérêt. Toutefois, afin de ne pas extraire les valeurs propres de la matrice, Harris utilise une mesure reposant sur la trace et le déterminant de la matrice :

$$h(x,y) = \det(\boldsymbol{H}) - k \cdot (trace(\boldsymbol{H}))^2$$

La valeur recommandée pour k est $k = 0,04$.

Stabilisation du détecteur de Harris

Comme l'a montré C. Schmid [149], le calcul des dérivées est mal conditionné dans le sens où il manque de robustesse vis-à-vis du bruit dans les données d'entrée. Elle a proposé pour y remédier d'utiliser un opérateur de dérivation plus souple. De tels opérateurs s'obtiennent aisément en dérivant une fonction de lissage, par exemple une gaussienne. Dans notre cas, nous avons utilisé des filtre de Hermite dont la convolution correspond à une dérivation accompagnée d'un filtrage gaussien de paramètre σ_{He}.

Recherche des maxima locaux

Une fois la réponse $h(x, y)$ du détecteur de Harris obtenue, il faut extraire des points d'intérêts de cette image. Ceux-ci correspondent théoriquement, dans le domaine continu, aux maxima locaux de la réponse du détecteur. L'approche proposée par C. Schmid [149] pour extraire les maxima dans l'image discrète consiste à rechercher le pixel le plus intense de l'image et à forcer à zéro la valeur de l'intensité des pixels avoisinant. Cette opération est ensuite réitérée autant de fois que de points souhaités. L'inconvénient est que la fenêtre de voisinage choisie n'est jamais parfaitement adaptée au lobe du maximum local et que cela peut engendrer des points erronés sur les bords de la fenêtre.
Nous avons choisi, pour notre part, de conserver les maxima locaux discrets réels sur une fenêtre 3×3, après un filtrage supplémentaire de la réponse du détecteur. Le nombre de points ainsi obtenus est moindre mais cela ne constitue pas forcément un handicap pour notre application.

Nombre de points et seuillage

Parmi tous les maxima locaux extraits de la réponse du détecteur de Harris, un certain nombre ne sont pas pertinents. Les points ayant des valeurs très proches de zéro sont ainsi principalement dûs au bruit. Ils sont présents autant dans des zones homogènes que dans des zones texturées. Ils peuvent être très nombreux dans le cas où les paramètres réglant l'échelle d'analyse, σ_{He} et σ_{Ha}, sont faibles.
Certains autres points ayant des réponses faibles ne résultent pas du bruit mais sont très peu fiables en position (texture, ensemble de petits objets, coins peu contrastés, etc.). Globalement, plus la réponse du détecteur est forte, plus les points sont visuellement significatifs et plus ils sont répétables.
Contrairement à d'autres applications, il est important pour la détection de copies vidéo de ne conserver que les points les plus pertinents et non la totalité des maxima locaux. D'une part parce qu'un nombre trop important de points d'intérêt pose des problèmes de coûts de stockage et de recherche dans la base de signatures. D'autre part, parce que les points non pertinents (non répétables ou non informatifs) risquent de nuire à la discriminance des signatures locales.
Les deux approches classiques pour limiter le nombre de points et ne conserver que les plus pertinents consistent soit à ne conserver qu'un nombre constant des points ayant la réponse la plus forte, soit à seuiller la valeur de la réponse. Les deux approches présentent des avantages et des inconvénients :

- Conserver un nombre fixe de points d'intérêt par image permet de s'abstraire de la variabilité de la réponse du détecteur d'une image à l'autre et donc de garantir une certaine constance du débit de signatures. L'inconvénient majeur est de forcer la présence de points non pertinents dans certaines images et d'omettre certains points significatifs dans d'autres. Le fait que le critère soit global à l'image analysée est également un inconvénient pour notre application, notamment pour les incrustations ou les suppressions de certaines régions de l'image. Le risque est de voir les points d'intérêt se faire *aspirer* par l'ajout d'un logo ou d'un texte dans l'image.

- L'utilisation d'un seuil fixe rend la détection d'un point d'intérêt indépendante de celle des autres. Cela résout le problème de l'incrustation d'objets dans les images. Ce critère

FIG. 2.1 – Illustration d'une détection de points d'intérêt à deux échelles d'analyse différentes : en gauche $\sigma_h = 0,8$ - à droite $\sigma_h = 2,4$

a également l'avantage d'adapter le nombre de points au contenu réel de l'image et de ne pas forcer la détection de bruit dans des images homogènes. En revanche, la réponse du détecteur de Harris étant très dépendante des conditions globales d'illumination d'une image, l'utilisation d'un seuil engendre un débit de signatures très variable d'une séquence vidéo à une autre.

Nous avons finalement opté pour une approche hybride en limitant le nombre de points par image en plus d'un seuil fixe sur la réponse du détecteur pour éviter des points trop bruités.

Echelle d'analyse et recentrage des points d'intérêt

Les valeurs des paramètres σ_{He} et σ_{Ha} sont très importantes puisqu'elles caractérisent l'échelle d'analyse du détecteur. Dans la suite, on considérera qu'elles sont toujours égales et on parlera uniquement de l'échelle d'analyse $\sigma_h = \sigma_{He} = \sigma_{Ha}$. La figure 2.1 illustre l'influence de l'échelle d'analyse sur une détection de points dans une image réelle.

On peut observer que les points d'intérêt détectés ne sont globalement pas les mêmes pour les deux échelles d'analyse. Les coins diffus ne sont par exemple détectés qu'à l'échelle d'analyse la plus grande. Une échelle d'analyse trop faible présente plusieurs inconvénients pour notre application :

- Plusieurs points sont souvent très proches les uns des autres car ils correspondent aux variations haute fréquence de primitives visuelles plus globales (comme par exemple les points situés sur la bouche du personnage de la figure 2.1). Les signatures locales seront ainsi très redondantes.
- La répétabilité des points observés à faible échelle est moins bonne pour des transformations colorimétriques comme un changement de gamma ou pour la présence de bruit. Le détecteur est également plus sensible aux artefacts de compression ou la présence de poussières très fréquentes dans les archives anciennes.
- La dispersion spatiale des points est plus faible. Le nombre de points par image étant limité, ceux-ci ont tendance à se concentrer dans une région unique de l'image. Ceci limite la robustesse à l'incrustation de petits objets très saillants comme un logo ou un texte.

On comprend donc l'intérêt d'effectuer la détection à une échelle d'analyse plus grande. Cette dernière a cependant également des inconvénients. Le coût de l'extraction est d'abord beaucoup

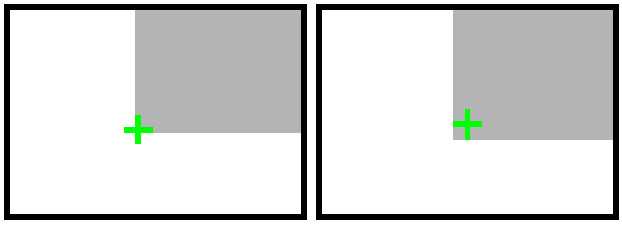


FIG. 2.2 – Illustration de la délocalisation du détecteur de Harris : à gauche $\sigma_h = 0,8$ - à droite $\sigma_h = 2,4$

plus élevé puisqu'il est quadratique par rapport à la taille de la fenêtre de calul. D'autre part comme l'a montré Deriche [56] au sujet de la plupart des détecteurs opérant directement sur les niveaux de gris, le détecteur de Harris souffre de délocalisation dans le cas de points observés à une échelle d'analyse trop grande, comme l'illustre la figure 2.2.

Cette délocalisation est croissante avec la valeur σ_h et perturbe la répétabilité du détecteur à un redimensionnement de l'image. En effet, si pour un σ_h donné on a un décalage d_{σ_h}, l'erreur de position dans une image redimensionnée d'un facteur s, pour une même échelle d'analyse, sera égale à $d_{\sigma_h} |1 - s|$.
Pour conserver les avantages d'une grande échelle d'analyse tout en limitant l'impact de la délocalisation, nous appliquons aux point détectés un algorithme de recentrage comme cela l'a déjà été suggéré entre autres par Deriche [56]. La détection est effectuée à une première échelle d'analyse assez grande σ_{h_0} (autant que le coût de calcul le permet), puis on applique aux points conservés l'algorithme de recentrage. Dans le domaine continu, si on faisait varier σ_h progressivement, chaque point se déplacerait de manière continue dans l'espace réduisant ainsi la délocalisation par rapport au point théorique. Le principe de la relocalisation est de poursuivre de manière discrète la position du maximum local spatial de la réponse du détecteur de Harris stabilisé en faisant varier l'échelle d'analyse du détecteur. On utilise pour cela une batterie de R_f filtres d'échelle d'analyse décroissante $\sigma_r = \sigma_{h_0} k^r$ avec $r \in [0, R_f - 1]$, et $k \in]0, 1[$.

Implémentation du détecteur de Harris

Nous avons réimplémenté notre version du détecteur de Harris afin d'accélérer les temps de calcul par rapport aux versions existantes. La propriété de séparabilité des filtres a tout d'abord été utilisée partout où cela était possible. Afin de profiter au maximum du cache mémoire de la machine, toutes les étapes du calcul sont effectuées successivement dans une boucle indépendante (calcul de la dérivée en x, calcul de la dérivée en y, mise au carré de la dérivée en x, mise au carré de la dérivée en y, calcul du produit des deux dérivées, filtrage gaussien horizontal, filtrage gaussien vertical, calcul de la valeur finale). Tous les filtres verticaux sont calculés en parcourant simultanément plusieurs lignes successives pour limiter le nombre de sorties de cache.
Cette implémentation a permis de diviser par 3 le temps de calcul. Sur un processeur pentium IV, 3,0 GHz avec 512 Ko de cache ; pour $\sigma_h = 1,6$ et des images de taille 352×288, le temps de calcul est d'environ 50 ms par image.

Expérimentations

Nous avons réalisé des expérimentations afin d'évaluer la répétabilité de notre version du détecteur de Harris après transformation de l'image. Ce type d'étude a déjà été réalisé dans de nombreux travaux [151, 153, 82] mais nous avons utilisé des transformations plus spécifiques à

la détection de copies (changement de gamma, changement de contraste, redimensionnement de l'image, ajout de bruit gaussien). Les résultats complets de ces expérimentations n'étant pas indispensables à la compréhension de ce chapitre descriptif, ils sont présentés en annexe B. Ils montrent clairement l'intérêt de la relocalisation des points d'intérêt : l'utilisation d'une échelle d'analyse de départ assez grande ($\sigma_h = 2,0$) au lieu d'une échelle faible ($\sigma_h = 0,8$) permet d'améliorer de 25 % la répétabilité à un ajout de bruit gaussien d'écart-type $20,0$ et de 10 % la répétabilité à un changement de gamma de facteur $0,6$. La relocalisation permet de conserver quasiment la même répétabilité au redimensionnement pour ces deux échelles d'analyse, alors qu'elle peut chuter de 25% pour l'échelle la plus grande lorsqu'il n'y a pas de relocalisation.

2.3.3 Caractérisation locale à l'aide de descripteurs différentiels

Description

Nous avons vu que l'étude comparative de Mikolajczyk et Schmid [126] concluait par la suprématie des descripteurs SIFT et des filtres différentiels orientables pour la caractérisation du signal autour de points d'intérêt. Celle-ci ayant cependant été publiée après le développement de notre système de détection de copies, nous n'avons pas initialement retenu ces descripteurs. Rappelons également que les transformations expérimentées ne sont pas les mêmes que dans notre contexte applicatif et qu'il faut donc relativiser ces conclusions.
Dans un souci d'adaptation de la caractérisation à l'information détectée, nous avons choisi d'utiliser des descripteurs différentiels qui semblent appropriés à l'information existant autour des points détectés par le détecteur de Harris, puisque celui-ci est basé sur une maximisation bi-directionnelle des dérivées premières. Nous avons vu précédemment qu'il était possible d'obtenir des invariants différentiels pour un certain nombre de transformations. Selon le principe évoqué dans le paragraphe 1.4.3 (page 23), il faut cependant prendre garde à ne pas rendre la signature invariante à des transformations inutiles, car cela risquerait d'être contre-performant en terme de discriminance. Notons également, que dans leur étude comparative de descripteurs, Mikolajczyk et Schmid [126] stipulent que les performances moins bonnes des descripteurs différentiels qu'ils utilisent sont probablement dûs à l'instabilité des dérivées d'ordre élevé, nécessaires à la construction d'invariants à la rotation ou aux transformations affines. L'invariance à ces deux transformations n'étant pas nécessaire dans notre contexte, nous pouvons nous passer des dérivées d'ordre élevé. Il existe également une solution théorique pour se rendre invariant à un changement d'échelle dans le domaine continu. La division par les dérivées premières alors mise en œuvre, s'avère cependant très instable dans la pratique ([150]) et cette solution n'a pas été retenue non plus.
La signature utilisée est basée uniquement sur les cinq premières composantes du jet local gaussien (dérivées stabilisées en les lissant par une gaussienne) :

$$\boldsymbol{S_I} = \left(S_I^1, S_I^2, S_I^3, S_I^4, S_I^5\right) = \left(\frac{\delta I}{\delta x}, \frac{\delta I}{\delta y}, \frac{\delta^2 I}{\delta x \delta y}, \frac{\delta^2 I}{\delta x^2}, \frac{\delta^2 I}{\delta y^2}\right)$$

Les dérivées gaussiennes utilisées, sont les polynômes de Hermite et dépendent donc de l'écart type σ_{sig} de la gaussienne déterminant l'échelle d'analyse. Toujours dans le souci d'adapter l'information caractérisée à l'information détectée, σ_{sig} doit être du même ordre de grandeur que σ_h, définissant l'échelle d'analyse du détecteur de Harris stabilisé.

Pour rendre la signature invariante aux changements de contraste (de la forme $I'(x,y) = kI(x,y)$), on divise les composantes de $\boldsymbol{S_I}$ par la norme L_2 du vecteur et on définit alors

$$\boldsymbol{S_{II}} = \frac{\boldsymbol{S_I}}{\|\boldsymbol{S_I}\|_{L_2}}$$

L'invariance aurait aussi pu être obtenue en divisant $\boldsymbol{S_I}$ par n'importe quelle combinaison linéaire des composantes S_I^j, ou encore par la norme du gradient ou plus généralement par n'importe quelle fonction $r(\boldsymbol{S_I}) : \Re^5 \rightarrow \Re$ ayant la propriété : $r(k \times \boldsymbol{S_I}) = k \times r(\boldsymbol{S_I})$. Quel que soit l'opérateur $r(\boldsymbol{S_I})$ utilisé, cette normalisation correspond à une perte de dimension puisque l'on projette l'espace de dimension $D = 5$ sur une hyper-surface de dimension $D = 4$ (une hyper-sphère dans le cas de $\boldsymbol{S_{II}}$, un hyper-plan si on avait divisé par une seule dérivée). L'utilisation de la norme L_2 permet d'être plus stable à la présence de faibles dérivées (une ou plusieurs composantes S_I^j proches de zéro) et permet également d'avoir toutes les composantes de $\boldsymbol{S_{II}}$ bornées. La perte d'une dimension doit être mise en relation avec la perte de discriminance qui accompagne en général un gain d'invariance.

L'invariance à un décalage de niveau de gris est directement obtenue par l'utilisation même d'opérateurs différentiels puisque

$$\delta\left(I(x,y) + a\right) = \delta\left(I(x,y)\right)$$

L'utilisation de dérivées gaussiennes a pour avantage de réaliser un filtrage passe-bas du signal et donc d'être robuste aux bruits hautes fréquences. Celles-ci permettent également une certaine tolérance à un décalage de quelques pixels (erreur de localisation du détecteur de Harris par exemple), à un redimensionnement léger, ou à un changement de gamma léger.

Afin d'augmenter la discriminance et d'utiliser l'information temporelle de la vidéo, la signature finale S définie pour chaque point d'intérêt est constituée de quatre signatures de type $\boldsymbol{S_{II}}$ extraites à des coordonnées et des instants différents autour du point d'intérêt. Ainsi pour un point d'intérêt détecté à l'instant t aux coordonnées (x,y) on définit la signature $\mathbf{S}$ de dimension $D = 4 \times 5 = 20$, comme un ensemble de sous-signatures $\boldsymbol{S_{II}}$ extraites à différents instants $t + t_i$:

$$\boldsymbol{S}(x,y,t) = \left\{\boldsymbol{S_{II}}\left(x + x_i, y + y_i, t + t_i\right)\right\}_{i\epsilon[1:4]}$$

Les décalages spatiaux (x_i, y_i) ont pour but de limiter la redondance des composantes $\boldsymbol{S_{II}}$ dans le cas d'un plan statique ou de points d'intérêt détectés sur un décor fixe. De la même manière, pour éviter de retrouver les mêmes vecteurs $\boldsymbol{S_{II}}$ aux différents instants $t + t_i$, dans le cas d'un plan panoramique, il faut éviter que la trajectoire définie par l'ensemble des points $\{(x + x_i, y + y_i, t + t_i)\}$ soit linéaire.

Les décalages spatiaux x_i et y_i et le décalage temporel temporel t_i doivent être suffisamment importants pour éviter la redondance entre les différentes composantes $\boldsymbol{S_{II}}$. En revanche, plus ces écarts seront importants et moins la signature sera tolérante aux redimensionnements. En effet, pour un redimensionnement d'un facteur k, la position théorique des points sera $(kx + kx_i, ky + ky_i)$, alors que la signature sera calculée aux coordonnées $(kx + x_i, ky + y_i)$. L'écart de position sera de $((k-1)x_i, (k-1)y_i)$ et donc proportionnel aux décalages x_i et y_i.

2.4 Décision globale sur la cohérence spatio-temporelle des signatures locales

Dans cette section, nous nous intéressons à la dernière étape de la méthode proposée pour la détection de copies, celle de la fusion des résultats individuels des signatures locales d'un même objet candidat. Cette stratégie est applicable aussi bien dans le cas d'images fixes que dans le cas de la vidéo, et nous avons essayé de rester le plus générique possible dans la description. Nous allons tout d'abord voir comment la robustesse et la discriminance peuvent être améliorées en ramenant cette étape à un problème particulier d'estimation de paramètres visant à recaler temporellement et spatialement l'objet candidat par rapport aux objets de référence. Nous décrirons ensuite l'équation de minimisation proposée pour résoudre cette estimation particulière ainsi que plusieurs techniques pour effectuer la minimisation. Nous discuterons enfin de la mesure de similarité finalement utilisée pour l'étape de décision proprement dite.

2.4.1 Du vote à l'estimation

Les signatures locales précédemment décrites étant des vecteurs de dimension fixe, il est possible de les indexer via un système d'indexation vectoriel multidimensionnel. Ainsi, lors de la phase *hors-ligne* de construction et d'indexation de la base de signatures, la signature locale va servir d'index pour la stratégie d'indexation que nous détaillerons dans la deuxième partie de ce mémoire. A chaque signature locale sont associées des données complémentaires qui sont également stockées dans la base. La première information est un identifiant Id, qui caractérise l'objet dans lequel les signatures ont été extraites (l'image dans le cas d'une détection d'images fixes ; un fichier, un extrait, une émission, etc. dans le cas de la vidéo). La deuxième information concerne la position du point d'intérêt dans l'objet : les coordonnées (x, y) du point dans l'image et le code temporel tc. Dans le cas de la vidéo, cette position peut être uniquement le code temporel ou le code temporel et les coordonnées du point. Dans le cas général, nous désignerons cette position par un unique vecteur $\boldsymbol{P}$, pouvant avoir une, deux ou trois dimensions suivant les informations conservées.
Ainsi, la base de signatures est constituée d'un ensemble de N signatures $\boldsymbol{S_n}$, auxquelles sont attachées un identifiant Id_n et un vecteur de position $\boldsymbol{P_n}$, pour $n \in [1, N]$.
Lors de la phase *en-ligne* de détection, les signatures locales sont extraites du flux d'objets candidats avant d'être recherchées dans la base via le système d'indexation. Soient $\boldsymbol{S_i}$ et $\boldsymbol{P_i}$ les signatures et les vecteurs position associés. Pour chaque signature $\boldsymbol{S_i}$, un ensemble de K_i signatures similaires notées $\boldsymbol{S_{ik}}$ pour $k \in [1, K_i]$ sont retrouvées dans la base. Pour chacune d'elle, le système retourne les données associées, à savoir Id_{ik} et $\boldsymbol{P_{ik}}$.
La phase de fusion des résultats consiste à prendre une décision sur un ensemble de signatures candidates $\{\boldsymbol{S_i}\}_{i\in[1,N_{cand}]}$ correspondant à un segment temporel du flux vidéo candidat (ou une image dans le cas fixe). Il s'agit de décider s'il existe effectivement une copie ou un original de ce segment dans le catalogue de référence, et, dans le cas positif, de déterminer quels sont les identifiants et les codes temporels correspondant. Cette décision est prise uniquement en utilisant les données Id_{ik} et $\boldsymbol{P_{ik}}$ pour $i \in [1, N_{cand}]$ et non les signatures elles mêmes. En pratique, la phase d'identification est réalisée avant la décision. Il faut d'abord identifier les différents objets présents dans les résultats et calculer ensuite une mesure de similarité pour chacun d'entre eux. Le score de cette mesure permet ensuite de décider quels sont les objets qui sont effectivement des copies ou des originaux de l'objet candidat.
Une première solution consiste a effectuer un simple vote pour chacun des identifiants présent dans $\{Id_{ik}\}_{i\in[i,N_{cand}],k\in[1,K_i]}$. C'est ce qu'utilisent Berrani et al. [20], dans leur méthode de dé-

tection de copies d'images fixes basée sur des signatures locales. Les identifiants retrouvés, sont triés selon le nombre de fois qu'ils sont présents dans les résultats et un test de Page-Hinkley est appliqué à cette liste de scores pour décider à partir de quel score les identifiants retrouvés ne sont plus des copies mais des fausses alarmes. L'avantage d'un vote est qu'il est totalement invariant aux transformations géométriques. L'inconvénient majeur est qu'il ne tient compte d'aucune information sur la position relative des points d'intérêt. L'information géométrique est ainsi perdue augmentant fortement le risque de fausses alarmes.
Pour tenir compte de la position relative des points d'intérêt, il faut mesurer la cohérence géométrique de l'ensemble des points d'intérêt de l'objet candidat avec l'ensemble des points d'intérêt de chaque objet de référence présent dans les résultats. Cette cohérence géométrique doit être invariante aux transformations géométriques prises en compte. Une manière consiste à modéliser les transformations géométriques possibles et d'estimer les paramètres de ce modèle pour chaque objet de référence présent dans les résultats. On suppose alors que les transformations géométriques sont globales à tout l'objet candidat et donc communes à toutes les signatures. Une fois les paramètres de transformation estimés pour chaque objet, la mesure de similarité est alors calculée par un vote sur le nombre d'appariements de signatures qui respectent effectivement la transformation estimée.

Le problème de l'estimation des transformations géométriques se rapproche de la problématique plus générale du recalage d'images. Étant données deux images d'une même scène avec des points de vue différents, il s'agit de trouver la transformation géométrique entre les deux images. Ces techniques sont également beaucoup utilisées en estimation de mouvement dans les vidéos. Un état de l'art récent et complet du domaine a été réalisé par Zitova et al. [181]. Parmi les techniques les plus utilisées, on peut citer la transformée de Hough [97], l'algorithme *Iterative Closest Point* [21] ou encore l'algorithme *Random Sample Consensus* [70]. Une autre technique, le hachage géométrique permet de recaler des images candidates par rapport à une base d'images de référence. Elles utilisent souvent des points d'intérêt [174]. La position de tous les points d'intérêt d'une image est recalculée dans différents espaces définis par tous les couples possibles de points d'intérêt dans l'image. Un couple de points d'intérêt permet en effet de définir un repère dans lequel la position des autres points est invariante à une rotation ou à un changement d'échelle de l'image. Toutes ces positions sont ensuite indexées via une table de hachage. La phase de recherche consiste alors à calculer la position des points d'intérêt de l'image candidate dans les différents espaces et à interroger la table de hachage. Cette phase de recherche peut être consolidée par l'utilisation de signatures stockées comme données complémentaires dans la table. Un hachage géométrique peut donc être vu comme une méthode inverse de notre approche : la position des points sert d'index alors que la signature est une donnée annexe qui permet de rendre le recalage plus robuste. Dans notre cas, c'est la signature qui est indexée et le recalage est effectué grâce aux données annexes après le recherche dans la base. Un hachage géométrique est en effet difficilement envisageable pour notre problématique car l'utilisation de tous les couples de points d'intérêt conduirait à des tailles de base vertigineuses.

Dans notre application, nous limitons les transformations géométriques à la translation et aux changements d'échelles. Nous laissons volontairement de côté la rotation et les transformations affines ou projectives, celles-ci étant très rares dans le cadre de la détection de copies. On peut

donc se contenter du modèle général suivant :

$$\boldsymbol{P'} = \boldsymbol{AP} + \boldsymbol{B} \Longleftrightarrow \begin{pmatrix} x' \\ y' \\ t' \end{pmatrix} = \begin{pmatrix} a_x & 0 & 0 \\ 0 & a_y & 0 \\ 0 & 0 & a_t \end{pmatrix} \begin{pmatrix} x \\ y \\ t \end{pmatrix} + \begin{pmatrix} b_x \\ b_y \\ b_t \end{pmatrix} \tag{2.1}$$

où la position $\boldsymbol{P'}$ correspond à la position $\boldsymbol{P}$ après transformation. La matrice $\boldsymbol{A}$ contient les coefficients d'échelle tandis que le vecteur $\boldsymbol{B}$ contient les paramètres de translation. En ce qui concerne la coordonnée temporelle t, le changement d'échelle correspond à un ralenti ou un accéléré. La translation temporelle est systématique puisque les codes temporels du flux vidéo candidat ne sont pas les mêmes que les codes temporels des extraits vidéos référencés. Ce modèle est général et peut être en pratique encore plus simple. Dans le cas d'images fixes par exemple, il faut exclure la coordonnée temporelle. Dans le cas de la vidéo on peut vouloir diminuer les coûts de l'estimation en se limitant à la coordonnée temporelle. Si l'on considère que les ralentis et les accélérés sont très rares pour l'application envisagée, on peut fixer le paramètre a_t à 1.

2.4.2 Estimation

La première étape de notre méthode consiste à lister les différents identifiants présents dans l'ensemble des résultats de la recherche $\{Id_{ik}\}$. L'estimation des paramètres $\boldsymbol{A}$ et $\boldsymbol{B}$ est ensuite réalisée pour chaque identifiant id présent dans l'ensemble des résultats. L'estimation proposée est définie par l'équation suivante :

$$\left(\hat{\boldsymbol{A}}(id), \hat{\boldsymbol{B}}(id)\right) = \underset{\boldsymbol{A},\boldsymbol{B}}{\arg\min} \left(\sum_{i=1}^{N_{cand}} \min_{\substack{k \in K_i \\ Id_{ik}=id}} \rho\left(\|\boldsymbol{P_{ik}} - (\boldsymbol{AP_i} + \boldsymbol{B})\|\right) \right) \tag{2.2}$$

où $\rho(u)$ est une fonction de coût croissante lorsque u croit et $\|.\|$ est la fonction résiduelle, restreinte ici à l'ensemble des normes L_p.
La minimisation s'effectue uniquement sur les vecteurs position ayant le même identifiant ($Id_{ik} = id$). L'originalité de cette fonction de minimisation, comparée à une estimation classique est de ne conserver que le résidu minimum des signatures voisines de chaque signature candidate. Parmi les K_i signatures les plus proches de chaque signature candidate $\boldsymbol{S_i}$, seule celle ayant le résidu le plus faible contribue à la fonction de coût ($\min_{k \in K_i}$). L'avantage est de rendre l'estimation plus robuste et plus précise dans le cas de signatures redondantes spatialement et temporellement. La redondance correspond à des points d'intérêt d'un même objet ayant des signatures similaires mais des positions spatiales et/ou temporelles différentes. Parmi l'ensemble des $\boldsymbol{S_{ik}}$ signatures retrouvées pour une signature candidate $\boldsymbol{S_i}$, plusieurs auront alors le même identifiant mais des vecteurs positions $\boldsymbol{P_{ik}}$ différents, dont la plupart seront erronés. Si les résidus de tous ces points contribuaient à la fonction de coût, la solution serait biaisée, c'est pourquoi seul le résidu ayant le coût le plus faible, est conservé.

Cette estimation étant effectuée pour tous les identifiants présents dans les résultats, plusieurs copies (ou originaux) d'un même objet candidat peuvent être détectés. Cette particularité de notre système est très intéressante puisqu'elle permet de lier plusieurs séquences du catalogue de référence entre elles, en plus de les lier à la séquence candidate. En revanche s'il existe plusieurs copies d'un objet candidat à l'intérieur d'un même objet de référence (par exemple deux extraits identiques dans un même fichier vidéo du catalogue de référence), seul un objet sera détecté. Pour

résoudre ce problème, il faudrait estimer simultanément plusieurs modèles, avec un algorithme EM par exemple (Expectation Maximisation [171]). Cependant ce cas est bien moins fréquent que celui de plusieurs copies avec des identifiants différents.

Notre application entre typiquement dans le cadre de l'estimation robuste et le choix de la fonction de coût $\rho(u)$ doit tenir compte de la présence d'exceptions dans le bruit des observations. Une simple minimisation de l'erreur quadratique ($\rho(u) = u^2$) n'est en effet optimale que dans le cas d'un bruit gaussien sur les observations. Lorsque certaines observations incorrectes isolées ont un comportement tout à fait différent du modèle, leur contribution à l'erreur quadratique globale est en effet trop importante et la solution obtenue peut être fortement biaisée. Pour remédier à cela, on utilise un M-estimateur [160, 24] qui diminue la contribution des points les plus éloignés de la solution testée. Dans notre cas, les exceptions peuvent être dues à plusieurs causes. Elles peuvent provenir des fausses alarmes du détecteur de points d'intérêt, ou bien de signatures similaires dans un même objet de référence. Le M-estimateur que nous avons utilisé est le bi-poids de Tukey. C'est un estimateur descendant, c'est-à-dire que la contribution des exceptions tend vers zéro lorsque le résidu augmente. Il est défini par l'équation suivante :

$$\rho_r(u) = \begin{cases} \frac{r^4 u^2}{2} - \frac{r^2 u^4}{2} + \frac{u^6}{6} & si \ |u| < r \\ \frac{r^6}{6} & si \ |u| \geq r \end{cases} \tag{2.3}$$

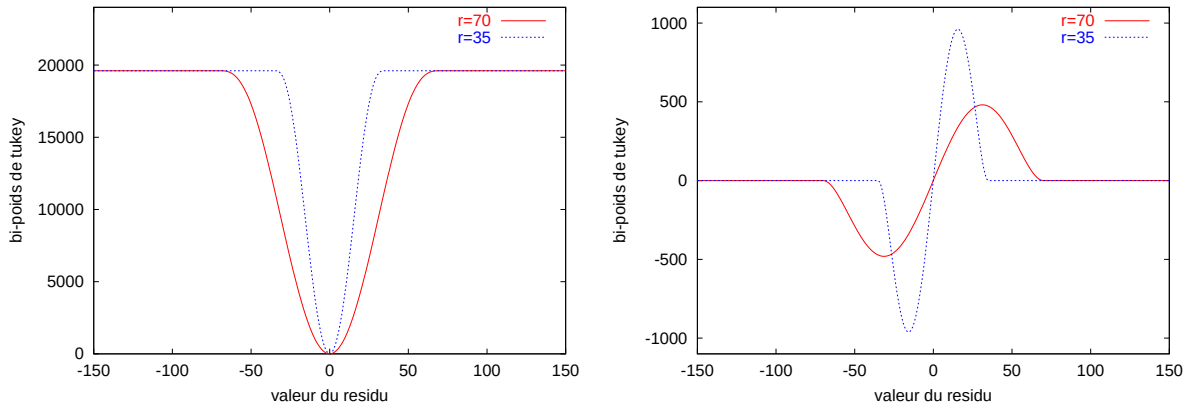


FIG. 2.3 – M-estimateur de Tukey *à gauche* : bi-poids de Tukey *à droite* : dérivée du bi-poids de Tukey

L'analyse du comportement d'un estimateur peut être faite en regardant sa fonction d'influence, c'est-à-dire la dérivée de l'estimateur. Celle-ci caractérise le biais qu'une observation particulière entraîne dans la solution. La figure 2.3 représente le bi-poids de Tukey et sa fonction d'influence pour deux valeurs du paramètre r. On constate que la contribution des résidus supérieurs à r est nulle. Le paramètre r permet donc de régler le seuil de rejet des exceptions.

2.4.3 Minimisation

En pratique, il existe plusieurs méthodes pour approximer la solution idéale décrite par l'équation 2.2.

Exploration de l'espace des paramètres

La première solution consiste à parcourir l'espace des paramètres. Cela peut être fait en quantifiant l'espace des paramètres et en calculant systématiquement la fonction de coût globale pour chaque cellule. Une autre manière moins coûteuse pour explorer l'espace des paramètres, est de calculer la fonction de coût globale pour l'ensemble des transformations défini par toutes les combinaisons de k_{cand} observations, k_{cand} étant le nombre minimum d'observations nécessaires pour définir entièrement un jeu de paramètres. Le modèle des transformations géométriques utilisé (cf. equation 2.1) étant séparable pour les coordonnées x, y et t, il suffit de deux observations pour définir la totalité des six paramètres $(a_x, a_y, a_t, b_x, b_y, b_t)$. Si on se limite aux translations dans le modèle des transformations, une seule observation est nécessaire pour définir un jeu de paramètres.

Autres méthodes de minimisation

Les méthodes de descente de gradient ou de minimisation stochastique traditionnellement utilisées pour les problèmes de minimisation ne sont pas directement applicables à notre fonction globale de coût (équation 2.2). L'introduction de la fonction *min* sur les résidus des signatures voisines rend en effet la fonction de coût globale non continue et ne permet pas de calculer simplement les variations de la fonction suite à une perturbation des paramètres. Pour remédier à cela on peut utiliser une méthode itérative basée sur la minimisation suivante, à laquelle on peut appliquer les algorithmes classiques :

$$\left(\hat{\boldsymbol{A}}(id), \hat{\boldsymbol{B}}(id)\right) = \underset{\boldsymbol{A},\boldsymbol{B}}{\arg\min} \left(\sum_{i=1}^{N_{cand}} \sum_{k=1}^{K_i} \delta(Id_{ik}, id) \rho \left(\| \boldsymbol{P_{ik}} - (\boldsymbol{A P_i} + \boldsymbol{B}) \| \right) \right) \tag{2.4}$$

avec $\delta(u,v) = \begin{cases} 1 \; si \; u = v \\ 0 \; si \; u \neq v \end{cases}$

Lors de cette minimisation, la totalité des résidus des plus proches signatures contribuent à la solution avec les problèmes déjà évoqués que cela engendre en termes de précision et de robustesse de la solution. En revanche, on peut se rapprocher de la minimisation idéale (équation 2.2) en appliquant plusieurs itérations. L'idée est de supprimer de l'ensemble des observations, après chaque itération, les positions ayant les résidus maximum. Parmi les K_i vecteurs de position initiaux, pour chaque signature $\boldsymbol{S_i}$, la deuxième itération sera effectuée uniquement sur les $(K_i - 1)$ vecteurs ayant les résidus les plus faibles. Ce processus peut ensuite être réitéré jusqu'à ce que $K_i = 1 \;\; \forall i \in [1, N_{cand}]$.

2.4.4 Mesure de similarité

Une fois l'estimation de la transformation réalisée pour tous les identifiants présents dans les résultats, il faut calculer une mesure de similarité et décider pour chacun d'entre eux s'il s'agit d'une copie ou non. Signalons ici que, pour limiter le coût des estimations, on peut faire une présélection des identifiants en éliminant ceux ayant un score de présence vraiment trop faibles. La première solution la plus évidente pour la mesure de similarité est de prendre la valeur de la fonction de coût globale minimisée lors de l'étape d'estimation. Si l'on préfère avoir une mesure plus lisible, on peut également se ramener à un vote, en comptant le nombre de signatures candidates qui ont effectivement une de leurs signatures les plus proches qui respecte le modèle estimé. Ce comptage doit être fait à une erreur de position près, typiquement du même ordre

de grandeur que le seuil du M-estimateur (paramètre r du bi-poids de Tukey). Cette mesure de similarité sera notée n_{sol} (sol pour solution) :

$$n_{sol}(id) = \sum_{i=1}^{N_{cand}} \sum_{k=1}^{K_i} \delta(Id_{ik}, id)\ \delta_r(\boldsymbol{P_{ik}}, \hat{\boldsymbol{A}}(id)\boldsymbol{P_i} + \hat{\boldsymbol{B}}(id)) \tag{2.5}$$

avec

$$\delta(u,v) = \begin{cases} 1\ si\ u = v \\ 0\ si\ u \neq v \end{cases} \qquad \delta_r(u,v) = \begin{cases} 1\ si\ |u-v| \leq r \\ 0\ si\ |u-v| > r \end{cases}$$

Elle représente le nombre de points d'intérêt qui ont effectivement contribué au recalage de l'objet candidat par rapport à l'objet de référence. Dans le cas d'une copie exacte, elle est théoriquement égale au nombre de points d'intérêt contenus dans l'objet. Dans le cas d'un objet candidat n'appartenant pas à la base, elle sera très faible pour tous les objets de référence présents dans les résultats de la recherche, puisqu'il n'y a théoriquement pas de cohérence spatio-temporelle entre les points d'intérêts candidats et ceux des objets de référence.

2.4.5 Mise en œuvre dans le cadre de la détection d'archives télévisées

Le système complet de monitorage devant être rapidement fonctionnel pour un grand nombre de chaînes et des catalogues de référence très volumineux, la méthode de fusion des résultats locaux est utilisée de manière assez simple dans le système actuellement exploité à l'INA. La construction de la base de signatures pour des catalogues importants est un processus très long bien qu'effectué de manière continue sur plusieurs machines. Les débits réseaux, les débits disque, le temps de calcul des descripteurs, les arrêts sur coupure électrique ou maintenance du réseau sont autant de contraintes qui ralentissent l'indexation. Il a fallu ainsi environ 7 mois pour construire une base de signatures correspondant à 20 000 heures de vidéo. Dans ces conditions, il était difficile de changer la technologie en cours de route.
Le choix initial avait été de minimiser le plus possible les coûts de stockage des signatures et des données associées. C'est pourquoi seuls l'identifiant et le code temporel des signatures ont été conservés. Le modèle de transformation utilisé ne tient donc compte que de la coordonnée temporelle et pas des positions spatiales. Les ralentis et les accélérés ont été également laissés de côté pour éviter de déstabiliser l'estimation, alors que ce type de transformation est assez rare dans notre contexte applicatif. **L'estimation consiste donc uniquement en un recalage temporel entre la séquence candidate et les séquences de référence** (seul le paramètre du décalage temporel b_t est estimé). Le gain de discriminance dû à cette contrainte temporelle, par rapport à un simple vote, est cependant déjà très significatif et donne de très bons résultats comme nous le verrons dans la partie III de ce mémoire.
La méthode de minimisation utilisée est la deuxième méthode d'exploration de l'espace des paramètres (cf. section 2.4.3). Le paramètre b_t est calculé pour chaque observation (c'est à dire pour chaque couple de code temporel candidat et de code temporel retourné par la recherche) et la fonction de coût globale est calculée pour chacune de ces valeurs. Dans l'espace des codes temporels candidats en fonction des codes temporels de référence, cela revient en fait à estimer, pour chaque identifiant représenté dans les résultats de la recherche, la droite ayant la fonction de coût la plus faible parmi toutes celles passant par au moins un point.

Un objet candidat est défini par un ensemble de 9 images clé, soit en moyenne environ $N_{cand} \approx 170$ signatures pour une durée moyenne de $11,7$ secondes.
Pour rendre la détection plus robuste aux redécoupages temporels, les objets sont définis de

manière glissante et non pas en découpant le flux vidéo candidat en tronçons fixes. Pour chaque nouvelle image clé, un *buffer* tournant contenant les résultats de la recherche des images clé précédentes est mis à jour et l'estimation du recalage temporel est recalculé pour tous les identifiants. Pour chaque image clé, on dispose ainsi d'un ensemble d'identifiants affectés d'un code temporel et d'une mesure de similarité dont le seuillage détermine s'il s'agit ou non de copies de l'objet candidat.
En poursuivant les solutions d'image clé en image clé, on peut ensuite déterminer les codes temporels de début et de fin d'une détection pour rendre les résultats plus exploitables qu'une simple liste d'images clé identifiées.

2.5 Synthèse

Ce chapitre a été consacré à la description de la méthode proposée pour la traçabilité d'un catalogue de séquences vidéo. La première originalité par rapport aux autres méthodes de détection de copies proposées dans la littérature est que cette méthode est basée sur des signatures locales et non sur une signature globale caractérisant chaque objet du catalogue de référence. Nous avons également justifié le choix du détecteur de Harris et de la signature proposée en tenant compte des contraintes applicatives liées au contexte de monitorage temps réel d'archives télévisées. L'autre originalité est la méthode proposée pour la fusion des résultats locaux de la recherche dans la base. Cette étape a été traitée comme un problème de recalage et non comme un simple vote. L'équation de minimisation proposée pour résoudre l'estimation des paramètres de recalage prend en compte la nature particulière des observations. Celles-ci sont en effet issues de la recherche des signatures dans la base et ne peuvent pas être traitées comme un ensemble d'observations homogènes. Ainsi, parmi toutes les signatures de la base similaires à une signature candidate, seule celle ayant le résidu le plus faible contribue à la fonction de coût de l'estimation. Dans la partie suivante, nous allons présenter la méthode proposée pour la recherche des signatures similaires dans la base.

Deuxieme partie

Recherche des signatures similaires dans la base

Chapitre 3

État de l'art de la recherche de descripteurs par similarité

3.1 De la similarité des images à la recherche de descripteurs

L'objectif d'un système de recherche par le contenu est de retrouver des objets par le biais de leurs descripteurs. Le principe est de retrouver dans la base des descripteurs, ceux qui sont similaires à un descripteur candidat, selon une certaine mesure de similarité. L'approche naïve consiste à calculer, pour chaque descripteur candidat, la mesure de similarité avec tous les descripteurs de la base. On parlera alors d'une *recherche séquentielle* ou d'une *recherche exhaustive*. La complexité de calcul d'une telle méthode est alors proportionnelle au nombre N de descripteurs dans la base ($O(N)$).
Lorsque les volumes des catalogues de référence sont importants, une recherche exhaustive devient beaucoup trop coûteuse et on fait alors appel à des méthodes issues du domaine des bases de données pour tenter d'accélérer les interrogations. Le principe général, est de limiter le nombre de descripteurs à comparer avec le descripteur candidat en éliminant des paquets entiers de descripteurs par des règles de filtrage.
Dans la suite de cet état de l'art, nous reviendrons plus en détail sur les différentes méthodes de recherche de descripteurs similaires dans une base. Cette première section a pour objectif de préciser quelques notions préliminaires à notre étude et de donner une vue d'ensemble du chapitre.

3.1.1 Les requêtes

Les deux requêtes les plus couramment utilisées pour la recherche de descripteurs similaires sont la recherche de K-plus proches voisins et la recherche à un rayon près. Nous en donnons ici les définitions respectives :

Définition K-plus proches voisins - *Soit $\Delta = \{\boldsymbol{D_n}\}_{n\in[1,N]}$ une base de N descripteurs et $d : \Delta \times \Delta \rightarrow \Re$ une mesure de disimilarité associée. Les **K-plus proches voisins** d'un descripteur candidat $\boldsymbol{D_{cand}}$ sont un sous ensemble de Δ, noté $\Delta_{pp} = \{\boldsymbol{D_k}\}_{k\in[1,K]}$ pour lequel :*

$$d(\boldsymbol{D_k}, \boldsymbol{D_{cand}}) \leq d(\boldsymbol{D_n}, \boldsymbol{D_{cand}}),\ \forall k \in [1, K],\ \forall \boldsymbol{D_n} \in \Delta \setminus \Delta_{pp}$$

Définition Recherche à ϵ-près - *Soit $\Delta = \{\boldsymbol{D_n}\}_{n\in[1,N]}$ une base de N descripteurs et $d : \Delta \times \Delta \rightarrow \Re$ une mesure de disimilarité associée. Le résultat de la recherche à ϵ-près d'un descripteur*

candidat $\boldsymbol{D_{cand}}$*, est un sous-ensemble de* Δ*, noté* Δ_ϵ *défini par :*

$$\Delta_\epsilon = \{\boldsymbol{D_n} \mid d(\boldsymbol{D_n}, \boldsymbol{D_{cand}}) \leq \epsilon\}_{n\in[1,N]}$$

Définition Recherche par similarité *On parlera de recherche par similarité lorsqu'il s'agit d'une recherche de* K*-plus proches voisins ou d'une recherche à* ϵ*-près.*

Dans le cadre de la recherche d'objets par le contenu, la recherche des K-plus proches voisins est plus souvent utilisée que la recherche à un rayon près bien que les deux approches possèdent des avantages et des inconvénients. Lorsqu'un descripteur global unique est utilisé, les K-plus proches voisins d'un descripteur candidat correspondent directement aux K-plus proches objets de l'objet candidat. Dans le cas d'un système de recherche d'images par le contenu cela garantit à l'utilisateur de pouvoir systématiquement visualiser K images. Certaines de ces réponses peuvent cependant être très éloignées de la requête et ne pas être pertinentes.
La recherche à ϵ-près permet de palier ce problème, en seuillant la valeur de la mesure de disimilarité à partir de laquelle on considère que les descripteurs ne sont plus similaires. Le nombre de réponses est en revanche variable, ce qui peut poser des problèmes dans les cas extrêmes. Pour fixer la valeur de ϵ, il est nécessaire d'avoir une certaine connaissance de la distribution des descripteurs.
Dans le cas des descripteurs locaux, la mesure de similarité finale est prise sur un ensemble de candidats recherchés individuellement dans la base. La question de la présentation des résultats à un utilisateur n'est donc plus directement liée à la requête soumise au système d'indexation. Le choix de la requête et la pertinence des résultats est dans ce cas uniquement guidé par l'étape finale fusionnant les résultats individuels.
Le type de requête n'est pas seulement important pour la pertinence des résultats retournés, il l'est également vis-à-vis des performances du système. L'évolution du temps de recherche en fonction de la taille de la base ou de la dimension des descripteurs n'est pas la même pour une requête de type K-plus proches voisins et pour une requête à ϵ-près [83].
Nous reviendrons plus en détail sur l'influence de la nature des requêtes dans le chapitre suivant. Nous analyserons leur pertinence dans le cadre de nos signatures locales, avant de décrire les requêtes statistiques proposées (cf. section 4.2.2).

3.1.2 La nature des descripteurs

Il est possible de classer les méthode de recherche suivant la nature des descripteurs et la mesure de disimilarité associée à la requête. Nous complétons ici la classification proposée par Ciaccia et al. [47], par ordre croissant d'applicabilité :

1. VS_{L_p} **(Espace vectoriel, Norme** L_p**)** : Il s'agit des méthodes qui ne s'appliquent qu'à des espaces vectoriels et en utilisant une distance L_p [77, 17],. Tous les objets sont des vecteurs de même dimension D. Si on instancie p, on obtient des classes encore plus restrictives. La classe VS_{L_1} correspond, par exemple, aux structures ne permettant d'indexer que des espaces munis de la distance L_1.
2. VS **(Espace vectoriel)** : Il s'agit des méthodes ne s'appliquant qu'aux espaces vectoriels [84, 99]. Les objets sont ici encore des vecteurs de dimension fixe et l'indexation se fait en utilisant explicitement les coordonnées de ces vecteurs. La mesure de similarité peut en revanche être n'importe quelle distance.
3. MS **(Espace métrique)** : Il s'agit des structures d'indexation s'appliquant à tout ensemble d'objets muni d'une distance. Les objets n'ont pas forcément une représentation

vectorielle. On peut aussi indexer des vecteurs ayant des dimensions variables, ou bien encore des chaînes de caractères. Deux types d'approches sont utilisées pour ce type d'indexation. La première consiste à utiliser les propriétés de définition d'une distance pour construire un arbre dont les branches sont parcourues pendant le traitement d'une requête. La deuxième solution consiste à transformer l'espace métrique en un espace vectoriel puis à utiliser une structure d'indexation de type VS. Cette approche est parfois appelée GEMINI (*GEneric Multimedia INdexIng*) [64, 44].
Contrairement aux méthodes de la classe VS, les structures de type MS n'utilisent que les distances et non les coordonnées des points dans l'espace. L'information utilisée est donc beaucoup moins riche et les structures résultantes sont en général moins performantes [27, 47]. Lorsque l'espace des descripteurs est réellement un espace vectoriel, il est donc préférable d'utiliser une structure d'indexation vectorielle bien que toutes les méthodes métriques soient applicables.

4. NMS **(Espace non métrique)** : Il s'agit des structures permettant d'indexer des ensembles d'objets munis d'une mesure de disimilarité qui n'est pas forcément une distance [78]. En général, ce type de structure est développé spécifiquement pour une mesure particulière, pour laquelle aucune autre structure n'est applicable.

3.1.3 Les différentes méthodes

Historiquement, la première solution retenue pour effectuer des recherches par similarité dans des bases de descripteurs, a été l'utilisation de structures d'indexation multidimensionnelle. Elles ont été proposées pour permettre un accès et une interrogation efficace dans des bases de données multidimensionnelles stockées sur disque. Elles couvrent un domaine beaucoup plus large que la seule recherche de descripteurs par similarité dont l'apparition est plus tardive. Il s'agit de structures d'indexation complètes et dynamiques, qui en plus des opérations de recherche permettent des opérations d'insertion ou de suppression d'éléments *en-ligne* (en général en un temps de complexité $O(log(N))$. L'ensemble des requêtes envisagées peut également être plus vaste (requête par point, par fenêtre rectangulaire, etc.). Elles reposent sur le même principe que les index monodimensionnels tels que le B-tree [9]. Nous revenons plus en détail sur ces structures d'indexation multidimensionnelle dans la section 3.2.
Les limites des ces structures d'indexation, en particulier la dégradation des performances pour les descripteurs de dimension élevée ont conduit à l'émergence de nouvelles approches. Les effets non intuitifs observables dans les espaces de dimension élevée, ont soulevé des problèmes dans la plupart des domaines manipulant des données multidimensionnelles. Cette problématique est connue depuis les années 60 sous l'appellation de *malédiction de la dimension*. Il aura, en revanche, fallu l'apparition de quelques travaux expérimentaux récents pour que des méthodes alternatives aux structures d'indexation classiques soient proposées. Le travail de référence est celui de Weber et al. [170] qui a montré que les performances de toutes les structures d'indexation de l'époque se dégradaient de manière exponentielle lorsque la dimension augmente. Elles deviennent plus mauvaises qu'une recherche séquentielle pour les dimensions supérieures à $D = 16$.
La première solution envisagée pour traiter les descripteurs de dimension élevée a été d'améliorer les performances de la recherche séquentielle par des méthodes de compression des descripteurs (section 3.3). Un deuxième axe d'investigations très fécond pour la recherche par similarité dans des espaces de grande dimension est l'utilisation de méthodes de recherche approximative. Il a en effet été montré qu'une faible dégradation de la qualité des résultats pouvait conduire à des gains conséquents pour le temps de recherche (section 3.4). Cet état de l'art a été en partie guidé par les travaux de Berrani [19], de Böhm et al. [27] et de Ciaccia et al. [47].

3.2 Structures d'indexation multidimensionnelle

3.2.1 Les structures d'indexation vectorielle *(classe VS ou VS_{L_p})*

L'utilisation des structures d'indexation vectorielle est recommandée lorsque la base atteint une taille importante et que les données doivent impérativement être stockées sur disque.
Pour une description détaillée de la plupart des structures d'indexation vectorielle proposées dans la littérature depuis les années 80, le lecteur pourra se référer à l'état de l'art constitué par Gaede et Günther [76]. Cependant, cette revue ne s'intéresse pas particulièrement aux bases de données multimédia et à la particularité des requêtes par similarité propres à la recherche de descripteurs. Le problème de la recherche des K-plus proches voisins n'y est par exemple pas abordé. De plus, la plupart des structures d'indexation multidimensionnelle qui y sont détaillées ont été utilisées pour des espaces de dimension relativement faible, typiquement inférieures à 5. Leur première utilisation était destinée à la gestion d'informations géographiques bi- ou tri-dimensionnelles. Or, les effets de la *malédiction de la dimension* ne commencent à être vraiment significatifs qu'à partir des dimensions supérieures. Pour un état de l'art plus récent, s'intéressant plus particulièrement aux bases multimédia et aux espaces de descripteurs de dimension plus élevée, le lecteur pourra se référer à l'état de l'art dressé par Böhm et al. [27].

Principe des structures d'indexation vectorielle

Les méthodes d'indexation vectorielle sont basées sur le principe du regroupement hiérarchique des données de l'espace. Structurellement, ils ressemblent à un arbre de type B-tree [50] : les vecteurs sont stockés dans les feuilles d'un arbre, de manière à ce que les vecteurs qui sont proches dans l'espace des descripteurs résident dans une même feuille. Chaque vecteur est stocké uniquement dans une feuille et il n'y a pas de duplication de vecteurs. Les feuilles contenant les vecteurs sont organisées par une arborescence structurée. Chaque nœud contient un ensemble de clés et de pointeurs vers un sous-arbre ou une feuille. Chaque clé correspond à un codage d'une région de l'espace multidimensionnel dépendant de la structure utilisée (par analogie, dans un B-tree, la clé correspond à un intervalle). Toutes les clés contenues dans un sous-arbre correspondent à des régions de l'espace incluses dans la région correspondant à la clé d'entrée du sous arbre. Dans un R-tree [84] par exemple, la clé correspond à un hyper-rectangle englobant toutes les clés et tous les vecteurs du sous-arbre ; dans un SS-tree [172], il s'agit d'une hyper-sphère. Les régions sont ainsi également appelées formes englobantes.
Le fait de regrouper les vecteurs d'une même région de l'espace dans une même feuille permet de traiter la recherche en deux étapes. Une première étape consiste à déterminer les feuilles qui sont susceptibles de contenir des vecteurs répondant à la requête. La deuxième étape consiste à calculer la distance entre la requête et les vecteurs contenus dans les pages sélectionnées. Ceci permet de réduire sensiblement le nombre de vecteurs qui doivent effectivement être comparés à la requête, réduisant ainsi le coût CPU de calcul des distances ainsi que le nombre d'entrées/sorties.
Identiquement au B-tree, un des objectifs de l'indexation vectorielle est que l'arbre soit équilibré, c'est-à-dire que la profondeur de l'arbre (longueur du chemin entre la racine et les feuilles) soit constante pour toutes les feuilles. Ceci permet de garantir que le nombre d'opérations pour effectuer un accès soit de l'ordre de $O(log(N))$. La capacité des feuilles et des nœuds est un paramètre important des structures vectorielles. Pour permettre des opérations dynamiques d'insertion et

de suppression de vecteurs, et de mise à jour des clés, les feuilles et les nœuds ne doivent pas être pleins. Lorsque l'insertion d'un vecteur provoque un remplissage trop élevé d'une feuille (ou qu'une suppression provoque un remplissage trop faible), certaines feuilles doivent être découpées (ou fusionnées) et les nœuds doivent alors être mis à jour pour conserver l'équilibre de l'arbre.
Originellement, chaque feuille de l'arbre correspondait à une page sur le disque. Dans les systèmes modernes, la taille réelle des pages du disque est considérée comme un détail *hardware* caché pour les programmeurs et les utilisateurs. La lecture consécutive des données sur le disque est cependant toujours plus efficace qu'un accès aléatoire. Dans les implémentations modernes, les structures d'indexation utilisent une sorte de pagination artificielle en déterminant une taille fixe de données consécutives pour les nœuds et les feuilles (typiquement autour d'une centaine de kilo-octets).
Il existe trois catégories de structures d'indexation vectorielle : celles basées sur le partitionnement des données, celles basées sur le partitionnement de l'espace et celles basées sur une courbe remplissant l'espace. Avant de décrire plus en détail ces trois types d'approches, nous décrivons les algorithmes de recherche communs à ces approches.

Les algorithmes de recherche

Les algorithmes de recherche sont différents suivant qu'il s'agit d'une recherche de K-plus proches voisins ou une recherche à ϵ-près.

1. Pour traiter les requêtes à ϵ-près, il suffit de pouvoir traiter efficacement les deux tests suivants :
   - Est-ce que la région correspondant à une clé donnée possède une intersection avec la région correspondant à la requête ?
   - Est-ce qu'un vecteur donné appartient à la région correspondant à la requête ?

   L'algorithme consiste alors à tester hiérarchiquement les clés aux différents niveaux de l'arbre avec le premier test, pour éliminer les branches n'ayant pas d'intersection avec la requête. La deuxième étape consiste à tester toutes les feuilles résultantes avec le deuxième test. Ce type d'algorithme est théoriquement applicable à toute distance tant que les deux tests peuvent être résolus de manière efficace. Le premier test n'est cependant pas toujours trivial et le type de distance exploitable dépend en général des régions utilisées par la structure d'indexation.

2. Pour expliquer les algorithmes de recherche des K-plus proches voisins, il est nécessaire de définir quelques notions importantes :

   Définition MinDist et MaxDist - *Soit un vecteur requête $\boldsymbol{Q}$ dans un espace vectoriel E de dimension D. Soit R une région de E dépendant de la structure utilisée et caractérisée par sa clé dans l'arborescence. Soit $d : E \times E \rightarrow \Re$, la distance considérée pour la recherche des K-plus proches voisins de $\boldsymbol{Q}$. Alors,*

   $$MinDist(\boldsymbol{Q}, R) = \min_{\boldsymbol{X} \in R} d(\boldsymbol{X}, \boldsymbol{Q})$$

   $$MaxDist(\boldsymbol{Q}, R) = \max_{\boldsymbol{X} \in R} d(\boldsymbol{X}, \boldsymbol{Q})$$

   Les algorithmes de recherche des K-plus proches voisins sont basés sur des règles de filtrage permettant d'éliminer directement des branches de l'arbre, lorsqu'il est impossible que la

région correspondante puisse contenir des points plus proches que la solution actuelle.
La première règle de filtrage stipule que si, pour deux régions R_1 et R_2,

$$MaxDist(\boldsymbol{Q}, R_1) \leq MinDist(\boldsymbol{Q}, R_2)$$

alors R_2 peut être éliminée. Cette règle suppose qu'il y ait au moins K vecteurs dans chaque région.
La deuxième règle de filtrage stipule que si

$$MinDist(\boldsymbol{Q}, R) \geq d(\boldsymbol{Q}, \boldsymbol{X_K})$$

alors R peut être éliminée. $\boldsymbol{X_K}$ représente le K^{eme} plus proche voisin actuellement trouvé. Les sous-régions d'un nœud étant en général d'abord triées par ordre croissant de $MinDist(\boldsymbol{Q}, R)$, cette règle met fin à la recherche dans le nœud.

Les deux algorithmes de recherche des K-plus proches voisins les plus connus sont l'algorithme **RKV** [140] et l'algorithme **HS** [92]. L'algorithme RKV, proposé originellement dans le cas du R-tree, s'applique lorsque les régions sont des hyper-rectangles. La première règle de filtrage est renforcée par l'utilisation d'une distance, appelée MinMaxDist, plus restrictive que MaxDist. Elle correspond au minimum, sur les différents côtés de l'hyper-rectangle, de la MaxDist de chaque côté. La règle de filtrage suppose alors que chaque côté de l'hyper-rectangle contient au moins un point, ce qui est vrai dans le cas du R-tree par construction.

L'algorithme HS propose d'éviter de parcourir l'arbre de manière classique, en profondeur ou en largeur d'abord. Pour cela il gère une Liste de Régions Actives (LRA), indépendamment du niveau de l'arbre auquel elles se situent. Cette liste est initialisée par la région correspondant à la racine de l'arbre (ainsi que sa MinDist), puis réitère les opérations suivantes :
- Sélectionner la région R ayant la plus petite MinDist dans la LRA.
- Si R correspond à une feuille, chercher les plus proches voisins.
- Sinon, calculer la MinDist des régions filles et les insérer dans la LRA.
- Supprimer R de la LRA.

L'algorithme HS a l'avantage de ne nécessiter que la connaissance de MinDist. Il a de plus été montré qu'il est plus performant en termes de nombre de nœuds ou de feuilles visités [27]. L'inconvénient est qu'il nécessite plus de mémoire pour maintenir la LRA et que l'espace mémoire nécessaire varie en $O(N)$.

Techniques basées sur le partitionnement des données

Les formes englobantes peuvent, dans le cas de ces techniques, être disjointes ou bien se chevaucher. Considérons d'abord les régions correspondant uniquement aux feuilles. Les vecteurs ne sont présents que dans une et une seule de ces régions. En revanche, l'union de toutes ces régions ne couvre pas systématiquement tout l'espace. Les régions englobantes des niveaux supérieurs peuvent également être disjointes mais elles peuvent en plus se chevaucher. Un vecteur peut ainsi appartenir à plusieurs régions des niveaux supérieurs mais ne sera stocké que dans un seul des sous-arbres. L'avantage du partitionnement des données est qu'il ne structure que des régions où il y a effectivement des points (régions disjointes). En revanche, le recouvrement est un inconvénient puisqu'il faudra parfois parcourir plusieurs sous-arbre pour une seule position dans l'espace. Nous donnons ci-après quelques précisions sur certaines des structures les plus utilisées :

1. **Le R-tree, le R^+-tree, le R^*-tree et le Hilbert-R-tree**
   Le R-tree [84] est l'extension directe du B-tree aux espaces multidimensionnels. Les intervalles sont remplacés par des formes englobantes hyper-rectangulaires dont les côtés sont parallèles aux axes. On appelle ces hyper-rectangles des MBR (*Minimum Bounding Region*) car ce sont les hyper-rectangles englobant minimaux de l'ensemble des vecteurs ou des hyper-rectangles du niveau inférieur. C'est-à-dire qu'il n'existe pas d'hyper-rectangle plus petit pouvant englober ces données. La figure 3.1 illustre la structuration résultante d'un R-tree dans le plan.

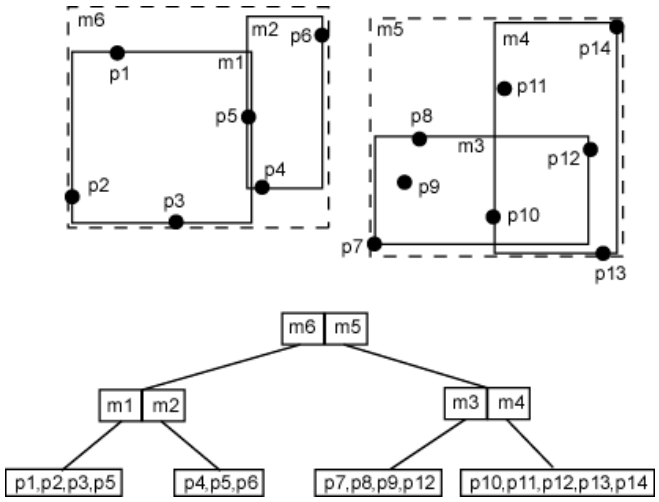


FIG. 3.1 – Structure du R-tree

La principale difficulté d'un R-tree réside dans sa construction. L'algorithme initialement proposé construit le R-tree par insertions dynamiques successives. L'arbre contient au début une seule feuille vide et les vecteurs sont insérés un à un. Le processus d'insertion d'un vecteur pose deux problèmes. Il faut d'une part déterminer dans quelle feuille le point doit être inséré et il faut d'autre part gérer la saturation des feuilles et des nœuds. Plusieurs scénarios sont envisageables pour le choix de la feuille, suivant que le vecteur à insérer appartient seulement à une, à plusieurs ou à aucune MBR. Dans le premier cas, le vecteur est simplement inséré dans la feuille correspondante. Dans le cas d'un recouvrement, la MBR ayant le plus petit volume est choisie. Dans le cas où aucune des MBR existantes ne contient le vecteur, on choisit la région nécessitant la plus petite extension de volume.
Lorsqu'une feuille ou un nœud est saturé, la MBR correspondante est divisée en deux MBR de même cardinalité afin de préserver l'équilibre de l'arbre. Le choix de l'axe parmi D, suivant lequel la MBR est divisée en deux est fait de manière à ce que le volume de l'union des deux MBR résultantes soit minimum. La minimisation du volume couvert par chaque MBR permet d'augmenter la performance des règles de filtrage mises en œuvre dans les algorithmes de recherche (plus le volume est faible et moins le nombre de régions ayant une intersection avec la requête est important).

Le défaut principal du R-tree est le très fort taux de recouvrement entre les MBR. Il est en effet important de minimiser le taux de couverture global en minimisant le volume des MBR, mais il est tout aussi important de minimiser le taux de recouvrement entre les MBR pour éviter qu'une trop grande partie de l'arbre ne soit explorée. Pour remédier au

chevauchement des MBR, Sellis et al. [154] ont proposé le R^+-tree. Lorsque le recouvrement ne peut être évité, lors du fractionnement d'une MBR, certaines des MBR du niveau inférieur sont simplement scindées en deux. L'inconvénient de ce découpage forcé est qu'il peut se répercuter jusqu'aux feuilles de l'arbre, augmentant ainsi fortement le coût d'insertion des vecteurs.

Beckmann et al. [10] ont remarqué que le fait de supprimer des vecteurs puis de les réinsérer améliorait sensiblement les performances globales d'un R-tree. Ils ont alors proposé le R^*-tree [10], dans lequel, avant de fractionner une MBR saturée, ils proposent de supprimer puis de réinsérer une partie des données ce cette MBR. Il s'avère alors que dans de nombreux cas les données réinsérées se répartissent différemment et que le fractionnement n'est plus nécessaire. Le taux d'exploitation de l'espace alloué est en conséquence sans cesse amélioré. De plus, les procédures d'insertion et de fractionnement sont améliorées.

Le Hilbert R-tree [105] a été proposé afin d'améliorer le taux d'exploitation de l'espace alloué par le biais d'une méthode de fractionnement différé. Au lieu de fractionner directement un nœud saturé en deux nœuds, certaines entrées de ce nœud sont transférées à un nœud frère (nœud ayant un parent commun). Le fractionnement est réalisé lorsqu'un certain nombre s de nœuds frères sont également saturés. On parle alors d'un fractionnement s-to-$(s+1)$ puisque $(s+1)$ nœuds seront créés à partir de s nœuds. Ce fractionnement différé est facilité par un ordonnancement des nœuds selon une courbe de Hilbert. Chaque nœud possède une clé représentant sa position sur une courbe de Hilbert. Cette clé permet de déterminer rapidement, de manière unique et systématique un ensemble de $(s-1)$ nœuds frère pouvant accueillir les entrées dépassant la capacité du nœud saturé.

2. **Le X-tree, le SS-tree, le SR-tree et TV^*-tree**
Pour résoudre le problème de la croissance exponentielle du taux de chevauchement en fonction de la dimension, Berchtold et al. [18] ont proposé le X-tree. La principale différence avec un R-tree est l'utilisation d'une capacité variable pour les nœuds de l'arbre. Cette souplesse permet de rejeter les fractionnements de MBR qui provoquent trop de chevauchements et de préférer agrandir le nœud en un seul super-nœud de capacité supérieure.

Le SS-tree [172] est une structure semblable au R^*-tree en ce qu'elle utilise également la réinsertion des vecteurs. En revanche, les régions employées ne sont pas des hyper-rectangles mais des hyper-sphères. Pour des raisons de complexité de calcul, ces hyper-sphères englobantes ne sont pas des formes englobantes minimales. Le centre de l'hyper-sphère est le barycentre des vecteurs englobés. Lors d'une insertion, le vecteur est assigné à la sphère la plus proche. L'avantage théorique des sphères est qu'elles préservent mieux la proximité spatiale qu'un hyper-rectangle à volume équivalent. D'autre part les clés sont plus légères puisqu'il suffit d'un vecteur de dimension D et d'un rayon pour définir une hyper-sphère, alors que deux vecteurs de dimension D sont nécessaires pour définir un hyper-rectangle. En revanche, le fractionnement des nœuds provoque encore plus de recouvrement.

Le TV-tree *Telescopic vector tree* [117], est également un dérivé du R^*-tree basé sur des régions englobantes hyper-sphériques. La métrique utilisée pour définir les sphères peut cependant être n'importe quelle distance L_p. Pour limiter les effets liés aux grandes dimensions, l'idée est d'utiliser une méthode de réduction de la dimension. Au niveau de chaque

nœud, les dimensions sont ordonnées selon leur contenu informatif et seul un sous-ensemble d'entre elles est pris en compte. Le nombre de composantes conservées varie d'un nœud à l'autre. La sélection de ces composantes se fait en éliminant les dimensions selon lesquelles les composantes des vecteurs sont redondantes. Cependant, la réduction de la taille de l'arbre est fortement dépendante de la distribution des vecteurs dans l'espace.

Pour prendre avantage des deux types de formes englobantes, le SR-tree [106] utilise des régions englobantes qui sont l'intersection d'hyper-rectangles et d'hyper-sphères. Il peut être vu comme une combinaison du SS-tree et du R^*-tree, puisque les formes englobantes utilisées pour faire l'intersection sont les mêmes que dans ces deux structures. L'inconvénient des hyper-rectangles est que les points diagonaux peuvent être très éloignés les uns des autres, spécialement lorsque la dimension augmente. L'inconvénient des hyper-sphères est qu'il est très difficile de les fractionner sans qu'il en résulte un recouvrement. L'intersection des deux formes permet de supprimer les points diagonaux éloignés tout en conservant certaines faces parallèles aux axes qui facilitent le fractionnement sans recouvrement. Il est à noter que ces considérations sont surtout vraies pour la distance L_2 et que le SR-tree ne présente pas forcément un avantage pour d'autres types de métriques. L'autre inconvénient majeur du SR-tree est que la gestion des formes englobantes est beaucoup plus complexe, accroissant sensiblement les coûts d'insertion, de mise à jour et des algorithmes de recherche (RKV ou HS en général).

Techniques basées sur le partitionnement de l'espace

Contrairement, aux structures précédentes, les structures d'indexation vectorielle basées sur un partitionnement de l'espace n'entraînent aucun recouvrement entre les régions englobantes. Elles ont également l'avantage d'être beaucoup plus simple à construire et à manipuler. Sans recouvrement, l'insertion d'un point et la gestion du fractionnement des nœuds et des feuilles deviennent en effet beaucoup plus simples. L'inconvénient est que l'espace entier est structuré, même dans les régions où il n'y a pas de vecteur et celles-ci seront donc inutilement explorées. La plupart des méthodes proposées dans la littérature sont inspirées du KD-tree [13]. Le KD-tree est un arbre binaire multidimensionnel, partitionnant les régions en deux autres régions, suivant une seule dimension, à chaque niveau de l'arbre. En cela, il garantit que la partition de l'espace obtenue est complète et disjointe (cf. figure 3.2).

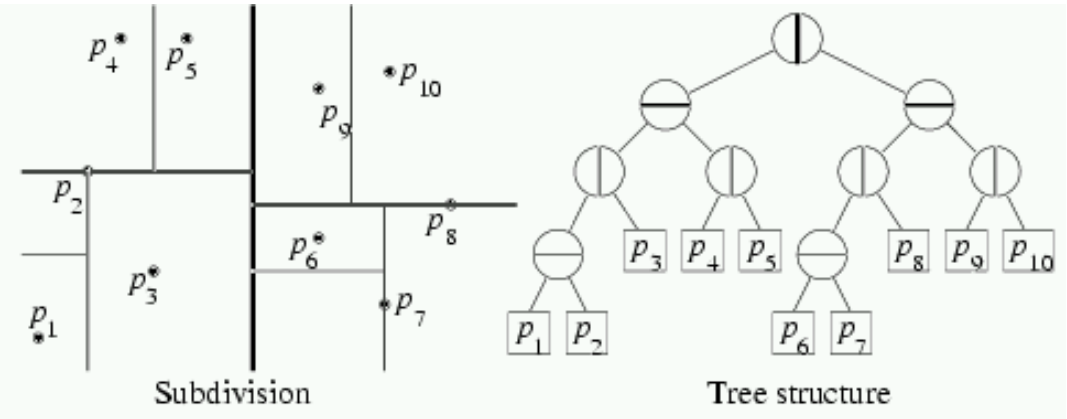


FIG. 3.2 – Structure d'un KD-tree

1. **Le LSD-tree :** Le problème des KD-tree est qu'ils ne sont pas équilibrés par principe (contrairement aux arbres basés sur un partitionnement des données). La structure binaire de l'arbre ne permet pas de fractionner une région à l'intérieur d'un même nœud père. Le principe est de rajouter un niveau localement, déséquilibrant ainsi l'arbre. Ainsi le LSD-tree (Local Split Decision tree, [91]), est une structure d'indexation binaire non équilibrée. La clé contenue dans chaque nœud caractérise uniquement la dimension et la position de la partition. L'arbre est initialement entièrement stocké en mémoire, sauf les feuilles contenant les données qui sont stockées sur disque. Lorsque la base de données est trop volumineuse, la structure est répartie sur trois niveaux. Le premier niveau est maintenu en mémoire et ne contient que des nœuds. Le deuxième niveau contient également des nœuds mais qui sont stockés sur disque et le troisième niveau contient les données stockées sur disque. L'originalité est que les nœuds du deuxième niveau sont stockés par paquets contenant uniquement des sous-arbres de même hauteur réduisant ainsi les désavantages liés au déséquilibre de l'arbre (temps de recherche très variable).

2. **Le KDB-tree :** Le KDB-tree [139] a été proposé pour résoudre le problème de l'équilibrage d'un KD-tree. Lorsqu'un nœud est saturé, la région correspondante est partitionnée selon un hyper-plan choisi. Ne pouvant rajouter une troisième feuille au nœud père, la solution retenue est de propager le partitionnement jusqu'à la racine de l'arbre. Ce type d'approche rend la complexité d'une insertion beaucoup plus importante, comme dans le cas du R^+-tree. De plus, l'utilisation minimale de l'espace disque n'est pas garantie et il est difficile de prévoir la taille effective de l'arbre.

3. **Le LSD_h-tree :** Le LSD_h-tree [90], est une amélioration du LSD-tree palliant les problèmes liés à l'utilisation de vecteurs de grande dimension. Le problème des KD-tree (et plus globalement de toutes les méthodes de partitionnement de l'espace) est que toutes les régions vides de l'espace sont inutilement prises en charge par la structure augmentant ainsi considérablement le nombre de feuilles à visiter. L'importance de ces régions vides croît exponentiellement avec D. Le LSD_h-tree propose de modifier les clés obtenues lors d'une première passe par un LSD-tree, en réduisant tous les hyper-rectangles de la partition à l'hyper-rectangle englobant minimum (le plus petit hyper-rectangle englobant tous les points contenus dans l'hyper-rectangle d'origine). Pour faciliter le codage de ces nouvelles régions, celles-ci ne sont en fait qu'une approximation supérieure de l'hyper-rectangle minimal. Cette stratégie permet d'accélérer très sensiblement les algorithmes de recherche, en éliminant des branches de l'arbre beaucoup plus rapidement.

Techniques basées sur une courbe remplissant l'espace

Les courbes remplissant l'espace (*space filling curves*, voir [146] pour une revue), comme la courbe de Hilbert ou l'ordonnancement en Z, sont des courbes définies de manière récursive dans l'espace multidimensionnel et dont le tracé tend vers un recouvrement complet de l'espace lorsque l'ordre de la courbe tend vers l'infini (cf. figure 3.3). Elles permettent d'établir une transformation bijective entre un espace de dimension D et un espace monodimensionnel (coordonnée curviligne d'un point). Les distances entre points ne sont pas préservées mais une certaine forme de localité est préservée. Ainsi deux points voisins sur la courbe sont proches dans l'espace. Inversement tous les points proches dans l'espace ne le sont pas systématiquement sur la courbe. La méthode que nous proposons utilisant également une courbe de ce type, nous reviendrons plus en détail

sur la construction, l'encodage et les propriétés des courbes remplissant l'espace dans le chapitre suivant.
Le principe des structures d'indexation basées sur des courbes remplissant l'espace [63, 100] est de convertir les vecteurs en leur position sur la courbe, et d'indexer les clés monodimensionnelles ainsi obtenues dans n'importe quelle structure d'indexation monodimensionnelle, typiquement un B^+-tree.
Les opérations d'insertion et de suppression sont ainsi considérablement facilitées puisqu'il suffit d'encoder les vecteurs avant de les soumettre au B^+-tree.
Les courbes remplissant l'espace étant définies à un certain ordre k, elles sont définies sur une grille contenant 2^k cellules. Chaque feuille du B^+-tree correspond en fait à un ensemble de cellules de cette grille ayant des positions successives sur la courbe, mais correspondant à des formes qui peuvent être complexes dans l'espace. Pour les recherches par similarité, il est nécessaire de pouvoir déterminer la distance minimum entre une requête et l'ensemble des cellules contenues dans une même feuille du B^+-tree. Pour éviter d'avoir à parcourir toutes les cellules de la feuille, des algorithmes de type *divide and conquer* sont alors mis en œuvre. L'idée est de diviser récursivement l'intervalle correspondant à une feuille, en deux parties, de manière à ce que la partition correspondante dans l'espace soit exploitable pour déterminer laquelle des deux régions est la plus proche.

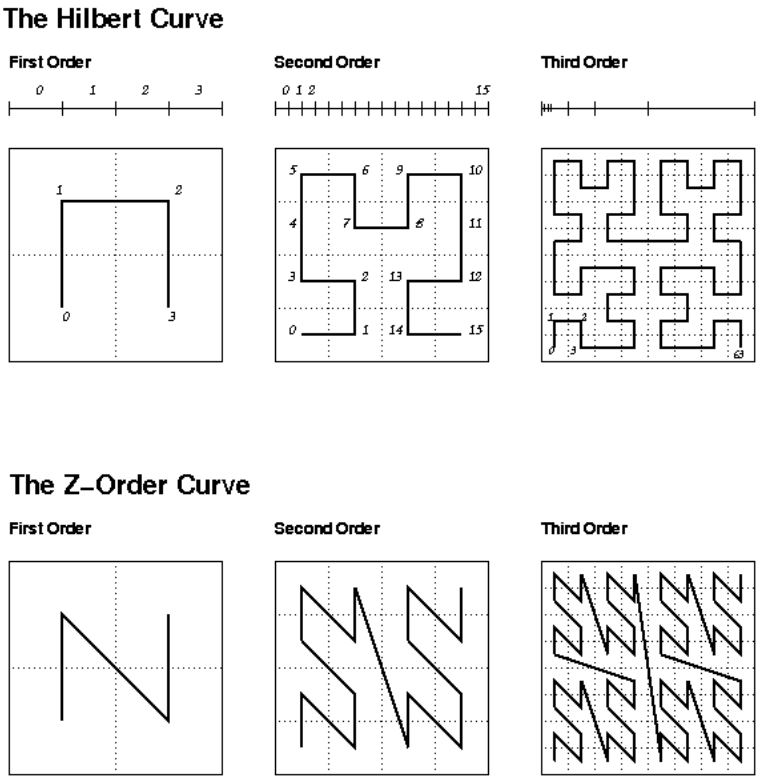


FIG. 3.3 – Deux courbes remplissant l'espace

Le Pyramid-tree

Le Pyramid-tree [17] est une structure d'indexation basée sur un partitionnement de l'espace, mais qui présente des différences importantes par rapport aux autres méthodes vues précédemment, c'est pourquoi nous le présentons à part. Il a été spécialement conçu pour contrer la *malédiction de la dimension*, comme l'indique le titre de l'article. Toutes les structures proposées jusqu'alors présentaient en effet une dégradation exponentielle des performances lorsque la dimension augmente. Le Pyramid-tree repose sur une partition de l'espace dont le nombre de régions croît linéairement avec la dimension. La partition consiste d'abord à diviser l'espace vectoriel en $2D$ pyramides ayant leurs sommets communs au centre de l'espace. Chaque pyramide est ensuite partitionnée par des hyper-plans parallèles à la base de la pyramide. Contrairement aux autres méthodes, les points ne sont pas stockés directement dans les feuilles d'un arbre où les clés seraient une caractérisation de la région à laquelle ils appartiennent.
Le principe du Pyramid-tree est de calculer une clé monodimensionnelle pour chaque vecteur afin de les stocker dans un B^+-tree. En cela, le Pyramid-tree est très proche des méthodes basées sur une courbe remplissant l'espace décrites précédemment. Un point de l'espace peut être caractérisé par le numéro de la pyramide à laquelle il appartient et par la hauteur entre le sommet de la pyramide et le point (distance entre les deux points projetée sur l'axe perpendiculaire à la base de la pyramide). Ces deux valeurs sont ensuite ajoutées de manière à n'obtenir qu'un seul nombre réel utilisé comme entrée du B^+-tree.
La recherche à ϵ-près se fait en déterminant toutes les pyramides ayant une intersection avec la requête puis toutes les hauteurs affectées, par un algorithme assez lourd comportant de nombreux tests de cas. Le B^+-tree est ensuite interrogé et tous les vecteurs contenus dans les feuilles retrouvées sont comparés à la requête.
Il a été montré que, pour une distribution uniforme et une requête à un rayon près constant, les performances du Pyramid-tree s'améliorent lorsque la dimension augmente [17]. Le gros inconvénient du pyramid tree, est qu'il ne permet de faire que des recherches à ϵ-près et non des recherches de K-plus proches voisins.

3.2.2 Les structures d'indexation métriques *(classe MS)*

Dans certaines applications, les objets ne peuvent pas être représentés dans un espace vectoriel, mais il existe tout de même une distance permettant de les comparer. C'est le cas lorsque l'on utilise des descripteurs de dimension variable, par exemple. C'est également le cas lorsque les objets ne sont pas des vecteurs mais des chaînes de caractères ou bien des ensembles d'hypothèses. Dans d'autres cas, les objets sont bien des vecteurs de même dimension, mais la distance entre ceux-ci est trop complexe pour que les structures d'indexation vectorielle soient applicables. Le problème est qu'il n'y a plus assez de corrélation entre la proximité spatiale des vecteurs et la distance que l'on souhaite indexer. Prenons l'exemple, d'une image que l'on peut représenter comme un point dans un espace vectoriel de très grande dimension. La mise en place de distances complexes a justement pour but de se rendre invariant à certaines transformations qui engendrent de très fortes variations dans cet espace de représentation.
Dans tous ces cas, l'indexation doit donc se baser uniquement sur la distance entre les objets. Deux méthodes sont utilisées pour ce genre de données. La première méthode consiste à trouver une transformation qui permet de représenter les objets dans un espace vectoriel et à utiliser une structure d'indexation vectorielle classique. La deuxième méthode consiste à utiliser une structure d'indexation réellement basée sur les distances.

Transformation dans un espace vectoriel

Ce type d'approche consiste à construire une base de points dans un espace vectoriel à partir des objets de la base et des distances entre ces objets. Une structure d'indexation vectorielle peut ensuite être utilisée, c'est pourquoi nous citons ce genre d'approche dans cette section bien qu'il s'agisse en fait de méthodes approximatives. Les distances dans *l'espace d'encastrement* ainsi obtenu (*embedding space* [29, 67]) ne sont en effet jamais parfaitement conservées. L'objectif de ce genre d'approches est d'essayer d'optimiser la transformation pour que les résultats d'une requête par similarité soient le plus exacts possibles dans le nouvel espace. Nous reviendrons sur ces approches dans la section consacrée aux méthodes approximatives de recherche par similarité.

Structures d'indexation basées sur la distance

Ces structures d'indexation sont plus génériques que les structures d'indexation vectorielle et peuvent très bien être appliquées à des données vectorielles. Elles sont parfois également appelées structures d'indexation basées sur des pivots [35] car le principe général consiste à utiliser certains objets de la base (les pivots) comme référence pour indexer les autres points en fonction de leur distance à ces points. Pour un état de l'art plus complet, le lecteur pourra se référer à [42, 31].
La plus ancienne référence de structure permettant d'indexer des espaces métriques à partir de la distance est le Burkhard-Keller-tree [34]. Il repose sur la quantification des valeurs de la distance, qui ne peut prendre qu'un petit nombre n_d de valeurs possibles. Un des objets de la base est choisi arbitrairement comme la racine de l'arbre (un pivot) et la distance quantifiée est utilisée pour partitionner les objets restant en n_d branches. Cette procédure est répétée pour chaque sous-ensemble non vide pour construire l'arbre. La structure résultante est donc un arbre statique (pas d'insertion ou de suppression dynamique) et non équilibré (le temps d'un d'accès n'est donc pas garanti).

Le Vantage-Point-tree (VP-tree, [177]) est un arbre binaire dont chaque nœud est constitué uniquement par un des points de la base (un pivot) et par une distance d_m égale à la médiane des distances entre le pivot et les objets contenus dans le sous-arbre pointé par ce nœud. La branche gauche de l'arbre indexe tous les points dont la distance avec le pivot est inférieure où égale à d_m, la branche droite de l'arbre ceux dont la distance avec le pivot est supérieure à d_m. Le VP-tree est un arbre équilibré puisque le même nombre de points est réparti sur chaque branche pour tous les nœuds. La construction de l'arbre se fait directement sur l'ensemble des points de la base avec un nombre de distance à calculer variant en $O(Nlog(N))$. Le VP-tree est donc une structure statique.

Le M-tree [48], est une structure d'indexation métrique dynamique, basée sur un arbre équilibré. C'est la première méthode qui permet de faire, avec un espace métrique non vectoriel, tout ce qu'il est possible de faire avec une structure vectorielle classique. Sa conception a été guidée à la fois par la réduction des coûts d'entrée/sortie sur disque mais également par la réduction du nombre de distances à calculer. L'utilisation d'un espace métrique non vectoriel va en effet souvent de pair avec des distances complexes, dont le coût CPU ne peut être négligé.
Le principe du M-tree est d'éliminer rapidement des branches de l'arbre en utilisant l'inégalité triangulaire et la MinDist entre la requête et les formes englobantes constituées par des hypersphères centrées atour des pivots. Sa construction se fait par insertions successives de manière similaire aux structures d'indexation vectorielles.

3.2.3 Comparaisons, évaluations et optimisations

Aucune étude comparative complète et objective de toutes ces structures n'a permis de véritablement les différencier dans l'absolu [27]. Les expérimentations publiées ont tendance a être très dépendantes des données utilisées. Plus la dimension augmente plus il y a de différences engendrées par différentes distributions ou différentes requêtes.
En général, seules les expérimentations comparant une structure avec une version améliorée de celle-ci sont plus concluantes. Il est ainsi communément admis que le R^*-tree est de 10 % à 70 % plus performant que le R-tree.
Le pyramid-tree est également reconnu pour donner de meilleures performances que les autres structures dans le cas de dimensions très élevées.
Le choix d'une structure plutôt qu'une autre repose donc avant tout sur les caractéristiques de ces structures et la connaissance des données : dimension des données, type de requête ou encore nécessité ou non des insertions dynamiques. Le résumé, sous forme de tableau, de la comparaison qualitative réalisée par Böhm et al. [27] est fourni en annexe C.

Certains travaux se sont occupés de l'optimisation des structures d'indexation multidimensionnelle dans un cadre général. La première approche concerne la construction des arbres. Au lieu d'insérer les vecteurs dynamiquement, processus très coûteux pour la construction initiale de l'arbre, des techniques dites de *bulk-loading* ont été proposées pour construire l'arbre directement sur un ensemble de vecteurs. Dans [16], Berchtold et al. ont ainsi montré qu'outre l'accélération très significative de la construction de l'arbre, ce genre de technique peut également permettre d'accélérer les temps de recherche. En permettant un déséquilibre contrôlé de l'arbre, leur technique permet d'accélérer les temps de recherche d'un facteur 192.
Pour pallier la dégradation des performances dans les espaces de dimension élevée, il a été proposé des méthodes dites de *tree-striping* dont le principe est de découper les vecteurs en des sous-vecteurs de dimension plus faible, qui sont alors indexés par des techniques classiques. Lors d'une recherche, la requête est également découpée en plusieurs sous-vecteurs qui sont recherchés dans la base et une étape de fusion des résultats permet de retrouver la réponse à la requête globale. A la limite, lorsqu'un vecteur de dimension D est découpé en D scalaires, cette technique se rapproche d'une liste inversée. Dans [15], il a ainsi été montré qu'un optimum pour la dimension des sous-vecteurs pouvait être évaluée théoriquement et que les temps de recherche pouvaient être fortement améliorés.
Des modèles de coût ont également été proposés pour optimiser la taille des pages d'un arbre ou bien la taille des blocs d'une partition spatiale [14], conduisant encore une fois à des gains significatifs de performances.

3.2.4 Structures d'indexation multidimensionnelle en mémoire

La plupart des structures précédentes ont été conçues pour une utilisation où les données sont stockées en mémoire secondaire (sur disque). L'objectif est principalement de minimiser le nombre de nœuds et de feuilles à visiter pour diminuer le nombre d'entrées/sorties dont le coût est considéré comme le plus important. Les modèles de coût sont donc généralement basés uniquement sur le nombre de feuilles et de nœuds visités, assimilé au nombre d'accès au disque. L'utilisation des ces techniques en mémoire principale n'est cependant pas proscrite. Historiquement, les structures basées sur le partitionnement de l'espace, qui sont les plus anciennes, étaient même utilisées en mémoire avant d'être adaptées à une utilisation sur disque, en même

temps qu'apparaissaient les structures basées sur un partitionnement des données. Dans le cas d'un fonctionnement en mémoire, au lieu de réduire le nombre d'entrées/sorties, l'objectif est de réduire le nombre de comparaisons et de distances à calculer, ainsi que le nombre d'accès mémoire.
Les supports de type RAM devenant de plus en plus capacitifs et de moins en moins chers, l'hypothèse selon laquelle les bases de données ne peuvent pas être stockées en mémoire principale des calculateurs n'est aujourd'hui plus valide. Cela a provoqué un regain d'intérêt pour les systèmes fonctionnant exclusivement en mémoire [25, 107, 53]. Certains travaux se sont notamment attachés à optimiser les structures d'indexation vectorielle classiques pour un fonctionnement en mémoire. Ils partent de la constatation que les mémoires actuelles à deux niveaux de cache ne peuvent plus être modélisées par un modèle de mémoire à plat et que les sorties du cache de deuxième niveau (cache $L2$) ne peuvent plus être négligées. C'est pourquoi ces approches sont qualifiées de *cache-conscious*. L'objectif principal est de réduire l'espace des entrées et donc de l'arbre, afin de diminuer le nombre de sorties de cache. Ainsi, le CR-tree [107] a été proposé afin de compresser les clés des régions englobantes minimum d'un R-tree. Cette réduction permet un gain de temps de recherche d'un facteur 2,5.
Les approches *cache-conscious* sont intéressantes uniquement pour des dimensions très faibles voir unique dans le cas d'optimisation du B-tree [25]. Dans les espaces de plus grande dimension, le coût CPU de calcul des distances a un poids tout aussi important que le nombre de sorties de cache L2 [53]. Le Δ-tree [53] permet ainsi de limiter le coût de calcul des distances par une technique de réduction de la dimension basée sur une analyse en composantes principales. Un comparatif des performances de plusieurs techniques de recherche de plus proches voisins, réimplémentées et optimisées pour fonctionner en mémoire, est réalisé dans cette même publication. Ces structures sont le CR-tree (R-tree avec clés compressées), le TV-tree, le M-tree, le Δ-tree ainsi qu'une méthode de recherche séquentielle accélérée que nous détaillerons dans la section suivante, le VA-file. Dans le contexte de dimension élevée (D>14), le CR-tree est clairement moins performant que toutes les autres méthodes. Le Δ-tree semble sensiblement plus performant que les autres, en particulier lorsque la taille de la base augmente.

3.3 Les méthodes séquentielles accélérées *(classe VS)*

Dans [170], Weber et al. ont montré que, pour une recherche de K-plus proches voisins, la plupart des structures d'indexation multidimensionnelle deviennent moins performantes qu'une recherche séquentielle lorsque la dimension devient supérieure à environ $D = 16$. Ce comportement a par ailleurs été interprété théoriquement par des études sur la *malédiction de la dimension*, comme celle de Beyer et al. [22]. Deux phénomènes principaux permettent d'expliquer la dégradation des performances dans les espaces de dimension élevée :

- Les vecteurs ayant tendance à devenir tous équidistants les uns des autres lorsque la dimension augmente, le plus proche voisin d'une requête est aussi loin que tous les autres vecteurs de la base. Il est alors nécessaire de comparer la requête avec toute la base et une méthode séquentielle est forcément plus performante.
- Plus la dimension augmente, plus l'espace semble vide. Le nombre de points à l'intérieur d'une hyper-sphère de rayon constant tend ainsi vers 0 très rapidement avec la dimension. Autrement dit, en plus de tendre vers l'équidistance, les points sont de plus en plus loin les uns des autres.

Weber et al. ont alors proposé une technique de recherche par similarité, appelée le VA-file [170].

Le principe est d'effectuer dans un premier temps une recherche séquentielle rapide, non pas sur les vecteurs eux-mêmes mais sur une version fortement comprimée de ceux-ci. Cette étape permet d'éliminer rapidement une très grande partie des vecteurs de la base. Une deuxième passe sur les vecteurs originaux restant permet ensuite de raffiner la recherche. Les vecteurs comprimés sont obtenus simplement par une quantification scalaire des différentes composantes du vecteur d'origine.
La méthode n'est cependant rentable que si le VA-file peut être entièrement contenu en mémoire. Dans le cas contraire, le fichier doit être rechargé entièrement en mémoire à chaque requête et le coût engendré ne justifie plus son utilisation. Ce résultat a été vérifié expérimentalement dans [4] où il est montré que dans ce cas, il est à nouveau moins performant qu'une recherche séquentielle. Un deuxième inconvénient est que les temps de recherche, bien qu'accélérés, restent linéaires en fonction de la taille de la base [4]. Enfin, Le VA-file n'a aucune utilité lors d'une utilisation strictement en mémoire principale. Le principal gain du VA-file est en effet d'économiser les temps de chargement de la base lors d'une recherche séquentielle. Lorsque ces coûts d'entrées/sorties n'existent plus, le coût CPU de la quantification et de la comparaison avec les vecteurs approximés devient plus important que le coût économisé sur le calcul des distances. Une implémentation du VA-file en mémoire a été testée dans [53], où il a été vérifié qu'il était moins performant qu'une recherche séquentielle classique.

Une amélioration du VA-file pour des distributions non uniformes a été proposée sous l'appellation VA^+-file [69]. Les données sont transformées par une analyse en composantes principales avant d'être quantifiées. Les pas de quantification sont déterminés de manière plus subtile par un algorithme de classification de type *K-means*.

Le LPC-file [40] est une technique inspirée du VA-file, où les règles de filtrage ont été enrichies par l'ajout, dans le fichier d'approximation, d'une information de position des vecteurs originaux dans leur cellule d'approximation. Cette information est codée par deux composantes : la distance entre le vecteur et le coin inférieur gauche de la cellule d'approximation le contenant, et l'angle entre le vecteur et la diagonale de la cellule. Lors de la recherche, ces deux paramètres sont utilisés pour restreindre les valeurs de MinDist et MaxDist.

Le *Landmark file* [23] est également une technique basée sur un fichier contenant une version comprimée des vecteurs mais n'est pas vraiment une méthode séquentielle, puisque la totalité de ce fichier n'est pas parcourue. Le fichier comprimé et le fichier original sont en effet triés physiquement selon une clé calculée pour chacun des vecteurs permettant de borner l'intervalle qui sera parcouru séquentiellement. La dite clé correspond à la distance entre les vecteurs de la base et un point de référence dans l'espace. Les valeurs de cette clé sont stockées dans un troisième fichier, uniquement pour certains vecteurs situés à intervalles réguliers dans le fichier des vecteurs ordonnés. Il s'agit, d'après les auteurs, d'une des rares techniques tirant avantage d'un contexte statique pour accélérer la recherche par similarité. L'hypothèse est que la base ne change jamais ou très rarement et qu'il est donc possible de stocker les vecteurs de manière ordonnée, ce qui simplifie énormément les correspondances d'adresses entre les 3 fichiers. Toute insertion ou suppression dynamique d'un vecteur est alors impossible mais cela permet en contre partie d'accélérer les temps de recherche d'un facteur croissant en fonction de la taille de la base.

3.4 Les méthodes de recherche approximatives

Toutes les techniques d'indexation vues précédemment sont des méthodes de recherche exactes, c'est-à-dire que les résultats obtenus sont exactement les mêmes que si l'on avait utilisé une méthode de recherche exhaustive. La limitation des performances de ces structures, en particulier pour les dimensions élevées, a conduit récemment les chercheurs à s'intéresser de plus en plus à des méthodes de recherche approximatives. Dans de nombreux cas, il a en effet été montré qu'une faible erreur par rapport aux résultats exacts pouvait conduire à des gains de performances très élevés. L'idée est d'offrir à l'utilisateur un compromis entre le temps de calcul et la qualité des résultats obtenus. Ce paradigme est en fait assez récent car l'impact de l'erreur introduite n'est pas le même pour tous les types de données.
Dans le cas de données géographiques par exemple, historiquement les premières à avoir alimenté le besoin de structures d'indexation multidimensionnelle, son intérêt est moindre. Dans les applications nécessitant ce type de données, la distance entre les vecteurs est généralement la traduction exacte de la requête de l'utilisateur (quelles sont les villes qui se situent dans un rayon de 10 km autour de Paris ?). La perte de qualité dans la recherche se traduit donc directement par une erreur dans le résultat qu'attend l'utilisateur.
Dans le cas de la recherche d'images par le contenu la notion de distance entre les descripteurs n'est qu'une interprétation particulière de la notion de similarité. Il n'existe donc pas de résultat exact à une requête donnée. Une recherche exacte dans la base des descripteurs est donc déjà une approximation de la requête utilisateur. Il y a même des cas où l'introduction d'une erreur dans la qualité de la recherche n'a aucun impact sur le résultat final de l'application. Comme l'ont montré Berrani et al. [20], ceci est particulièrement vrai dans le cas de descripteurs locaux où la recherche dans la base est suivie d'une étape de fusion des résultats qui peut se satisfaire de résultats incomplets.
On peut séparer les méthodes de recherche approximatives en deux grandes catégories, en fonction de la nature de l'erreur introduite [47]. Celles qui sont basées sur un changement d'espace (section 3.4.1) et celles qui sont basées sur une réduction du nombre de comparaisons (section 3.4.2). De la même manière que les structures d'indexation vues précédemment, les méthodes approximatives peuvent également être classifiées par la nature de la mesure de similarité et des descripteurs de la base (cf. section 3.1.2 : VS_{L_p}, VS, MS ou NMS). Nous spécifierons les catégories auxquelles appartiennent les méthodes décrites par la suite.
Enfin, une autre caractéristique primordiale d'un système de recherche approximative, est la garantie du résultat. Certaines techniques permettent en effet à l'utilisateur de contrôler explicitement l'imprécision sur la recherche, tandis que d'autre ne peuvent fournir qu'une mesure de l'erreur a posteriori. Certaines, enfin, ne fournissent ni garantie, ni mesure de l'erreur. Nous discuterons de cette caractéristique pour les méthodes présentées ci-après.

3.4.1 Les méthodes basées sur un changement d'espace

Le changement d'espace peut se faire soit par une transformation des descripteurs, soit par une transformation de la mesure de similarité. L'objectif de la transformation est généralement d'obtenir un problème exact plus simple à résoudre. Pour mesurer l'erreur introduite par ce genre de techniques, on considère généralement la liste ordonnée des objets de la base par leur degrés de similarité à la requête. L'idée est que, si l'ordre des objets est préservé d'un espace à l'autre, la technique n'introduit pas d'erreur pour les requêtes de K-plus proches voisins. Ainsi, la mesure de rang normalisée [169], est définie pour une requête $\boldsymbol{Q}$ donnée, par :

$$m_{rn}(\boldsymbol{Q}) = \frac{K(K+1)}{2\sum_{i=1}^{K} rang(\boldsymbol{Q}, \boldsymbol{nn_i(Q)})}$$

où K est le nombre de voisins, $\boldsymbol{nn_i(Q)}$ est le i^{eme} vecteur retourné par la méthode approximative et $rang(\boldsymbol{Q}, \boldsymbol{X})$ est le rang qu'aurait l'objet $\boldsymbol{X}$ avec une méthode exacte. Cette mesure vaut 1 lorsque l'ordre est préservé et diminue lorsque les points retrouvés sont de plus en plus loin des plus proches voisins théoriques.

Pour les requêtes à un rayon près, l'hyper-sphère dans l'espace d'origine se traduit généralement par une forme plus complexe dans le nouvel espace. Lorsque celle-ci n'est pas calculable ou difficilement exploitable, on utilise une approximation de la région ou bien une requête à un rayon près dans ce nouvel espace. Les critères utilisés dans ce cas pour quantifier l'erreur introduite, sont plutôt du type précision/fausses alarmes. Notons toutefois que la grande majorité des méthodes de recherche approximative se concentrent uniquement sur la recherche des plus proches voisins [47].

Techniques vectorielles de réduction de la dimension *(classe VS)*

Longtemps, les techniques vectorielles de réduction de la dimension ont été les seules solutions apportées au problème de la dimension élevée dans les systèmes réels de recherche d'images par le contenu. Le principe est de transformer un espace vectoriel de grande dimension en un espace de dimension faible, puis d'utiliser une structure d'indexation vectorielle classique (R-tree, SR-tree, etc.).
Ce paradigme est intimement lié au concept de **dimension intrinsèque des données**, c'est-à-dire le nombre de composantes indépendantes dans la base ou encore le nombre de degrés de liberté. La distribution de descripteurs réels dans un espace vectoriel n'est en effet jamais uniforme, et la dépendance statistique entre les vecteurs fait supposer qu'ils pourraient être représentés dans un espace de dimension moins élevée tout en conservant la même information [19]. Si tous les vecteurs sont, par exemple, distribués le long d'une courbe, la dimension intrinsèque sera égale à $D' = 1$ quelle que soit la dimension D de l'espace. Si les points sont répartis selon une distribution uniforme la dimension intrinsèque sera égale à la dimension de l'espace. Les données réelles se situent entre ces deux extrêmes. Dans un système linéaire parfaitement déterministe, la dimension intrinsèque correspond au rang de la matrice des données. Dans le cas de données réelles bruitées, la dimension intrinsèque est généralement reliée au concept de dimension fractale qui en plus de prendre en compte la corrélation entre les données, intègre également leur auto-similarité [66]. En effet, si la répartition des données est la même à plusieurs échelles d'observation de l'espace, il s'agit également d'une dépendance statistique.
Le principe des techniques de réduction de la dimension est de projeter les données dans un espace de dimension réduite pour se rapprocher au mieux de la dimension intrinsèque. Autrement dit, il s'agit de supprimer l'information redondante.
Les deux techniques vectorielles les plus utilisées pour la réduction de la dimension sont des méthodes linéaires : l'analyse en composantes principales (ACP) et l'analyse en composantes indépendantes (ACI). La transformation d'un vecteur $\boldsymbol{X}$ de l'espace d'origine (dimension D) en un vecteur $\mathbf{S}$ dans un espace de dimension F, s'écrit alors :

$$\boldsymbol{S} = \boldsymbol{AX}$$

où $\boldsymbol{A}$ est une matrice $D \times F$ contenant les coefficients de la transformation linéaire.

La construction de la matrice $\boldsymbol{A}$ dans le cas de l'ACP, résulte de la diagonalisation de la matrice de covariance $\boldsymbol{\Sigma}$ des données de la base. Les moments statistiques utilisées par cette méthode sont donc limités à l'ordre 2, et seule la corrélation des données est réduite (ou supprimée dans le cas d'une dépendance linéaire non bruitée). Géométriquement, il s'agit d'un changement de repère uniquement par translation et rotation, dans le but de maximiser la variance de chacune des composantes.
L'ACI permet d'intégrer des contraintes sur les moments statistiques d'ordre plus élevé. La matrice de transformation est construite sur un critère d'indépendance des composantes projetées et non pas seulement de non-corrélation comme dans le cas de l'ACP. Géométriquement, le nombre de degrés de liberté possibles pour le changement de repère est plus important. Les axes du nouveau repère ne sont par exemple pas forcément orthogonaux. Pour plus de précisions sur les techniques d'ACI, le lecteur pourra se référer à [51].
Les techniques linéaires de réduction de la dimension souffrent de plusieurs limitations :

- L'application de l'ACP ou de l'ACI à la recherche par similarité approximative est une méthode statique, dans le sens où l'insertion de nouvelles données dans la base nécessite de recalculer la nouvelle matrice $\boldsymbol{A}$ de transformation.
- Pour conserver une qualité acceptable des résultats, la réduction de la dimension n'est souvent pas suffisante pour que les structures d'indexation vectorielle soient optimales. De plus, aucune garantie sur la qualité finale des résultats n'est fournie.
- Le fait que ces techniques soient basées sur une projection linéaire, n'est pas adapté à de nombreuses distributions réellement observées. Elles supposent que les vecteurs sont tous regroupés en un seul nuage de points ayant des propriétés statistiques similaires. Elles n'apportent pas de solution pour des distributions plus complexes, notamment lors de la présence de plusieurs agrégats de données. Il s'agit en fait d'une estimation globale qui n'est pas adaptée aux variations locales qui peuvent être observées dans une distribution réelle.

Changement d'espace basé sur la distance *(classe MS)*

Ces méthodes permettent de construire un ensemble de points dans un espace vectoriel uniquement grâce à la connaissance de la distance entre les objets d'origine. Elles sont donc souvent utilisées pour permettre l'utilisation de structures d'indexation vectorielle dans le cas d'espaces métriques non vectoriels. Cette approche est parfois appelée GEMINI (*GEneric Multimedia INdexIng*, [64, 44]). Mais elles peuvent également être utilisées pour transformer un espace vectoriel en un autre espace vectoriel, généralement dans le but de réaliser une réduction non linéaire de la dimension. L'inconvénient de ce genre de techniques, est que la dimension de l'espace d'arrivée doit être fixé a priori. Or, s'il existe bel et bien des fonctions de coût pour quantifier l'erreur des recherches dans le nouvel espace (cf. section 3.4.1), aucune technique ne permet de fixer la dimension a priori. Qui plus est, l'estimation de l'erreur a posteriori ne constitue pas une garantie de la qualité des résultats.
Toutes les méthodes proposées dans la littérature, pour transformer un espace métrique en un espace vectoriel ne sont pas applicables à la recherche de descripteurs par similarité. Certaines transformations utilisent en effet la totalité des objets pour construire chaque vecteur dans le nouvel espace. Il n'est pas possible de projeter un seul nouvel objet dans l'espace ainsi construit sans refaire toute la construction. Une requête ne peut donc pas être convertie individuellement rendant toute recherche impossible. C'est le cas par exemple des techniques de *multidimensional*

scaling [110]. Ces méthodes sont surtout utiles pour comparer entre eux des objets déjà indexés, pour des problématiques de classification par exemple.

D'autres méthodes plus spécifiques à la recherche par similarité ont été proposées dans la littérature [67, 168, 29, 96]. On peut en trouver un état de l'art récent dans [103]. A titre d'illustration, nous décrivons les principes de base de la technique FastMap [67], utilisée dans plusieurs travaux de recherche d'images ou de vidéos par le contenu [136, 108]. L'entrée de l'algorithme FastMap est une matrice $\mathbf{W}$ de taille $(N_W \times N_W)$ où chaque élément w_{ij} représente la valeur de la distance entre le i^{eme} et le j^{eme} objet de la base. Ces objets sont supposés avoir effectivement une représentation dans un espace vectoriel de dimension D qu'il s'agit de projeter dans un espace de dimension $F < D$. Dans cette hypothèse, F objets définissent un espace de dimension au plus égal à F. En choisissant dans la base deux objets pivots $\boldsymbol{O_1}$ et $\boldsymbol{O_2}$, et en appliquant les formules classiques de projection, il est possible de calculer les coordonnées de tous les autres objets sur la droite définie par ces deux pivots. Si on choisit $\boldsymbol{O_1}$ comme origine, et que l'on note d la distance dans l'espace d'origine, on peut alors calculer la coordonnée x_i d'un objet $\boldsymbol{O_i}$ sur cette droite par (cf. figure 3.4) :

$$x_i = \frac{d(\boldsymbol{O_1}, \boldsymbol{O_i})^2 + d(\boldsymbol{O_1}, \boldsymbol{O_2})^2 - d(\boldsymbol{O_2}, \boldsymbol{O_i})^2}{2d(\boldsymbol{O_1}, \boldsymbol{O_2})}$$

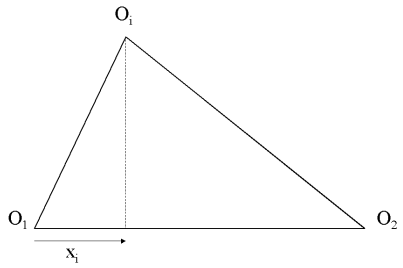


FIG. 3.4 – Illustration de la projection d'un objet réalisée par la méthode FastMap

Cette projection donne directement les coordonnées des objets dans un espace de dimension $F = 1$. Pour une dimension plus élevée, les objets sont ensuite projetés dans un hyper-plan perpendiculaire à la première droite, puis les distances sont recalculées dans ce nouvel espace. Deux nouveaux pivots sont alors choisis dans cet espace et on réitère les mêmes opérations. La construction de l'espace vectoriel à partir d'une base d'objets est rapide puisque sa complexité est linéaire ($O(N)$). Lors d'une phase de recherche ou d'insertion, un objet peut être projeté individuellement, exactement de la même façon, à condition d'avoir stocké les pivots. Le vecteur obtenu peut ensuite être inséré dynamiquement ou recherché via la structure d'indexation vectorielle utilisée.

Utilisation de vecteurs approximés

Ici, le changement d'espace est uniquement induit par une représentation différente des descripteurs de la base, facilitant le calcul des distances ou la recherche dans la base.

1. **VA-LOW** *(classe VS)* : dans [169], Weber et al. proposent deux versions approximatives du VA-file [170]. Une d'elle, le VA-LOW peut être classée dans les méthodes utilisant un

changement d'espace. Le principe est, comme pour le VA-file, de calculer une version compressée des vecteurs de la base en quantifiant scalairement les composantes du vecteur. La différence dans cette version est que seule la première étape calculant la distance entre la requête compressée et les vecteurs compressés est effectuée. L'erreur induite sur le calcul des distances est donc uniquement due à la quantification. Plus le nombre de bits utilisés pour quantifier les composantes est grand, plus la précision sera bonne, au détriment du temps de calcul. Puisqu'il est facile de borner l'écart entre la distance exacte et la distance approximée, la qualité de la recherche peut être garantie géométriquement. Cette technique est strictement linéaire avec la taille de la base, puisque le nombre de comparaisons est toujours égal à N.

2. **Hachage multidimensionnel** *(classe VS_{L_1})* Il s'agit d'une extension des techniques de hachage monodimensionnel. Cette technique a été introduite par Indyk et al. [99] puis améliorée par Gionis et al. [77]. Le principe est de calculer plusieurs fonctions de hachage par vecteur en espérant que certaines d'entre elles pointeront vers une liste contenant les vecteurs recherchés. L'objectif de chacune de ces fonctions de hachage est donc d'obtenir des collisions dans le cas de vecteurs proches dans l'espace tout en différenciant les vecteurs éloignés. L'utilisation de plusieurs fonctions de hachage permet de maximiser la probabilité de retrouver les vecteurs exacts. La transformation correspondant au calcul de la fonction de hachage correspond en fait à un changement d'espace. Dans la version proposée par Gionis et al. [77], les vecteurs sont transformés dans un espace binaire de dimension $C \times D$. La transformation d'un vecteur $\boldsymbol{X} = (x_1, x_2, ..., x_d)^T$ est défini par :

$$v(\boldsymbol{X}) = unaire(x_1)\ unaire(x_2)\ ...\ unaire(x_d)$$

où la fonction $unaire(x)$ est la représentation unaire de x, c'est-à-dire un mot binaire constitué de x uns suivis de $(C - x)$ zéros, C représente la borne maximale sur toutes les composantes. Ils montrent que ce nouvel espace associé à la distance de Hamming correspond en fait à une distance L_1 dans l'espace d'origine. Cette méthode est donc de la classe VS_{L_1} et ne permet pas d'utiliser d'autres métriques.
Les fonctions de hachage sont des projections orthogonales de $v(\boldsymbol{X})$ sur un sous-ensemble d'axes de cet espace. Elles peuvent être directement calculées à partir de $\boldsymbol{X}$ sans passer par $v(\boldsymbol{X})$. Les expérimentations effectuées montrent que cette technique est sous-linéaire en fonction de la taille de la base et qu'elle ne souffre pas non plus de l'augmentation de la dimension.
Le choix du nombre de fonctions de hachage et la dimension des sous-espaces de projection pour les calculer sont des paramètres critiques de la méthode, qui influencent directement la qualité des résultats obtenus. Le lien entre le réglage de ces paramètres et une mesure de l'erreur n'est toute fois pas clairement identifié, et c'est là le principal inconvénient de cette technique.

3. La technique proposée par Oostveen et al. [133], se rapproche du hachage multidimensionnel bien qu'elle soit plutôt apparentée à une technique de liste inversée. Les descripteurs sont là aussi transformés en des vecteurs binaires mais il s'agit, ici, uniquement de la quantification binaire de chaque composante. La distance utilisée pour comparer ces vecteurs comprimés est la distance de Hamming, mais elle n'a cette fois pas de relation bijective avec une métrique de l'espace d'origine. Les vecteurs utilisés ayant des dimensions très élevées, le principe est de découper les vecteurs binaires en un nombre fixe de sous-vecteurs binaires de

dimension réduite et de les utiliser comme index dans une table. Ces sous-vecteurs peuvent être vus comme des fonctions de hachage dont les valeurs pointent vers une liste contenant tous les vecteurs contenant ce sous-vecteur. Le principe de l'approximation consiste uniquement à supposer qu'un des sous-vecteurs restera inchangé entre la représentation binaire de la requête et celles des vecteurs proches recherchés. Les auteurs montrent que cette probabilité est assez forte dans le cas d'une distribution uniforme mais cela parait plus discutable pour des données réelles. La méthode n'offre en outre aucun contrôle de la précision obtenue et aucune expérimentation ne montre l'influence de la dimension des données et de la taille de la base.

3.4.2 Les méthodes réduisant le nombre de comparaisons

Lors de la phase de recherche, ces méthodes calculent la mesure de disimilarité exacte entre la requête et les descripteurs comparés mais limitent le nombre de comparaisons qu'il aurait fallu faire pour une recherche exacte. Il s'agit en général de règles de filtrage, comme dans le cas de la recherche exacte, sauf que le critère de rejet d'une région est différent.

Approximation géométrique des requêtes

Les premières approches proposées pour effectuer des recherches approximatives ont été d'utiliser les techniques de recherche exacte mais en modifiant la requête. Pour approximer une recherche de plus proche voisin il a ainsi été proposé la recherche de ϵ-plus proches voisins. Un ϵ-plus proche voisin est un vecteur dont la distance avec la requête est inférieure à $(1+\epsilon)$ fois la distance du plus proche voisin exact :

Définition ϵ-plus proche voisin - *Soit $\Delta = \{\boldsymbol{D_n}\}_{n\in[1,N]}$ une base de N descripteurs et d : $\Delta \times \Delta \rightarrow \Re$ une mesure de disimilarité associée. Un descripteur $\boldsymbol{D_{\epsilon pp}}$ est un **ϵ-plus proche voisin** d'un descripteur candidat $\boldsymbol{D_{cand}}$ si :*

$$d(\boldsymbol{D_{\epsilon pp}}, \boldsymbol{D_{cand}}) \leq (1+\epsilon) d(\boldsymbol{D_{pp}}, \boldsymbol{D_{cand}})$$

où $\boldsymbol{D_{pp}}$ est le plus proche voisin exact de $\boldsymbol{D_{cand}}$.

ϵ est en fait l'erreur relative maximale entre le point trouvé et le plus proche voisin exact :

$$\frac{d(\boldsymbol{D_{\epsilon pp}}, \boldsymbol{D_{cand}}) - d(\boldsymbol{D_{pp}}, \boldsymbol{D_{cand}})}{d(\boldsymbol{D_{pp}}, \boldsymbol{D_{cand}})} \leq \epsilon$$

Ce paradigme peut être étendu aux K-plus proches voisins en remplaçant dans la définition, le plus proche voisin par le K^{eme} plus proche voisin.
En pratique, cette requête est très facilement transposable à la plupart des méthodes exactes vues dans ce chapitre, puisqu'il suffit de modifier, dans les règles de filtrage (cf. section 3.2.1), la $MinDist$ et la $MaxDist$ par $(MinDist+\epsilon)$ et $MaxDist-\epsilon$. Dans le cas des structures d'indexation multidimensionnelle, cela revient à modifier la taille des régions englobantes, en diminuant ainsi le taux de chevauchement et en renforçant la sélectivité du filtrage.
Le problème de ces approches est que le lien entre la valeur de ϵ et le pourcentage des plus proches voisins exacts effectivement retrouvés dépend fortement de la distribution des points dans la base et est donc difficilement estimable. L'influence de l'augmentation de la taille de la base de données sur ce lien nous semble critique à cet égard, mais nous n'avons trouvé aucun

travaux s'y intéressant.
Dans [178], Zezula et al. ont appliqué le principe de la recherche de ϵ-plus proches voisins à un M-tree. Ils montrent que des gains de performance d'un facteur 20 peuvent être obtenus en conservant 50 % des K-plus proches voisins exacts. Ils ont cependant constaté que la valeur de l'erreur relative mesurée était souvent très inférieure à ϵ et qu'il était donc difficile de contrôler l'erreur relative réellement obtenue. Pour remédier à cela, Ciaccia et al. [46] ont proposé une méthode approximative, utilisant également un M-tree, qui permet de contrôler la probabilité qu'un point retrouvé ait une erreur relative supérieure à ϵ. La technique est basée sur une analyse de la distribution des distances entre les points de la base, qui permet de déduire une condition d'arrêt lors de la recherche.
Dans [169], Weber et al. proposent une deuxième version approximative du VA-file pour la recherche approximative de K-plus proches voisins par des requêtes de type ϵ-plus proches voisin. Le choix de ϵ est déterminé empiriquement afin de trouver un bon compromis entre le gain de performances et la précision des résultats obtenus. Les gains de performance sont typiquement de l'ordre d'un facteur 5 en conservant 80 % de plus proches voisins exacts.

L'équivalent de cette approximation géométrique à une recherche à ϵ-près n'a pas vraiment d'intérêt puisque cela revient en fait à réduire directement le rayon de recherche ϵ et à appliquer normalement les méthodes de recherche exacte. Il serait pourtant intéressant d'étudier l'impact de cette réduction en fonction de la distribution des points dans la base et de l'augmentation de la taille de la base, ce qui à notre connaissance n'a pas été traité dans le cadre de la recherche par similarité approximative, qui se focalise généralement uniquement sur les requêtes de K-plus proches voisins.

Le P-sphere Tree *(classe VS)*

Le P-sphere Tree [79] est une structure d'indexation vectorielle dont l'originalité est que les vecteurs de données peuvent être dupliqués. Comme dans les structures d'indexation classiques, un arbre permet de stocker les données dans les feuilles et les nœuds contiennent des clés correspondant à des régions englobantes, en l'occurrence des hyper-sphères. La différence majeure, est que les vecteurs appartenant à plusieurs régions sont dupliqués dans toutes les feuilles correspondantes. Ainsi, dans le cas d'un chevauchement de plusieurs formes englobantes, il suffit de visiter une seule d'entre elles pour trouver le point recherché. Pour les recherches par similarité, la méthode se limite à la comparaison des vecteurs contenus dans la forme englobante la plus proche de la requête. L'approximation est donc due à l'élimination des vecteurs les plus proches contenus dans d'autres hyper-sphères, dans des régions où il n'y a pas de recouvrement avec la sphère sélectionnée. L'avantage de cette technique, outre sa simplicité, est qu'il est possible d'affirmer de manière déterministe si le vecteur le plus proche ainsi approximé est le plus proche voisin exact ou s'il est possible qu'un autre voisin plus proche existe dans les autres hyper-sphères. Pour être sûr que le vecteur trouvé est effectivement le plus proche voisin, il suffit en effet que la distance entre ce point et la requête ajoutée à la distance entre la requête et le centre de la sphère soit inférieure au rayon de la sphère. Cette propriété n'est vraie que parce que tous les points contenus dans la sphère ont effectivement été comparés à la requête lors du parcours de la feuille.
La probabilité de retrouver le plus proche voisin exact peut être estimée par un échantillon de vecteurs requêtes, sans avoir à faire une recherche exacte par une autre méthode. Le paramètre permettant de régler cette probabilité est la taille des feuilles de l'arbre qui ont un impact direct

sur l'étendue de la couverture des hyper-sphères. L'inconvénient de cette technique est que le coût de stockage peut devenir prohibitif à cause de la duplication des données. Ainsi, pour avoir des probabilités élevées, la base peut être multipliée par un facteur 100 dans les cas extrêmes.

Techniques basées sur une classification des données

Les phénomènes liés à la *malédiction de la dimension* (cf. section 3.3) sont particulièrement contraignants dans le cas d'une distribution uniforme des descripteurs de la base. Dans le cas de données naturellement regroupées en agrégats, l'impact de la dimension est moindre. Dans de telles distributions, en effet, certains ensembles de points ont tendance à rester proches les uns des autres malgré l'augmentation de la dimension. Dans le cas de la recherche d'images par le contenu, il est légitime de supposer que ce type de distribution est effectivement observé en raison de l'objectif même de l'heuristique : lorsque l'on tente de généraliser un concept de similarité ou de se rendre invariant à certaines transformations de l'image, on s'attend en effet à ce que les descripteurs de plusieurs images similaires soient très proches, tout en étant éloignés des autres ensembles.
L'utilisation de techniques basées sur une classification des données est donc naturellement apparue dans le cadre de la recherche par similarité. L'idée de base est que des descripteurs similaires ont de grandes chances de se situer dans un même agrégat. Ces approches étant pour la plupart assez récentes, l'intérêt de la recherche approximative pour les requêtes par similarité y est presque systématiquement exploité. C'est pourquoi, bien que des recherches exactes soient également envisageables, nous n'avons pas cité ces méthodes auparavant. La grande majorité d'entre elles ne s'intéressent qu'à l'approximation des requêtes de type K-plus proches voisins.
Le principe de ces méthodes est de regrouper les données de la base lors d'une phase hors-ligne. Les agrégats ainsi formés sont ensuite stockés individuellement sur le disque, dans un fichier. Lors de la recherche, une première étape sélectionne les agrégats les plus pertinents pour la requête, charge les fichiers correspondant en mémoire et traite séquentiellement les descripteurs contenus dans ces fichiers. L'approximation se fait au moment de la sélection des agrégats. Plus le nombre d'agrégats traités est grand et meilleure est la précision des résultats.
L'inconvénient principal de ce genre d'approche est qu'elles ne sont performantes que lorsque les données sont effectivement fortement regroupées sous forme d'agrégats, ce qui n'est pas toujours le cas, y compris dans certaines applications de recherche d'images par le contenu. De plus, le coût de la phase hors ligne de regroupement des données est assez élevé (généralement en $O(N^2)$). Enfin les coûts de recherche par similarité sont généralement linéaires avec la taille de la base [114, 19].

Les méthodes se différencient entre elles, par l'algorithme de regroupement de données utilisé et par l'étape de filtrage des agrégats. Dans certaines approches, la réduction du nombre d'agrégats se fait uniquement par arrêt prématuré de la recherche. Tous les agrégats sont d'abord ordonnés par un critère de proximité de la requête, puis, seul un nombre arbitraire d'agrégats sont visités. Ce genre d'approche n'offre aucun contrôle a priori de la qualité des résultats. C'est le cas de la méthode CLINDEX [114] *(classe VS)* dont l'algorithme de regroupement de données est en revanche très rapide. Lors de la recherche, les agrégats y sont ordonnés par la distance entre leur centroïde et la requête.
La technique proposée par Tuncel et al. [165] *(classe VS)* est également basée sur un arrêt prématuré de la recherche. En revanche, cette approximation de type réduction du nombre de comparaisons, est combinée à une approche de type compression par quantification vectorielle. A notre connaissance, il s'agit de la seule approche combinant avantageusement les deux types

d'approximations. L'algorithme de regroupement de données employé dans cette technique sont les *K-means*.

D'autres méthodes proposées récemment dans la littérature, utilisent une approche probabiliste pour l'étape de sélection des agrégats. Elles permettent d'effectuer un contrôle a priori de la précision des résultats obtenus dans le cas d'une recherche de plus proches voisins.
La méthode DBIN (*Density-Based Indexing* [12], *classe* VS) repose sur un regroupement des données basé sur l'algorithme d'estimation de paramètres EM (*Expectation Maximisation*). La distribution des vecteurs de la base est modélisée par un mélange de lois gaussiennes, chacune représentant un agrégat. L'algorithme EM est appliqué pour estimer la matrice de covariance et le vecteur moyen de ces lois gaussiennes. Lors de la recherche, les agrégats sont d'abord ordonnés selon la probabilité que la requête appartienne à chacun d'entre eux. Les résultats de la recherche séquentielle de l'agrégat courant sont alors utilisés pour mettre à jour la probabilité qu'un vecteur meilleur que le plus proche actuellement trouvé, se trouve dans les agrégats suivants. La recherche s'arrête lorsque cette probabilité est inférieure au seuil fixé par l'utilisateur. L'inconvénient de cette technique est liée à l'utilisation même de l'algorithme EM. Outre sa complexité de calcul, il y a en effet une limitation du nombre de paramètres estimables en fonction du nombre de données. Cela se traduit par une limitation du nombre d'agrégats, alors que certaines données réelles sont constituées d'un grand nombre d'agrégats de petites tailles.
Dans [19], Berrani propose également une méthode basée sur une sélection probabiliste des agrégats. L'algorithme de regroupement utilisé est une adaptation de l'algorithme hiérarchique BIRCH ([179]). Les agrégats ainsi obtenus ont la propriété d'être contenus dans des hyper-sphères englobantes. L'approximation effectuée lors de l'étape de filtrage des agrégats consiste à réduire le rayon de ces hyper-sphères englobantes, comme dans le cas d'une recherche de ϵ-plus proches voisins. La différence majeure est qu'ici, les rayons réduits sont calculés de manière probabiliste, hors-ligne, en fonction des différentes valeurs que peut prendre un paramètre global sur la précision des résultats. De plus, la réduction du rayon de chaque agrégat est estimée de manière locale en fonction du paramètre global. Le calcul de la probabilité suppose uniquement que la distribution des points à l'intérieur d'un agrégat est isotrope et il est montré que même lorsque ce n'est pas le cas, l'estimation de la précision reste très juste. Un comparatif avec la méthode CLINDEX montre que cette méthode est, à précision égale, de 10 à 70 fois plus performante. L'inconvénient majeur est que le temps de recherche est linéaire en fonction de la taille de la base.

Toutes les méthodes de recherche présentées jusqu'ici ne permettent d'indexer que des données métriques (VS ou MS). Il existe cependant des méthodes non métriques souvent spécifiquement développées pour une mesure de similarité particulière. La structure d'indexation *DynDex* [78] a par exemple été proposée pour indexer la fonction dynamique partielle, qui n'est pas une distance (*DPF*, cf. section 1.3.3). Le principe est le même que les méthodes basées sur une classification des données présentées dans cette section, sauf que l'algorithme de regroupement de données est un algorithme non métrique.

3.5 Synthèse

Ce chapitre a été dédié à l'état de l'art des méthodes de recherche par similarité. Nous avons d'abord détaillé certaines des structures d'indexation multidimensionnelle les plus utilisées. Dans ces techniques, les données sont stockées dans les feuilles d'un arbre généralement équilibré. Elles ont principalement été développées pour gérer des bases de données de grande taille stockées sur

disque, en permettant plusieurs types d'opérations dynamiques comme les insertions, les suppressions, les requêtes par point ou bien les requêtes par similarité. Elles ne sont en revanche pas optimales pour la recherche de descripteurs par similarité et leurs performances se dégradent rapidement lorsque la dimension des données dépassent $D = 16$.
Pour les données de dimension très élevée, nous avons vu que la recherche séquentielle était bien souvent meilleure. De ce constat sont nées des méthodes séquentielles accélérées, utilisant des versions compressées des vecteurs. Ces techniques garantissent un gain de performance par rapport à une recherche séquentielle mais restent linéaires en fonction de la taille de la base. Elles n'ont de sens que pour un stockage des données sur disque.
Nous avons ensuite vu que les méthodes de recherche approximatives étaient particulièrement encourageantes pour des applications de recherche par similarité de descripteurs d'images. Elles permettent un gain significatif des performances contre de faibles pertes de qualité de la recherche, ayant peu d'impact sur le résultat final. Elles permettent en outre de réduire les effets de la dimension élevée des descripteurs. La grande majorité des travaux portent uniquement sur la recherche approximative de K-plus proches voisins et à la manière de contrôler la précision de ce type de requêtes. Ces méthodes sont généralement utilisées sur disque, mais un fonctionnement en mémoire est clairement envisageable.
Dans le chapitre suivant, où nous présentons notre méthode de recherche approximative de descripteurs, nous rediscuterons de l'applicabilité des méthodes vues dans ce chapitre dans le cadre plus spécifique de la détection de copies. Nous verrons ainsi pourquoi la recherche approximative de K-plus proches voisins ne satisfait pas notre application. Nous verrons également que l'augmentation de la taille de la base est une contrainte majeure qui ne doit pas être négligée devant l'influence de la dimension des descripteurs. Nous verrons enfin pourquoi une méthode sur disque ne nous semble pas envisageable dans le contexte vidéo temps réel.

Chapitre 4

Description de la méthode proposée pour la recherche de signatures

4.1 Introduction

Les méthodes approximatives de recherche de K-plus proches voisins, récemment proposées dans la littérature dans le cadre de la recherche par similarité, ont montré que de faibles pertes sur la qualité des résultats pouvait permettre des gains de performances significatifs. Ce principe est particulièrement justifié pour la recherche d'images par le contenu : la recherche des K-plus proches voisins dans une base de descripteurs est en effet déjà une heuristique de l'objectif réel, à savoir trouver les images les plus pertinentes pour l'utilisateur.

Définition Heuristique - *Une **heuristique** est une méthode de résolution de problèmes, non fondée sur un modèle formel et qui n'aboutit pas nécessairement à une solution optimale*

Une légère erreur dans les résultats de la recherche ne dégradera donc pas forcément les résultats présentés à l'utilisateur. Pis, ceux-ci peuvent même être meilleurs car rien ne garantit, par exemple, que le 2^{eme} plus proche voisin ne soit, à ses yeux, plus pertinent que le 4^{eme} plus proche voisin. La recherche approximative est simplement une heuristique différente.
Si les K-plus proches voisins dans la base des descripteurs correspondaient systématiquement aux K résultats les plus pertinents pour l'utilisateur, une erreur dans les K-plus proches voisins serait plus ennuyeuse. En revanche, un contrôle précis de cette erreur prendrait alors tout son sens, puisqu'il contrôlerait directement la qualité des résultats présentés à l'utilisateur. Dans le cas d'une heuristique, le contrôle de la précision de la recherche ne donne pas d'indication directe sur la pertinence des résultats.
Le cas de la détection de copies est différent en ce que l'ensemble des transformations de l'image auxquelles on doit se rendre invariant est plus facilement identifiable. L'influence de cet ensemble de transformations sur la distorsion des signatures peut donc être analysée de manière statistique et on peut tenter d'en tirer un modèle probabiliste. L'approche que nous proposons consiste à effectuer la recherche par similarité des signatures en utilisant un tel modèle. Il nous semble en effet que plutôt que d'approximer précisément une requête géométrique dont on ne connaît pas le lien réel avec les résultats attendus, il vaut mieux essayer de modéliser le lien entre la similarité des images et celles des signatures. Le principe général est de retourner comme résultat, l'ensemble des signatures de la base ayant le plus de chances d'être issues d'une copie du motif correspondant à la signature candidate. Ce nouveau type de requêtes, dit **statistique**, est introduit à la section 4.2.

La section 4.3 s'intéresse à la mise en œuvre de ces requêtes dans une structure d'indexation. La méthode originale proposée comporte une phase *hors-ligne*, consistant à trier physiquement les signatures selon leur position sur une courbe de Hilbert. Lors de la phase *en ligne*, la base ainsi triée est chargée en mémoire. Cet ordonnancement permet d'accéder très simplement à la base par une simple table d'index monodimensionnelle, tout en limitant les sauts dans la mémoire. Il s'agit d'une technique statique, c'est-à-dire que les insertions et les suppressions dynamiques ne sont pas possibles. C'est à ce prix que les données peuvent être physiquement ordonnées. Il s'agit principalement d'un système fonctionnant en mémoire centrale, bien que le principe soit facilement transposable sur disque.
Pour les tailles de base dépassant la taille mémoire, nous proposons un mode de fonctionnement par requêtes groupées reposant sur le même principe mais chargeant la base en mémoire en plusieurs blocs. Le système ne fonctionne plus en temps réel instantané mais en temps réel différé, c'est à dire que les résultats sont disponibles après un certain laps de temps pouvant atteindre plusieurs heures pour les bases les plus volumineuses. Cela permet cependant de conserver un temps de recherche moyen du même ordre de grandeur que celui d'un système fonctionnant entièrement en mémoire avec des requêtes uniques.
Dans la section 4.4, nous proposons une modélisation des coûts de recherche de notre technique, permettant d'optimiser le principal paramètre de la méthode, à savoir la profondeur de la partition de l'espace. Nous discuterons les hypothèses de cette modélisation et nous simulerons l'évolution des coûts en fonction de l'évolution de la base de données.

4.2 Requêtes statistiques

Cette section est structurée de la manière suivante : nous verrons d'abord pourquoi les requêtes à ϵ-près sont plus appropriées à la recherche de signatures locales que les requêtes de type K-plus proches voisins (section 4.2.1). Nous montrerons ensuite que l'augmentation de la base de données n'a pas la même influence sur la complexité de la recherche pour les deux types de requêtes (section 4.2.2). Dans la section 4.2.3, nous discuterons de l'extension du principe de recherche approximative aux requêtes à ϵ-près avant d'introduire le concept de requêtes statistiques dans la section 4.2.4.

4.2.1 Quelles requêtes pour la détection de copies à l'aide de signatures ?

Nous avons vu dans les chapitres précédents que parmi les deux types de requêtes utilisés en recherche par similarité, les requêtes de type K-plus proches voisins avaient été largement plus étudiées et utilisées, que les requêtes à ϵ-près. Dans le cas de la recherche approximative notamment (section 3.4), seules les requêtes de type K-plus proches voisins sont envisagées. Cette préférence se justifie d'un point de vue applicatif : dans un système de recherche par le contenu classique, la recherche des K-plus proches voisins dans la base des descripteurs est en général directement assimilée à la recherche des K-plus proches objets. Le principal avantage de ces requêtes est de garantir à l'utilisateur un nombre constant de résultats sans aucun a priori sur la base de descripteurs. Ils sont présentés par ordre décroissant de similarité, laissant ainsi à l'utilisateur le soin de l'analyse de leur pertinence. Le choix du paramètre K est plus guidé par des contraintes d'affichage et d'ergonomie que par une cohérence des résultats.
Dans ce contexte, une recherche à ϵ-près est beaucoup plus contraignante puisque le nombre de résultats n'est pas fixe. Elle peut donner lieu à un ensemble de résultats vide ou à un nombre de résultats très important. Le réglage de ϵ est ainsi problématique et nécessite d'avoir une certaine

connaissance de la base des descripteurs.

Pour la détection de copies par le contenu, une recherche à un rayon près est cependant beaucoup plus cohérente qu'une recherche de K-plus proches voisins. L'objectif n'est en effet pas de retrouver un certain nombre d'objets similaires pour tous les objets du flux candidat mais avant tout de décider quels sont les objets de la base qui sont effectivement des copies (ou des originaux) de l'objet candidat. Une notion de seuil sur la mesure de disimilarité des signatures est donc plus adaptée.
Le principal argument en défaveur des requêtes de type K-plus proches voisins pour la détection de copies est la **variabilité du nombre de signatures similaires pertinentes**. Celle-ci est due à plusieurs raisons :

- Le nombre de copies d'un même objet dans le catalogue de référence peut être très variable. Dans une base contenant près de 10 000 heures de vidéo, nous avons trouvé certains extraits présents plus de 700 fois (introduction du journal télévisé par exemple) tandis que d'autres ne sont présents qu'une fois. Que les signatures soient globales ou locales, cette variabilité se répercute systématiquement sur la base des signatures dans laquelle le nombre de signatures pertinentes peut être lui aussi très variable. Si la valeur de K est trop faible, certaines copies ne seront pas identifiées car leur signature n'aura pas été retrouvée. Si K est trop élevé, le nombre de fausses alarmes augmente de manière conséquente. Notons également que la plupart des objets candidats n'appartiennent pas à la base et qu'on s'attend pour ceux-ci à retrouver beaucoup moins de signatures similaires que dans le cas d'une copie.
- Le nombre de signatures pertinentes dépend de la taille de la base. La signature d'une copie peut être le plus proche voisin dans une petite base et le 50^{eme} plus proche voisin dans une base très volumineuse.

- Dans le cas des signatures locales, la variabilité du nombre de signatures pertinentes est accentuée par la variabilité des motifs locaux. Certaines signatures sont en effet présentes un grand nombre de fois (par exemple des points d'intérêt dans le décors) tandis que d'autres sont quasiment uniques.

Nous pensons donc qu'une recherche à un rayon-près est plus adaptée qu'une recherche des K-plus proches voisins pour la détection de copies car elle permet de prendre en compte la variabilité du nombre de signatures similaires pertinentes.

4.2.2 Comportements asymptotiques de la recherche

En plus de son influence sur l'efficacité de la détection de copies, le choix du type de requête est aussi décisif pour le temps de la recherche. Comme nous l'avons vu au chapitre 3, la plupart des techniques de recherche de descripteurs similaires possèdent une étape de filtrage, consistant à sélectionner certaines régions englobantes de l'espace, et une étape de raffinement consistant à comparer le descripteur candidat avec tous les descripteurs contenus dans les régions sélectionnées. Sans entrer dans les détails de chaque méthode, le coût d'une recherche dépend principalement de l'étendue spatiale de la requête [11, 1]. C'est en effet elle qui déterminera le nombre d'opérations de l'étape de filtrage et le nombre de distances à calculer lors de l'étape de raffinement. Dans le cas d'une requête à ϵ-près, l'étendue spatiale de la requête est une hyper-sphère de rayon ϵ et elle est constante. Dans le cas d'une requête de type K-plus proches voisins, l'étendue de la requête est l'hyper-sphère minimale englobante, c'est-à-dire l'hyper-sphère ayant pour rayon la distance du K^{eme} plus proche voisin. Elle dépend donc dans ce cas de la distribu-

tion des points dans la base. Le but de cette section est d'étudier l'influence de la dimension de l'espace et du nombre d'éléments dans la base sur la complexité de la recherche pour ces deux types de requêtes.

Comportement asymptotique en fonction de la dimension

L'influence de la dimension des données sur les performances d'un système d'indexation multidimensionnelle a été un enjeu majeur des travaux sur ce sujet. Dans le cas de la recherche d'images par le contenu, la complexification progressive des descripteurs est allée de pair avec une augmentation de leur dimension. Parallèlement à cela, on a constaté que les performances des structures d'indexation multidimensionnelle classiques se dégradaient rapidement avec l'augmentation de la dimension, avec une complexité de recherche croissant de manière exponentielle. Il s'agit là d'un des aspects auxquels fait allusion le terme de *malédiction de la dimension* [17, 19]. Cela a conduit à de nouvelles structures d'indexation tentant de contourner le phénomène [17, 170] et également à des travaux théoriques étudiant les propriétés des espaces de grande dimension et leurs influences sur des requêtes par similarité [22].
Un des résultats surprenant pour la recherche de K-plus proches voisins est le théorème de Beyer et al. [22], qui sous certaines conditions sur la distribution des données et des requêtes, énoncent que la distance du plus proche voisin tend vers la distance du point le plus éloigné de la base, lorsque la dimension augmente. Les auteurs affirment que, pour ce type de distributions et au delà d'une dimension de l'ordre de $D = 15$, la recherche des K-plus proches voisins n'a pas de sens puisque tous les points sont à peu près à la même distance. Ils montrent notamment qu'une distribution uniforme et indépendante suivant toutes les dimensions satisfait les conditions requises. De nombreux modèles de coût pour la recherche de K-plus proches voisins étant basés sur cette distribution [26, 117, 7], leur pertinence en est remise en cause. Il semblerait en effet que, pour qu'une recherche de K-plus proches voisins ne respectent pas le théorème de Beyer, il faille que la distribution soit bien différente d'une distribution uniforme : une distribution pour laquelle les descripteurs sont regroupés sous forme d'agrégats par exemple, ou bien une distribution dont la dimension intrinsèque des données est bien inférieure à la dimension réelle. Quoi qu'il en soit, il y a de forts risques pour que les modèles de coût calculés sur une distribution uniforme soient erronés pour des distributions ne satisfaisant pas le théorème de Beyer.
Notons que ce dernier n'est pas en contradiction avec le principe même de l'heuristique de recherche des K-plus proches voisins pour la recherche d'images par le contenu. En effet, l'objectif d'une dimension élevée pour les descripteurs n'est pas d'obtenir une distribution uniforme des vecteurs. Il s'agit plutôt de concentrer les descripteurs d'objets visuellement similaires dans des agrégats, tout en augmentant la dispersion des agrégats entre eux. Les distributions observées sur des données réelles semblent ainsi beaucoup moins sensibles à la dimension. Dans [53], il est observé que les performances des systèmes de recherche des K-plus proches voisins s'améliorent nettement lorsque l'on passe de données synthétiques uniformes à des données réelles ; le phénomène est en général inversé pour les requêtes à ϵ-près.
Pour la détection de copies, les choses sont différentes puisque l'objectif est de discriminer chacun des objets de la base. Ainsi, contrairement à la recherche d'images par le contenu, une distribution uniforme des signatures dans la base peut être un objectif cohérent si tous les objets référencés sont distincts les uns des autres. Dans ce cas idéal, le théorème de Beyer s'appliquerait ce qui plaide encore une fois contre l'utilisation d'une recherche des K-plus proches voisins pour ce type d'application.

L'influence de la dimension sur une requête à ϵ-près est tout à fait différente, puisque la com-

plexité de la recherche est beaucoup moins dépendante de la distribution des points, du fait que le volume couvert par la requête est fixé uniquement par ϵ. Le volume d'une requête à ϵ-près dans un espace de dimension D est donné par le volume d'une hyper-sphère :

$$V_{HS}(\epsilon, D) = \frac{\pi^{\frac{D}{2}} \epsilon^D}{\Gamma(\frac{D}{2} + 1)} \tag{4.1}$$

où $\Gamma(.)$ représente la fonction gamma. La propriété remarquable suivante est vérifiée :

$$\lim_{D \to \infty} V_{HS}(\epsilon, D) = 0$$

La fonction passe généralement par un maximum dans les faibles dimensions avant d'être strictement décroissante. A ϵ constant, la complexité d'une recherche à ϵ-près semble donc décroître avec la dimension puisque l'on tend vers une recherche ponctuelle. Il faut cependant relativiser cette constatation par le fait qu'ϵ n'a en réalité aucune raison de rester constant. La distance maximale dans un espace de dimension croît en effet en $\sqrt{D}$ et il semble intuitivement logique qu'ϵ croisse également. Il faut également replacer la requête à ϵ-près dans sa signification vis-à-vis de l'application. La valeur de ϵ est censée représenter la variation maximale d'une signature lorsque l'image ou le motif correspondant est soumis à l'ensemble des transformations tolérées. Sa valeur ne peut donc être fixée que par une étude statistique ou une modélisation des distorsions de la signature vis à vis des transformations. L'ajout de nouvelles composantes à une signature nécessite donc de fixer une nouvelle valeur de ϵ. Nous reviendrons sur la modélisation des distorsions de la signature dans la suite de ce chapitre.

Comportement asymptotique en fonction de la taille de la base

L'étude des effets de l'augmentation de la dimension s'est souvent faite au détriment de l'influence de la taille de la base. Le théorème de Beyer n'est par exemple vrai que dans le cas d'un nombre fixe d'éléments dans la base. L'influence de la taille de la base sur le rapport entre la distance du plus proche voisin et celle du point le plus éloigné n'est pas étudiée. Seule la limite en $D \to \infty$ est considérée. Combien de fois faut-il multiplier la taille de la base pour compenser l'effet de la dimension ? Cette question reste sans réponse.
Or, les tailles des bases d'images présentes dans la réalité étant sans cesse croissante, c'est un paramètre que l'on ne peut négliger. Depuis quelques années, certaines études théoriques sur la modélisation du coût d'une recherche des K-plus proches voisins ont pris en compte l'influence des deux paramètres. Berchtold et al. [14] ont ainsi montré que la valeur de la dimension à partir de laquelle les structures d'indexation multidimensionnelle deviennent moins performantes qu'un simple parcours séquentiel dépend du nombre d'éléments dans la base. Notons toutefois que cette influence est limitée par rapport à celle de la dimension et que pour les très grandes dimensions (> 60), un parcours séquentiel reste plus efficace même si la taille des bases actuelles est multipliée par 1 million.

L'influence de la taille de la base n'en reste pas moins un paramètre important à étudier. D'un point de vue applicatif, la dimension de la signature est en effet une constante tandis que la base est susceptible de croître significativement. Nous allons maintenant illustrer l'influence de l'augmentation de la taille de la base sur les deux types de requêtes utilisées en recherche par similarité.
Dans la suite, nous considérons que la base de signatures est un ensemble de N vecteurs dans un espace vectoriel $E = [0, 255]^D$ de dimension D. La distance associée est la norme L_2. Nous

supposons que les vecteurs sont répartis selon une distribution de densité de probabilité $p(\boldsymbol{X})$, $\boldsymbol{X} \in E$. Celle-ci est supposée continue et **indépendante de** N. Cette dernière hypothèse est fondamentale et nous reviendrons sur sa pertinence par la suite.
La densité locale $e(\boldsymbol{X})$, représentant localement le nombre de vecteurs par unité de volume peut être estimée par :

$$e(\boldsymbol{X}) = p(\boldsymbol{X})N$$

L'espérance $nb(V)$ du nombre de vecteurs contenus dans un hyper-volume quelconque V, est alors :

$$nb(V) = \int_V e(\boldsymbol{X})\boldsymbol{dX} = N \int_V p(\boldsymbol{X})\boldsymbol{dX} \tag{4.2}$$

L'espérance du nombre de vecteurs contenus dans un hyper-volume quelconque de E est donc proportionnel à N. Ceci est en particulier vrai dans le cas d'une requête à ϵ-près où V représente alors une hyper-sphère de rayon ϵ centrée sur la signature candidate (de volume $V_{HS}(\epsilon, D)$). La figure 4.1 illustre cette linéarité pour une requête dans des bases réelles de signatures locales, de tailles croissantes, et pour différentes valeurs de ϵ. La requête a été sélectionnée aléatoirement parmi les signatures de la plus petite base, contenant 77 131 signatures soit environ une heure de vidéo. Chaque base contient toutes les signatures des bases de tailles inférieures.

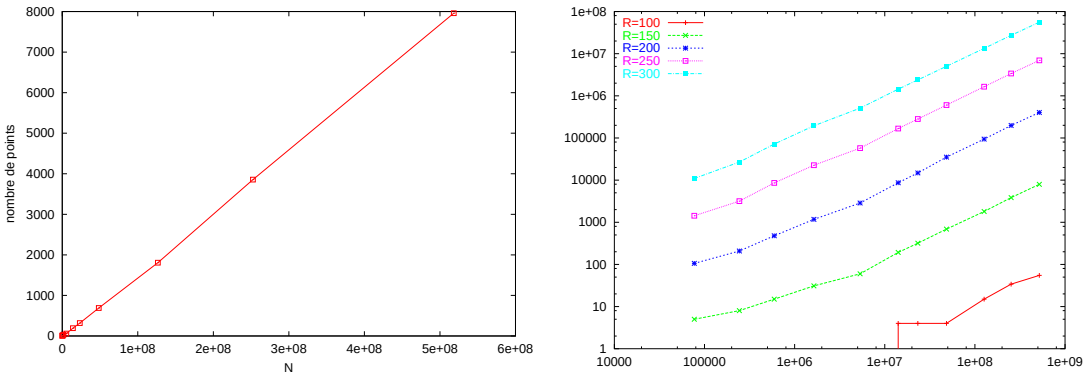


FIG. 4.1 – Nombre de signatures dans une requête à ϵ-près en fonction de la taille N de la base - à gauche : ϵ=150 - à droite : plusieurs valeurs de ϵ, coordonnées logarithmiques

Quelle que soit la structure d'indexation utilisée, tous les résultats d'une recherche à ϵ-près sont comparés avec la signature candidate lors de l'étape de raffinement. D'après l'équation 4.2 cela introduit nécessairement un coût linéaire en fonction de N dans le coût global de la recherche. Ce coût peut être dans un premier temps négligé devant d'autres plus importants et éventuellement sous-linéaires. Le coût global pourra alors être sous-linéaire pour des valeurs de N inférieures à un certain seuil mais sera asymptotiquement linéaire : Pour toute répartition des vecteurs de la base indépendante de N et quelle que soit la technique utilisée, le coût d'une recherche à ϵ-près sera asymptotiquement linéaire (ou surlinéaire). Ce coût linéaire résultant de la comparaison de la signature candidate avec les résultats de la requête sera désormais appelé **coût linéaire minimal d'une recherche à ϵ-près**.
Dans le cas d'une distribution uniforme, l'espérance du nombre de vecteurs contenus dans un hyper-volume V est directement estimé par

$$nb(V) = \frac{V\ N}{256^D}$$

Dans le cas d'une requête à ϵ-près et en considérant que l'espace des signatures est illimité (sans effet de bords), on a :

$$nb\left(V_{HS}(\epsilon, D)\right) = \frac{\pi^{\frac{D}{2}}\epsilon^D}{\Gamma(\frac{D}{2}+1)}\frac{N}{256^D}$$

Pour les requêtes à ϵ-près, le nombre de résultats est linéaire en fonction de la taille de la base tandis que le volume couvert par la requête reste constant. Pour la recherche de K-plus proches voisins, le nombre de résultats est constant et égal à K tandis que le volume de l'hyper-sphère englobante minimale tend vers 0. Ceci peut être facilement montré pour le plus proche voisin dans le cas d'une distribution uniforme. La distance moyenne d_{NN} du plus proche voisin dans une base uniforme [14] peut en effet être estimée par :

$$d_{NN} = 256\sqrt[D]{\frac{\Gamma\left(\frac{D}{2}+1\right)}{N\pi^{\frac{D}{2}}}}$$

et on a :

$$\lim_{N\to\infty} d_{NN} = 0$$

Notons cependant que la vitesse de décroissance est faible puisque celle-ci varie en $N^{-(1/D)}$ (il faut multiplier la taille de la base par 2^D pour que la distance du plus proche voisin soit divisée par 2). Ainsi pour les très grandes dimensions, l'influence de la taille de la base est négligeable, comme nous l'avons déjà évoqué précédemment.
Dans le cas général, on peut constater que l'estimation de la densité locale (nombre de points par unité de volume) tend vers l'infini en tout point, puisque,

$$\lim_{N\to\infty} e(\boldsymbol{X}) = \lim_{N\to\infty} p(\boldsymbol{X})N = \infty \quad \forall \boldsymbol{X}\ tq\ p(\boldsymbol{X}) \neq 0$$

La distance espérée des plus proches voisins ne peut donc que tendre vers 0, en tout point où la densité de probabilité est non nulle. La décroissance de la distance du plus proche voisin a été vérifiée expérimentalement pour des bases réelles de signatures locales, de tailles croissantes.
On constate donc que le volume de l'hyper-sphère englobante minimale d'une requête de type K-plus proches voisins est décroissant avec N et tend vers 0 lorsque N tend vers l'infini. Ainsi, plus la base sera grande, plus la complexité de ce type de requête sera faible par rapport à une requête à ϵ-près. La décroissance de la distance des plus proches voisins avec la taille de la base illustre encore l'inconvénient de ce type de requête pour la détection de copies. **Il est en effet clair que si le paramètre K reste constant alors que la base augmente, de plus en plus de signatures pertinentes seront exclues des résultats de la recherche**. Il faudrait donc augmenter le nombre de plus proches voisins en fonction de la taille de la base. Le réglage de K nécessiterait une connaissance de la distribution des distorsions comme dans le cas de la recherche à ϵ près, mais il faudrait également avoir une connaissance de la distribution des signatures dans la base. L'avantage des requêtes à ϵ près est que le réglage de ϵ ne dépend que de l'ensemble des transformations. Il est indépendant du nombre de signatures dans la base et de la distribution des points dans celle-ci.

4.2.3 Requêtes à ϵ-près et recherche approximative

Nous avons expliqué dans la section 4.2.1 pourquoi une requête à ϵ-près nous semble plus appropriée qu'une requête de type K-plus proches voisins pour la détection de copies à l'aide

de signatures. Le choix de la valeur de ϵ n'a cependant pas encore été discuté, et c'est l'un des objectifs de cette section. Partant de la constatation que la recherche approximative de K-plus proches voisins a permis des gains de performance significatifs, il faut se demander également comment le principe d'une faible perte de qualité pourrait être transposé au cas d'une recherche à ϵ-près.

Rappelons d'abord que l'étape de recherche des signatures locales de la méthode que nous proposons pour la détection de copies vidéo, peut être vue comme un système de détection de copies de motifs locaux. Nous appelons motif local la région spatio-temporelle de la vidéo autour d'un point d'intérêt, caractérisée par une signature locale unique. Il s'agit en fait d'un objet vidéo de taille réduite auquel s'appliquent toutes les transformations globales. La copie d'un motif local est la région spatio-temporelle de même dimension centrée autour du même point d'intérêt dans une copie de l'objet vidéo global. Nous considérons ainsi que le catalogue de référence est un ensemble M de N motifs locaux :

$$M = \{m_n\}, n \in [1, N]$$

caractérisés par un ensemble Σ de N signatures locales constituant la base de signatures :

$$\Sigma = \{\boldsymbol{S_n}\}, n \in [1, N]$$

La fonction permettant de calculer une signature locale $\mathbf{S}$ à partir d'un motif local m est notée :

$$s : M \rightarrow \Sigma \;\; avec \;\; S = \boldsymbol{s(m)}$$

Un motif m_2 est la copie d'un motif m_1 si $m_2 = t(m_1)$, $t \in T$ l'ensemble des transformations tolérées (cf. définition 1.2).
L'objectif d'une recherche à ϵ-près étant de retrouver toutes les signatures issues d'un éventuel motif original du motif candidat, la valeur de ϵ doit être supérieure à la valeur théorique naïve suivante :

$$\epsilon_{max} = \max_{t \in T, m \in M} d\left[\boldsymbol{s}\left(\boldsymbol{m}\right), \boldsymbol{s}\left(\boldsymbol{t}(\boldsymbol{m})\right)\right]$$

où d représente la fonction de disimilarité associée à la recherche à ϵ-près. ϵ_{max} est donc déterminé par la transformation la plus sévère au sens de la fonction de disimilarité. Cette valeur théorique est en réalité difficile à déterminer pour plusieurs raisons :

- T est difficilement connaissable (cf. section 1.2).
- T n'est pas un ensemble fini. Les paramètres de la plupart des transformations envisageables peuvent en effet évoluer de manière continue, même à l'intérieur d'un intervalle borné. Il n'est donc pas possible de parcourir systématiquement cet ensemble.
- certaines transformations ne sont pas déterministes, comme l'ajout de bruit par exemple ; $d\left[\boldsymbol{s}\left(\boldsymbol{m}\right), \boldsymbol{s}\left(\boldsymbol{t}(\boldsymbol{m})\right)\right]$ est dans ce cas une variable aléatoire.
- enfin, dans les bases de très grande taille, il n'est pas envisageable de tester un ensemble de transformations pour tous les motifs du catalogue de référence.

Il serait cependant possible d'estimer une valeur cohérente de ϵ_{max} en considérant un ensemble fini de transformations faisant office de cahier des charges et en l'appliquant à un sous-ensemble de motifs du catalogue de référence sélectionnés aléatoirement.
Une telle valeur de ϵ risque malheureusement d'être assez élevée et de ne pas être vraiment rentable pour le système global. Le taux de fausses alarmes de la recherche, c'est-à-dire le nombre de

signatures non pertinentes situées à une distance inférieure à ϵ, est en effet fortement croissant avec ϵ (dans un espace vectoriel de dimension D, cela s'explique par la croissance en $O(\epsilon^D)$ du volume d'une hyper-sphère de rayon ϵ). Or, une augmentation du taux de fausses alarmes de la recherche peut déstabiliser l'estimation du recalage lors de l'étape de fusion des résultats ou bien augmenter le taux de fausses alarmes de cette même étape. Une réduction du rayon de recherche ϵ peut donc avoir le double avantage d'augmenter les performances globales de la détection tout en diminuant fortement les temps de recherche.
La réduction de ϵ suppose cependant que l'on puisse contrôler l'erreur de l'approximation ainsi introduite. Si on s'en tenait uniquement à un critère géométrique, la mesure de l'erreur consisterait à mesurer la perte volumique engendrée par l'approximation. Si cette mesure a un sens dans le cas de données géométriques, comme c'est le cas en géométrie algorithmique par exemple [6] elle n'est pas adaptée à la recherche de signatures par similarité.
La véritable erreur doit en effet estimer le pourcentage de signatures réellement perdues et non la perte volumique. Dans le cas général, la mesure de l'erreur entre une requête à ϵ-près et une requête à ϵ'-près ($\epsilon' < \epsilon$) doit donc tenir compte de la distribution des descripteurs pertinents pour la requête. Dans le cas de la détection de copies, l'objectif n'est pas de retrouver toutes les signatures similaires à la requête, mais uniquement celles qui sont effectivement issues d'un original du motif candidat. La mesure de l'erreur doit donc tenir compte de la distribution des distorsions engendrées par les transformations et non de la distribution des signatures de la base.

Dans la suite de notre étude, nous considérons que l'espace E des signatures est un espace vectoriel de dimension D muni de la distance euclidienne d et nous définissons la *distorsion* par :

Définition Distorsion - *Soit un motif m et $t(m)$ une copie de ce motif. La distorsion, notée $\boldsymbol{\Delta S}$, est le vecteur différence entre la signature de m et la signature de $t(m)$, soit*

$$\boldsymbol{\Delta S} = \boldsymbol{s}\,(\boldsymbol{m}) - \boldsymbol{s}\,(\boldsymbol{t}(\boldsymbol{m}))$$

$\|\boldsymbol{\Delta S}\| = d\left[\boldsymbol{s}\,(\boldsymbol{m}), \boldsymbol{s}\,(\boldsymbol{t}(\boldsymbol{m}))\right]$ représente la norme euclidienne de la distorsion et $\boldsymbol{s}\,(\boldsymbol{t}(\boldsymbol{m}))$ sera qualifiée de signature distordue.

Nous introduisons un rayon de recherche ϵ_α, permettant d'espérer retrouver α % des signatures pertinentes pour la requête. Nous définissons ainsi une requête à ϵ_α-près par :

Définition Requête à ϵ_α-près - *Une requête à ϵ_α-près est une requête à ϵ-près pour laquelle $\epsilon = \epsilon_\alpha$ et :*

$$\int_0^{\epsilon_\alpha} p_{\|\boldsymbol{\Delta S}\|}(r)dr = P_{\|\boldsymbol{\Delta S}\|}(\epsilon_\alpha) = \alpha \tag{4.3}$$

où $p_{\|\boldsymbol{\Delta S}\|}(r)$ représente la densité de probabilité de la norme de la distorsion et $P_{\|\boldsymbol{\Delta S}\|}(\epsilon_\alpha)$ la fonction de répartition de la norme de la distorsion.

Une approche naïve pour contrôler la précision d'une requête approximative à ϵ-près consisterait à considérer la perte volumique engendrée. Cela équivaudrait en fait à considérer que la densité de probabilité de la distorsion est uniforme à l'intérieur d'une hyper-sphère de rayon ϵ. Dans la suite, nous étudions l'hypothèse d'une telle loi de probabilité afin de discuter sa pertinence et illustrer les effets non intuitifs liés aux espaces de grande dimension. A titre comparatif et pour la suite de notre discussion, nous étudions également l'hypothèse d'une loi de probabilité gaussienne, indépendante pour chaque composante. Enfin, nous confronterons ces deux modèles à quelques distributions réelles estimées en appliquant certaines transformations à des vidéos

réelles.

Les valeurs numériques de cette étude correspondent à l'espace vectoriel $E = [0, 255]^D$ muni de la distance euclidienne. Nous rappelons que la distance euclidienne maximale dans un tel espace vaut :

$$d_E^{max}(D) = 255\sqrt{D}$$

Les signatures locales de notre méthode ayant une dimension égale à $D = 20$, on a

$$d_E^{max}(20) = 255\sqrt{20} \approx 1140,39$$

Loi de probabilité sphérique uniforme

Toutes les signatures pertinentes sont dans ce cas équiprobables à l'intérieur d'une hyper-sphère centrée autour de la requête. La probabilité qu'un point se situe dans une région donnée de l'espace est proportionnelle au volume intercepté par cette région (cf. équation 4.1). La fonction de répartition de la norme de la distorsion est alors :

$$P_{\|\boldsymbol{\Delta S}\|}(r) = F_{HS}(r) = \begin{cases} 0 & si \ \ |r| > \epsilon \\ \frac{V_{HS}(r,D)}{V_{HS}(\epsilon,D)} & si \ \ |r| \leq \epsilon \end{cases}$$

La densité de probabilité correspondante est :

$$p_{\|\boldsymbol{\Delta S}\|}(r) = f_{HS}(r) = \frac{dF_{HS}(r)}{dr} = \begin{cases} 0 & si \ si \, |r| > \epsilon \\ \frac{A_{HS}(r,D)}{V_{HS}(\epsilon,D)} & si \ \ |r| \leq \epsilon \end{cases}$$

où $A_{HS}(r, D)$ représente la surface d'une hyper-sphère de rayon r dans un espace de dimension D :

$$A_{HS}(r, D) = \frac{dV_{HS}(r, D)}{dr} = \frac{2\pi^{\frac{D}{2}} r^{D-1}}{\Gamma(\frac{D}{2})}$$

On peut également exprimer la densité de probabilité d'une seule composante de la distorsion ΔS_j $(j \in [1, D])$ par :

$$p_{\Delta S_j}(x) = f_{HS}^j(x) = \begin{cases} 0 & si \ si \, |x| > \epsilon \\ \frac{V_{HS}(\sqrt{\epsilon^2 - x^2}, D-1)}{V_{HS}(\epsilon,D)} & si \ si \, |x| \leq \epsilon \end{cases}$$

Ces trois fonctions sont représentées sur la figure 4.2, pour $D = 20$ et $\epsilon = 40$.

Il est possible de montrer que, lorsque la dimension augmente, la densité de probabilité d'une seule composante converge vers la densité de probabilité d'une loi normale monodimensionnelle de moyenne nulle et d'écart type décroissant :

$$\sigma_{Hx} = \frac{\epsilon}{\sqrt{D}} \tag{4.4}$$

Démonstration cf. annexe D.1

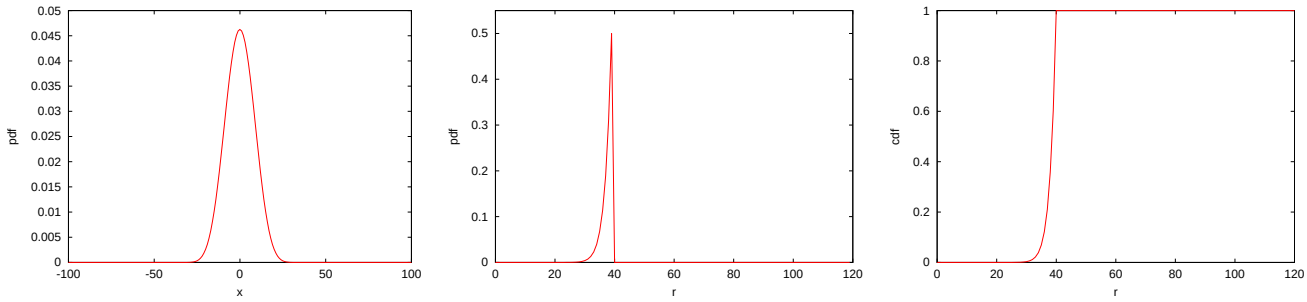


FIG. 4.2 – Loi de probabilité uniforme dans une hyper-sphère - à gauche : densité de probabilité d'une seule composante de la distorsion - au milieu : densité de probabilité de la norme de la distorsion - à droite : fonction de répartition de la norme de la distorsion

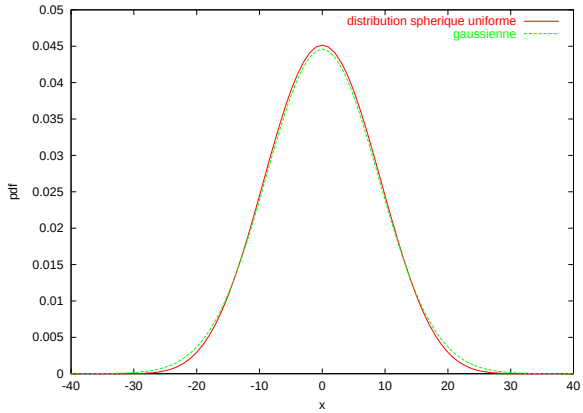


FIG. 4.3 – $f^i_{HS}(x)$ et $f_{\mathcal{N}(0,\sigma_{Hx})}(x)$ pour $\epsilon = 40$ et $D = 20$

Ceci est illustré par le figure 4.3 représentant les deux densités de probabilité pour $\epsilon = 40$ et $D = 20$. Cette remarque est intéressante puisqu'elle montre que, dans des espaces de grandes dimensions, une loi uniforme sphérique et une loi normale ne peuvent être différenciées en observant uniquement la distribution d'une composante seule. Notons également que lorsque la dimension augmente, σ_{Hx} tend vers 0, et la distribution se rapproche alors d'un dirac en zéro.

Une autre propriété remarquable d'une telle loi dans les espaces de grande dimension est que la probabilité qu'un point se situe près de la surface de l'hyper-sphère tend vers 1 lorsque la dimension augmente. En effet, si on note $P_{\epsilon-\eta}(\eta)$ la probabilité qu'une signature se situe à une distance comprise entre ϵ et $\epsilon - \eta$ du centre de l'hyper-sphère, on a :

$$P_{\epsilon-\eta}(\eta) = F_{HS}(\epsilon) - F_{HS}(\epsilon - \eta)$$

et

$$\lim_{D\to\infty} P_{\epsilon-\eta}(\eta) = \lim_{D\to\infty} 1 - \left(\frac{\epsilon - \eta}{\epsilon}\right)^D = 1$$

Cette propriété est illustrée par la figure 4.4 représentant la fonction de répartition de la norme de la distorsion $F_{HS}(r)$ pour plusieurs dimensions. Elle est intéressante dans le cadre de notre discussion sur l'approximation d'une requête à ϵ-près puisque l'on constate que, pour une distribution uniforme sphérique dans un espace de grande dimension, la grande majorité des vecteurs d'un échantillon seront situés à la surface de l'hyper-sphère. Si la distorsion des signatures suivait réellement une telle loi, une réduction de ϵ, même faible, engendrerait donc la perte d'une part importante des signatures pertinentes, sans pour autant diminuer significativement le temps de recherche.

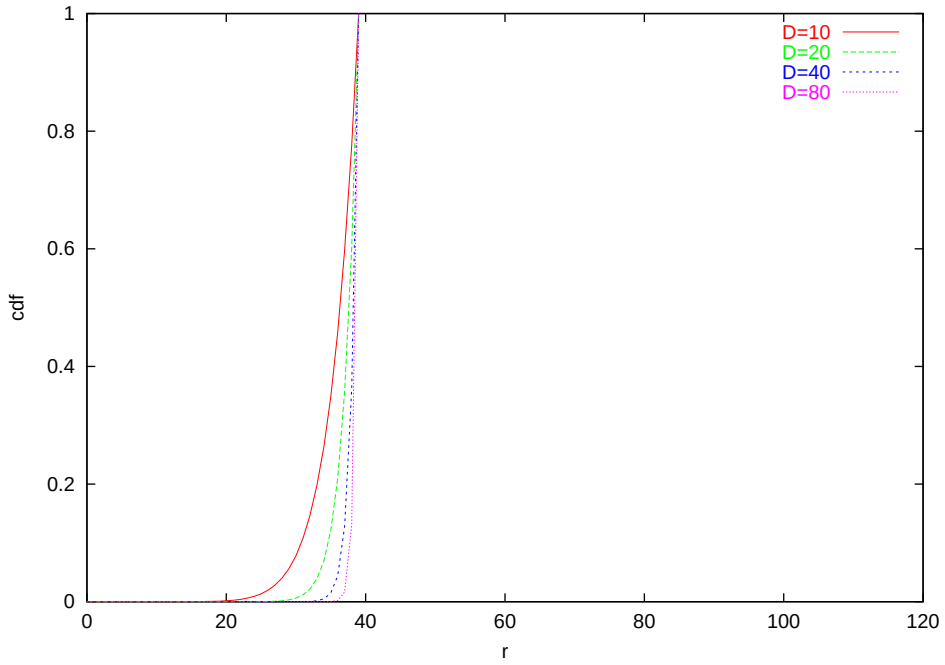


FIG. 4.4 – Effet de la dimension sur une distribution sphérique uniforme : $F_{HS}(r)$ pour quatre dimensions et $\epsilon = 40$

Loi de probabilité gaussienne

On considère dans ce cas que la distorsion suit une loi de probabilité gaussienne avec indépendance des D composantes. Chaque composante est supposé suivre une loi normale de même écart-type σ_g et de moyenne nulle :

$$p_{\Delta S_j}(x) = f^j_{Gauss}(x) = f_{\mathcal{N}(0,\sigma_g)}(x)$$

La densité de probabilité de la norme de la distorsion vaut alors :

$$p_{\|\boldsymbol{\Delta S}\|}(r) = f_{Gauss}(r) = \frac{f_{\mathcal{N}(0,\sigma_g)}(r)}{(2\pi\sigma_g)^{\frac{D-1}{2}}} \frac{\pi^{\frac{D}{2}} D}{\Gamma\left(\frac{D}{2}+1\right)} r^{D-1}$$

et la fonction de répartition associée :

$$P_{\|\boldsymbol{\Delta S}\|}(r) = F_{Gauss}(r) = \int_0^r f_{Gauss}(\rho) d\rho$$

La figure 4.5 représente ces trois fonctions pour $\sigma_g = 10$ et $D = 20$. On constate sur la courbe représentant la densité de probabilité de la norme de la distorsion que l'effet de concentration des signatures sur une hyper-sphère n'est pas observable dans le cas d'une distribution gaussienne. Avec une telle loi, le rayon de recherche ϵ d'une requête à ϵ-près, peut être réduit de manière beaucoup plus significative que dans le cas d'une distribution uniforme sphérique. Ceci est dû à la propriété de décroissance stricte, en fonction du rayon, de la densité de probabilité de la loi gaussienne multidimensionnelle.
L'autre propriété intéressante d'une telle loi est qu'elle possède à la fois la propriété d'isotropie et la propriété de séparabilité des composantes (due à l'indépendance des composantes).

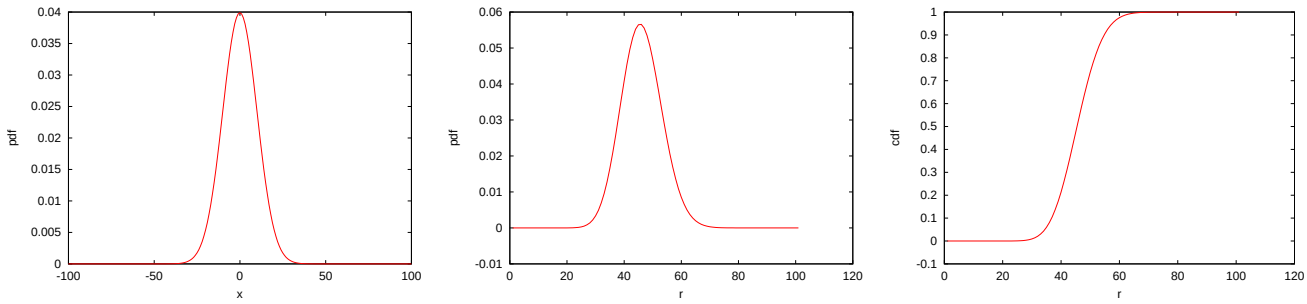


FIG. 4.5 – Loi de probabilité gaussienne avec indépendance des composantes - à gauche : densité de probabilité d'une seule composante de la distorsion - au milieu : densité de probabilité de la norme de la distorsion - à droite : fonction de répartition de la norme de la distorsion

Estimation de densités de probabilité réelles

Nous avons mis en place un protocole pour estimer la densité de probabilité $p_{\|\boldsymbol{\Delta S}\|}(r)$ de la norme de la distorsion dans le cas de transformations réelles. Le principe est de calculer les signatures locales pour un ensemble de vidéos puis de calculer ces mêmes signatures pour une version transformée de ces vidéos. Lors de la deuxième étape cependant, pour s'abstenir de l'influence du détecteur de points d'intérêt, les signatures sont directement extraites autour des positions correspondant aux points détectés lors de la première étape. Dans le cas des transformations

géométriques comme le redimensionnement, la position des points est ainsi recalculée de manière théorique en fonction des paramètres de la transformation. Pour simuler une imprécision spatiale dans la détection des points d'intérêt, une des transformations consiste à décaler l'image d'un ou deux pixels sans modifier la position des points.
Un histogramme de la distance entre les signatures originales et les signatures distordues peut ainsi être calculé. L'estimation de $p_{\|\boldsymbol{\Delta S}\|}(r)$ se fait alors en divisant les valeurs de l'histogramme par le nombre total de signatures. L'estimation de la fonction de répartition est réalisée en intégrant numériquement les valeurs de $p_{\|\boldsymbol{\Delta S}\|}(r)$.
Durant ce protocole, nous estimons également la densité de probabilité et la fonction de répartition selon chacune des 20 composantes.
Plusieurs transformations de l'image ont été étudiées :

- un redimensionnement d'un facteur $0,85$
- un changement de gamma d'un facteur $0,5$
- un ajout de bruit gaussien d'écart-type $20,0$
- un décalage d'1 pixel
- la combinaison des quatre transformations précédentes

L'ensemble des vidéos utilisé représente un total d'environ 5 heures de vidéo et le nombre de signatures locales extraites est de 19 883 signatures.
La figure 4.6 représente l'estimation de la densité de probabilité de la norme de la distorsion pour les cinq transformations.

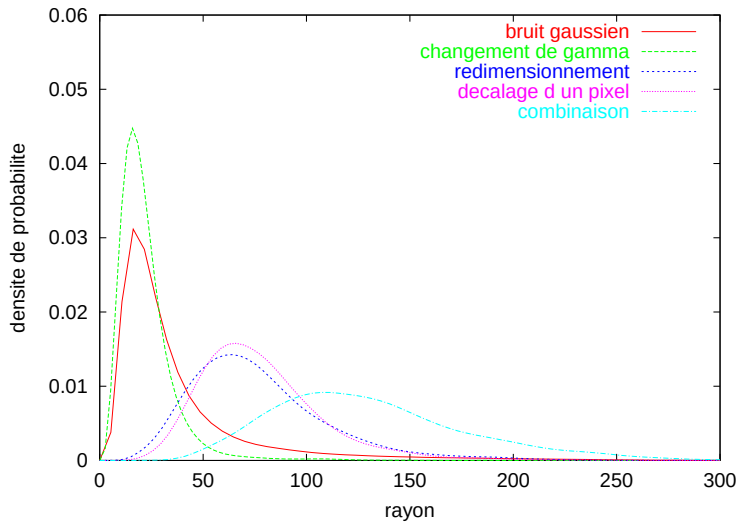


FIG. 4.6 – Estimation de $p_{\|\boldsymbol{\Delta S}\|}(r)$ pour cinq transformations réelles

Les courbes montrent clairement qu'il est possible de réduire sensiblement le rayon de recherche, tout en conservant un pourcentage important des signatures pertinentes. On constate en effet que le nombre de signatures pour lesquelles la norme de la distorsion atteint les valeurs maximales, est très faible. Cette propriété de $p_{\|\boldsymbol{\Delta S}\|}(r)$ rend la réduction du rayon de recherche ϵ très intéressante, car un faible pourcentage de signatures perdues peut permettre une réduction conséquente du rayon de recherche. La table 4.1 illustre ces propos, en donnant pour chacune des transformations, la valeur de ϵ_{max} (sur l'échantillon des 19 883 signatures) et la valeur de ϵ_α pour trois valeurs de α : 99 %, 95 % et 80 %. Nous rappelons que α représente l'espérance du pourcentage de signatures qui seront effectivement retrouvées.

Transformation	ϵ_{max}	ϵ_{99}	ϵ_{95}	ϵ_{80}
redimensionnement	290	191	136	100
gamma	202	79	43	28
bruit gaussien	319	199	113	49
décalage	275	181	134	98
combinaison	346	264	221	167

TAB. 4.1 – Influence de l'espérance sur le rayon de recherche d'une requête à ϵ_α-près

Afin de confronter les deux modèles théoriques décrits précédemment à une distribution réelle, nous avons tracé sur la figure 4.7 la densité de probabilité $p_{\|\boldsymbol{\Delta S}\|}(r)$ pour les deux modèles (soit $f_{Gauss}(r)$ et $f_{HS}(r)$), ainsi que l'estimation de la densité de probabilité à partir de l'histogramme réel. Les paramètres de chacune des deux lois théoriques ont été choisis de manière à ce que l'écart type d'une seule composante de la distorsion soit égal à l'écart-type mesuré sur les distorsions réelles.

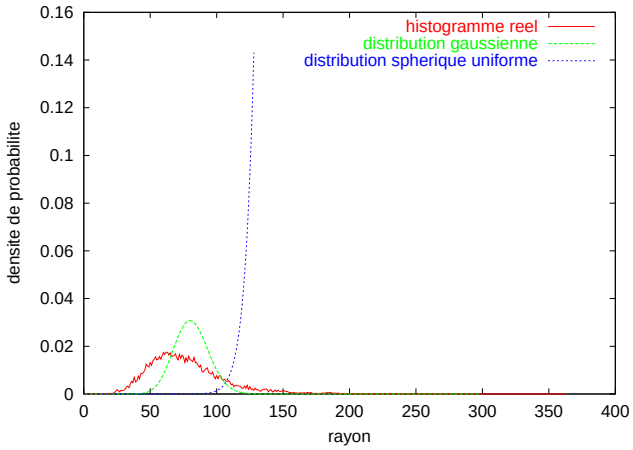


FIG. 4.7 – Densité de probabilité de la norme de la distorsion : comparaison des deux modèles théoriques et d'une distribution réelle

Aucune conclusion formelle ne peut être tirer sur la distribution réelle des distorsions, mais la figure montre bien que l'hypothèse de distribution uniforme sphérique présente une distribution très particulière et très éloignée d'une distribution réelle, alors qu'une distribution gaussienne est plus semblable. La propriété commune de la loi réelle et de la loi gaussienne est en fait la décroissance de la densité de probabilité en fonction du rayon. Pour s'en convaincre, on peut retracer ces trois courbes en divisant la valeur en chaque point par une grandeur proportionnelle à une unité de volume élémentaire soit r^{D-1}. On obtient alors une estimation de la densité de probabilité multidimensionnelle en fonction du rayon. Les courbes de la figure 4.8 montrent ainsi que la densité de probabilité de la loi uniforme est bien constante tandis que la densité de probabilité réelle et la densité de probabilité de la loi gaussienne sont décroissantes. Cette opération a été réalisée pour divers types de transformations et combinaisons de transformations (changements de gamma, redimensionnements d'autres facteurs, changements de contraste, ajout de bruit gaussien, décalage d'un ou deux pixels). La propriété de décroissance de la densité

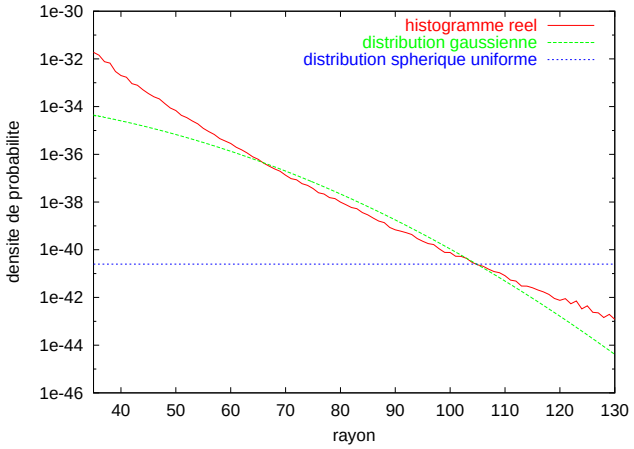


FIG. 4.8 – Estimation de la densité de probabilité de la distorsion en fonction du rayon : comparaison des deux modèles théoriques et d'une distribution réelle

de probabilité réelle a systématiquement été observée et c'est là la principale justification de l'utilisation de la loi gaussienne dans la suite de nos travaux. L'autre raison étant que la loi gaussienne possède à la fois la propriété de séparabilité et d'isotropie.

4.2.4 Requêtes statistiques

Principes et Définitions

Nous avons précédemment introduit la notion de requête à ϵ_α-près afin de contrôler l'espérance du nombre de signatures retrouvées après réduction du rayon de recherche d'une requête à ϵ-près. Le contrôle était basé seulement sur la densité de probabilité de la norme de la distorsion $(p_{\|\boldsymbol{\Delta S}\|}(r))$. Si l'on se restreint à des requêtes hyper-sphèriques, il n'est en effet pas nécessaire d'avoir une connaissance plus approfondie de la densité de probabilité de la distorsion. Les requêtes à ϵ_α-près restent, d'un point de vue du système de recherche, une requête géométrique exacte mais de rayon réduit.

Par l'utilisation de requêtes statistiques, nous proposons de s'abstenir de la contrainte géométrique de la requête et de rechercher directement les régions de l'espace maximisant la probabilité de retrouver des signatures issues d'un éventuel original du motif candidat.
Nous supposons désormais que l'on connaît directement la densité de probabilité du vecteur de distorsion $\boldsymbol{\Delta S}$. Étant donnée une signature $\boldsymbol{s}(\boldsymbol{t}(\boldsymbol{m}))$ issue d'une copie d'un motif m de la base, la probabilité de retrouver la signature $\boldsymbol{s}(\boldsymbol{m})$ du motif original dans une région quelconque V de l'espace est alors donnée par :

$$P(V) = \int_V p_{\boldsymbol{\Delta S}}\left(\boldsymbol{X} - \boldsymbol{s}\left(\boldsymbol{t}(\boldsymbol{m})\right)\right) d\boldsymbol{X} = \int_V p_{\boldsymbol{\Delta S}}\left(\boldsymbol{Z}\right) d\boldsymbol{Z} \tag{4.5}$$

où $p_{\boldsymbol{\Delta S}}(\boldsymbol{Z})$ est la densité de probabilité du vecteur de distorsion et $\boldsymbol{Z} = \boldsymbol{X} - \boldsymbol{s}\left(\boldsymbol{t}(\boldsymbol{m})\right)$ représente la coordonnée d'un point quelconque de l'espace dans le repère centré sur la requête $\boldsymbol{s}\left(\boldsymbol{t}(\boldsymbol{m})\right)$. Dans le cas d'une requête à ϵ-près, on impose une contrainte géométrique à la région V qui ne peut être qu'une hyper-sphère centrée en $\boldsymbol{Z} = 0$. Ainsi pour une probabilité α fixée a priori, il existe un seul rayon ϵ_α pour lequel une requête à ϵ-près satisfait l'équation :

$$P(V_{HS}(\epsilon, D)) = \alpha \tag{4.6}$$

Si l'on n'impose aucune forme particulière à la requête, le nombre de régions distinctes satisfaisant l'équation 4.6 est a priori non borné. Nous appelons requête statistique le fait de rechercher tous les points appartenant à une région V_α satisfaisant l'équation :

$$P(V_\alpha) = \alpha \tag{4.7}$$

Définition Requête statistique d'espérance α - *Soit* $\Delta = \{\boldsymbol{S_n}\}_{n\in[1,N]}$ *une base de* N *signatures ayant une représentation dans un espace vectoriel* E*. Le résultat d'une requête statistique d'espérance* α *pour une signature candidate* $\boldsymbol{Q}$*, est un sous-ensemble de* Δ*, noté* Δ_α*, représentant toutes les signatures qui appartiennent à une région* V_α *de l'espace, telle que :*

$$\int_{V_\alpha} p_{\boldsymbol{\Delta S}}(\boldsymbol{X} - \boldsymbol{Q})\, \boldsymbol{dX} = \alpha \tag{4.8}$$

où la fonction $p_{\boldsymbol{\Delta S}}(\boldsymbol{Z})$ *représente le densité de probabilité de la distorsion.*

La recherche à ϵ_α-près est un cas particulier de requête statistique d'espérance α.

En pratique, parmi les multiples possibilités pour V_α, le choix doit se faire selon un critère maximisant les performances du système. Le fait de s'abstenir de la contrainte sphérique de la requête laisse supposer qu'il est possible dans le cas général de trouver une région plus rentable. Sans connaître a priori le système d'indexation utilisé, on est tenté d'utiliser comme critère le volume de la région, partant du principe que plus le volume de la requête est faible moins la complexité de la recherche est importante. La recherche d'une région de volume minimum satisfaisant l'équation 4.8 dans un espace continu de grande dimension n'est toutefois pas envisageable dans le cas général et il s'agit là d'un objectif purement théorique. De plus, en complexifiant la forme géométrique de la requête, on complexifie les règles de filtrage du système d'indexation et les performances ne seront au final pas forcément améliorées.
Le choix pratique de V_α dépend donc avant tout du système d'indexation multidimensionnelle utilisé. Nous proposons de remplacer les règles de filtrage géométrique classiques par des règles de filtrage probabilistes. On a vu dans le chapitre 3 que la plupart des systèmes de recherche par similarité sont basées soit sur un partitionnement des données, soit sur un partitionnement de l'espace. Dans toutes ces méthodes, les descripteurs sont indexés par une clé caractérisant une forme englobante. Ces formes englobantes sont en général assez simples pour que les règles de filtrage géométrique puissent s'appliquer de manière efficace. Il s'agit pour l'essentiel d'hyper-sphères ou bien d'hyper-rectangles de faces parallèles aux axes. La probabilité que la signature d'un original du motif candidat se situe dans une forme englobante quelconque est donnée par l'équation 4.5 dans laquelle V représente alors la forme englobante elle-même. La mise en œuvre pratique d'une requête statistique d'espérance α consiste à ne visiter que les signatures contenues dans les M formes englobantes les plus probables et de manière à ce que la somme de leurs probabilités soit la plus petite possible au dessus de la valeur de α :

$$\sum_{m=1}^{M} P(V_m) \geq \alpha \geq \sum_{m=1}^{M-1} P(V_m) \tag{4.9}$$

où V_m représente la m^{eme} forme englobante la plus probable. **Cette équation n'est valable que dans le cas où il n'y a pas de chevauchement entre les M formes englobantes**. Dans le cas d'un système où les formes englobantes se chevauchent, il faudrait soustraire de la probabilité totale les probabilités des régions communes, ce qui complexifie grandement la résolution d'une telle requête. Tout système de recherche ne conviendra donc pas à la mise en œuvre

de requêtes statistiques.

Expérimentation

Pour illustrer le gain potentiel d'une requête statistique par rapport à une requête à ϵ_α-près, nous considérons un partitionnement de l'espace vectoriel en 2^D blocs hyper-cubiques égaux, délimités par les D hyper-plans médians de chaque axe. L'ensemble de ces blocs $B = \{b_u\}_{u\in[1,2^D]}$ est supposé être l'ensemble des formes englobantes d'une structure d'indexation. Nous supposons que la distorsion suit la loi gaussienne étudiée dans la section 4.2.3, les composantes étant indépendantes et de densité de probabilité $f_{\mathcal{N}(0,\sigma_g)}(x)$. L'objectif est de comparer le nombre moyen de blocs interceptés par une requête à ϵ-près et le nombre moyen de blocs interceptés par une requête statistique. Les paramètres des deux requêtes sont réglés de manière à avoir la même espérance α de retrouver une signature distordue :

- La connaissance de la fonction de répartition de la norme de la distorsion (cf. section 4.2.3), nous permet de déterminer facilement, pour une valeur de α donnée, le rayon de recherche ϵ_α pour lequel la requête à ϵ_α-près est vérifiée. Les propriétés d'isotropie et de décroissance de la loi gaussienne font que le volume intercepté par cette requête est en fait le volume minimal permettant de répondre à l'équation définissant une requête statistique (cf. équation 4.8). Une règle de filtrage géométrique classique permet ensuite de déterminer tous les blocs ayant une intersection avec la requête.

- Pour la requête statistique, l'hypothèse d'indépendance nous permet de calculer facilement la probabilité d'un bloc hyper-cubique b_i :

$$Pr(b_i) = \int_{b_i} p_{\boldsymbol{\Delta S}}(\boldsymbol{Z})\, \boldsymbol{dZ} = \prod_{j=1}^{D} \int_{z1_j}^{z2_j} f_{\mathcal{N}(0,\sigma_g)}(z)$$

  En classant les blocs par ordre de probabilité décroissante, il est ensuite possible de déterminer les Q blocs satisfaisant l'inégalité 4.9 et répondant donc à la requête statistique d'espérance α.

Le nombre de blocs interceptés par les deux méthodes a été moyenné sur un ensemble de 1 000 requêtes aléatoires uniformément réparties dans l'espace, et l'opération a été réitérée pour plusieurs valeurs de α. Les courbes représentant le nombre de blocs en fonction de α pour chacune des deux méthodes sont présentées à la figure 4.9.

On constate que la réduction du nombre de blocs est importante et qu'elle l'est d'autant plus que l'approximation est forte. Elle varie ainsi d'un facteur 24 pour une espérance de 99 % à un facteur 82 pour une espérance d'un peu plus de 70 %. Si l'on parvient à trouver un algorithme de filtrage probabiliste ayant un coût du même ordre de grandeur qu'un filtrage géométrique, on peut donc espérer réduire très fortement les temps de recherche en utilisant une requête statistique plutôt qu'une requête sphérique exacte, sans dégrader l'espérance du nombre de signatures retrouvées.

On peut expliquer simplement cette réduction du nombre de blocs interceptés par le fait que la contrainte sphérique de la requête à ϵ-près entraîne un grand nombre d'intersections avec des blocs dont la probabilité effective de contenir une signature pertinente est très faible. Sur l'illustration de la figure 4.10, on constate que certains blocs (les n°6,8,14 et 16) ont une intersection

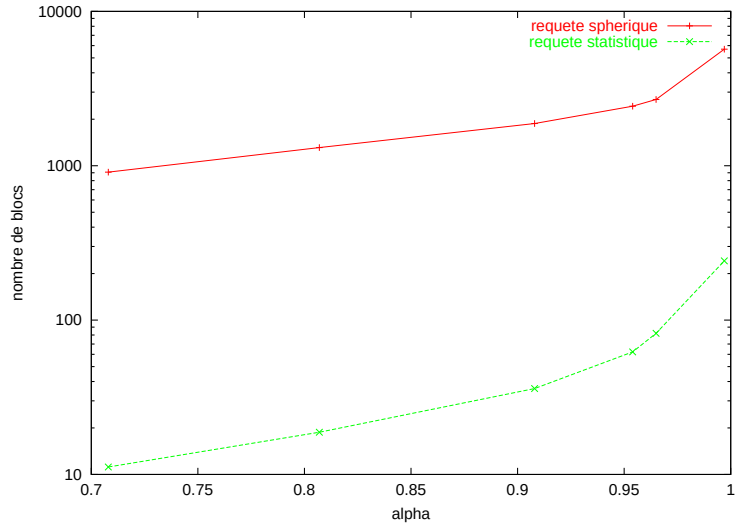


FIG. 4.9 – Comparaison du nombre de blocs interceptés dans le cas d'une requête à ϵ_α-près et dans le cas d'une requête statistique

très faible avec l'hyper-sphère. Ils seront pourtant parcourus intégralement lors de l'étape de raffinement au même titre que les autres blocs. Le nombre de ce type d'intersections augmente considérablement avec la dimension de l'espace en même temps que leurs volumes diminuent très fortement. Ces deux remarques sont vraies dans le cas général quelles que soient les formes englobantes utilisées. Dans un premier temps on serait tenté d'établir une règle éliminant les blocs dont le volume de l'intersection avec la requête est trop faible. Cela reviendrait cependant à considérer que la distribution des solutions à l'intérieur de l'hyper-sphère est uniforme et nous avons déjà vu que ce type de distribution ne correspond pas à la réalité. Le filtrage de ces blocs doit donc être basé sur une distribution se rapprochant au mieux de la réalité et c'est justement là le principe des requêtes statistiques.

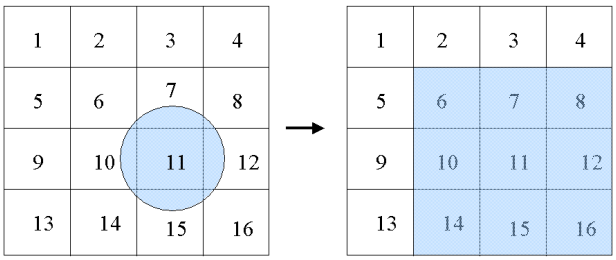


FIG. 4.10 – Illustration des intersections non rentables entre une requête sphérique et une grille régulière dans un espace de dimension 2

Il n'est bien sûr pas possible de généraliser ces conclusions à tous les types de partitionnement ou de formes englobantes par cette seule expérimentation, mais nous pensons que le principe reste général. L'étude des problèmes liés à *la malédiction de la dimension* a en effet montré que quelles que soient les formes englobantes utilisées, la dégradation du temps de recherche est en grande partie due à la forte augmentation du nombre d'intersections entre la requête et les formes

englobantes [106, 17]. Nous pensons que l'utilisation de requêtes statistiques s'abstenant de la contrainte d'une forme géométrique constante et exacte peut réduire significativement ce nombre d'intersections dans la plupart des cas. Notons de plus qu'une requête géométrique exacte étant une heuristique, il n'y a pas de raison de conserver cette contrainte. La force d'une requête statistique est de s'adapter aux formes englobantes sans intermédiaire géométrique.

Mise en œuvre

La principale difficulté de mise en œuvre des requêtes statistiques est liée aux calculs de probabilité lors de la phase de recherche. Il faut en effet que la densité de probabilité de la distorsion soit intégrable sur les formes englobantes utilisées dans la structure, afin de calculer leur probabilité. Ce calcul de probabilité doit de plus pouvoir se faire de manière efficace, si l'on veut que le gain de sélectivité dû à la requête statistique ne soit pas rendu inutile par un coût de sélection trop élevé. L'intégration de la densité de probabilité sur une région englobante hyper-sphèrique est par exemple problématique.
Par souci d'efficacité, la technique de recherche présentée dans ce mémoire fait l'hypothèse de l'**indépendance des composantes du vecteur de distorsion**. Cette hypothèse permet de simplifier considérablement le calcul des probabilités sur des régions rectangulaires puisqu'il se résume à un produit de probabilités monodimensionnelles qui peuvent elles mêmes être rapidement calculées grâce à l'utilisation de tables (*Look Up Table*). La question de la généralisation des requêtes statistiques à des modèles probabilistes plus complexes et à d'autres types de formes englobantes n'est pas traitée dans cette thèse et reste à notre connaissance une question ouverte. Une des principales contraintes des études qu'il faudrait mener reste à notre avis l'efficacité des calculs.

Sous l'hypothèse d'indépendance, on peut exprimer la densité de probabilité de la distorsion sous la forme :

$$p_{\boldsymbol{\Delta S}}(\boldsymbol{X}) = \prod_{j=1}^{D} p_{\Delta S_j}(x_j) \tag{4.10}$$

la fonction de répartition correspondante est alors :

$$P_{\boldsymbol{\Delta S}}(\boldsymbol{X}) = \prod_{j=1}^{D} P_{\Delta S_j}(x_j) \tag{4.11}$$

En pratique, la loi de probabilité actuellement utilisée dans notre système est la loi gaussienne indépendante pour chaque composante (la section 4.2.3). Son utilisation se justifie par la propriété de décroissance de la densité de probabilité et par la conservation de l'isotropie malgrè la séparabilité des composantes. La pertinence de ce modèle sera discutée au regard des résultats de la partie expérimentale de cette thèse (cf. section 6.2).
La densité de probabilité de la distorsion ne dépend en réalité pas uniquement de sa répartition dans l'espace des signatures pour une transformation donnée. Elle dépend également de la répartition des transformations. Il nous est à l'heure actuelle difficile d'estimer cette répartition mais on peut tout de même remarquer que les transformations les plus sévères sont en général moins fréquentes que les transformations les plus bénignes. Cette constatation est en accord avec la propriété de décroissance de la densité de probabilité en fonction du rayon.
En pratique, le modèle probabiliste sera paramétré par la distribution des distorsions d'une seule transformation de référence considérée comme la plus sévère au sens d'un critère de sévérité sur la distorsion des signatures. Nous en rediscuterons dans la partie expérimentale (cf. section 6.2).

4.3 Système de recherche par requêtes statistiques

4.3.1 Argumentation

La structure d'indexation utilisée pour permettre d'effectuer les requêtes statistiques décrites précédemment est basée sur un ordonnancement des signatures selon une courbe de Hilbert. Avant de rentrer plus en détail dans la description de la technique mise en œuvre, nous revenons ici sur les considérations qui nous ont amené à faire ce choix.

Dimension des signatures et taille de la base

L'efficacité d'un système de recherche par similarité dépend à la fois de la dimension des signatures, de la taille de la base, de la distribution des signatures dans l'espace et du type de requête (cf. section 4.2.2). Ainsi pour les dimensions très élevées et une distribution uniforme des descripteurs dans la base tous les systèmes deviennent moins performants qu'une recherche séquentielle pour la recherche de K-plus proche voisins. Seule une distribution particulière des descripteurs, comme la présence d'agrégats par exemple, justifie l'utilisation de structures d'indexation pour les dimensions très élevées. Ceci a notamment été exploité dans de nombreuses méthodes récentes de recherche approximative de K-plus proches voisins dans des espaces de très grande dimension. L'inconvénient majeur est que ces techniques conduisent généralement à une croissance linéaire du temps de recherche en fonction de la taille de la base. Dans notre cas la dimension des signatures est moyenne $D = 20$ tandis que la taille de la base peut devenir très importante. Nous nous situons ainsi dans un intervalle de caractéristiques pour lesquelles une structure d'indexation peut encore être plus efficace qu'une recherche séquentielle même pour une distribution uniforme des signatures dans la base [14]. De plus les requêtes à ϵ-près ou les requêtes statistiques dans notre cas souffrent moins de la dimension que les requêtes de type K-plus proches voisins (cf. section 4.2.2). L'utilisation d'une structure d'indexation classique ne peut donc être écartée surtout si elle permet de conduire à un coût de calcul sous linéaire en fonction de la taille de la base.

Contrainte des requêtes statistiques

Rappelons d'abord que notre concept de requête statistique est basé sur une représentation vectorielle des signatures qui élimine l'utilisation de systèmes d'indexation non vectoriels.
Nous avons vu dans le chapitre précédent que pour traiter efficacement les requêtes statistiques, nous avons fait l'hypothèse d'indépendance des composantes de la distorsion afin de calculer efficacement la probabilité d'une forme englobante rectangulaire. La structure d'indexation choisie doit donc respecter cette contrainte géométrique pour les formes englobantes. Dans notre description des requêtes statistiques (section 4.2.4), nous avons également vu que la résolution du problème se trouve fortement simplifiée lorsqu'il n'y a pas de chevauchement entre les formes englobantes. Les techniques vectorielles basées sur un partitionnement de l'espace et celles basées sur une courbe remplissant l'espace permettent de répondre à ces deux contraintes. Dans les deux cas, le partitionnement de l'espace résultant est en effet complet et disjoint.

Méthode statique

Le *landmark file* [23] présenté dans la section 3.3, est l'une des seules techniques tirant avantage d'un contexte statique pour accélérer les performances de la recherche par similarité. Ce type de techniques est applicable lorsque la base de descripteurs peut rester inchangée pendant

un certain temps. Dans le cas du monitorage d'une chaîne de télévision, nous avons retenu ce type de fonctionnement. Le catalogue de référence croît en réalité quotidiennement mais nous considérons qu'une mise à jour ponctuelle de la base des signatures est largement suffisante. Seules quelques semaines ou quelques mois d'extraits vidéo récents, sur un total qui couvre plusieurs dizaines d'années de vidéo, seront ainsi négligés par la recherche.
Similairement au *landmark file*, ce contexte statique nous permet d'ordonner physiquement les vecteurs de la base. Dans notre cas, les signatures sont triées selon leur coordonnée sur une courbe de Hilbert. Notons qu'il s'agit d'une approche différente des structures d'indexation classiques basées sur une courbe remplissant l'espace. Dans ces dernières, la coordonnée des signatures sur la courbe est utilisée comme index dans une structure d'indexation dynamique monodimensionnelle tandis qu'elle nous sert de clé pour trier physiquement les vecteurs dans un fichier ou dans un tableau. L'objectif est de profiter à la fois de l'avantage d'un parcours séquentiel et de la sélectivité d'une étape de filtrage dans un partitionnement de l'espace. La recherche comporte une première étape de filtrage consistant à déterminer les intervalles de la base ordonnée qui doivent être parcourus, puis une deuxième étape consistant à parcourir séquentiellement ces différents intervalles. L'accès à la base ne nécessite pas de structure arborescente mais seulement une table d'index. De plus, il s'agit d'une approche adaptative, qui dans le pire des cas sera un parcours séquentiel de la base entière.

Fonctionnement en mémoire

Nous avons rapidement écarté l'hypothèse d'un fonctionnement sur disque. Dans notre contexte vidéo temps réel, un très faible nombre d'accès au disque par requête est déjà trop coûteux. Nous avons donc opté pour un fonctionnement en mémoire en minimisant au maximum la taille des signatures et des données associées. Lorsque la base excède la taille mémoire, nous proposons un fonctionnement par requêtes groupées qui charge la base en mémoire par morceaux et qui permet de conserver des temps de recherche proches d'un fonctionnement en mémoire, comme nous le verrons par la suite.
Lorsque la taille de la base le permet, ce qui est le cas dans la plupart des expérimentations présentées dans la littérature, beaucoup de structures d'indexation multidimensionnelle fonctionnent implicitement déjà entièrement en mémoire [180]. Ceci est dû au fait que la mémoire vive est utilisée comme un niveau de cache du disque dur par le système. Cependant, un régime transitoire peut être nécessaire avant que cela soit effectif et l'organisation des données en mémoire n'est pas contrôlable. Dans notre cas, le système fonctionne explicitement entièrement en mémoire dès le démarrage, puisque la base de données et la table d'index sont affectées dans une variable de type tableau.

4.3.2 Principe général

La figure 4.11 résume le fonctionnement général du système de recherche. Il comporte deux phases principales : une phase d'indexation hors-ligne et une phase de recherche proprement dite qui ne peut être effectuée qu'après avoir terminé la phase d'indexation.

Phase d'indexation

La phase d'indexation prend en entrée un fichier contenant toutes les signatures de la base accompagnées de leurs données associées (Identifiant de la séquence vidéo, code temporel et éventuellement position du point d'intérêt). Les signatures y sont stockées dans l'ordre où elles

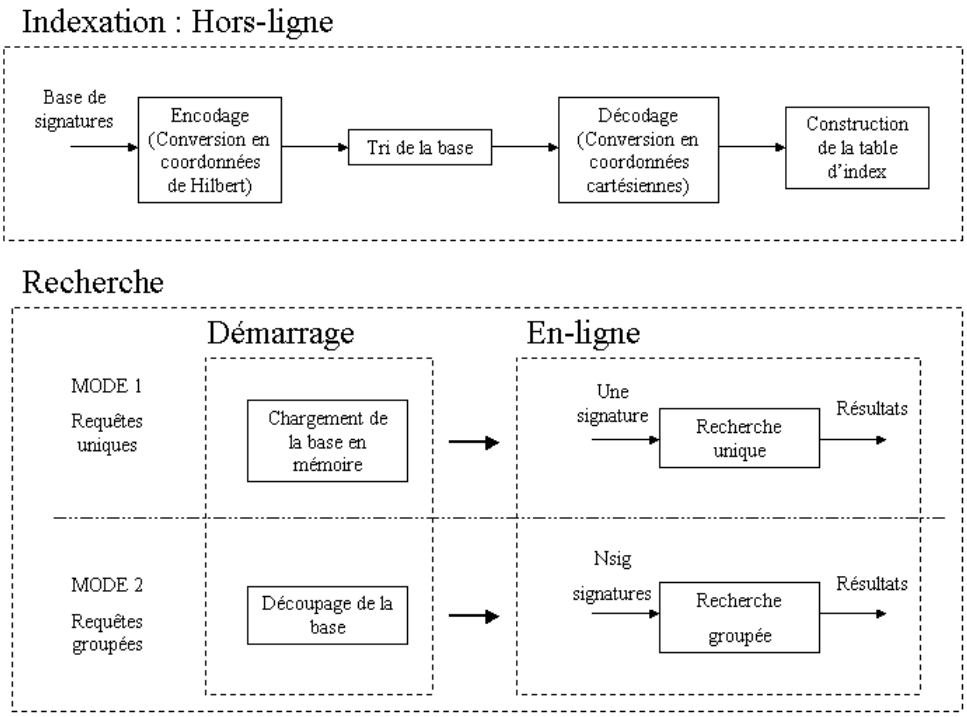


FIG. 4.11 – Organigramme du fonctionnement de la technique de recherche

ont été préalablement calculées. Chaque composante d'une signature est codée sur un octet et les données associées sont stockées immédiatement après.
La phase d'indexation consiste à trier ces signatures suivant leur position sur la courbe de Hilbert (cf. section 4.3.3). Elle nécessite de convertir toutes les signatures en une coordonnée sur la courbe de Hilbert (encodage), de trier les valeurs obtenues, puis de les reconvertir en coordonnées cartésiennes (décodage). Le processus d'encodage d'une signature est parfaitement réversible et seul l'ordre des signatures dans le fichier de sortie est modifié. Les données associées aux signatures sont déplacées en même temps que les signatures et restent stockées immédiatement après chaque signature.
Le tri des valeurs encodées est effectué par l'algorithme `qsort()` en mémoire. Lorsque la taille de la base dépasse la taille mémoire, celle-ci est d'abord triée par morceaux, en mémoire, puis les différents morceaux triés sont parcourus parallèlement afin de réécrire les signatures de manière ordonnée dans un nouveau fichier.
La dernière étape de la phase d'indexation consiste à construire une table d'index qui contient la position de certaines signatures dans la base triée. Nous détaillerons son contenu exact ultérieurement. Cette table d'index est également stockée dans un fichier.
La phase d'indexation étant relativement coûteuse, elle n'est effectuée que périodiquement, et c'est à ce moment là seulement que d'éventuels suppressions ou ajouts de signatures seront pris en compte ; il s'agit d'une méthode statique.

Phase de recherche

La phase de recherche comporte deux modes de fonctionnement suivant la possibilité de stocker entièrement la base de données et la table d'index en mémoire. Lorsque c'est le cas, le système fonctionne en mode *requête unique* (MODE 1 sur la figure 4.11). La recherche comporte une étape de démarrage qui consiste à charger en mémoire sous forme de tableau les fichiers contenant la base des signatures triées et la table d'index. Une fois le chargement effectué, les signatures peuvent être continuellement recherchées, une par une, grâce à l'algorithme de recherche par requêtes statistiques détaillé à la section 4.3.4.
Lorsque la base des signatures et la table d'index ne peuvent pas être contenues en mémoire, nous proposons un mode par *requêtes groupées* qui permet, si l'on dispose d'un ensemble suffisamment grand de signatures candidates (soit N_{sig} signatures), de les rechercher simultanément avec un temps de recherche moyen relativement proche de celui que l'on aurait eu en mémoire. Le principe de cette recherche, ainsi que la condition sur le nombre de signatures N_{sig}, sont détaillés dans la section 4.3.4.

4.3.3 Indexation des signatures basée sur une courbe de Hilbert

Tri des signatures

La première étape de l'indexation des signatures consiste à les trier physiquement dans un fichier selon leurs positions sur une courbe de Hilbert remplissant l'espace. Une telle courbe est une application bijective entre l'espace multidimensionnel et une coordonnées curviligne monodimensionnelle. Le principe de construction de la courbe de Hilbert ainsi que les différentes techniques pour passer d'un espace à l'autre sont décrit en annexe E. Outre le fait d'ordonner l'espace multidimensionnel, l'intérêt de la courbe de Hilbert est qu'elle est continue et qu'elle préserve une certaine localité entre les coordonnées curvilignes de deux points proches dans l'espace. Certaines signatures voisines dans l'espace pourront donc être parcourues séquentiellement dans le fichier trié.

Parmi les trois principales techniques permettant de calculer la coordonnée curviligne d'une signature sur la courbe de Hilbert nous avons retenu l'algorithme de Butz. Contrairement aux arbres et aux diagrammes (cf. annexe E.2), l'algorithme de Butz est purement calculatoire et ne nécessite que très peu d'espace mémoire. Sa description détaillée est donnée en annexe E.3.
La coordonnée curviligne d'une signature $\mathbf{S}$, résultant de l'algorithme de Butz, est en fait exprimée sous forme d'un mot binaire $H_Q^D(\boldsymbol{S})$ de longueur QD bits, où D représente la dimension de l'espace et Q l'ordre de l'approximation de la courbe de Hilbert. Ce mot binaire, parfois appelé clé de Hilbert peut éventuellement être converti en un réel dans l'intervalle $[0, 1[$ ou bien en un entier dans l'intervalle $[0, 2^{QD}[$ afin de simplifier la comparaison entre deux clés.

Partition de l'espace

Avant de décrire la procédure de construction de la table d'index permettant d'accéder rapidement aux signatures de la base triée, il est important de comprendre la correspondance entre une section de la courbe de Hilbert et la région de l'espace correspondante. On s'intéresse ainsi à la partition de l'espace résultant d'une quantification de la clé de Hilbert binaire H_Q^D. Si l'on ne conserve que les p premiers bits de la clé de Hilbert, cela revient à découper la courbe en 2^p

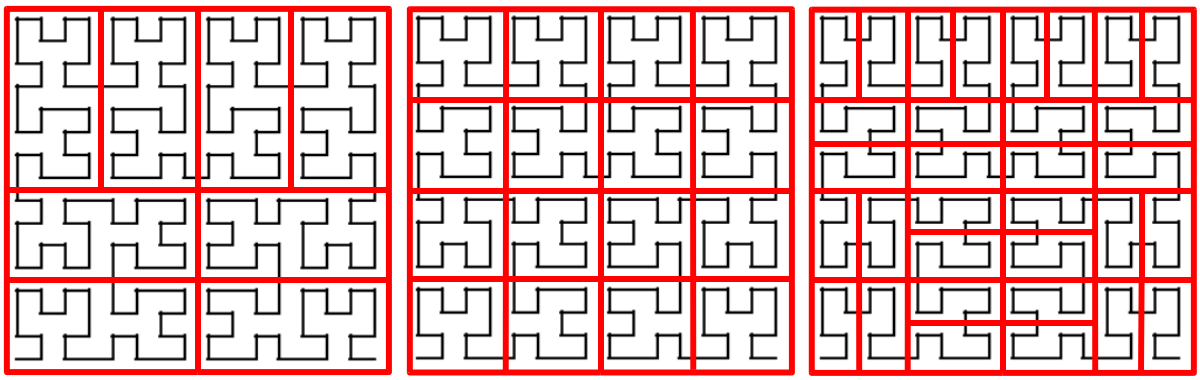


FIG. 4.12 – Partition de l'espace pour $D = 2$ et $Q = 4$ à différentes profondeurs - de gauche à droite : p=3, 4, 5

intervalles égaux. La partition de l'espace résultant de ce découpage est un ensemble de 2^p blocs hyper-rectangulaires de même forme et de même volume, mais avec des orientations différentes, comme l'illustre la figure 4.12. Par analogie à un KD-tree, nous appelons le paramètre p la **profondeur de la partition**. Les blocs résultant d'une telle partition sont appelés des **p-blocs**. Lorsque p est un multiple de D, on retombe sur une partition hyper-cubique similaire à celle obtenue par un quad-tree. Le plus grand multiple de D inférieur à p, soit $q = \lceil \frac{p}{D} \rceil$, est appelé l'ordre de la partition.

Toutes les signatures appartenant à un même p-bloc ont les p premiers bit de leur clé de Hilbert égaux et sont **consécutives dans la base triée**. Ces p premiers bits égaux constituent le préfixe de la clé Hilbert de profondeur p noté $H_p(\boldsymbol{S})$. La valeur de $H_p(\boldsymbol{S})$ en base-(10) de toutes les signatures appartenant au i^{eme} p-bloc est égale à i :

$$H_p(\boldsymbol{S})_{(10)} = i \;\; \forall \boldsymbol{S} \in b_i \;\; \forall i \in [1, 2^p] \tag{4.12}$$

où b_i dénote le i^{eme} p-bloc de la partition, lorsque ceux-ci sont ordonnés par leur position sur la courbe.

Toute région de l'espace, par exemple l'hyper-sphère correspondant à une requête à ϵ-près, peut être approximée par l'ensemble des p-blocs ayant une intersection avec cette région et donc par un ensemble d'intervalles sur la courbe. Plus la profondeur sera importante, plus l'ensemble des p-blocs se rapprochera de la région exacte. Le nombre d'intervalles sur la courbe sera alors plus important mais le nombre de signatures appartenant à ces intervalles sera plus faible. Pour illustrer l'effet de la profondeur de la partition, la figure 4.13 montre, pour $D = 2$, l'ensemble des intervalles correspondant aux p-blocs ayant une intersection avec une requête à ϵ-près, pour plusieurs profondeurs de la partition.
Le principe de l'algorithme permettant de déterminer tous les p-blocs ayant une intersection avec une région donnée de l'espace est décrit en annexe E.4.

Construction de la table d'index

Lors de la phase de recherche, l'accès à la base ordonnée des signatures stockée en mémoire, se fait via une table d'index également stockée en mémoire sous forme d'un tableau. Cette table

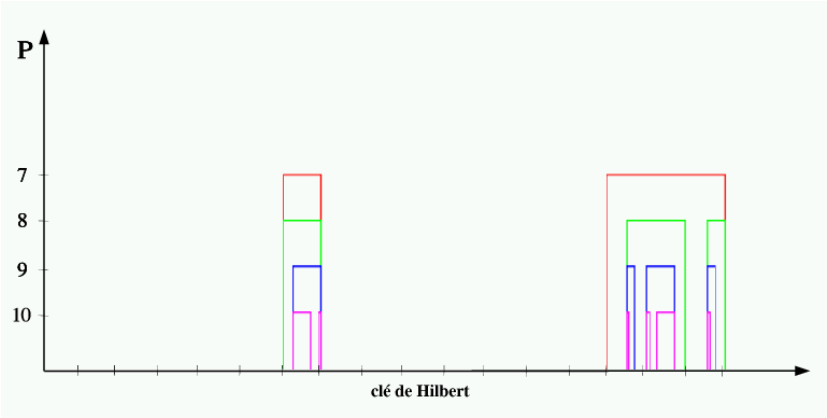


FIG. 4.13 – Sections de la courbe de Hilbert interceptées par une requête sphérique pour différentes profondeurs de la partition

permet de déterminer la position, dans la base, de l'ensemble des signatures consécutives appartenant à un même p-blocs de la partition de l'espace à une profondeur p. La profondeur de la partition permet de régler le compromis entre le nombre de signatures qu'il faudra parcourir et le nombre d'intervalles non consécutifs auxquels il faudra accéder. Pour une profondeur nulle, toute région sera approximée par l'espace entier, et la base toute entière sera parcourue séquentiellement.

La taille de la table d'index dépend de la profondeur de la partition de l'espace et elle doit être recalculée si l'on souhaite en changer. Elle doit également être recalculée chaque fois que la base des signatures a été elle même reconstruite, typiquement après l'insertion d'un ensemble de nouvelles signatures.

Concrètement, la table contient 2^p entiers qui correspondent à la position de certaines signatures dans la base ordonnée. Les signatures en question sont celles ayant la plus petite clé de Hilbert à l'intérieur de chacun des 2^p p-blocs de la partition de l'espace. Le i^{eme} élément de la table, $Table(i)$, contient ainsi la position de la signature ayant la plus petite clé de Hilbert parmi toutes celles contenues dans le i^{eme} bloc de la partition.

Une signature **S** quelconque est située dans l'intervalle de positions suivant :

$$I_p = \Big[Table(H_p(\boldsymbol{S})_{(10)}), Table(H_p(\boldsymbol{S})_{(10)} + 1)\Big[$$

où $H_p(\boldsymbol{S})$ est le préfixe binaire de la clé de Hilbert de **S** (cf. équation 4.12).

Si l'on veut effectuer un accès ponctuel à la base, il est nécessaire de faire une recherche dichotomique à l'intérieur de l'intervalle I_p. Chaque comparaison nécessite alors le calcul de la clé de Hilbert de la signature de la base. Notons toutefois que notre système de recherche étant de toute manière un système statique, aucun accès ponctuel n'est effectué durant le fonctionnement $en - ligne$ puisqu'aucune insertion ou suppression de signature ne peut avoir lieu. Lors de la phase de recherche, l'accès à la base se fera toujours par p-blocs entiers ou par p-blocs successifs. L'accès à un p-bloc ou un ensemble de p-blocs se fait en lisant dans la table, la positon initiale et la position finale de l'intervalle de signatures correspondant. L'étape de raffinement consistera

alors à parcourir séquentiellement ces portions de la base en mémoire.

4.3.4 Algorithme de recherche par requête statistique

Principe

Nous rappelons que la recherche par requêtes statistiques que nous proposons est basée sur l'hypothèse d'indépendance des D composantes du vecteur de distorsion $\boldsymbol{\Delta S} = (\boldsymbol{s}(\boldsymbol{m}) - \boldsymbol{s}(\boldsymbol{t}(\boldsymbol{m})))$, soit :

$$p_{\boldsymbol{\Delta S}}(\boldsymbol{X}) = \prod_{j=1}^{D} p_{\Delta S_j}(x_j) \tag{4.13}$$

Étant donnée une signature candidate $\boldsymbol{Q}$ et une partition de l'espace de profondeur p, satisfaire une requête statistique d'espérance α, revient à trouver un ensemble B_α de p-blocs, $B_\alpha = \{b_i : i \in [1, Card(B_\alpha)]\}$, satisfaisant :

$$\sum_{i=1}^{Card(B_\alpha)} \int_{b_i} p_{\boldsymbol{\Delta S}}(\boldsymbol{X} - \boldsymbol{Q})\, \boldsymbol{dX} \geq \alpha \tag{4.14}$$

où $0 \leq Card(B_\alpha) \leq 2^p$.

Pour optimiser le coût d'une recherche, $Card(B_\alpha)$ doit être minimum. Il faut en effet que l'ensemble des blocs respecte l'inégalité 4.9 (page 95), c'est-à-dire que les $Card(B_\alpha)$ blocs soient les plus probables de l'espace. Nous notons cette solution particulière B_α^{min} et nous verrons par la suite l'algorithme pour approximer cette solution optimale.
Une fois B_α^{min} déterminé, le préfixe de Hilbert $H_p(b_i)$ est calculé avec l'algorithme de Butz pour chaque p-bloc de B_α^{min}. Les positions $Table(H_p(b_i)_{(10)})$ et $Table(H_p(b_i)_{(10)} + 1)$ sont alors lues dans la table d'index pour chaque bloc et les éventuels intervalles de position contigus sont fusionnés. Le processus complet consistant à déterminer ces intervalles de position dans la base triée constitue ce que nous appelons *l'étape de filtrage*.
Les sections de la base triée correspondant à ces intervalles sont ensuite parcourues séquentiellement pour calculer les distances exactes entre la requête et les signatures de la base. Ce processus est exactement le même que si l'on avait utilisé une méthode séquentielle classique, mais appliquée uniquement à certaines sections de la base. Cette étape est appelée *l'étape de raffinement*.

Étape de filtrage

La problématique principale de cette étape est de déterminer l'ensemble minimal de blocs B_α^{min}, satisfaisant l'inégalité 4.14. Ceci n'est pas une tâche triviale puisqu'il n'est pas envisageable de calculer la probabilité des 2^p p-blocs de l'espace et de les trier par probabilité décroissante. Cela serait trop coûteux à la fois en temps de calcul mais également en stockage. Cependant, il est possible d'identifier rapidement tous les blocs ayant une probabilité supérieure à un certain seuil fixe τ :

$$B(\tau) = \left\{ \{b_i\} : \int_{b_i} p_{\boldsymbol{\Delta S}}(\boldsymbol{X} - \boldsymbol{Q})\, \boldsymbol{dX} > \tau \right\}$$

La probabilité cumulée de $B(\tau)$ est alors :

$$P_{sum}(\tau) = \sum_{i=1}^{Card(B(\tau))} \int_{b_i} p_{\boldsymbol{\Delta S}}(\boldsymbol{X} - \boldsymbol{Q})\, \boldsymbol{dX}$$

Sous l'hypothèse d'indépendance des composantes de la distorsion, la probabilité d'un p-bloc peut être calculée facilement par :

$$\int_{b_i} p_{\boldsymbol{\Delta S}}(\boldsymbol{X} - \boldsymbol{Q})\, \boldsymbol{dX} = \prod_{j=1}^{D} \left[P_{\Delta S_j}(u_i^j - q_j) - P_{\Delta S_j}(w_i^j - q_j) \right]$$

où $P_{\Delta S_j}(x)$ est la fonction de répartition de la j^{eme} composante du vecteur de distorsion (tabulée dans une *Look Up Table*, cf. section 4.2.4, page 98) et u_i^j et w_i^j sont les bornes supérieures et inférieures du p-bloc suivant le j^{eme} axe.

L'algorithme que nous proposons pour déterminer $B(\tau)$ est décrit en annexe F. Il est basé sur le principe de partitions hiérarchiques que nous avons déjà étudié à la section 4.3.3 (page 102). L'idée de base est que si la probabilité d'un p-bloc est inférieure à τ à une profondeur p_1, alors tous les sous-blocs aux profondeurs $p > p_1$ auront également une probabilité inférieure à τ.

Comme $Card(B(\tau))$ est décroissant en fonction de τ, trouver B_α^{min} est équivalent à trouver le seuil de probabilité τ_{max} pour lequel :

$$\begin{cases} P_{sum}(\tau_{max}) \geq \alpha \\ P_{sum}(\tau) < \alpha \qquad \forall \tau > \tau_{max} \end{cases} \tag{4.15}$$

Il est donc nécessaire d'étudier la fonction $P_{sum}(\tau)$. Un exemple d'une telle fonction est donné à la figure 4.14 pour une requête arbitraire Q et une loi de probabilité de la distorsion gaussienne, indépendante suivant chaque composante : $p_{\Delta S_j}(x) = f_{\mathcal{N}(0,20)}(x)$.

$P_{sum}(\tau)$ possède les propriétés suivantes :
- $P_{sum}(0) = 1,0$ puisque $B(0)$ est l'espace entier.
- $P_{sum}(\tau)$ est une fonction constante par morceaux, globalement décroissante en fonction de τ puisque $B(\tau_1) \subset B(\tau_2), \forall \tau_2 \leq \tau_1$.
- $P_{sum}(\tau) = 0 \;\; \forall \tau > P_{max}$ où P_{max} est la probabilité du bloc le plus probable de l'espace.

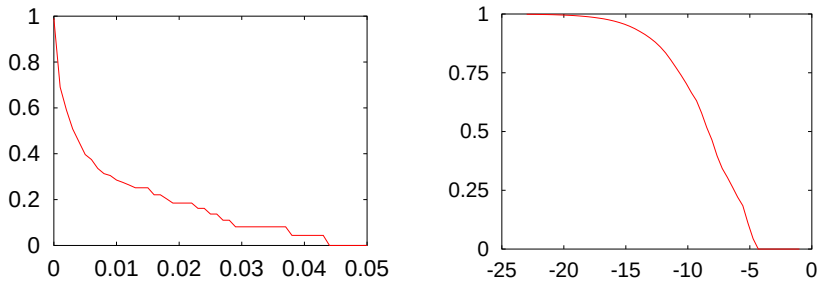


FIG. 4.14 – $P_{sum}(\tau)$ (à gauche) et $P_{sum}(log_{10}(\tau))$ (à droite)

La figure 4.14 montre également le tracé de la fonction $P_{sum}(\tau_{log})$ où $\tau_{log} = log_{10}(\tau)$. Cette transformation est intéressante puisqu'elle permet de limiter la longueur des intervalles sur lesquels la

fonction est constante par morceaux. Le principe de notre algorithme pour déterminer τ_{max} est de considérer $P_{sum}(t_{log})$ comme une fonction continue strictement décroissante et de résoudre l'équation $P_{sum}(t_{log}) = \alpha$ par une technique itérative de Newton-Raphson. La m^{eme} itération de cet algorithme est définie par :

$$\tau_{log}(m+1) = \tau_{log}(m) + \frac{\alpha - P_{sum}(\tau_{log}(m))}{\frac{dP_{sum}}{d\tau_{log}}(\tau_{log}(m))}$$

où $\frac{dP_{sum}}{d\tau_{log}}$ est la dérivée numérique calculée comme suit :

$$\frac{dP_{sum}}{d\tau_{log}}(\tau_{log}) = \frac{P_{sum}(\tau_{log} + \Delta\tau_{log}) - P_{sum}(\tau_{log})}{\Delta\tau_{log}}$$

La valeur initiale de cet algorithme, pour chaque requête, peut être fixée $\tau_{log}(0) = log_{10}(P_{max})$. Comme le coût de calcul de $P_{sum}(\tau)$ augmente rapidement lorsque τ décroît, le coût des premières itérations est négligeable. Une autre solution consiste à apprendre, au démarrage du système, la valeur moyenne de τ_{max} sur un ensemble de requêtes représentatives et de conserver cette valeur comme valeur initiale de chaque requête. Une ou deux itérations sont alors en général suffisantes pour avoir une bonne précision. C'est cette solution que nous avons retenue. Notons qu'à la limite, on peut s'abstenir de la technique de Newton-Raphson et conserver directement cette valeur moyenne de τ_{max} comme seuil de probabilité. Le contrôle de la recherche ne sera alors pas garanti pour une requête individuelle mais l'espérance moyenne sur l'ensemble des requêtes sera proche de α.

Une fois B_α^{min} déterminé, le préfixe de Hilbert $H_p(b_i)$ est calculé avec l'algorithme de Butz pour chaque p-bloc de B_α^{min}. Les positions $Table(H_p(b_i)_{(10)})$ et $Table(H_p(b_i)_{(10)} + 1)$ sont alors lues dans la table d'index pour chaque bloc et les éventuels intervalles de position contigus sont fusionnés.

Étape de raffinement

L'étape de raffinement est un parcours séquentiel des intervalles de signatures dans la base triée, déterminés par l'étape de filtrage. Théoriquement, l'ensemble complet de ces signatures pourrait être considéré comme résultat de la recherche puisque les intervalles ont été déterminés de manière à ce que l'espérance des p-blocs correspondant soit égale à α. Cependant, le nombre de signatures est en pratique trop élevé pour que l'algorithme de fusion des résultats appliqué en sortie de la recherche des signatures locales dans la base soit efficace. Cela est dû au fait que certaines régions à l'intérieur des p-blocs sélectionnés ont en réalité une probabilité quasiment nulle. Pour éliminer les signatures les moins probables, la requête statistique d'espérance α_1 est couplée à une requête à $\epsilon_{\alpha 2}$-près lors de l'étape de raffinement.
Par considération ensembliste, l'espérance α de la recherche complète respectera alors l'inégalité suivante :

$$\alpha \geq \alpha_1 + \alpha_2 - 1$$

En pratique une valeur très proche de 1 pour α_2 est suffisante pour éliminer un grand nombre de signatures. Ainsi, dans le cas d'une loi de probabilité de la distorsion gaussienne, telle que $p_{\Delta S_j}(x) = f_{\mathcal{N}(0,\sigma_g)}(x)$, la valeur du rayon de recherche garantissant une espérance de $\alpha_2 = 99,9\ \%$ est par exemple égale égale à $\epsilon_{\alpha_2} = 6,7\sigma_g$. Quelques expérimentations nous ont permis de montrer qu'un tel rayon étaient suffisant pour éliminer une très grande partie des signatures sélectionnées

par l'étape de filtrage. Ainsi, sur une base réelle de 126 562 273 signatures (environ 2500 heures de vidéo), un rayon de recherche de $6,7\sigma_g = 6,7 \times 18,0 = 120,6$, permet d'éliminer 97,02 % des signatures sélectionnées par l'étape de filtrage (pourcentage moyenné sur 1 000 requêtes sélectionnées aléatoirement dans une base de signatures réelles). La même expérimentation a été réalisée pour des tailles de base différentes et des écart-types différents, et le pourcentage de signatures éliminées reste très important (supérieur à 94,86 % pour des bases allant de 5 320 367 à 252 470 803 de signatures et des écart-types allant de $\sigma_g = 14,0$ à $\sigma_g = 24,0$).
La valeur de ϵ_{α_2} étant toujours choisie de manière à ce que $99,0\ \% < \alpha_2 < 100\ \%$, on considérera par la suite que :

$$\alpha \approx \alpha_1$$

L'étape de raffinement est donc une recherche séquentielle à un rayon près classique appliquée uniquement aux intervalles sélectionnées par l'étape de filtrage. Elle consiste simplement à calculer la distance (L_2 dans notre cas) entre la signature candidate et les signatures parcourues et à stocker dans un tableau les signatures (et les données associées) pour lesquelles la distance est inférieure à ϵ_{α_2}.

4.3.5 Algorithme de recherche par requêtes groupées

Ce mode de recherche est utilisé lorsque la base de signatures et la table d'index ne peuvent être contenues en mémoire. Le principe est de cumuler N_{sig} signatures candidates en mémoire dans une première phase puis de les rechercher dans la base lors d'une seconde phase. Au niveau du système de monitorage d'une chaîne de télévision, cela signifie que les résultats du flux vidéo correspondant aux signatures cumulées, ne seront disponibles qu'à la fin de ces deux phases. Il y aura donc un différé, mais le fonctionnement global sera encore assez rapide pour suivre la diffusion temps réel des images.
La phase de recherche des N_{sig} signatures est basée sur un découpage de la base triée des signatures en plusieurs pages pouvant être contenues en mémoire. Les pages sont chargées successivement en mémoire et les N_{sig} signatures sont recherchées à l'intérieur de chacune des pages. Le coût de recherche des N_{sig} signatures inclut donc le temps de chargement de toutes les pages. L'idée est que ce temps étant fixe, plus le nombre de signatures cumulées sera important plus le temps de chargement moyen par requête sera faible.

L'étape de filtrage de la recherche par requêtes statistiques est totalement indépendante de la base et elle peut être calculée uniquement avec la table d'index. On pourrait donc commencer par charger uniquement la table d'index en mémoire pour calculer les intervalles de positions pour chacune des N_{sig} signatures. Les pages de la base de signatures pourraient alors être chargées successivement et l'étape de raffinement pourrait être réalisée pour chacune des N_{sig} signatures dans chacune des pages. Cette méthode n'est en pratique pas réalisable car l'espace mémoire nécessaire pour stocker tous les intervalles des N_{sig} signatures candidates est trop important.
Notre solution consiste donc à recalculer les intervalles de toutes les signatures à chaque nouvelle page. Cependant, l'algorithme de l'étape de filtrage mis en œuvre permet de restreindre le calcul aux intervalles qui appartiennent effectivement à la page courante. Cela évite de multiplier le coût de l'étape de filtrage par le nombre de pages. La différence avec l'étape de filtrage vue dans le cas de la base en mémoire, se situe au niveau de l'algorithme permettant de déterminer $B(\tau)$ (cf. section 4.3.4 et annexe F), c'est-à-dire l'ensemble des p-blocs ayant une probabilité supérieure à un seuil τ. Le nouvel algorithme permet de déterminer directement l'ensemble des p-blocs ayant une probabilité supérieure à τ et appartenant à la page courante. Il est également décrit dans

l'annexe F.

Cet algorithme nécessite un découpage particulier pour les pages de la base triée. Celui-ci doit en effet correspondre à un découpage régulier de la courbe de Hilbert en 2^θ pages, avec $\theta < p$. Cela permet de faire coïncider la région couverte par une page avec un θ-bloc, c'est à dire un p-bloc à la profondeur θ. Ce découpage de la base est calculé automatiquement lors du démarrage de la phase de recherche du système, en fonction de l'espace mémoire disponible. La valeur de θ est alors déterminée par :

$$\theta = \left\lceil log_2 \left(\frac{N}{N_{mem}} \right) \right\rceil \tag{4.16}$$

où N représente le nombre de signatures dans la base et N_{mem} le nombre de signatures qui peuvent être logées en mémoire. N_{mem} est calculé de manière à n'occuper qu'une partie de la mémoire totale de la machine, typiquement 75 %. Il faut noter que l'optimisation du seuil de probabilité τ par la technique de Newton-Raphson ne peut pas être effectuée sur une seule page puisque le critère porte sur la probabilité cumulée des blocs dans tout l'espace. Cette étape étant cependant indépendante de la base, les valeurs de τ pour chacune des N_{sig} signatures candidates peuvent être calculées et mémorisées avant de commencer à charger la première page. Ces valeurs mémorisées sont ensuite réutilisées lors de la recherche des signatures dans chacune des pages.

4.4 Modélisation et Optimisation

La profondeur p de la partition est un paramètre très important de notre système de recherche puisqu'elle influence le coût des deux étapes de la recherche, à savoir l'étape de filtrage et l'étape de raffinement. Le temps de recherche global du système peut ainsi s'exprimer sous la forme :

$$T(p) = T_f(p) + T_r(p) \tag{4.17}$$

Le coût de l'étape de filtrage $T_f(p)$ est strictement croissant en fonction de p puisque le nombre de p-blocs et donc le temps de calcul pour déterminer B_α^{min} augmente avec p. D'un autre côté, le coût de l'étape de raffinement $T_r(p)$ est décroissant avec p puisque la sélectivité de l'étape de filtrage augmente et que le nombre de signatures sélectionnées diminue. Nous allons voir, grâce à la modélisation proposée par la suite que $T(p)$ présente un minimum en p_{min} que l'on va tâcher d'estimer en fonction des différents paramètres.
La distorsion est modélisé par la loi gaussienne, indépendante sur chaque composante :

$$p_{\Delta S_j}(x) = f_{\mathcal{N}(0,\sigma_g)}(x) \;\; \forall j \in [1, D]$$

La section 4.4.1 présente tout d'abord une modélisation empirique du nombre moyen de p-blocs interceptés par une requête statistique en fonction de la profondeur de la partition. A la section 4.4.2 nous décrivons les hypothèses et le modèle de coût lui même. La section 4.4.3 traite plus spécifiquement de l'optimisation de la profondeur de la partition pour minimiser le coût moyen d'une recherche. Enfin, la section 4.4.4 présente une analyse de l'évolution du coût d'une recherche en fonction de la dimension des signatures et de la taille de la base. Une discussion du modèle est présentée à la section 4.4.6. Le lecteur égaré pourra se référer à la table des notations de ce manuscrit (cf. page xviii).

4.4.1 Nombre moyen de p-blocs interceptés par une requête statistique

L'objectif de cette section est d'évaluer l'espérance de $Card(B_\alpha^{min}(p))$, pour une requête Q uniformément distribuée dans $[0,255]^D$:

$$n(p) = E(Card(B_\alpha^{min}(p)))$$

L'expression de $n(p)$ que nous allons développer est basée sur **une hypothèse empirique vérifiée expérimentalement**.
Compte tenu de la décroissance de la densité de probabilité de la loi gaussienne et puisque chaque bloc de la partition à une profondeur $p-1$ est composé de deux blocs à la profondeur p, il est légitime de supposer que

$$n(p-1) \leq n(p) \leq 2n(p-1)$$

Ceci est équivalent à écrire que

$$n(p) = 2^{\gamma(p)} n(p-1) \quad 0 \leq \gamma(p) \leq 1$$

La partition de l'espace liée à la courbe de Hilbert ne privilégiant aucune direction plutôt qu'une autre, il est également légitime de supposer que la valeur de $\gamma(p)$ reste constante tant que l'ordre de la partition, $q = \lceil \frac{p}{D} \rceil$ (cf. section 4.3.3), reste le même :

$$\gamma(p) = \gamma_q, \quad \forall p \in [(q-1)D, qD]$$

On a alors :

$$log_2(n(p)) = log_2(n(p-1)) + \gamma_q \quad p \in [(q-1)D, qD]$$

et γ_q sera appelé la raison logarithmique de la suite $n(p)$ à l'ordre q.

Cette hypothèse a été vérifiée expérimentalement, comme l'illustre la figure 4.15 montrant le nombre de blocs moyen $n(p)$ pour différentes profondeurs ($D = 20$, $\sigma = 20,0$ et $\alpha = 0,95$), calculé sur 1 000 requêtes uniformément réparties dans l'espace). On constate que $log(n(p))$ est linéaire par morceau, chaque intervalle correspondant à un ordre constant de la partition ($q = 1$ et $q = 2$ sur la figure). Cela signifie que le rapport $\frac{n(p)}{n(p-1)} = 2^{\gamma_q}$ est bien constant par morceau. Les valeurs de la raison γ_q sont les pentes des droites en coordonnées logarithmiques et ont été dans ce cas estimées à $\gamma_1 = 0,271$ and $\gamma_2 = 0,464$.

La figure 4.16 montre l'évolution de la série γ_q pour des ordres q plus élevés et pour deux valeurs de σ_g. On constate que la série γ_q tend vers 1 lorsque l'ordre de la partition augmente ce qui signifie que le ratio $\frac{n(p)}{n(p-1)}$ tend vers 2, lorsque l'ordre de la partition augmente.

4.4.2 Modèle de coût

Afin de trouver la profondeur p_{min} la plus optimale, il est nécessaire de modéliser le coût moyen $T(p)$ d'une recherche par requête statistique. On a vu (cf. equation 4.17, page 109) que celui-ci peut s'exprimer comme la somme du coût de l'étape de filtrage et de celui de l'étape de raffinement.

Si l'on suppose que les signatures sont uniformément réparties dans les blocs de la partition, le temps moyen de l'étape de raffinement peut être modélisé par :

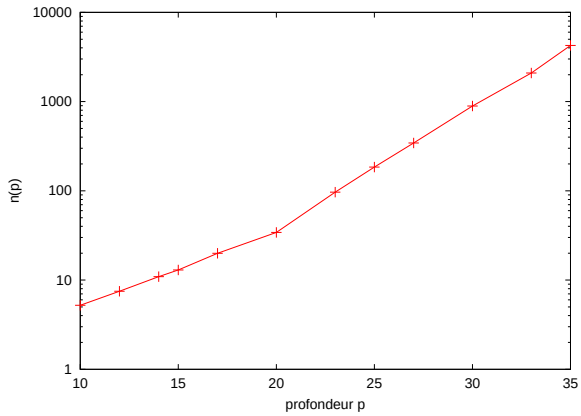


FIG. 4.15 – Nombre moyen de p-blocs $n(p)$ en fonction de la profondeur p (coordonnées logarithmiques)

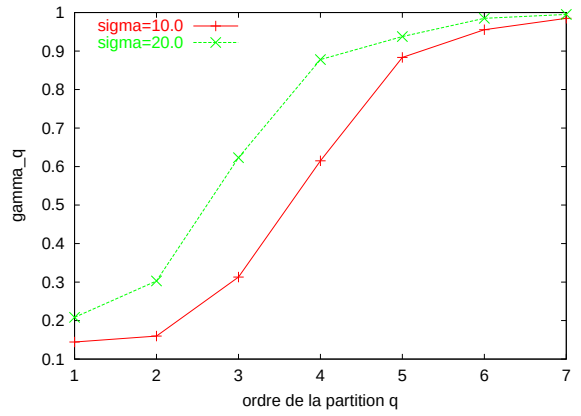


FIG. 4.16 – Valeurs de la suite γ_q pour $\sigma_g = 10,0$ et $\sigma_g = 20,0$

$$T_r(p) = t_s \; n(p) \; \frac{N}{2^p} \tag{4.18}$$

où $n(p)$ est le nombre moyen de p-blocs contenus dans B_α^{min}, N est le nombre de signatures dans la base, $\frac{N}{2^p}$ est le nombre moyen de signatures dans un bloc, et t_s, supposé constant, est le débit d'un parcours séquentiel des signatures (temps de calcul divisé par le nombre de signatures parcourues). Le temps de calcul de l'étape de raffinement dépendant principalement du nombre de distances calculées, il est en effet légitime de supposer qu'il est linéaire en fonction du nombre de signatures. Les coûts supplémentaires provoqués par les sauts en mémoire, lors du passage d'un intervalle à un autre, sont supposés être linéaires en fonction du nombre de blocs, et seront intégrés au modèle du coût de l'étape de filtrage.
Notons que l'hypothèse d'uniformité du nombre de signatures à l'intérieur des blocs est moins contraignante que celle de répartition uniforme des signatures dans l'espace. Notons également que l'expression 4.18 peut être vérifiée alors même que l'hypothèse d'uniformité du nombre de signatures à l'intérieur d'un bloc est fausse. Il suffit que le nombre de signatures moyen à l'intérieur des blocs sélectionnés par la requête soit effectivement égal $\frac{N}{2^p}$. Nous reviendrons sur la pertinence de cette hypothèse dans le cas d'une distribution réelle à la section 4.4.7.

Le coût moyen de l'étape de filtrage est modélisé par :

$$T_f(p) = t_a \; n(p)$$

où t_a regroupe tous les coûts qui sont supposés être linéaires en fonction du nombre de blocs : détermination de B_α^{min}, calcul du préfixe de Hilbert de chaque bloc, fusion des intervalles contigus, accès à la table d'index et coût mémoire de l'accès aux intervalles de signatures dans la base triée. Une discussion sur la pertinence de cette hypothèse comparativement à des coûts réels sera présentée à la section 4.4.6.

Finalement, le temps de recherche moyen total est modélisé par :

$$T(p) = t_a \; n(p) + t_s \; n(p) \; \frac{N}{2^p} \tag{4.19}$$

4.4.3 Optimisation

En substituant l'expression du nombre moyen de p-blocs interceptés $n(p)$ dans l'équation 4.19 du modèle de coût, on peut obtenir une expression par morceau du temps de recherche moyen du système. Il est alors possible de montrer que celui-ci possède un unique minimum global dont une valeur approchée est donnée par l'expression suivante :

$$p_{min} = log_2 \left(\frac{t_s}{t_a} \right) + log_2 \left(N \right) \tag{4.20}$$

Démonstration section D.2, page 208

La valeur de p_{min} ainsi approximée dépend uniquement de la taille de la base et de constantes temporelles fixes et peut donc être calculée facilement. Cette valeur approximative de la profondeur optimale est systématiquement utilisée dans les expérimentations présentées dans ce mémoire et dans le système de monitorage. Nous verrons dans la section 4.4.6, comment les temps t_s et t_a sont estimés au démarrage du système.

4.4.4 Analyse

Nous allons maintenant étudier l'évolution du temps de recherche lorsque la valeur approchée p_{min} définie précédemment est utilisée (cf. équation 4.20).
Étudions d'abord l'évolution de T_{min} en fonction de la taille de la base N. Celle-ci est donnée par la fonction suivante $T_{min}(N)$, définie par morceaux :

$$T_{min}(N) = A_q \; N^{\gamma_q} \tag{4.21}$$

lorsque

$$N \in \left[2^{(q-1)D} \; \frac{t_a}{t_s}, 2^{qD} \; \frac{t_a}{t_s} \right]$$

où A_q est une constante ne dépendant pas de N sur un intervalle.

Démonstration cf. annexe D.3, page 209

La figure 4.17 présente cette fonction pour des valeurs réalistes des paramètres. La modélisation du coût d'une recherche séquentielle dans la base a également été représentée à titre de comparaison (complexité O(N)). On peut tirer les conclusions suivantes :

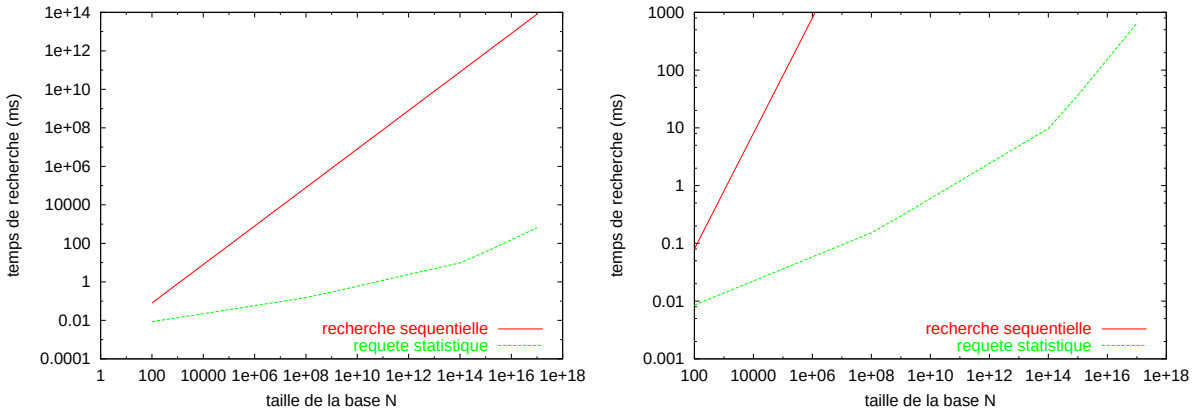


FIG. 4.17 – Temps de recherche modélisé $T_{min}(N)$ en fonction de la taille de la base N ($D = 20$) - à droite : zoom de la figure de gauche

1. Le temps de recherche est toujours sous-linéaire en fonction de N, puisque $\gamma_q < 1, \forall q$. Notons que ce résultat est en cohérence avec l'étude théorique d'A. Gray [83] qui montre que le coût espéré d'une requête à ϵ-près a une complexité en $O(N^\eta)$ ($\eta \in [0, 1]$) quelle que soit la distribution des vecteurs dans la base.
2. Le temps de recherche est de moins en moins sous-linéaire chaque fois que la taille de la base est multipliée par 2^D.
3. Le temps de recherche est asymptotiquement linéaire lorsque N tend vers l'infini puisque γ_q tend vers 1. Notons que cela corrobore notre discussion sur le coût linéaire minimal d'une recherche à ϵ-près (cf. section 4.2.2, page 84). Lorsque la taille de la base tend vers l'infini, la profondeur optimale de la partition tend également vers l'infini et la taille des p-blocs tends vers 0. Notre modélisation de la densité de probabilité de la distorsion étant isotrope, la région de l'espace correspondant à $B_\alpha^{min}(p)$ tend alors vers une hyper-sphère de rayon ϵ_α. L'évolution du nombre de signatures sélectionnées par l'étape de filtrage est alors linéaire en fonction de la taille de la base, en conséquence le coût de l'étape de raffinement aussi. Le coût de l'étape de filtrage évolue quant à lui en $2^{p_{min}}$ et est donc également linéaire en fonction de la taille de la base. Nous verrons cependant, lors des expérimentations, que ce comportement asymptotique linéaire est loin d'être atteint avec des tailles de base réelles.

Regardons maintenant l'évolution du temps de recherche en fonction de la dimension D des signatures. Elle s'exprime par la fonction $T_{min}(D)$, définie par morceau :

$$T_{min}(D) = C_q \; D \; 2^{(q-1) \; D \, \gamma_q} \prod_{r=1}^{q-1} 2^{D \, \gamma_r} \tag{4.22}$$

lorsque

$$D \in \left[\frac{log_2(N \frac{s}{a})}{q-1}, \frac{log_2(N \frac{s}{a})}{q} \right]$$

où C_q est une constante ne dépendant pas de D sur un intervalle.

Démonstration cf. annexe D.3, page 209

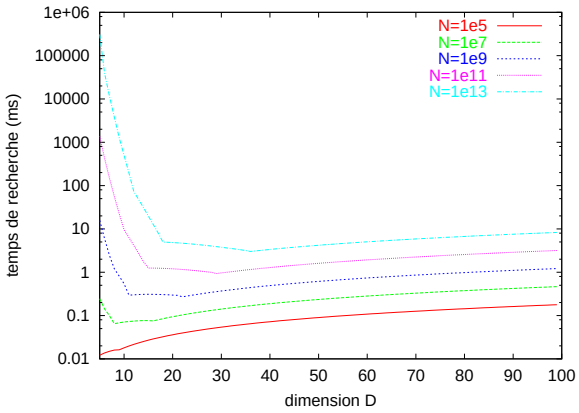


FIG. 4.18 – Temps de recherche modélisé $T_{min}(D)$ en fonction de la dimension D pour plusieurs tailles de base N

La figure 4.18 représentent l'évolution du temps de recherche en fonction de la dimension des signatures et pour différentes tailles de base N. On constate que le temps de recherche est décroissant par morceau en fonction de la dimension lorsque $D \leq log_2(\frac{s\,N}{a})$ et croissant par morceaux lorsque $D > log_2(\frac{s\,N}{a})$). Lorsque la dimension tend vers l'infini avec une taille de base constante, le temps de recherche est linéaire en fonction de la dimension (car $q = 1$) et ne souffre donc pas de la malédiction de la dimension.

4.4.5 Cas de la recherche par requêtes groupées

Nous nous intéressons maintenant à la modélisation du temps de recherche moyen d'une requête statistique lorsque le mode de fonctionnement est celui des requêtes groupées. Le temps total pour rechercher les N_{sig} signatures peut être modélisé par la somme du temps total pour charger toutes les 2^θ pages de la base en mémoire et du temps nécessaire pour rechercher toutes les signatures dans ces pages. En ramenant ce temps au nombre de signatures pour avoir un temps moyen par requête, on a :

$$T_{tot}(p,\theta) = T_{rech}(p,\theta) + \frac{T_{load}}{N_{sig}} \tag{4.23}$$

où T_{tot} représente le temps total moyen par requête, T_{rech} est le temps de recherche moyen dans les pages et $\frac{T_{load}}{N_{sig}}$ est le temps moyen de chargement de toutes les pages. Nous supposons que ce dernier ne dépend que du débit de lecture séquentielle sur le disque et qu'il ne dépend pas du nombre de pages ou de la profondeur de la partition de la base. Le chargement de la table d'index est supposé négligeable devant le chargement de la base. On aura ainsi $T_{load} = N\ t_{load}$ où t_{load} est une constante représentant le débit de lecture du disque en unité de temps par signature.

Le coût moyen de la recherche d'une signature dans les pages de la base est modélisé quant à lui par la somme du coût que l'on aurait eu si la base était entièrement en mémoire et d'un coût linéaire en nombre de pages, censé représenter certains des coûts de calcul de l'étape de filtrage qui sont répliqués à chaque page ainsi que les sorties de caches mémoires provoquées par le découpage de l'algorithme de recherche en 2^θ étapes. Ainsi, $T_{tot}(p,\theta)$ s'exprime de la manière

suivante :

$$T_{tot}(p,\theta) = T(p) + 2^{\theta} t_{page} + \frac{N\ t_{load}}{N_{sig}} \tag{4.24}$$

où $T(p)$ est le même que le temps de recherche moyen en mémoire défini par l'équation 4.19. Il est facile de montrer que cette nouvelle fonction comporte encore un minimum en p_{min} et finalement le temps de recherche total peut être approximé par (cf. annexe D.3) :

$$T_{tot} \approx T_{min} + \frac{N}{N_{mem}} t_{page} + \frac{N\ t_{load}}{N_{sig}} \tag{4.25}$$

où T_{min} est la modélisation du temps de recherche optimal en mémoire (cf équation 4.21).

On constate que le passage du mode par requête unique au mode par requêtes groupées rajoute deux coûts qui sont linéaires en fonction de la taille de la base.

Celui qui est dû au chargement de la base peut cependant être stabilisé si on se permet d'augmenter le nombre N_{sig} de signatures cumulées lorsque la taille de la base augmente. La solution retenue est de choisir $N_{sig} = \kappa\sqrt{N}$, on a alors :

$$\frac{N\ t_{load}}{N_{sig}} = \frac{t_{load}}{\kappa}\sqrt{N}$$

Cela permet d'avoir un coût de recherche supplémentaire sous-linéaire en fonction de la taille de la base avec une augmentation du nombre de signatures cumulée elle aussi sous-linéaire en fonction de la taille de la base.

Les coûts linéaires en fonction du nombre de pages, regroupés dans le deuxième terme de l'équation 4.25, resteront pour leur part linéaires en fonction de la taille de la base. On ne peut en effet pas augmenter indéfiniment la capacité N_{mem} de la mémoire. Nous verrons dans la partie expérimentale, que ces coûts sont négligeables devant T_{min} pour un nombre de pages limité puis qu'ils deviennent prépondérants pour un nombre important de pages. Nous verrons que malgré cette évolution linéaire, le temps de recherche reste cependant très faible comparé à celui d'un parcours séquentiel de la base (la pente de la linéarité est très faible).

4.4.6 Discussion sur le modélisation des coûts intermédiaires

Le but de cette section est de discuter la modélisation du coût de l'étape de filtrage $T_f(p)$ et de celui de l'étape de raffinement $T_r(p)$ (cf. section 4.4.2).

Pour apprécier la cohérence de l'hypothèse de linéarité du coût de l'étape de filtrage en fonction du nombre de blocs sélectionnés, il nous faut mesurer les coûts des différentes opérations qui constituent cette étape, à savoir :

- la sélection des p-blocs constituant B_{α}^{min} par la technique de Newton-Raphson,
- le calcul de la clé de Hilbert de chacun des p-blocs,
- le tri de ces clés permettant de fusionner celles qui sont contigues,
- les coûts d'accès mémoire qui comprennent les accès à la table d'index et les accès à la base de signatures.

Le coût de l'algorithme de sélection des blocs a été mesuré et moyenné sur 1 000 requêtes pour différentes profondeurs de la partition variant de 10 à 35, avec $\alpha = 0,90$ et $\sigma_g = 22$.

Le coût de calcul de la clé de Hilbert et le coût de calcul pour trier un ensemble de clés ont été pour leur part mesurés isolément en itérant ces deux fonctions un certain nombre de fois.

En ce qui concerne les coûts mémoire, nous proposons ici un modèle grossier. Dans [123], Manegold et al. proposent un modèle générique des coûts d'accès aux mémoires hiérarchiques des systèmes modernes. Le modèle est basé sur la latence des différents niveaux de cache et le nombre de sorties de cache, chaque niveau de cache étant traité individuellement. Dans le cadre de cette étude, ils ont développé un outil permettant de déterminer automatiquement les caractéristiques des différents niveaux de cache d'une machine. Nous en avons récupéré une version *open source* afin de déterminer les paramètres des niveaux de cache de notre système.
Si on considère qu'une sortie de cache advient à tous les niveaux de cache pour tous les accès mémoire dans des données volumineuses, c'est-à-dire dans la table d'index et dans la base. Pour chaque bloc, deux accès mémoires sont nécessaires pour lire la borne supérieure et la borne inférieure de l'intervalle dans la table d'index. Un accès mémoire est ensuite nécessaire pour accéder à la borne inférieure dans la base de signatures. Le coût mémoire du parcours séquentiel de l'intervalle sera quant à lui pris en compte dans le coût de l'étape de raffinement. Le coût mémoire total de l'accès à la base peut donc être modélisé par trois accès mémoire par bloc, chaque accès étant modélisé par la somme des temps de latence de chaque niveau de cache.

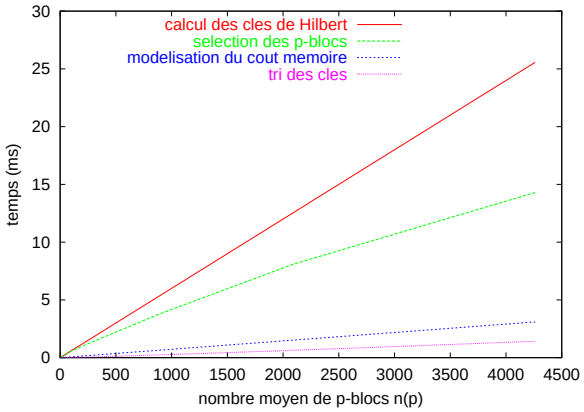


FIG. 4.19 – Coût des différentes opérations de l'étape de filtrage en fonction du nombre de p-blocs sélectionnés

La figure 4.19 montre l'évolution du coût des quatre opérations principales de l'étape de filtrage, en fonction du nombre de p-blocs sélectionnés. On constate que le coût principal est le calcul de la clé de Hilbert des p-blocs et que celui-ci est comme prévu linéaire en fonction du nombre de blocs. Le second coût le plus important correspond à l'algorithme de sélection des blocs et semble sous-linéaire en fonction du nombre de blocs. L'hypothèse de linéarité pour la somme des quatre coûts n'est cependant pas aberrante. Il est intéressant de noter que le goulet d'étranglement n'est pas le coût des accès mémoires comme on aurait pu s'y attendre. Toute optimisation algorithmique sera donc pleinement profitable.

Voyons maintenant le cas de l'étape de raffinement. D'après l'étude de Manegold [123], le coût

mémoire de plusieurs parcours séquentiels non consécutifs peut se modéliser ainsi :

$$T_{mem} = \sum_{r=1}^{n_c} \sum_{i=1}^{Card(B^{\alpha}_{min})} \left\lceil \frac{C_i}{C_r} \right\rceil l_r$$

où n_c est le nombre de niveaux de cache, C_r est la taille d'une ligne de cache de niveau r, et C_i est la longueur en octet du i^{eme} p-bloc. En appliquant la règle $\lceil x \rceil \leq x + 1$, il est facile de montrer que :

$$T_{mem} \leq \sum_{r=1}^{n_c} \left\lceil \sum_{i=1}^{Card(B^{\alpha}_{min})} \frac{C_i}{C_r} \right\rceil + Card(B^{\alpha}_{min}) \sum_{r=1}^{n_c} l_r$$

Le premier terme est le coût qu'aurait engendré la recherche si tous les blocs avaient été consécutifs. Le second terme représente le coût d'un accès mémoire dans le pire des cas, c'est-à-dire lorsqu'il y a une sortie de cache à chaque niveau. Ce coût a déjà été modélisé dans l'étape de filtrage et n'entre donc pas dans le coût de l'étape de raffinement. Celui-ci est donc bien linéaire en fonction du nombre de signatures parcourues. Il en est de même pour le coût CPU puisqu'une distance et une comparaison sont calculées pour chaque signature. Le temps global de l'étape de raffinement est donc bien linéaire en fonction du nombre de signatures parcourues. Nous avons évalué le coût mémoire à environ 20 % du coût total de l'étape de raffinement (qui est lui-même d'environ par signature parcourue).

4.4.7 Cas d'une distribution réelle

Le but de cette section est de discuter, dans le cas d'une distribution réelle, l'hypothèse selon laquelle le nombre moyen de signatures à l'intérieur d'un p-bloc sélectionné est égal à :

$$\bar{n_b}(p) = \frac{N}{2^p}$$

Il faut tout d'abord remarquer que cette estimation est systématiquement fausse si la dimension intrinsèque des données est inférieure à la dimension réelle. Les algorithmes pour déterminer la dimension intrinsèque d'un ensemble de descripteurs sont précisément basés sur un comptage du nombre de descripteurs contenus à l'intérieur des blocs d'une partition hiérarchique de l'espace [65, 19]. On peut ainsi montrer que dans le cas d'un ensemble de signatures ayant une dimension intrinsèque $D' < D$, le nombre moyen de signatures à l'intérieur d'un p-bloc est égal à :

$$\bar{n_b}(p) = \frac{N}{2^{p\frac{D'}{D}}} \tag{4.26}$$

Nous avons évalué le nombre moyen de signatures à l'intérieur d'un p-bloc dans le cas d'une base de nos signatures locales. La base est constitué de 5 320 367 signatures (environ 100 heures de vidéo) et les requêtes sont constituées de 10 000 signatures locales extraites du flux vidéo d'une chaîne de télévision. La figure 4.20 représente le nombre moyen de signatures à l'intérieur des p-blocs sélectionnés par une requête statistique, en fonction de la profondeur de la partition. L'axe des ordonnées du graphique est en coordonnées logarithmiques.

On constate que la courbe a une évolution linéaire et le nombre de signatures par bloc sélectionné peut donc bien s'exprimer sous la forme :

$$\bar{n_b}(p) = \frac{A}{2^{\varphi p}}$$

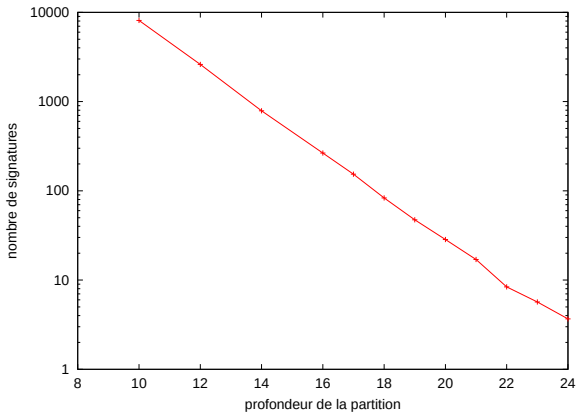


FIG. 4.20 – Nombre moyen de signatures à l'intérieur d'un p-bloc sélectionné par l'étape de filtrage en fonction de la profondeur de la partition

En estimant la pente de la courbe, on trouve que $\varphi = 0,794$, soit en identifiant avec le modèle 4.26 et en se rappelant que la dimension réelle est $D = 20$, on trouve que :

$$D' = 20 \times 0,794 = 15,88$$

Il s'agit d'une valeur cohérente pour la dimension intrinsèque de nos données. Rappelons en effet, que la signature globale est constituée de 4 sous-signatures de dimension 5 et que chacune d'entre elles a été normée. Chacune de ces normalisations provoquant la perte d'une dimension, la dimension intrinsèque théorique des signatures est forcément inférieure ou égale à 16. Le fait que la dimension estimée soit très proche de 16, ne signifie pas forcément que les signatures sont uniformément réparties dans l'espace normalisé, car il s'agit uniquement d'un nombre moyen de signatures par bloc.

En intégrant la dimension intrinsèque des signatures à notre modèle de coût et en conservant la valeur de p_{min} comme profondeur optimale, il est possible de montrer que le temps de recherche moyen optimal en mémoire reste de la forme :

$$T_{min}(N) = A'_q \; N^{\gamma_q}, \tag{4.27}$$

où A'_q est encore une constante ne dépendant pas de N sur un intervalle d'ordre constant, mais de valeur plus élevée que A_q :

$$A'_q = \frac{A_q}{2} \left(1 + \left(\frac{t_a}{t_s} \right)^{(1-\frac{D'}{D})} \right)$$

Démonstration cf. annexe D.3, page 209

Ainsi, la sous-linéarité en fonction de la taille de la base est conservée avec le même facteur de sous-linéarité γ_q. Le temps de recherche est uniquement multiplié par une constante. Étant donné que $1 \leq D' \leq D$, A'_q est forcément supérieur à A_q. Dans notre cas, si l'on se fie à la dimension intrinsèque des données estimée à $D' = 15,88$, le rapport $\frac{D'}{D}$ est assez élevé et le surcoût est limité. Si on remplace t_s et t_a par leur valeur typiquement observée lors des expérimentations, on obtient :

$$A'_q \approx 1,5 A_q$$

Dans le cas d'une base de signatures où la dimension intrinsèque des données est très inférieure à la dimension réelle, et que l'on est capable d'estimer la dimension intrinsèque des données, on peut prendre comme approximation de la profondeur optimale la valeur suivante (cf. annexe D.3, page 209) :

$$p'_{min} = \frac{D}{D'}\left(log_2\left(\frac{t_s}{t_a}\right) + log_2\left(N\right)\right) = \frac{D}{D'}p_{min} \tag{4.28}$$

Le temps de recherche est alors de nouveau de la forme

$$T_{min}(N) = A''_q \; N^{\gamma_q}$$

mais la constante A''_q est alors plus faible que A'_q.

Ces conclusions sont importantes, puisqu'elles montrent que la sous-linéarité du temps de recherche en fonction de la taille de la base ne nécessite pas l'hypothèse d'uniformité des signatures dans la base comme nous l'avions fait dans un premier temps. La condition nécessaire pour respecter le modèle proposé est que le nombre moyen de signatures à l'intérieur d'un p-bloc sélectionné ait une décroissance exponentielle en fonction de la profondeur.

4.5 Synthèse

Dans ce chapitre, nous avons présenté la méthode de recherche approximative des signatures par requête statistique que nous proposons. Nous avons d'abord introduit le concept de requête statistique dont le principe est d'adapter la recherche à la modélisation de la distorsion des signatures. Contrairement aux recherches approximatives classiques, le contrôle de l'erreur ne concerne pas les résultats d'une requête géométrique mais directement l'espérance du nombre de signatures pertinentes retrouvées. Nous avons vu que l'abandon de la contrainte géométrique d'une requête pouvait améliorer très significativement les temps de recherche.
Nous avons ensuite présenté la méthode proposée pour traiter les requêtes statistiques. Celle-ci profite du contexte statique de notre application, pour ordonner physiquement les signatures selon une courbe de Hilbert et éviter ainsi de parcourir une structure d'indexation pour accéder à la base. La recherche elle-même n'est en fait qu'un parcours séquentiel de certaines parties de cette base triée de signatures. Ces intervalles sont déterminés lors d'une première étape dite de filtrage qui déterminent un ensemble de blocs de l'espace partitionné. Cet ensemble est déterminé de manière à garantir une espérance α de retrouver une signature pertinente au sens du modèle de distorsion.
Nous avons ensuite proposé une modélisation du coût de recherche de notre technique qui nous a permis de définir une valeur optimale pour la profondeur de la partition qui est un paramètre essentiel. C'est en effet ce paramètre qui réalise le compromis entre la sélectivité de l'étape de filtrage et son coût de calcul. La modélisation nous a également conduit à analyser l'évolution du temps de recherche en fonction de différents paramètres. Nous avons ainsi montré que sous l'hypothèse d'indépendance des composantes de la distorsion et de décroissance exponentielle du nombre de signatures moyen à l'intérieur d'un p-bloc, le coût d'une recherche était sous-linéaire en fonction de la taille de la base et qu'il ne souffrait pas de la *malédiction de la dimension*.
Nous avons enfin proposé un mode de fonctionnement par requêtes groupées qui permet d'effectuer des recherches dans des bases de taille supérieure à la mémoire disponible. Ce fonctionnement ajoute des coûts linéaires en fonction de la taille de la base au temps de recherche moyen.

Chapitre 5

Evaluation de la méthode proposée pour la recherche de signatures

L'objectif de ce chapitre est d'étudier les performances de la méthode proposée pour la recherche de signatures par requêtes statistiques. Cette étude est faite indépendamment des autres parties du système complet de détection de copies, comme la détection des points d'intérêt ou l'étape de fusion des résultats locaux de la recherche. Le lien avec ces autres étapes et l'évaluation du système complet seront étudiés dans la partie III de ce mémoire.
En revanche, les bases de données réelles et les requêtes réelles utilisées pour une partie des expérimentations de ce chapitre sont constituées des signatures locales proposées dans ce mémoire.
La section 5.1 donne des précisions sur les conditions expérimentales et les données utilisées. La section 5.2 décrit les expérimentations réalisées sur des bases de données synthétiques, tandis que la section 5.3 décrit les expérimentations réalisées sur des bases de signatures réelles. Pour ces deux sections, seul le mode de fonctionnement en mémoire par requête unique est étudié. Le mode de fonctionnement par requêtes groupées sera étudié dans la section 5.4 sur des bases de signatures réelles.

5.1 Conditions expérimentales

5.1.1 Modélisation de la distorsion pour l'étape de filtrage

L'objectif des expérimentations de ce chapitre n'est pas de valider un modèle probabiliste plutôt qu'un autre pour modéliser au mieux les distorsions réelles. L'objectif est d'évaluer les performances de la méthode de recherche en supposant la densité de probabilité de la distorsion parfaitement connue.
La modélisation de la distorsion choisie pour l'ensemble des expérimentations de ce chapitre est la loi gaussienne déjà étudiée. Toutes les composantes du vecteur de distorsion sont supposées indépendantes et ont une densité de probabilité égale à :

$$p_{\Delta S_j}(x) = f_{\mathcal{N}(0,\sigma_g)}(x) \quad \forall j \in [1, D]$$

Les valeurs choisies pour σ_g sont cohérentes avec les écart-types estimés sur des ensembles de distorsions réelles, dans notre contexte applicatif. Le lien entre ce choix de modélisation et la précision effective d'une recherche dans le cas de distorsions réelles sera étudié dans la section 6.2 du chapitre suivant. Nous verrons en particulier que l'espérance d'une requête statistique basée sur cette simple modélisation permet un contrôle satisfaisant de la qualité réelle de la recherche.

Cette modélisation de la distorsion est utilisée dans le système de monitorage d'une chaîne de télévision que nous étudierons par la suite. Les temps de recherche mesurés lors des expérimentations sur des bases réelles et des requêtes réelles de ce chapitre sont donc à mettre directement en correspondance avec les temps de recherche du système de monitorage.

La valeur du rayon de recherche de l'étape de raffinement est choisie de manière à ce que $\alpha_2 = 99,9$ % et il est égale à :

$$\epsilon_{\alpha_2} = 6,7\sigma_g$$

garantissant une espérance réelle de la recherche α' telle que (cf. section 4.3.4) :

$$\alpha' \geq 0,999\alpha$$

où α représente l'espérance de l'étape de filtrage.

5.1.2 Machines et implémentation

Deux machines ont été utilisées pour ces expérimentations :

Monit01

- Processeur Pentium IV
- CPU de fréquence $1,8GHz$
- $1,0Go$ de mémoire centrale (RAM)
- $256Ko$ de taille de cache

Monit02

- Processeur Pentium IV
- CPU de fréquence $3,0GHz$
- $2,0Go$ de mémoire centrale (RAM)
- $512Ko$ de taille de cache

Les algorithmes ont été implémentés en langage C et C++ sous un système *Linux* avec une distribution *RedHat 7.2*.
Les temps de recherche sont mesurés par le temps utilisateur de la commande `getrusage()`.
Pour les mesures de temps relatives au chargement de la base en mémoire ou à l'étape d'indexation, c'est le temps absolu de la commande `getrusage()` qui est utilisé.
Les composantes des signatures sont codées sur un octet et celles-ci sont directement suivies des données associées, codées ici sur 6 octets. Toutes les distances dans l'espace des signatures sont donc exprimées dans l'espace $[0,255]^D$.
Le calcul des distances de l'étape de raffinement a été implémenté en langage assembleur SSE, afin d'accélérer cette étape critique.

5.1.3 Méthode de recherche séquentielle

Dans un but comparatif, nous avons implémenté une méthode de recherche séquentielle comparable avec l'implémentation de notre méthode. Celle-ci est en fait le même algorithme que l'étape de raffinement, mais appliqué à la base entière. Il suffit en fait de supprimer l'étape de

filtrage de notre méthode. Le calcul des distances est donc également implémenté en langage assembleur SSE.
Cette méthode de recherche peut également fonctionner par requêtes groupées en appliquant l'étape de raffinement à la totalité des pages chargées en mémoire (il suffit de supprimer l'étape de filtrage de notre méthode de recherche par requêtes groupées).

5.1.4 Les bases de signatures

Plusieurs bases de signatures distinctes ont été utilisées lors de ces expérimentations. Les premières sont des bases synthétiques, générées selon une loi de probabilité uniforme dans l'espace $[0, 255]^D$. Une première série comporte 9 bases qui contiennent un nombre N croissant de signatures de dimension fixe $D = 20$:

$$N \in \{1.10^4 \ ; \ 1.10^5 \ ; \ 1.10^6 \ ; \ 2,5.10^6 \ ; \ 5.10^6 \ ; \ 7,5.10^6 \ ; \ 1.10^7 \ ; \ 1.10^8 \ ; \ 1.10^9\}$$

Cette série sera nommée $UNI(20, N)$ où N représente la taille de la base.
Une deuxième série comporte 5 bases qui contiennent $N = 10\ 000\ 000 = 1.10^7$ signatures de dimensions D croissantes :

$$D \in \{10; 15; 20; 25; 30\}$$

Cette série sera nommée $UNI(D, 1.10^7)$ où D représente la dimension des signatures.

Des bases de signatures locales réelles seront également utilisées lors de ces expérimentations. Elles ont été construites à partir d'une base de l'INA (nommée base SNC), qui contient, à l'heure actuelle, plus de 100 000 heures d'extraits vidéo au format $MPEG1$ (1 Mbit/sec, taille d'image 352×288). Durant les deuxième et troisième années de la thèse, les signatures locales proposées dans ce mémoire ont été calculées sur environ 80 000 heures d'extraits vidéo issus de la base SNC, en traitant séquentiellement plusieurs des serveurs mettant à disposition les fichiers $MPEG1$. Le contenu de la base SNC est très varié. On peut classer très succinctement l'essentiel des extraits de la base de la manière suivante :

- Les commandes numérisées : il s'agit de la numérisation des extraits d'archives de l'INA stockées sur films qui ont un jour été commandées par un diffuseur.
- Des cassettes montées : il s'agit d'enregistrements d'émission, de reportages ou de films fournis par les chaînes de télévision elles mêmes. Dans le cas de reportages diffusés lors d'un journal télévisé par exemple, ces enregistrements ne contiennent que les reportages tels que la chaîne les possédait avant leur diffusion. Ces enregistrements ne contiennent donc pas les interventions du présentateur par exemple.
- Des parallèles antenne : il s'agit d'enregistrement d'émissions, de reportages ou de films lors de leur diffusion. Dans le cas d'un journal télévisé, ces enregistrements contiendront les interventions du présentateur ainsi que les génériques de début et de fin d'émission. Les grilles des programmes de diffusion n'étant pas précises, ces enregistrements sont effectués plusieurs minutes avant le début ou la fin de l'émission visée. Pour cette raison, ils peuvent contenir tout ce qui peut être diffusé lors des intervalles de diffusion inter-émission, à savoir des publicités, des habillages de chaîne, des bandes annonces d'autres émissions, des habillages d'émission (jingle, etc.).

Au niveau de la variété du contenu des images, la base contient tout ce que l'on peut observer à la télévision française hier et aujourd'hui : des émissions de variété, des journaux télévisés, du sport, des reportages, des films, des clips musicaux, des publicités, des *jingle* de chaînes,

des cartes météo, des génériques d'émissions, des reportages photographiques, etc. Certaines archives anciennes sont en noir et blanc, très bruitées, ou dégradées par des traces de scotche, des poussières, etc. D'autres en revanche ont été capturées au cours des dernières années, ou de l'année en cours et sont de bien meilleure qualité, avec en général plus de mouvements. La maintenance d'une base de cette dimension étant compliquée, et les technologies de l'audiovisuel n'ayant pas toujours été celles d'aujourd'hui, cette base contient également des mires fixes, des extraits noirs ou bruités de quelques dizaines de secondes, ou encore des comptes à rebours. Il est cependant difficile de quantifier la proportion de tel ou tel type d'images. Le nombre de copies des mêmes images à l'intérieur de ces bases est également difficilement évaluable. Nous verrons plus tard, dans ce mémoire, que le système que nous avons mis en place permet justement d'avoir une meilleure compréhension du contenu d'une telle base.

Chaque fichier $MPEG1$ de vidéo de la base SNC contient entre quelques minutes et quelques heures de vidéo. Pour chacun de ces fichiers, le robot calculant les signatures crée un fichier de signatures indépendant. Les bases utilisées lors des expérimentations de ce chapitre, ont été crées en fusionnant un grand nombre de ces fichiers dans des fichiers uniques, avant d'être indexées suivant la procédure décrite dans la section 4.3.2. 13 bases de tailles croissantes ont ainsi été construites. Chacune d'entre elles constitue un sous-ensemble des bases de tailles supérieures. Le tableau 5.1 récapitule les propriétés de ces bases. Nous rappelons que la dimension des signatures locales utilisées est $D = 20$.

nom de la base	nombre de signatures	nombre d'heures signées	taille en Mo
$REEL_1$	77 131	1,18	2Mo
$REEL_3$	243 704	3,54	6Mo
$REEL_{10}$	702 615	10,83	18Mo
$REEL_{30}$	1 956 558	29,18	51Mo
$REEL_{100}$	5 320 367	77,89	138Mo
$REEL_{250}$	14 098 729	250,33	367Mo
$REEL_{500}$	23 258 368	509,36	605Mo
$REEL_{1000}$	48 345 280	1051,55	1 256 Mo
$REEL_{2500}$	126 562 273	2546,52	3 291 Mo
$REEL_{5000}$	252 470 803	5055,45	6 564 Mo
$REEL_{10000}$	518 320 618	10 003,06	13 476 Mo
$REEL_{18000}$	940 021 221	17 845,22	24 440 Mo
$REEL_{30000}$	1 543 902 419	29 093,67	40 141 Mo

TAB. 5.1 – Description des bases de signatures locales réelles

5.1.5 Evaluation de la profondeur de la partition et construction de la table d'index

En dehors des expérimentations où nous évaluons l'influence de ce paramètre, la valeur de la profondeur de la partition est fixée **pour chaque base** par l'expression approchée de la valeur optimale théorique obtenue grâce à notre modélisation (cf. section 4.4.3, equation 4.20). Nous rappelons ici qu'elle vaut :

$$p_{min} = log_2\left(\frac{t_s}{t_a}\right) + log_2\left(N\right) \tag{5.1}$$

Les valeurs de t_s et t_a ont été estimées empiriquement **pour chacune des machines** utilisées :

- t_s est estimé en effectuant une recherche séquentielle en mémoire avec le même algorithme que l'étape de raffinement appliqué à la base $UNI(20, 10^7)$. On a alors

$$t_s = \frac{T_{sequ}}{10^7}$$

où T_{sequ} représente le temps total de cette recherche séquentielle. La valeur finalement retenue pour t_s est moyennée sur 100 requêtes tirées aléatoirement selon une loi uniforme dans l'espace $[0, 255]^D$.

- t_a est déterminé en évaluant indépendamment les 4 principaux coûts censés le composer comme cela a été fait lors de la section 4.4.6 relative à la modélisation des coûts intermédiaires (cf. page 115).

Les valeurs obtenues pour t_a et t_s sont résumées dans la table 5.2.

nom de la machine	t_a (μs)	t_s (μs)
Monit01	5,27	0,062
Monit02	3,03	0,034

TAB. 5.2 – Valeurs empiriques de t_a et t_s

La valeur théorique de p_{min} n'étant pas forcément entière, celle-ci est arrondi à l'entier le plus proche. Nous avons récapitulé la profondeur de la partition utilisée pour chaque base dans la table 5.3.

Nom de la base	p_{min}	Nom de la base	p_{min}
$UNI(20, 10^4)$	7	$REEL_1$	10
$UNI(20, 10^5)$	10	$REEL_3$	11
$UNI(20, 10^6)$	14	$REEL_{10}$	13
$UNI(20, 2, 5.10^6)$	15	$REEL_{30}$	14
$UNI(20, 5, 0.10^6)$	16	$REEL_{100}$	16
$UNI(20, 7, 5.10^6)$	16	$REEL_{250}$	17
$UNI(D, 10^7)$	17	$REEL_{500}$	18
$UNI(20, 10^8)$	20	$REEL_{1000}$	19
$UNI(20, 10^9)$	23	$REEL_{2500}$	20
		$REEL_{5000}$	21
		$REEL_{10000}$	22
		$REEL_{18000}$	23
		$REEL_{25000}$	24

TAB. 5.3 – Profondeur de la partition utilisée pour les différentes bases

Pour chaque base, une table d'index est construite selon la procédure décrite dans la section 4.3.3 (page 103) avec la valeur de p_{min} correspondante. Chaque table contient ainsi un nombre N_{ind}

d'entiers correspondant à une position dans le fichier de signatures, chacune d'entre elle étant codée sur 8 octets. On a alors

$$N_{ind} = 2^{p_{min}} = N\frac{t_s}{t_a}$$

Avec les valeurs de t_a et t_s données précédemment, l'espace mémoire occupé par la table d'index est environ 90 fois inférieur à celui occupé par la base.

5.2 Bases de données synthétiques

L'objectif de ces expérimentations sur des bases de signatures uniformément réparties dans l'espace est double :

1. Il s'agit premièrement de valider les algorithmes proposés pour la recherche par requête statistique dans le cas d'une distribution de la distorsion parfaitement maîtrisée. L'objectif est de vérifier que l'on retrouve effectivement les signatures avec une espérance égale à l'espérance α fixée pour la requête.

2. Le deuxième objectif est d'évaluer l'évolution du temps de recherche suivant plusieurs paramètres et de valider le modèle de coût proposé, dans le cas d'une distribution uniforme.

Les requêtes utilisées dans toute cette section sont obtenues à l'aide d'un générateur de données gaussiennes. Le principe est de sélectionner aléatoirement une signature de la base et de lui ajouter un vecteur de distorsion généré selon la loi gaussienne décrite précédemment (cf. section 5.1.1). La requête ainsi obtenue peut ensuite être recherchée dans la base et le temps de cette recherche est mesuré. Si la signature d'origine figure dans les résultats de la recherche, celle-ci est considérée comme correctement retrouvée.
Cette opération est répétée pour 1 000 requêtes et le temps d'une recherche est moyenné sur les 1 000 recherches. La pourcentage de requêtes correctement retrouvées est également moyenné sur ces 1 000 recherches et sera noté r_g.

5.2.1 Influence de la profondeur de la partition

Le but de cette expérimentation est d'étudier l'évolution du temps de recherche en fonction de la profondeur de la partition. La valeur p de la profondeur de la partition n'est ainsi pas fixée à p_{min} comme dans le cas des autres expérimentations, mais prend plusieurs valeurs comprises entre $p = 10$ et $p = 25$. La base utilisée est $UNI(D, 10^7)$ et une table d'index a été construite pour chaque valeur de p. Les paramètres de la requête statistique sont :

$$\alpha = 0,90 \quad \sigma_g = 18,0$$

La figure 5.1 montre l'évolution du temps de recherche moyen en fonction de la profondeur de la partition.

On constate bien qu'il existe un minimum comme cela a été affirmé dans la modélisation théorique. Le minimum de cette courbe est situé à une profondeur :

$$p_{min-exp} = 19$$

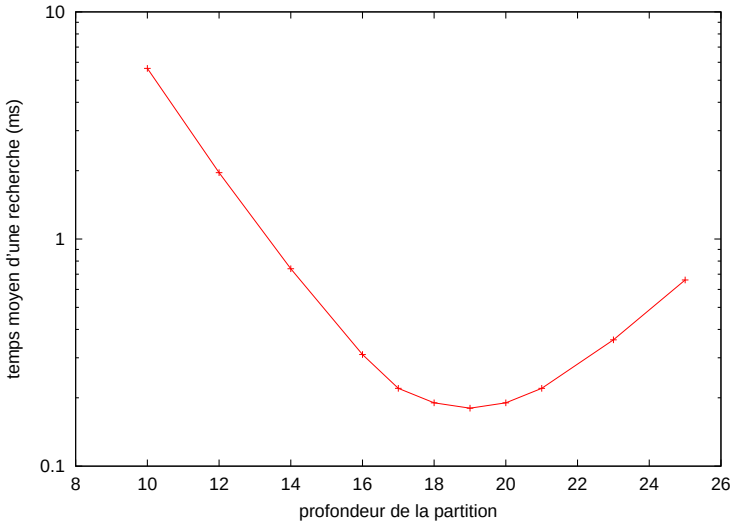


FIG. 5.1 – Temps de recherche moyen (ms) en fonction de la profondeur de la partition

et le temps de recherche moyen correspondant est $0,18\ ms$.

La valeur approchée de la profondeur optimale théorique (cf. table 5.3) vaut pour sa part :

$$p_{min} = 17$$

Le temps de recherche moyen correspondant est $0,21\ ms$. La perte liée à l'utilisation de cette valeur théorique approchée par rapport au minimum réellement obtenu est donc uniquement de 15 %. Cela justifie amplement l'utilisation de cette valeur théorique étant donné la difficulté d'obtention de la courbe réelle (une table d'index doit être construite pour chaque valeur de p).

Pour savoir si la différence entre $p_{min-exp}$ et p_{min} est due uniquement à l'approximation de la valeur optimale théorique exacte ou bien si elle est plutôt due à un biais du modèle, il est nécessaire d'estimer les valeurs de la série γ_q. Rappelons que la valeur théorique optimale exacte de la profondeur est en effet égale à

$$p_q^{min} = p_{min} + log_2\left(\frac{1-\gamma_q}{\gamma_q}\right) \tag{5.2}$$

où la valeur de q est l'unique ordre pour lequel on aura

$$p_q^{min} \in [(q-1)D, qD]$$

Les valeurs de la série γ_q peuvent être obtenues expérimentalement comme cela a déjà été vu dans la section 4.4.1 (page 110). Il faut pour cela mesurer le nombre moyen de blocs interceptés par la requête statistique en fonction de la profondeur de la partition. Les valeurs de γ_q correspondent alors aux coefficients directeur de cette courbe en coordonnées logarithmiques (cf. figure 4.15). Dans le cadre de cette expérimentation, nous avons mesuré

$$\gamma_1 = 0,31$$

soit

$$p_1^{min} = 18,23 \in [1, 20] \tag{5.3}$$

La profondeur optimale théorique ($p_1^{min} = 18,23$) est donc très proche de la profondeur optimale réelle ($p_{min-exp} = 19$) ce qui valide la modélisation théorique de cette valeur.

5.2.2 Validation de l'algorithme de recherche par requête statistique

Pour valider l'algorithme de recherche par requête statistique, il faut comparer l'espérance α de la requête statistique avec le taux r_g de signatures correctement retrouvées, sachant que la distorsion des requêtes est générée suivant la même loi que celle utilisée pour la recherche.
La base utilisée est $UNI(20, 10^7)$ et les paramètres de la requête statistique sont :

$$\sigma_g = 22,4$$

$$\alpha \in \{30\ \%\ ,\ 40\ \%\ ,\ 50\ \%\ ,\ 60\ \%\ ,\ 70\ \%\ ,\ 80\ \%\ ,\ 85\ \%\ ,\ 90\ \%\ ,\ 95\ \%\ ,\ 97,5\ \%\ ,\ 99\ \%\ ,\ 99,9\ \%\}$$

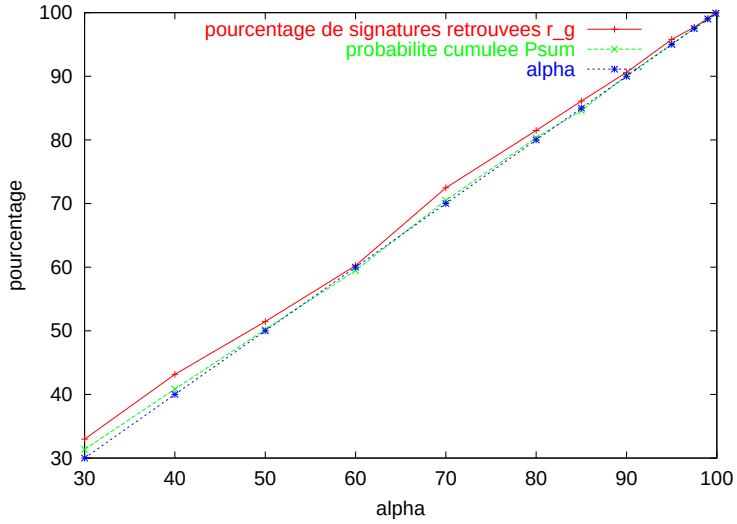


FIG. 5.2 – Probabilité cumulée et taux de signatures retrouvées en fonction de α

La figure 5.2 représente le taux de signatures correctement retrouvées r_g en fonction de l'espérance α de la requête statistique. Sur cette même figure, nous avons également tracé la courbe de référence $\alpha = \alpha$ ainsi que la valeur de la probabilité cumulée sur l'ensemble des blocs sélectionnés par la requête statistique (soit $P_{sum}(\tau_{max})$, cf. equation 4.15, page 106). Cette courbe montre que la probabilité de l'ensemble des blocs sélectionnés par l'étape de filtrage est très proche de l'espérance α fixée pour la requête. Ceci montre l'efficacité de la technique de Newton-Raphson pour trouver le seuil optimal de probabilité d'un bloc et contrôler ainsi l'espérance de la requête. Il est important de noter que cette courbe est totalement indépendante des données de la base et qu'elle serait donc identique dans le cas de données réelles, si l'on conserve cette même modélisation.
La courbe représentant le taux de signatures correctement retrouvées r_g montre que l'espérance de la requête statistique permet bien de contrôler la qualité réelle de la recherche, étant donnée une loi de probabilité parfaitement connue. L'erreur entre α et le taux de signatures retrouvées est ainsi comprise entre :

$$0,1\% < r - \alpha < 3,15\ \%$$

Le pourcentage de signatures réellement retrouvées est donc toujours supérieur à l'espérance de la requête et son contrôle est relativement précis. L'écart entre le taux de signatures correctement

retrouvées et la probabilité de l'ensemble des blocs sélectionnés par l'étape de filtrage peut être dû à plusieurs raisons. Le taux de signatures retrouvées est tout d'abord estimé uniquement sur 1 000 requêtes et il est normal que l'espérance théorique ne soit pas systématiquement atteinte. Il semble cependant qu'il y ait un biais supplémentaire qui tend à augmenter légèrement le taux de signatures correctement retrouvées et qui pourrait être dû à un bais du générateur de données gaussiennes.

5.2.3 Influence de la taille de la base

Regardons maintenant l'influence du nombre de signatures dans la base sur le temps de recherche moyen. Les bases utilisées sont celles de la série $UNI(20, N)$ pouvant être logées entièrement en mémoire principale de la machine **Monit01**, c'est-à-dire jusqu'à $N = 10^7$ signatures. Les paramètres de la requête statistique sont :

$$\sigma_g = 22,4 \quad \alpha = 96\ \%$$

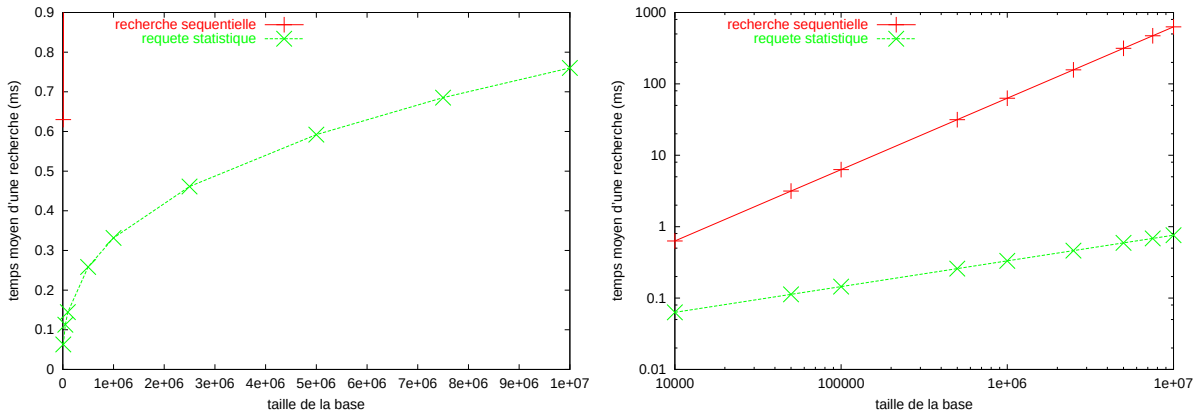


FIG. 5.3 – Comparaison du coût d'une recherche statistique et d'une recherche séquentielle - à gauche : coordonnées cartésiennes - à droite : coordonnées logarithmiques

La figure 5.3 représente le temps moyen d'une recherche en fonction de la taille de la base, pour notre méthode de recherche par requêtes statistiques et pour un parcours séquentiel de la base. La figure a été également représentée en coordonnées logarithmiques afin de représenter les deux techniques sur un même graphique.
Cette expérimentation montre clairement que le coût de recherche de notre technique est sous-linéaire par rapport à la taille de la base. Ainsi, le gain de temps par rapport à un parcours séquentiel de la base augmente lorsque la taille de la base augmente. Pour la plus grande base contenant $N = 10^7$ signatures, notre technique est ainsi 828 fois plus rapide qu'un parcours séquentiel, avec un temps de recherche égal à $0,76\ ms$.
La linéarité de la courbe en coordonnées logarithmiques montre que le temps de recherche moyen peut s'exprimer, conformément au modèle théorique (cf. equation 4.21, page 112), sous la forme :

$$T(N) = A_1\ N^{\gamma_1}$$

γ_1 est la raison logarithmique de la suite $n(p)$ et représente ici le coefficient directeur de la droite en coordonnées logarithmiques. Il a été estimé à :

$$\gamma_1 = 0,34$$

La constante A_1 a été estimé pour sa part à :

$$A_1 = 2,308.10^{-3}$$

Le temps moyen de la recherche peut ainsi être évalué par l'équation :

$$T(N) = 2,308.10^{-3}\ N^{0,34}\ ms$$

Il est important de noter que cette équation ne correspond qu'au premier intervalle de la définition par morceaux du modèle théorique du coût d'une recherche. Si on se réfère à la table 5.3 (page 125), on observe en effet, que pour toutes les bases utilisées pour cette expérimentation ($N \leq 10^7$), la profondeur optimale approchée de la partition, p_{min}, reste inférieur à $D = 20$ et reste donc du premier ordre ($q = 1$). La limitation de la mémoire principale ne nous permet pas de mesurer le temps de recherche par requête unique pour des bases plus grandes et l'on ne peut donc pas observer directement le changement de pente de la courbe en coordonnées logarithmiques.

Pour observer le changement de pente dû au changement d'ordre de la partition et valider ainsi le modèle théorique, nous avons refait la même expérience en augmentant arbitrairement la profondeur de la partition. Le principe est d'indexer les bases avec une profondeur, non pas égale à la profondeur optimale p_{min} mais égale à :

$$p_{min} + p_0$$

où p_0 représente une constante ne variant pas en fonction de taille de la base.
Si l'on redéveloppe le modèle de coût avec cette nouvelle valeur, on obtient toujours un coût théorique de la forme :

$$T(N) = A_q^0\ N^{\gamma_q}$$

mais avec une constante A_q^0 plus élevée que la constante A_q dans le cas optimal. La valeur expérimentale choisie pour l'augmentation de la profondeur est $p_0 = 7$. La table 5.4 récapitule les nouvelles valeurs de la profondeur pour chacune des bases utilisées dans cette nouvelle expérimentation.
La figure 5.4 représente le temps réponse moyen en fonction de la taille de la base pour ces nouvelles valeurs de la profondeur de la partition. Sur cette figure, nous avons également représenté de nouveau la courbe du temps de recherche moyen pour les valeurs optimales approchées de la profondeur de la partition ($p = p_{min}$).

Nom de la base	$p_{min} + p_0$
$UNI(20, 10^4)$	14
$UNI(20, 10^5)$	17
$UNI(20, 10^6)$	20
$UNI(20, 2, 5.10^6)$	22
$UNI(20, 5, 0.10^6)$	22
$UNI(20, 7, 5.10^6)$	23
$UNI(D, 10^7)$	24

TAB. 5.4 – Profondeurs expérimentales sur-evaluées ($p = p_{min} + 7$)

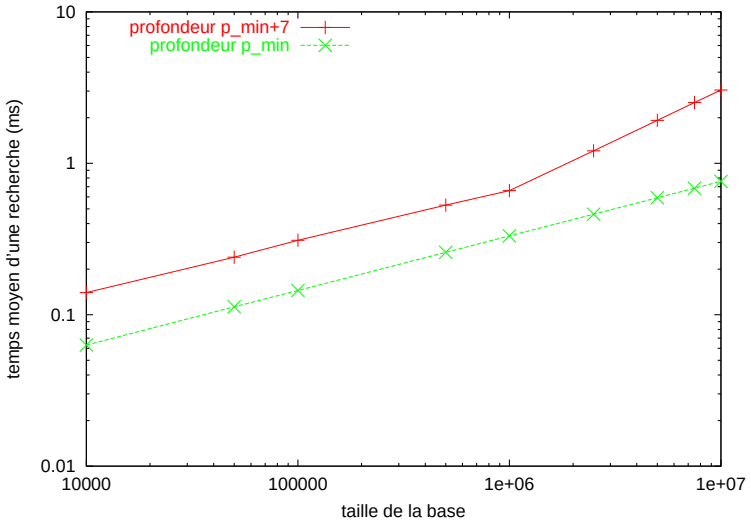


FIG. 5.4 – Temps de recherche moyen (ms) en fonction de la taille de la base pour la profondeur optimale $p = p_{min}$ et la profondeur sur-évaluée $p = p_{min} + 7$

Conformément au modèle théorique, la courbe montre clairement un changement de pente (en coordonnées logarithmiques) lorsque la profondeur de la partition passe de $q = 1$ à $q = 2$ ($p_{min} + p_0 = D = 20$ pour $UNI(20, 10^6)$). On vérifie bien que la valeur de γ_1 reste la même ($\gamma_1 = 0,34$) puisque la pente des deux courbes est la même lorsque la profondeur de la partition est inférieure à $D = 20$. On peut également désormais estimer la valeur de γ_2 par la pente de la droite définie dans l'intervalle correspondant à un ordre 2 de la profondeur et on a :

$$\gamma_2 = 0,64$$

Le coût d'une recherche pour des profondeurs de partition comprises entre 20 et 40, peut donc être évalué par :

$$T(N) = A_2 \; N^{0,64} \; ms$$

Celui-ci est donc toujours sous-linéaire en fonction de la taille de la base mais il augmente plus rapidement que pour des profondeurs de partition comprises entre 1 et 20.
L'évaluation de ce paramètre est importante puisqu'elle permet de prévoir l'évolution qu'aura le coût d'une recherche si l'on utilise des machines avec plus de mémoire, permettant de contenir en mémoire des bases de taille plus importante. En outre, ce coût est une partie du coût global du mode de recherche par requêtes groupées que nous expérimenterons par la suite pour des tailles de base dont la profondeur optimale est supérieure à $D = 20$.

Pour finir, on peut ajouter que la profondeur optimale approchée, p_{min}, restera du deuxième ordre ($20 \leq p_{min} \leq 40$), tant que la taille de la base sera inférieure à :

$$N_{2\rightarrow 3} = 93,45.10^{12}$$

Dans le cas de nos signatures locales, cela représenterait plus de 1 milliard d'heures de vidéo et l'on peut donc aujourd'hui considérer qu'elle ne sera pas atteinte avant plusieurs dizaines d'année.

5.2.4 Influence de la dimension des signatures

L'influence de la dimension des signatrues sur le temps de recherche moyen a été étudiée sur la série de bases $UNI(D, 10^7)$. Les paramètres utilisés pour la requête statistique sont :

$$\sigma_g = 22,4 \quad \alpha = 96\ \%$$

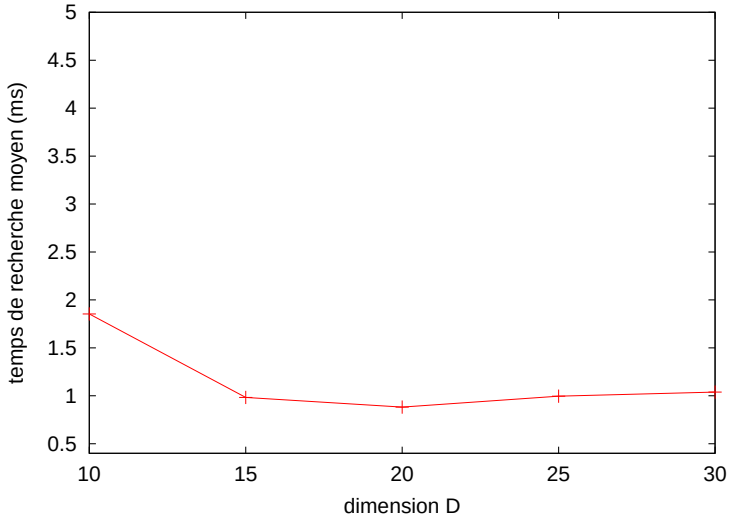


FIG. 5.5 – Temps de recherche moyen en fonction de la dimension D des signatures

La figure 5.5 représente le temps de recherche moyen en fonction de la dimension des signatures. Pour des raisons d'implémentation de l'algorithme de Butz, notre système est limité à une dimension maximale de $D = 32$ et nous ne pouvons pas étendre cette expérimentation à des dimensions considérées comme *très élevées* (D=100, 200, 500, etc.). Rappelons cependant que notre objectif principal était de pouvoir rechercher des signatures de dimension moyennement élevée dans le cas de base très volumineuses.
On peut toutefois vérifier la cohérence de cette portion de courbe avec le modèle théorique proposé (cf. figure 4.18). Ainsi, le minimum global théorique est censé se situer en $D_{min} = 16$, et l'on peut vérifier sur la courbe qu'un minimum global pourrait effectivement se situer entre 15 et 20.
On peut tout de même conclure que notre méthode ne semble pas souffrir de la malédiction de la dimension pour des paramètres constants de la requête statistique et pour des dimensions inférieures à $D = 32$. Le temps moyen d'une recherche est de plus théoriquement linéaire en fonction de la dimension, si $D > 16$. Notons, cependant que la constance du paramètre σ_g lorsque la dimension augmente n'a pas vraiment de sens, puisque, la signature n'étant plus la même, σ_g devrait en réalité être réévalué.

5.3 Bases de signatures réelles

L'objectif des expérimentations de cette section est d'étudier l'évolution du temps de recherche moyen dans le cas de bases de signatures réelles et de requêtes réelles. Les bases utilisées sont celles décrites dans la table 5.1 (page 124). Les requêtes utilisées sont les signatures locales

réelles, extraites d'une capture d'environ 16 minutes d'une chaîne de télévision française. Le nombre total de requêtes est ainsi égal à 19 723 signatures et les temps de recherche présentés lors des expérimentations suivantes sont moyennés sur l'ensemble de ces requêtes.
Notons que ces requêtes ne sont pas des signatures issues de copies des motifs de la base. Elles représentent uniquement un échantillon de la majorité des requêtes du système réel de monitorage d'une chaîne de télévision. L'aptitude du modèle probabiliste utilisé à retrouver une signature effectivement issue d'une copie d'un motif de la base n'est donc pas étudié ici. Cette étude sera menée dans le chapitre suivant, relatif à l'impact du système de recherche statistique sur la détection de copies. L'objectif, ici, est uniquement d'étudier les performances en terme de temps de recherche. Les paramètres de la requête statistique sont en revanche cohérents avec les statistiques réelles de la distorsion que nous étudierons par la suite.
Toutes les expérimentations sont réalisées sur la machine **Monit02**.

5.3.1 Influence de la profondeur de la partition

Comme dans le cas des bases de données synthétiques, nous étudions ici l'évolution du temps de recherche en fonction de la profondeur de la partition. La valeur p de la partition n'est ainsi pas fixée à p_{min} comme dans le cas des autres expérimentations, mais prend plusieurs valeurs comprises entre $p = 10$ et $p = 24$. La base utilisée est $REEL_{100}$ et une table d'index a été construite pour chaque valeur de p. Les paramètres de la requête statistique sont :

$$\sigma_g = 18,0 \quad \alpha = 0,90$$

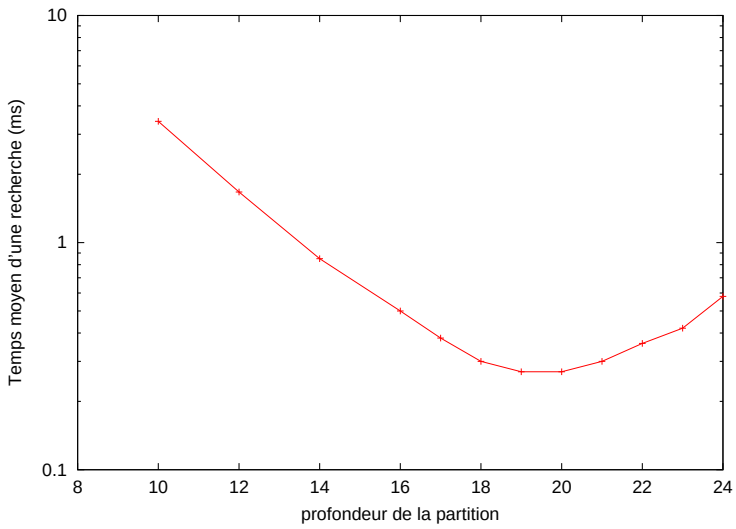


FIG. 5.6 – Temps de recherche moyen (ms) en fonction de la profondeur de la partition

La figure 5.6 montre l'évolution du temps de recherche moyen en fonction de la profondeur de la partition. On constate bien qu'il existe également un minimum comme cela a été affirmé dans la modélisation théorique et vérifié pour les données synthétiques. Le minimum de cette courbe est situé à une profondeur :

$$p_{min-exp} = 19$$

et le temps de recherche moyen correspondant est $0,28\ ms$. La valeur approchée de la profondeur optimale théorique (cf. table 5.3) vaut pour sa part :

$$p_{min} = 16$$

Le temps de recherche moyen correspondant est $0,49\ ms$. On constate que l'écart entre la profondeur optimale réelle et la profondeur optimale approchée théorique est plus important que dans le cas des données synthétiques uniformément réparties. L'utilisation de la valeur théorique induit ainsi une perte de performance plus importante puisque le temps de recherche est multiplié par $1,75$ tandis qu'il n'était multiplié que par $1,17$ dans le cas des données synthétiques. Cet écart est dû à la distribution réelle des signatures dans la base qui ne suit pas une loi uniforme.

Dans la section 4.4.7, nous avions redéveloppé notre modèle de coût dans le cas de données ayant une dimension intrinsèque D' inférieure à la dimension réelle D. Nous avions ainsi vu, que dans ce cas, une meilleure approximation de la profondeur optimale théorique pouvait s'exprimer par (cf. equation 4.28, page 119) :

$$p'_{min} = \frac{D}{D'} p_{min}$$

La dimension intrinsèque de nos signatures avait alors été estimée à $D' = 15,88$. Si on utilise cette valeur pour calculer la nouvelle approximation théorique de la profondeur optimale, on trouve :

$$p'_{min} = \frac{20}{15,88} p_{min} = 19,98$$

Cette nouvelle approximation de la profondeur optimale théorique issue de notre modèle coïncide avec la profondeur optimale réelle. L'utilisation de p'_{min} au lieu de p_{min} permet alors de diviser le temps moyen d'une recherche par $1,75$. On voit donc que la prise en compte de la dimension intrinsèque des données dans le modèle est assez rentable. Si la dimension intrinsèque des données était très inférieure à la dimension réelle, la prise en compte de celle-ci dans l'approximation de la profondeur optimale deviendrait indispensable si l'on ne veut pas dégrader fortement les performances optimales du système.

5.3.2 Influence de la taille de la base

Le but de cette expérimentation est d'étudier l'influence du nombre de signatures dans la base sur le temps de recherche moyen. Les bases utilisées sont celles de la série $REEL$ pouvant être logées entièrement en mémoire principale de la machine **Monit02**, c'est-à-dire jusqu'à la base $REEL_{1000}$ contenant $48,34.10^6$ signatures, représentant environ 1 000 heures de vidéo. Les paramètres de la requête statistique sont :

$$\sigma_g = 20 \quad \alpha = 90\ \%$$

La figure 5.7 représente le temps moyen d'une recherche en fonction de la taille de la base, pour notre méthode de recherche par requêtes statistiques et pour un parcours séquentiel de la base. La figure a été également représentée en coordonnées logarithmiques afin de représenter les deux techniques sur un même graphique.
Cette expérimentation montre clairement que le coût de recherche de notre technique reste sous-linéaire en fonction de la taille de la base pour des signatures réelles. Ainsi, le gain par rapport à un parcours séquentiel de la base augmente lorsque la taille de la base augmente. Pour la plus

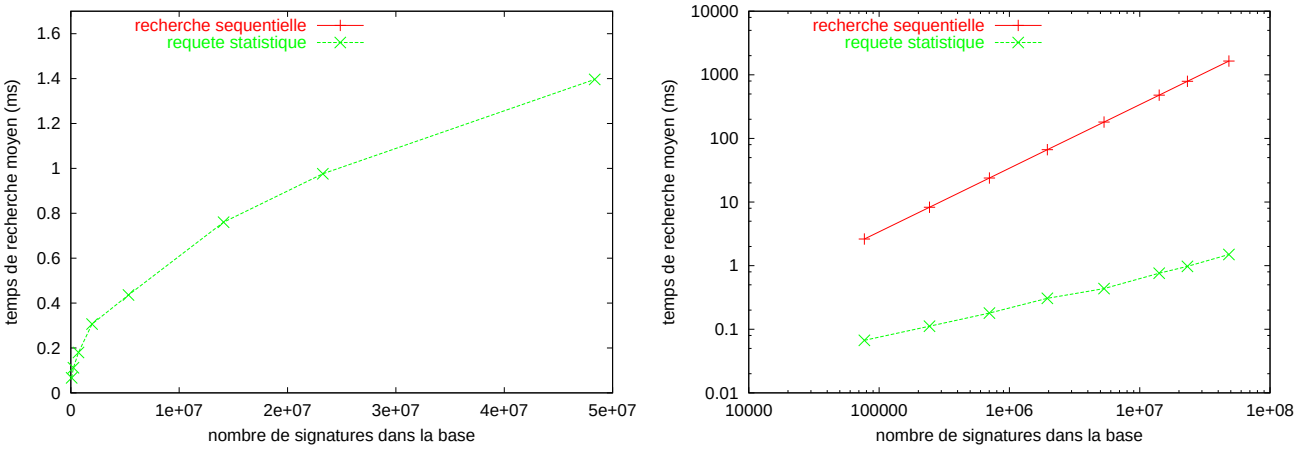


FIG. 5.7 – Comparaison du coût d'une recherche statistique et d'une recherche séquentielle - à gauche : coordonnées cartésiennes - à droite : coordonnées logarithmiques

grande base représentant environ 1 000 heures de vidéo, notre technique est ainsi 1610 fois plus rapide qu'un parcours séquentiel.
La linéarité de la courbe en coordonnées logarithmiques montre que le temps de recherche moyen peut s'exprimer, conformément au modèle théorique (cf. equation 4.21, page 112), sous la forme :

$$T(N) = A_1 \; N^{\gamma_1}$$

γ_1 représente le coefficient directeur de la droite en coordonnées logarithmiques et vaut :

$$\gamma_1 = 0,463$$

La constante A_1 a été estimée pour sa part à :

$$A_1 = 0,217.10^{-3}$$

et le temps moyen de la recherche peut être évalué par :

$$T(N) = 0,217.10^{-3} \; N^{0,463} \; ms$$

On peut noter que la valeur de la raison logarithmique γ_1 est sensiblement plus élevée que dans le cas des données synthétiques uniformes (on avait alors $\gamma_1 = 0,34$). Cela signifie que l'évolution du temps de recherche est plus rapide dans le cas des signatures réelles. Rappelons que la valeur de γ_1 est déterminée par l'évolution du nombre moyen de p-bloc interceptés par la requête statistique en fonction de la profondeur (γ_1 est la raison logarithmique de la suite $n(p)$ à l'ordre 1, cf. section 4.4.1). Son augmentation dans le cas des données réelles est donc très certainement due à la distribution réelle non uniforme des requêtes.

Malgré cette augmentation, la sous-linéarité de l'évolution du temps de recherche en fonction de la taille de la base reste cependant très rentable. Le temps de recherche moyen est ainsi multiplié seulement par 2,90 lorsque la taille de la base est multipliée par 10 (le temps de recherche était multiplié par 2,19 dans le cas des signatures uniformément réparties).

5.3.3 Influence de l'espérance α de la requête statistique

L'expérimentation présentée ici s'intéresse à l'influence de l'espérance de la requête statistique α sur le temps de recherche moyen. La base utilisée est $REEL_{250}$ et les paramètres de la requête statistique sont :

$$\sigma_g = 18,0$$

$$\alpha \in \{30\ \%\ ,\ 40\ \%\ ,\ 50\ \%\ ,\ 60\ \%\ ,\ 70\ \%\ ,\ 80\ \%\ ,\ 85\ \%\ ,\ 90\ \%\ ,\ 95\ \%\ ,\ 97,5\ \%\ ,\ 99\ \%\ ,\ 99,9\ \%\}$$

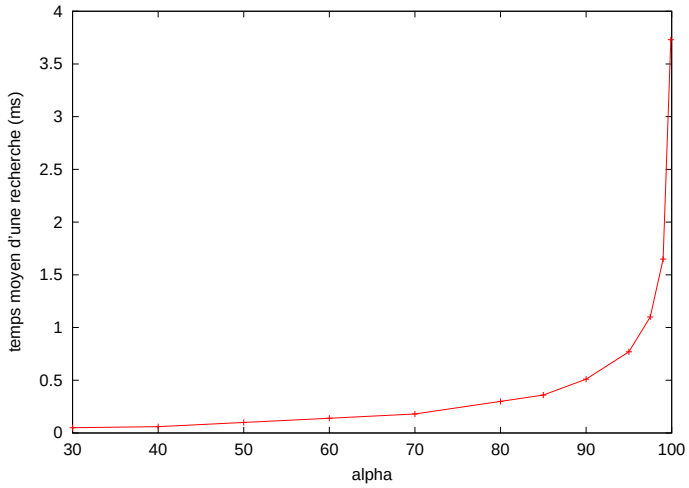


FIG. 5.8 – Temps de recherche moyen (ms) en fonction de l'espérance de la requête statistique α

La figure 5.8 représente le temps moyen d'une recherche en fonction de l'espérance α de la requête statistique. Elle montre qu'une faible perte de qualité de la recherche peut engendrer un gain de performances important. Ainsi, en passant d'une espérance de la requête statistique de 99 % à 80 %, le temps moyen d'une recherche est divisé par $5,5$.

Le gain temporel lié à une diminution constante de l'espérance a cependant tendance à diminuer lorsque l'espérance de la requête diminue. Cela montre que la dégradation de la qualité des résultats est de moins en moins rentable lorsque l'espérance de retrouver une signature pertinente est trop fortement diminuée.

5.3.4 Comparaison avec une requête à ϵ-près de même espérance

L'objectif de cette expérimentation est de comparer le temps de recherche d'une requête statistique d'espérance α avec celui d'une requête à ϵ-près de même espérance (requête à ϵ_α-près, cf. équation 4.3, page 87).

La valeur du paramètre σ_g de la requête statistique est fixée à

$$\sigma_g = 18,0$$

.

La valeur ϵ_α du rayon de la requête à ϵ_α-près est fixée de manière à garantir une espérance α

pour une distorsion modélisée par :

$$p_{\Delta S_j}(x) = f_{\mathcal{N}(0,\sigma_g)}(x) \quad \forall j \in [1, D]$$

Les signatures candidates sont les mêmes pour les deux types de requête (19 723 signatures issues d'une capture TV de 16 minutes).

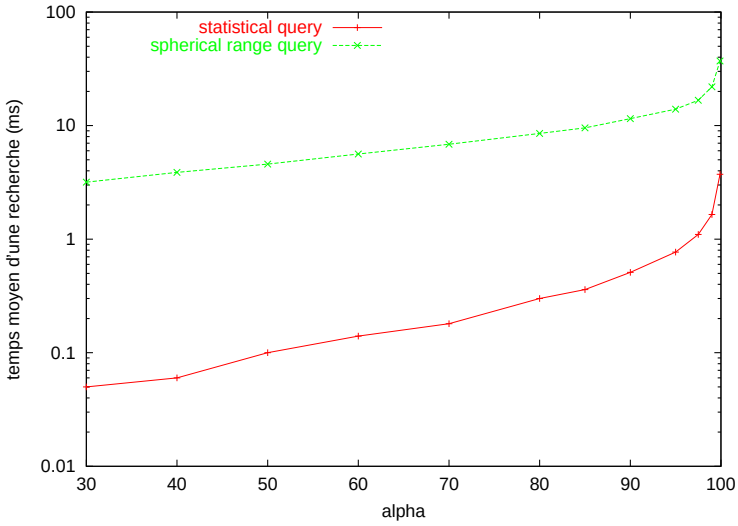


FIG. 5.9 – Temps de recherche moyen (ms) en fonction de α, pour une requête statistique d'espérance α et pour une requête à ϵ_α-près

La figure 5.9 représente le temps de recherche moyen des deux types de requête en fonction de l'espérance α de la requête (en coordonnées logarithmiques). La figure montre que pour une même espérance, le coût d'une requête statistique est très inférieur au coût d'une recherche à un rayon près. Le temps de recherche est ainsi divisé par un facteur allant de 78 à 11, suivant la valeur de l'espérance de la requête. Pour une espérance $\alpha = 80$ %, la recherche par requête statistique est 28 fois plus rapide. Cette expérimentation montre clairement que le fait de s'abstenir de la contrainte géométrique d'une requête sphérique permet des gains de performances très importants tout en garantissant la même espérance de retrouver correctement une signature dans les résultats de la requête.

5.3.5 Influence de σ_g

Nous nous intéressons ici à l'influence du paramètre σ_g de la requête satistique sur le temps de recherche moyen. Rappelons, que la valeur de ce paramètre est représentative de la sévérité des distorsions des signatures résultant de la transformation des images. La base utilisée est $REEL_{250}$ et les paramètres de la requête statistique sont :

$$\alpha = 90\ \%$$

$$\sigma_g \in \{5\ ,\ 10\ ,\ 15\ ,\ 20\ ,\ 25\ ,\ 30\ ,\ 35\ ,\ 40\}$$

La figure 5.10 représente le temps de recherche moyen en fonction de σ_g, en coordonnées logarithmiques. Le temps d'un parcours séquentiel est également représenté comme référence. La

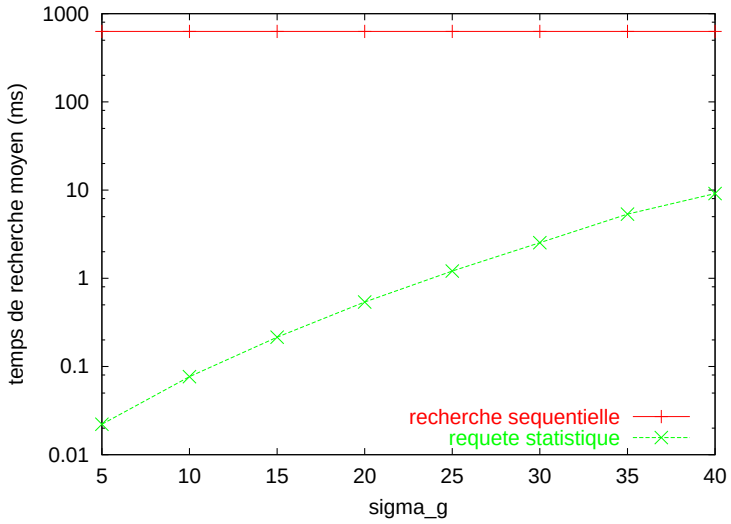


FIG. 5.10 – Temps de recherche moyen en fonction du paramètre σ_g de la requête statistique

forte augmentation du temps de recherche moyen en fonction de σ_g montre que la sévérité des distortions est décisive pour les performances du système de recherche. Le gain de performance par rapport à un parcours séquentiel est ainsi décroissant lorsque σ_g augmente. Notre méthode est 9460 fois plus rapide pour un faible écart-type de la distorsion ($\sigma_g = 10$) alors qu'elle sera 236 fois plus rapide pour un écart-type important ($\sigma_g = 30$).

5.4 Evaluation du mode de fonctionnement par requêtes groupées

L'objectif de cette expérimentation est d'étudier l'influence de la taille de la base sur le temps de recherche moyen dans le cas où le mode de recherche par requêtes groupées est utilisé. Nous nous intéresserons uniquement aux bases de signatures réelles interrogées par des requêtes réelles. Le protocole expérimental est donc le même que dans la section précédente hormis le fait que le nombre de requêtes sur lequel sont moyennés les temps de recherche sont plus importants. Ce mode de fonctionnement n'est en effet rentable que si l'on dispose de suffisamment de requêtes pouvant être recherchées de manière groupées. Comme cela a été proposé lors de la modélisation de ce mode de fonctionnement, le nombre de requêtes sera ainsi proportionnel à la racine carré du nombre de signatures dans la base. Plus précisément, nous choisirons :

$$N_{sig} = 10\sqrt{(N)}$$

Le principe de l'expérimentation de cette section est le même que celui du fonctionnement du système complet de monitorage d'une chaîne de télévision. Les signatures sont tout d'abord extraites du flux vidéo d'une chaîne télévision et cumulées en mémoire. Lorsque le nombre de ces signatures atteint N_{sig}, la recherche de ces signatures dans la base de référence est déclenchée. Nous rappelons que pour ce mode de fonctionnement, le temps total de recherche des N_{sig} signatures se décompose en deux parties distinctes. La première partie est constituée du temps qu'il faut pour charger les pages de la base en mémoire principale. La deuxième partie du coût total est constitué du temps de recherche en mémoire des N_{sig} signatures à l'intérieur de chacune des pages. Ces deux coûts ne sont pas mesurés de la même manière :
Le coût mémoire de la recherche est mesuré par le temps utilisateur retourné par la commande

`getrusage()` utilisée comme un chronomètre que l'on déclenche à la fin du chargement de chaque page et que l'on stoppe à la fin de la recherche des N_{sig} signatures à l'intérieur de chaque page. Le temps moyen de la recherche en mémoire est obtenu en divisant le temps ainsi obtenu par le nombre de requêtes N_{sig}.
Le temps de chargement des pages est mesuré par le temps absolu retourné par la commande `getrusage()` utilisée comme un chronomètre que l'on déclenche au début du chargement de chaque page et que l'on stoppe à la fin du chargement de chaque page.
Le temps moyen total par requête est obtenu en additionnant les deux temps ainsi obtenus et en divisant par le nombre de requêtes N_{sig}.

Avant de présenter les résultats de l'expérimentation, la table 5.5 récapitule les informations relatives au mode de fonctionnement par requêtes groupées pour chacune des bases et en particulier le nombre de requêtes groupées N_{sig}. Ces requêtes groupées devant être stockées en mémoire, nous donnons également l'espace mémoire correspondant ($Mem(N_{sig})$) ainsi que la durée approximative de vidéos que représentent ces signatures ($Heures(N_{sig})$). Ce temps est important puisqu'il s'agit de la longueur du différé à partir duquel les résultats seront disponibles. Ainsi, pour la plus grande base de nos expérimentations, les résultats ne pourront pas être disponibles avant un délai de 6 heures.
La commande Linux `hdparm` permet d'estimer le débit de lecture/écriture d'un disque. Nous avons appliqué cette commande au disque que nous utilisons pour ces expérimentations (monté sur la machine **Monit02**) et nous obtenons un débit de :

$$d_{disque} = 38,7 Mo/sec$$

On peut donc estimer le temps de chargement théorique total d'une base de signatures par :

$$T_{load} = \frac{26 \times N}{d_{disque}}$$

nom	N	N_{sig}	$Mem(N_{sig})$	$Heures(N_{sig})$	T_{load}	T_{load}/N_{sig}	n_{pages}
$REEL_1$	77 131	2 772	72 Ko	2,31 min	51,81 ms	0,018 ms	1
$REEL_3$	243 704	4 936	128 Ko	4,11 min	163,72 ms	0,033 ms	1
$REEL_{10}$	702 615	8 382	218 Ko	6,98 min	472,04 ms	0,056 ms	1
$REEL_{30}$	1 956 558	13 988	364 Ko	11,65 min	1,31 s	0,096 ms	1
$REEL_{100}$	5 320 367	23 065	600 Ko	19,22 min	3,57 s	0,154 ms	1
$REEL_{250}$	14 098 729	37 548	976 Ko	31,29 min	9,47 s	0,252 ms	1
$REEL_{500}$	23 258 368	48 226	1,25 Mo	40,18 min	15,62 s	0,323 ms	1
$REEL_{1000}$	48 345 280	69 530	1,81 Mo	57,94 min	32,48 s	0,467 ms	1
$REEL_{2500}$	126 562 273	112 499	2,92 Mo	1,56 heures	1,41 min	0,752 ms	4
$REEL_{5000}$	252 470 803	158 893	4,13 Mo	2,20 heures	2,83 min	1,068 ms	8
$REEL_{10000}$	518 320 618	227 666	5,92 Mo	3,16 heures	5,80 min	1,528 ms	16
$REEL_{18000}$	940 021 221	306 597	7,97 Mo	4,25 heures	10,52 min	2,058 ms	32
$REEL_{30000}$	1 543 902 419	392 925	10,21 Mo	5,45 heures	17,28 min	2,638 ms	64

TAB. 5.5 – Récapitulatif des informations relatives au mode de fonctionnement par requêtes groupées

Ce temps de chargement est donné dans la table 5.5 pour chacune des bases. Ce même temps divisé par le nombre de requêtes N_{sig} est également donné ; il représente l'estimation théorique de la partie du temps de recherche moyen d'une requête correspondant au chargement des pages

de la base. Enfin, la dernière colonne de la table 5.5 donne le nombre de pages de la base, n_{pages}.

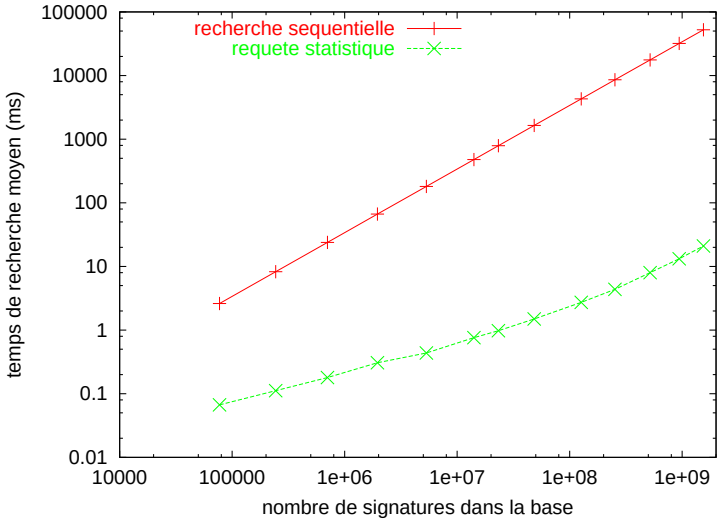


FIG. 5.11 – Temps de recherche moyen (sans le temps de chargement) en fonction du nombre de descripteurs dans la base N (coordonnées logarithmiques)

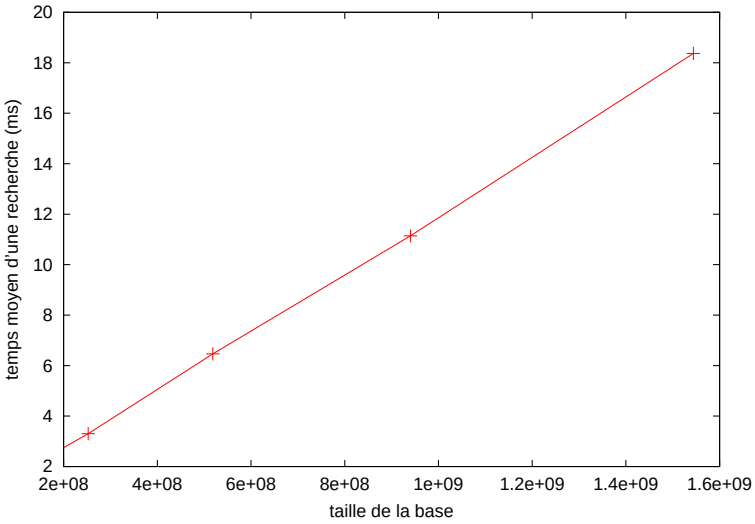


FIG. 5.12 – Temps de recherche moyen (sans le temps de chargement) en fonction du nombre de descripteurs dans la base N (coordonnées cartésiennes)

La figure 5.11 représente le temps de recherche moyen en mémoire en fonction de la taille de la base. Le temps de chargement n'est pas pris en compte dans cette figure. Sur cette même figure est représenté le temps de recherche moyen en mémoire d'un parcours séquentiel de la base par requêtes groupées. Il faut noter que pour toutes les bases pouvant être entièrement stockées en mémoire principale ($n_{pages} = 1$), c'est-à-dire jusqu'à la base $REEL_{1000}$ incluse (soit $N = 48,345.10^6$ signatures), le temps de recherche est le même qu'avec le mode de recherche par requête unique (étudié dans la section précédente 5.3.2). Le temps de recherche moyen est ainsi

sous-linéaire en fonction de la taille de la base et peut s'exprimer sous la forme :

$$T(N) = 0,217.10^{-3}\ N^{0,463}\ ms$$

Pour les bases de taille supérieures, on observe que l'évolution du temps de recherche moyen est de moins en moins sous-linéaire et qu'il tend vers une évolution linéaire, conformément à ce qui a été affirmé lors de la modélisation théorique de ce mode fonctionnement (cf. section 4.4.5). Il faut cependant distinguer les deux phénomènes contribuant à la diminution de la sous-linéarité :

- Le passage de la profondeur de la partition au deuxième ordre (lorsque $p_{min} > 20$). Celui-ci se traduit par un changement de pente de la courbe en coordonnées logarithmiques et le coût de recherche théorique en mémoire suit alors l'équation $T(N) = A_2\ N^{\gamma_2}$). Ce changement d'ordre de la profondeur intervient pour la base $REEL_{2500}$ ($N = 126,562.10^6$ signatures).
- L'ajout du coût linéaire en fonction du nombre de pages dû aux opérations qui sont répliquées pour chaque page (cf. équation 4.25, page 115). Ce dernier devient prépondérant pour les tailles de base les plus grandes, comme cela peut être observé sur la figure 5.12, montrant en cordonnées cartésiennes cette fois, la linéarité du temps de recherche moyen en fonction de la taille de la base pour les quatre bases les plus grandes. Le temps de recherche moyen en mémoire du mode de fonctionnement par requêtes groupées tend ainsi vers une évolution linéaire en fonction du nombre de signatures dans la base et on peut l'exprimer par :

  $$T(N) = 11,66.10^{-9}\ N\ ms$$

  Le gain par rapport à un parcours séquentiel devient ainsi constant et notre technique restera environ 3000 fois plus rapide qu'un parcours séquentiel pour les bases de tailles supérieures.

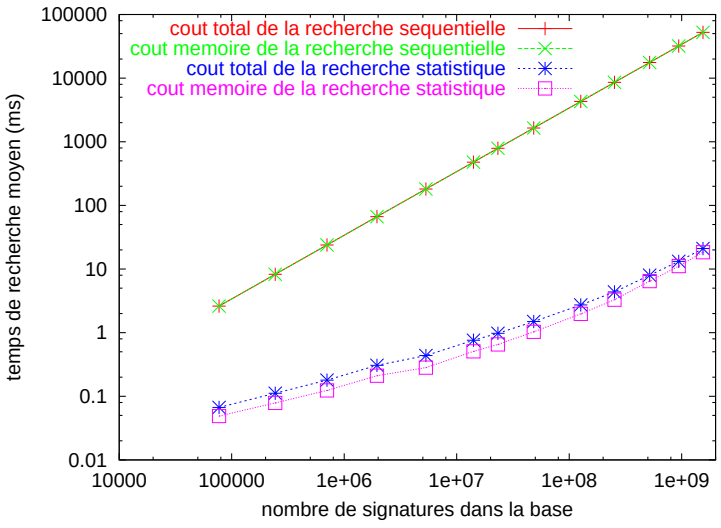


FIG. 5.13 – Temps de recherche moyen avec et sans le chargement de la base en fonction du nombre de descripteurs dans la base N

Nous nous intéressons maintenant au temps de recherche moyen total d'une recherche, prenant en compte le temps de recherche en mémoire et le temps de chargement des pages de la base en mémoire. Il est représenté sur la figure 5.13 en comparaison du temps de recherche en

mémoire uniquement. Ces deux courbes ont également été construites pour l'algorithme de recherche séquentielle.
On constate que l'influence du coût de chargement des pages reste limité par rapport au coût de la recherche des signatures en mémoire. Cette limitation est obtenue grâce à l'augmentation du nombre de requêtes groupées de manière proportionnelle à la racine carré du nombre de signatures dans la base. On peut vérifier par ailleurs que le surplus de coût dû aux chargements des pages en mémoire, est très proche de la valeur théorique estimée en fonction du débit disque (cf. table 5.5).

5.5 Evaluation du temps d'indexation des bases

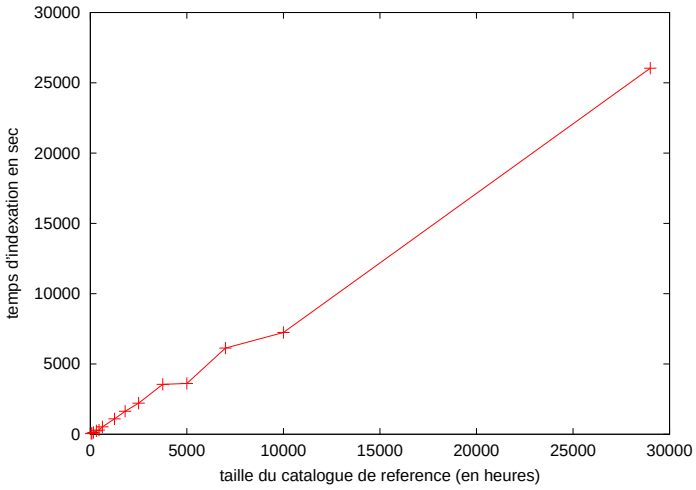


FIG. 5.14 – Temps d'indexation en fonction du nombre d'heures de vidéo contenues dans le catalogue de référence

L'objectif de cette section est d'évaluer l'influence de la taille de la base de signatures sur le temps d'indexation. Cette étape, rappelons-le, consiste principalement à trier les signatures en fonction de leur position sur la courbe de Hilbert et à construire la table d'index. Le temps d'indexation a été mesuré pour plusieurs bases de signatures réelles de taille croissante sur la machine **Monit02**. Le temps indiqué est le temps absolu retourné par la commande Linux `time`. La figure 5.14 et la table 5.6 donnent le temps d'indexation en fonction du nombre d'heures de vidéo contenues dans le catalogue de référence. On constate que l'évolution du temps d'indexation est linéaire en fonction de la taille de la base. La linéarité est due au fait que le coût de calcul des clés de Hilbert est prépondérant sur le coût du tri. Cette linéarité permet de conserver des coûts d'indexation tout à fait acceptables même dans le cas des bases très volumineuses puisque celui-ci ne dépasse pas 8 heures d'indexation. Une mise à jour hebdomadaire de la base est donc pleinement envisageable.

taille du catalogue (en heure)	temps d'indexation
40	46 sec
80	72 sec
160	2 min 5 sec
313	4 min 1 sec
450	4 min 57 sec
625	8 min 48 sec
1 250	18 min 15 sec
1 800	27 min 19 sec
2 500	36 min 52 sec
3 750	59 min 13 sec
5 000	1 h 0 min 14 sec
7 000	1 h 42 min 9 sec
10 000	2 h 0 min 38 sec
29 000	7 h 14 min 0 sec

TAB. 5.6 – Temps d'indexation en fonction du nombre d'heures de vidéo contenues dans le catalogue de référence

5.6 Synthèse

Dans ce chapitre, nous avons évalué notre méthode de recherche par requêtes statistiques, grâce à différentes expérimentations, à la fois sur des données synthétiques et des données réelles. Ces expérimentations ont tout d'abord valider les algorithmes de notre méthode en montrant qu'il y a bien adéquation entre l'espérance de la requête statistique et le pourcentage de signatures correctement retrouvées lorsque la loi de probabilité de la distorsion est parfaitement connue. Elles ont ensuite permis de créditer le modèle de coût théorique proposé, en particulier en ce qui concerne l'approximation de la profondeur optimale et la sous-linéarité du temps de recherche en fonction de la taille de la base (pour un fonctionnement par requête unique en mémoire). Cette sous-linéarité était un des objectifs principaux de nos travaux de recherche sur la détection de copies vidéo dans des bases de très grande dimension. Nous avons également démontré qu'au sein de notre système, une recherche par requête statistique était beaucoup plus rapide qu'une requête à un rayon près, et ce, pour une même espérance de la requête. Cela constitue en soi une validation de l'efficacité du principe de requêtes statistiques et il est tout à fait légitime de penser qu'une telle amélioration pourrait être étendue à d'autres structures d'indexation. Les temps de recherche de notre système semblent ainsi particulièrement faibles comparés à ce qui peut être lu dans la littérature. Il faudrait cependant réaliser une étude comparative sur les mêmes données et les mêmes machines pour pouvoir tirer des conclusions plus formelles. Nous avons enfin évalué le mode de fonctionnement par requêtes groupées qui permet de traiter des bases très importantes (jusqu'à 2 milliards de signatures) avec un temps de recherche moyen par signature très faible (quelques dizaines de millisecondes), et ce malgré l'évolution linéaire du temps de recherche moyen pour les bases de très grande taille. Nous n'avons trouvé, dans la littérature, aucune méthode qui annonce des temps aussi faibles avec des bases de signatures aussi grandes.

Troisieme partie

Expérimentations et Applications

Chapitre 6

Expérimentations

Ce chapitre présente les expérimentations effectuées pour évaluer notre méthode de détection de copies vidéo. Un soin particulier a été attaché à étudier les interactions entre la recherche des signatures similaires par requête statistique et l'efficacité globale de la détection de copies. La section 6.1 donne des précisions relatives aux conditions expérimentales. La section 6.2 évalue la capacité des requêtes statistiques à retrouver les signatures locales dans le cas de distorsions réelles (contrairement au chapitre précédent où les distorsions étaient simulées par un générateur gaussien). La section 6.3 présente une analyse de notre méthode de détections de copies par la construction des courbes opérationnelles du système. La section 6.4 évalue l'efficacité du système pour un large ensemble de transformations appliquées à des extraits du catalogue de référence et pour des vérités terrain construites à partir de séquences réellement diffusées. Enfin, la section 6.5 présente une analyse des bases de signatures locales afin de mettre en évidence les améliorations potentielles de la technique.

6.1 Conditions expérimentales

6.1.1 Modélisation de la distorsion

La modélisation de la distorsion choisie pour l'ensemble des expérimentations de ce chapitre est la loi gaussienne que nous avons déjà étudiée au cours des chapitres précédents. Toutes les composantes du vecteur de distorsion sont supposées indépendantes et ont une densité de probabilité égale à :

$$p_{\Delta S_j}(x) = f_{\mathcal{N}(0,\sigma_g)}(x) \quad \forall j \in [1, D]$$

6.1.2 Les bases de signatures

Les bases de signatures utilisées pour les expérimentations de ce chapitre sont les mêmes que celles utilisées lors de l'évaluation des performances du système de recherche par requêtes statistiques. Il s'agit de bases de signatures locales réelles construite à partir de la base SNC de l'INA (cf. section 5.1.4, page 124).

Paramètres

Les paramètres utilisés lors de l'étape de détection sont exactement les mêmes que ceux utilisés pour la construction des bases de référence :

- **Détection d'images clé** :
  Le paramètre de filtrage du signal d'activité de la vidéo est fixé à $\sigma_{activity} = 9,0$. Le taux moyen d'images clé par seconde est alors égal à $0,77$ images/sec.

- **Détecteur de Harris** :
  Les paramètres réglant l'échelle d'analyse du détecteur, avant l'étape de recentrage sont :

  $$\sigma_h = \sigma_{He} = \sigma_{Ha} = 1,4$$

  Le nombre de points maximum par image clé est fixé à 20.

- **Extraction des signatures** :
  Le paramètre de filtrage pour le calcul des dérivées est :

  $$\sigma_{sig} = 7,0$$

  La fenêtre utilisée pour calculer la signature autour d'un point d'intérêt a pour dimension 31×31 pixel.

Taux de compression

- Le nombre de signatures moyen par seconde est égal à $14,74$ **signatures/sec**.

- Le volume mémoire moyen occupé par les signatures et les données associées, est alors de 384 **octets/sec** soit $1,38$ **Moctets/h**.

- Le **taux de compression** par rapport au flux **MPEG1** initial (1 Mbit/sec) est de 325.

- Le **taux de compression** par rapport au flux vidéo non compressé utilisé pour l'extraction des signatures (352×288, 256 niveaux de gris) est de 6 600.

- Le **taux de compression** par rapport au flux vidéo initial non compressé (720×576, 3 couleurs) est de 81 000.

6.1.3 Les transformations étudiées

La plupart des expérimentations de ce chapitre sont basées sur la simulation d'un processus de post-production consistant à transformer des extraits vidéo appartenant à la base de référence. Seules les expérimentations de la section 6.4.4 sont réalisées sur des vérité-terrain incluant des vidéos candidates n'ayant pas été transformées par nos soins. La validation de notre méthode dans des conditions réelles sera également apportée lors du chapitre suivant présentant les résultats du système complet de monitorage d'une chaîne de télévision.
Nous simulerons ainsi cinq sortes de transformations que nous considérons comme représentatives des transformations les plus sévères **fréquemment** observées dans un contexte de diffusion télévisée d'archives. La suite de cette section s'attache à décrire et à illustrer ces transformations.

Décalage de l'image

La transformation consiste à décaler verticalement l'image d'un nombre de pixel égal à w_{shift} % la hauteur de l'image (cf. figure 6.1). Le reste de l'image est rempli par une région

uniforme d'un niveau de gris égal à 128. Le choix d'un décalage vertical n'est pas tout à fait arbitraire puisqu'il nous semble qu'il est beaucoup plus fréquent que le cas d'un décalage horizontal. Les effets sont cependant a priori les mêmes dans les deux cas. Une telle translation est observable dans de nombreux cas, comme par exemple l'ajout de bandes noires pour la diffusion d'un film, l'ajout d'un bandeau contenant des commentaires textuels, ou encore l'ajout d'un cadre qui personalise une émission particulière.
En allant plus loin, on pourrait considérer que cette transformation est représentative de toutes les transformations entraînant la suppression d'un certain pourcentage de l'image, comme les incrustations. Dans notre cas, la limitation du nombre de points d'intérêt dans une image clé engendrent cependant des effets différents selon le type d'information rajoutée puisque certains points de l'image d'origine peuvent être remplacés par de nouveaux points bien qu'ils soient encore présents dans l'image transformée. Dans le cas de la translation verticale, on verra ainsi apparaître certains points à l'intersection de la bande grise ajoutée et de l'image.

FIG. 6.1 – Translation verticale de l'image - de gauche à droite : w_{shift} = 0 % , 10 % , 20 % , 30 % , 40 %

Redimensionnement léger de l'image

FIG. 6.2 – Redimensionnement de l'image - de gauche à droite : w_{scale} = 0,75 ; 0,85 ; 1,0 ; 1,2 ; 1,4

La transformation consiste à redimensionner l'image d'un facteur w_{scale} par interpolation bilinéaire (cf. figure 6.2). Le centre de l'image reste fixe et, dans le cas d'une réduction de l'image, le reste de l'image est complété par une région uniforme de niveau de gris 128. Nous ne nous intéressons ici qu'à des redimensionnements relativement légers ($0,75 < w_{scale} < 1,4$) pour plusieurs raisons. Premièrement, notre système n'est par construction pas invariant aux redimensionnements forts (d'un facteur 2 ou 3 par exemple) et il n'est pas utile de tester des transformations aussi sévères. Il suffit de se fier aux courbes de répétabilité du détecteur de Harris fournies en annexe B pour en être convaincu. Deuxièmement, les redimensionnements légers sont beaucoup plus fréquents que les redimensionnements forts car ils sont liés à la *mise en page* des images à l'écran (cadres, bandes noires, etc.). Les redimensionnements forts, pour leur part, sont observés uniquement lorsque une version miniature d'un extrait est diffusé dans le fond de l'écran ou

qu'un zoom sur un détail particulier de l'extrait veut être mis en valeur. En général, la vidéo correspondante est diffusée en plein écran, soit juste avant, soit juste après et l'extrait sera donc détecté à ce moment là.

Modification du gamma de l'image

La transformation consiste à modifier le signal de l'image en niveau de gris par :

$$I'(x,y) = 255 \left[\frac{I(x,y)}{255} \right]^{w_{gamma}}$$

où $I(x,y)$ représente l'intensité du pixel (x,y) dans l'image d'origine et $I'(x,y)$ l'intensité du pixel (x,y) dans l'image transformée. Ce type de transformation est très fréquent lors des étapes d'étalonnage colorimétrique d'un processus de post-production.

FIG. 6.3 – Modification du gamma de l'image - de gauche à droite : $w_{gamma} = 0,4\ ,\ 0,7\ ,\ 1,0\ ,\ 1,4\ ,\ 2,5$

Modification du contraste de l'image

La transformation consiste à modifier le signal de l'image en niveau de gris par :

$$I'(x,y) = \begin{cases} w_{contrast} I(x,y) & si \ \ w_{contrast} I(x,y) \leq 255 \\ 255 & si \ \ w_{contrast} I(x,y) > 255 \end{cases}$$

où $I(x,y)$ représente l'intensité du pixel (x,y) dans l'image d'origine et $I'(x,y)$ l'intensité du pixel (x,y) dans l'image transformée. Ce type de transformation est également très fréquent en post-production.

FIG. 6.4 – Modification du contraste de l'image - de gauche à droite : $w_{contrast} = 0,4\ ,\ 0,7\ ,\ 1,0\ ,\ 1,4\ ,\ 2,5$

Ajout de bruit gaussien

La transformation consiste à ajouter, indépendamment pour chaque pixel de l'image, un bruit gaussien d'écart-type w_{noise}. Cette transformation est censée simuler les divers bruits liés à la diffusion des images et aux conversions entre supports analogiques et numériques. N'ayant pas fait d'étude sur la modélisation de ces bruits, nous avons choisi la loi normale.

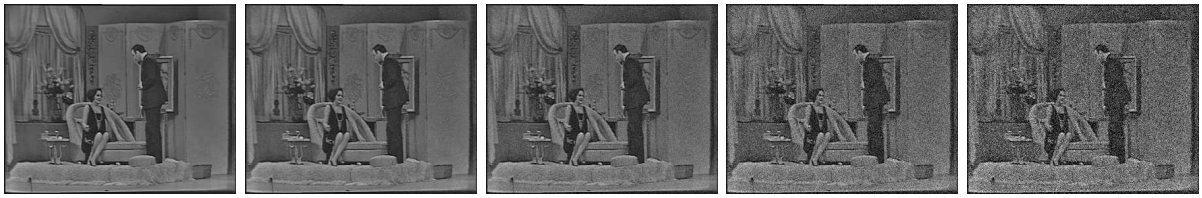


FIG. 6.5 – Ajout de bruit gaussien - de gauche à droite : $w_{noise} = 0,0\ ,\ 5,0\ ,\ 10,0\ ,\ 20,0\ ,\ 30,0$

6.1.4 Machine

Les expérimentations de ce chapitre ont été réalisées sur la machine **Monit02** ou sur des machines équivalentes (cf. section 5.1.2).

6.2 Evaluation des requêtes statistiques sur des distorsions réelles

L'objectif de cette section est de montrer la capacité des requêtes statistiques à retrouver correctement les signatures dans le cas de distorsions réelles et d'évaluer la pertinence du paramètre α (espérance de la requête statistique) pour le contrôle de la précision effective de la recherche. Le but de cette étude étant d'étudier uniquement la distorsion d'une signature locale suite à une transformation de l'image et non la répétabilité des détecteurs d'images clé et de points d'intérêt, nous avons simulé un détecteur parfait. Le principe des expérimentations est de détecter les images clé et les points d'intérêt dans un extrait vidéo de référence, d'appliquer certaines transformations à cet extrait et de recalculer les signatures dans l'extrait transformé aux positions théoriques où devraient se situer les points d'intérêt dans le cas d'un détecteur parfait. Dans le cas des transformations non géométriques, la position des points d'intérêt est en fait la même dans l'extrait de référence et dans l'extrait transformé. Dans le cas d'un redimensionnement de l'image, les positions doivent être recalculées en fonction du paramètre w_{scale} de la transformation.
La transformation consistant à décaler verticalement l'image ne nous intéresse pas ici puisque, dans le cas d'un détecteur parfait, soit les signatures seront exactement les mêmes, soit elles auront complètement disparu de l'image. En revanche pour simuler l'imprécision du détecteur de points d'intérêt, nous étudierons l'influence d'un décalage d'un ou deux pixel sur le calcul de la signature locale.

Toutes les expérimentations de cette section sont réalisées avec la base de signatures $REEL_{250}$ (environ 250 heures de vidéo, 14 millions de signatures, cf. table 5.1). Un extrait vidéo dont les signatures appartiennent à cette base, a été sélectionné aléatoirement. Sa durée est de 19 min 34 sec, et le nombre de signatures dans cet extrait est égal à 17584. Le principe général des expérimentations est de transformer cet extrait, de calculer les signatures dans l'extrait transformé sous l'hypothèse d'un détecteur de points d'intérêt parfait et de rechercher ces signatures

dans la base via notre méthode de recherche par requête statistique. Une signature est considérée comme correctement retrouvée si la signature originale appartient aux résultats de la requête.

6.2.1 Evaluation du modèle de distorsion

L'objectif des expérimentations de cette section est d'évaluer la pertinence de la loi gaussienne utilisée pour modéliser la distorsion des signatures. La valeur du paramètre σ_g de la loi de probabilité de la distorsion peut être estimée expérimentalement, pour une transformation donnée, par le protocole expérimental simulant un détecteur de points d'intérêt parfait. Il suffit pour cela d'estimer l'écart-type de $\Delta S_j \ \ \forall j \in [1, D]$, après transformation de l'image. L'estimation de σ_g sera notée $\overline{\sigma_g}$.

Pour la première expérimentation, nous avons transformé l'extrait de référence par un redimensionnement de facteur $w_{scale} = 0,85$. Les signatures de l'extrait transformé ont ensuite été recherchées dans la base pour différentes valeurs de l'espérance de la requête statistique α. Le pourcentage (ou taux) de signatures correctement retrouvées est noté r_g. La table 6.1 récapitule les résultats de l'expérimentation. La figure 6.6 montre l'évolution de r_g en fonction de α.

α (%)	40	50	60	70	80	90	95
r_g (%)	46,41	56,40	64,59	71,77	78,58	86,78	90,30

TAB. 6.1 – Taux de signatures réelles retrouvées en fonction de α

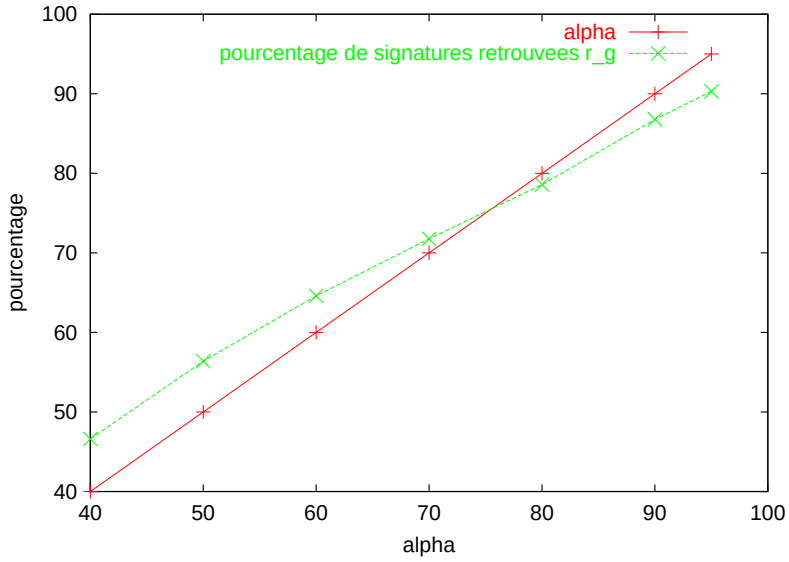


FIG. 6.6 – Taux de signatures réelles retrouvées en fonction de α

Les résultats montrent qu'il existe un biais entre l'espérance de la requête statistique et le pourcentage de signatures retrouvées. Ce biais est positif pour les valeurs de α inférieures à 75 % et négatif pour les valeurs de α supérieures à 75 %. La valeur absolue du biais varie ainsi de 0 % pour $\alpha = 75$ % à 6,41 % pour $\alpha = 40$ %. Le biais moyen (en valeur absolue), est de 4,07 %.

L'expérimentation montre cependant que, malgré l'hypothèse forte d'indépendance des composantes de la distorsion, le paramètre α offre un contrôle acceptable de la qualité de la recherche bien que celui-ci ne soit pas très précis. La précision n'est en effet pas d'une importance capitale : rappelons d'abord que la qualité de la recherche dans la base n'influence pas directement la qualité finale des résultats de la détection de copies puisque les résultats sont ensuite post-traités par les algorithmes d'estimation et de vote. Rappelons ensuite que la précision ne peut être contrôlée que pour une transformation donnée puisqu'il n'est pas possible de modéliser la fréquence relative des différentes transformations. Le modèle est donc estimé uniquement pour la transformation considérée comme la plus sévère. Pour les autres, il faut que le critère de sévérité garantisse que l'espérance du taux de signatures correctement retrouvées sera supérieure à celui de la transformation de référence.

La figure 6.7 représente le temps moyen d'une recherche en fonction du taux de signatures correctement retrouvées. Elle montre que le temps de recherche peut être sérieusement réduit si l'on se permet de réduire la qualité de la recherche. Le temps de recherche peut ainsi être divisé par 5 en passant de 90 % de signatures correctement retrouvées à 70 % de signatures correctement retrouvées.

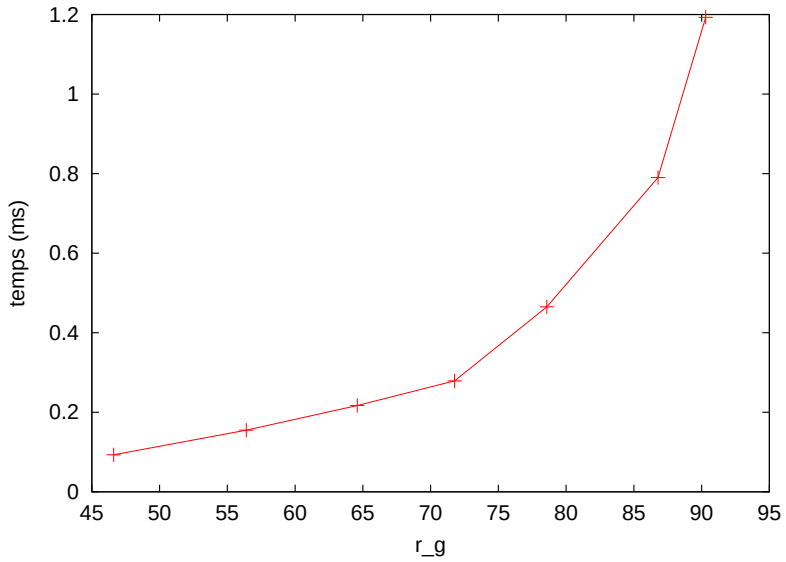


FIG. 6.7 – Temps moyen d'une recherche en fonction du taux de signatures retrouvées

Pour vérifier le fait que le modèle gaussien offre un contrôle acceptable de la qualité de la recherche dans le cas d'autres transformations de référence qu'un redimensionnement, nous avons renouvelé l'expérience pour d'autres transformations. Une courbe similaire à celle obtenue dans le cas d'un redimensionnement a été obtenue pour toutes les transformations testées. La table 6.2 récapitule les résultats de ces expérimentations pour $\alpha = 90$ %. La transformation notée δ_{pix}, correspond à un décalage de deux pixel simulant une imprécision du détecteur de point d'intérêt. La transformation notée *multi* correspond à une combinaison de transformations avec pour paramètres w_{scale}=0,9, w_{gamma}=1,5, w_{noise}=10,0, $w_{contrast} = 1,5$ et δ_{pix}=1 pixel. Pour chaque

transformation, le paramètre σ_g du modèle de distorsion a été choisi égal à :

$$\sigma_g = \overline{\sigma_g}$$

On constate que le bais entre le taux de signatures correctement retrouvées et l'espérance de la requête statistique ($\alpha = 90$ %) reste du même ordre de grandeur pour les différentes transformations. Il est toujours négatif et varie, en valeur absolue, entre $1,76$ % et $6,16$ %.

transformation	w_{scale}=0,8	w_{gamma}=2,0	w_{noise}=20,0	δ_{pix}=2 pixel	$w_{contrast} = 2,0$	multi
$\overline{\sigma_g}$	21,14	7,65	9,48	16,10	15,76	23,02
r_g (%)	86,11	88,24	87,65	83,84	87,28	86,09

TAB. 6.2 – Taux de signatures retrouvées pour différentes transformations ($\alpha = 0,90$, $\sigma_g = \overline{\sigma_g}$)

6.2.2 Paramétrage de σ_g

Nous avons montré dans la section précédente, que pour une transformation donnée, les requêtes statistiques basées sur un modèle gaussien permettaient de retrouver efficacement les signatures après transformation de l'image avec un contrôle acceptable de la qualité de la recherche. L'objectif des expérimentations de cette section est de discuter du choix d'une transformation de référence pour le calibrage du paramètre σ_g du modèle de distorsion. Nous nous proposons ainsi de vérifier la pertinence de l'écart-type $\overline{\sigma_g}$, comme critère de sévérité des transformations. $\overline{\sigma_g}$ a été estimé pour différents paramètres des transformations étudiées dans ce chapitre grâce au protocole expérimental décrit précédemment (simulation d'un détecteur de points d'intérêt parfait). Les résultats sont présentés dans les tables 6.3 et 6.4. Les résultats présentés dans la table 6.4 correspondent aux mêmes transformations que ceux de la table 6.3, mais un décalage systématique de 1 pixel a été réalisé avant le calcul de la signature dans l'image transformée, afin de simuler une imprécision du détecteur de points d'intérêt. Ces tables permettent de calibrer la valeur de σ_g en fonction des transformations auxquelles l'on souhaite être invariant. Il s'agit là d'un compromis entre la robustesse du système et les performances du système puisque la valeur de σ_g influence fortement le coût d'une requête statistique.

w_{scale}	0,7	0,77	0,84	0,91	0,98	1,05	1,12	1,19	1,26	1,33	1,4
$\overline{\sigma_g}$	33,74	26,20	18,85	12,22	7,88	8,15	12,20	16,82	21,09	24,81	28,40
w_{gamma}	0,4	0,61	0,82	1,03	1,24	1,45	1,66	1,87	2,08	2,29	2,50
$\overline{\sigma_g}$	5,55	3,55	1,70	0,46	2,51	4,03	5,72	7,16	8,92	10,26	11,48
$w_{contrast}$	0,4	0,61	0,82	1,03	1,24	1,45	1,66	1,87	2,08	2,29	2,50
$\overline{\sigma_g}$	2,05	1,60	1,39	0,58	3,24	6,15	9,02	12,18	14,67	16,58	18,0
w_{noise}	5,0	10,0	15,0	20,0	25,0	30,0	35,0	40,0			
$\overline{\sigma_g}$	4,62	6,60	8,10	9,70	11,13	12,54	13,95	15,08			

TAB. 6.3 – Sévérité des transformations étudiées ($\overline{\sigma_g}$) sans décalage ($\delta_{pix} = 0$)

Grâce à ces tables, nous nous proposons de valider la pertinence de $\overline{\sigma_g}$ comme critère de sévérité des transformations. L'objectif est de vérifier que, pour des transformations moins sévères

w_{scale}	0,7	0,77	**0,84**	0,91	0,98	1,05	1,12	1,19	1,26	1,33	1,4
$\overline{\sigma_g}$	36,76	29,86	**23,43**	18,13	14,91	13,92	15,30	18,50	21,94	25,22	28,58
w_{gamma}	0,4	0,61	0,82	1,03	1,24	1,45	1,66	1,87	2,08	2,29	2,50
$\overline{\sigma_g}$	10,80	9,82	9,23	8,96	9,19	9,60	10,31	11,05	12,17	13,10	13,98
$w_{contrast}$	0,4	0,61	0,82	1,03	1,24	1,45	1,66	1,87	2,08	2,29	2,50
$\overline{\sigma_g}$	9,26	9,15	9,13	8,99	9,64	10,97	12,74	15,17	17,26	18,92	20,18
w_{noise}	5,0	10,0	15,0	20,0	25,0	30,0	35,0	40,0			
$\overline{\sigma_g}$	10,28	11,35	12,25	13,35	14,42	15,52	16,64	17,58			

TAB. 6.4 – Sévérité des transformations étudiées ($\overline{\sigma_g}$) avec un décalage de 1 pixel ($\delta_{pix} = 1$)

($\overline{\sigma_g}$ plus faible) qu'une certaine transformation de référence, le pourcentage de signatures correctement retrouvées est supérieur.

Nous avons choisi comme transformation de référence, supposée être la plus sévère, un redimensionnement de paramètre $w_{scale} = 0,84$ suivi d'un décalage de 1 pixel pour simuler une imprécision du détecteur de points d'intérêt. La valeur correspondante de $\overline{\sigma_g}$ est égale à $23,43$ (en gras dans la table 6.4) et c'est donc cette valeur que nous retenons pour le calibrage du paramètre du modèle de distorsion :

$$\sigma_g = 23,43$$

La valeur de l'espérance de la requête statistique est fixée à :

$$\alpha = 85\ \%$$

Ces deux paramètres de la requête statistique étant fixés, nous avons ensuite recherché les signatures de l'extrait vidéo pour plusieurs transformations de sévérité décroissante et inférieure à $23,43$. Les résultats sont présentés dans la table 6.5.

transformation	$\overline{\sigma_g}$	r_g
$w_{scale} = 0,84,\ \delta_{pix} = 1$	**23,43**	**80,74**
$w_{scale} = 1,26,\ \delta_{pix} = 1$	21,95	81,69
$w_{scale} = 0,91,\ \delta_{pix} = 1$	18,13	92,49
$w_{scale} = 0,98,\ \delta_{pix} = 1$	14,92	97,30
$w_{gamma} = 2,08,\ \delta_{pix} = 1$	12,17	98,52
$w_{gamma} = 0,82,\ \delta_{pix} = 1$	9,23	99,79
$w_{noise} = 10,0,\ \delta_{pix} = 0$	6,60	99,65

TAB. 6.5 – Taux de signatures retrouvées pour des transformations de sévérité décroissante - $\alpha = 85\ \%$ - $\sigma_g = 23,43$

Le taux de signatures correctement retrouvées dans le cas de la transformation de référence (en gras dans la table), est égale à $80,74\ \%$ et on vérifie bien que le taux de signatures correctement retrouvées pour les transformations moins sévères ($\overline{\sigma_g}$ plus faible) est toujours supérieur à cette valeur. On constate également que moins la transformation est sévère et plus le pourcentage de signatures correctement retrouvées est important, à l'exception de la transformations la moins sévère ($w_{noise} = 10$, $\delta_{pix} = 0$), pour laquelle le taux de signatures est légèrement plus faible que

la deuxième transformation la moins sévère ($w_{gamma} = 0,82$, $\delta_{pix} = 1$).

En pratique, le choix de la transformation de référence la plus sévère résulte a priori d'un compromis entre temps de recherche et robustesse du système. Il faut toutefois remarquer que dans le cas du système complet de détection de copies vidéo à l'aide de signatures locales, il est inutile d'augmenter la sévérité tolérée par le système de recherche dans la base pour des transformations trop sévères. En effet, lorsque la répétabilité du détecteur de points d'intérêt est trop médiocre, et que la majorité des points détectés dans l'image transformée ne sont plus du tout les mêmes que dans l'image de référence, les signatures de référence ne seront pas retrouvées quoi qu'il arrive. Il en est de même de l'étape préalable de détection d'images clé qui diminue encore plus la répétabilité globale de la détection des points d'intérêt. La connaissance de la répétabilité du détecteur aux différentes transformations est donc très utile puisqu'elle permet de borner l'ensemble des transformations auxquelles il est utile de se rendre robuste lors de la recherche dans la base. Le détecteur de Harris utilisé dans notre méthode n'étant théoriquement pas invariant aux changements d'échelle, il est par exemple inutile de considérer des redimensionnements trop sévères pour l'étape de recherche des signatures dans la base (typiquement des redimensionnements d'un facteur supérieur à $1,5$ ou inférieur à $0,7$).

6.3 Analyse des courbes opérationnelles du système

L'objectif de cette section est d'analyser expérimentalement notre méthode complète de détection de copies vidéo. Les paramètres de l'extraction des signatures locales dans les vidéos candidates (détecteur d'images clé, détecteur de points d'intérêt, descripteurs) sont exactement les mêmes que ceux utilisés pour la construction des bases (cf. section 6.1.2). La recherche des signatures similaires dans la base se fait par notre méthode de requêtes statistiques. En ce qui concerne la dernière étape de notre méthode, consistant à fusionner les résultats de la recherche d'un ensemble de signatures locales cumulées dans le temps, des précisions sont données dans la section 6.3.1.

Dans le section 6.3.2, nous indiquons le mode opératoire de construction des courbes opérationnelles présentées par la suite et nous précisons notamment le mode opératoire pour déterminer la fréquence de fausses alarmes du système. La section 6.3.4 analyse l'influence de l'espérance de la requête α sur la détection. La section 6.3.5 analyse l'influence du nombre de signatures dans la base sur la détection. La section 6.3.5 analyse l'influence du paramètre σ_g du modèle de distorsion sur la détection.

6.3.1 Algorithme de décision globale

Dans la section 2.4 de la première partie de ce mémoire, nous avons proposé une méthode pour prendre une décision globale à partir des résultats de la recherche d'un ensemble de signatures locales cumulées dans le temps. L'utilisation de cette méthode dans le cas de la détection de copies d'archives vidéo a ensuite été détaillée dans la section 2.4.5 (page 49) et c'est dans ce cadre qu'ont été réalisées nos expérimentations.
Nous rappelons qu'une décision est prise pour chaque image clé du flux candidat en utilisant les résultats de la recherche de toutes les signatures locales contenues dans un *buffer* temporel centré autour de l'image clé courante. La taille de ce *buffer* est de 9 images clé. Le segment vidéo correspondant à ce buffer sera appelé **un segment clé**. La durée d'un segment clé est variable et il y a recouvrement entre deux segments clé successifs. Chaque segment clé est caractérisé par le

code temporel de son image clé centrale. Sachant que le taux moyen d'images clé par seconde est égal à 0,77 images/sec. (cf. section 6.1.2), la durée moyenne d'un segment clé $\overline{\Delta T}_{cle}$, est égale à :

$$\overline{\Delta T}_{cle} = \frac{9}{0,77} = 11,7 secondes$$

Nous rappelons également que seuls l'identifiant et le code temporel des signatures ayant été conservés comme données complémentaires des signatures, le modèle de transformation utilisé pour le recalage ne tient compte que de la coordonnée temporelle et pas des positions spatiales des points d'intérêt. L'estimation est donc faite uniquement sur le décalage temporel b_t existant entre les signatures candidates et les signatures de référence. Une solution est déterminée pour chacun des identifiants représentés dans les résulats d'un segment clé, et une mesure de similarité, notée n_{sol}, est calculée pour chacune de ces solutions.
Cette mesure de similarité consiste à compter le nombre de points d'intérêt du segment clé qui contribue effectivement à la solution estimée. La décision de savoir si les solutions ainsi estimées sont ou ne sont pas des copies du segment clé candidat, est prise par un seuillage de la mesure de similarité ; c'est-à-dire un seuillage sur le nombre de points d'intérêt qui contribuent effectivement à la solution estimée (n_{sol}). C'est en faisant varier la valeur de ce seuil que l'on obtient les courbes opérationnelles du système présentées dans la suite de cette section.
Il est important de noter qu'il peut y avoir plusieurs détections par segment clé candidat, ce qui correspond théoriquement à la présence de duplicata dans le catalogue des vidéos de référence.
Le nombre de points d'intérêt par image clé étant limité à 20, la valeur maximale de n_{sol} peut être bornée par :

$$0 \leq n_{sol} \leq 180$$

Pour des raisons algorithmiques et pour augmenter les performances, certaines barrières logicielles ont été introduites dans l'étape d'estimation des meilleures solutions :
Le nombre de signatures similaires retrouvées dans la base pour une signature candidate, est ainsi limité à 500 signatures (seules les 500 plus proches signatures sont conservées lorsqu'il y en a plus).
Le nombre d'identifiants pour lesquels le recalage du segment clé candidat est estimé, est limité à 100 (seuls les 100 identifiants les plus présents sont conservés). Dans le cas où il existe un grand nombre de duplicata d'un même extrait dans la base, on ne pourra ainsi pas en détecter plus de 100 pour un même extrait candidat.

6.3.2 Construction des courbes opérationnelles

L'approche classique pour construire la courbe opérationnelle d'un système consiste à mesurer le rappel et la précision (courbe *ROC*). Le rappel est défini comme la proportion des résultats pertinents retournés par le système par rapport à l'ensemble des résultats pertinents contenus dans la base. La précision est définie comme la proportion de résultats pertinents retournés par le système par rapport à l'ensemble des résultats retournés par le système.

L'estimation du rappel réel de notre système est cependant rendu difficile en raison de la grande taille des catalogues de référence. L'ensemble des résultats pertinents contenus dans la base, pour un document candidat donné, n'est en effet pas totalement connaissable car il est impossible de contrôler tous les extraits du catalogue de référence. On pourra vérifier que l'on retrouve bien des résultats connus à l'avance (extraits de référence transformés par exemple), mais on ne pourra

pas savoir si l'on en rate pas d'autres. Nous utiliserons donc uniquement le **taux de reconnaissance**, défini comme le pourcentage de résultats attendus correctement retrouvés par le système et noté t_{rn}. Un extrait est considéré comme correctement retrouvé si la solution estimée est la bonne et que la valeur de la mesure de similarité, n_{sol}, est supérieure au seuil de décision.

La précision, quant à elle, n'est pas vraiment pertinente pour notre système, pour deux raisons. Premièrement, lorsque l'on monitore une chaîne de télévision, la majorité des recherches ne retournent aucun résultat puisque la majorité des séquences diffusées ne sont pas des copies d'un extrait de la base. Ce qui nous intéresse est donc plutôt la fréquence des fausses alarmes et non le pourcentage des résultats retournés qui sont effectivement des bonnes détections.
Deuxièmement, la précision du système dépend de la fréquence des bonnes détections. Or, cette fréquence est d'une part très variable et d'autre part difficilement connaissable. Considérons, par exemple, qu'il y ait en moyenne dix fausses alarmes par jour. Si on utilise un catalogue de référence contenant un grand nombre de parallèles antenne (cf. section 5.1.4), celui-ci contiendra également des publicités, des habillages de chaîne, etc. et nous avons constaté que dans ce cas le nombre de résultats corrects pouvaient atteindre plus de 1 000 par jour. La précision du système sera alors très bonne ($\frac{1\,000}{1\,000+10} = 99$ %). Si on utilise, en revanche, un catalogue ne contenant que des archives anciennes, le nombre de résultats corrects peut être très faible, par exemple 1 par jour. Dans ce cas, la précision du système sera relativement mauvaise ($\frac{1}{11} = 9$ %), alors qu'il s'agit bien du même système.
Plutôt que la précision du système, nous utiliserons donc la **fréquence de fausses alarmes** définie comme le pourcentage de segments clé conduisant à au moins une fausse alarme. Elle sera noté t_{fa}.
Sachant que le nombre d'images clé est de $0,7$ par seconde, le nombre moyen de fausses détections par heure de vidéo, n_{fa} peut être obtenu par :

$$n_{fa} = 3600 \times 0,7 \times t_{fa} \tag{6.1}$$

Une fréquence de fausses alarmes $t_{fa} = 0,001$ conduit ainsi à une moyenne de $2,52$ fausses détections par heure soit environ 60 par jour.

Les courbes représentant le taux de reconnaissance t_{rn} en fonction de la fréquence de fausses alarmes t_{fa} seront appelées simplement **courbes opérationnelles**.

6.3.3 Estimation de la fréquence de fausses alarmes

L'objectif des expérimentations de cette section est d'estimer la fréquence de fausses alarmes t_{fa} de notre système en fonction du seuil de décision sur la mesure de similarité n_{sol}. La fréquence de fausses alarmes est définie par le pourcentage de segments clé conduisant à au moins une solution non pertinente, dont la mesure de similarité est au dessus du seuil de décision.

Pour estimer des faibles fréquences de fausses alarmes il est nécessaire de traiter suffisamment de requêtes. Pour connaître la valeur du seuil pour laquelle on a en moyenne moins d'une fausse alarme par heure, il est par exemple nécessaire de rechercher plusieurs heures de vidéo.
Or, le fait d'avoir un grand nombre de requêtes, pose un problème pour la vérification de la (non-)pertinence des résultats. Lorsque le seuil sur la mesure de similarité est bas, il n'est en effet pas envisageable de contrôler manuellement tous les résultats retournés par le système.
Pour automatiser entièrement l'estimation de la fréquence des fausses alarmes, nous avons recherché un flux vidéo nous garantissant le plus possible qu'aucune des séquences diffusées ne

puissent être une copie d'un extrait du catalogue de référence. Notre base étant constituée majoritairement d'archives de la télévision française, nous avons opté pour une chaîne étrangère. Il nous a fallu cependant éviter les chaînes diffusant des informations internationales, ou encore les chaînes qui sont susceptibles de diffuser des publicités identiques à celles que l'on trouve en France. Pour minimiser tous ces risques, nous avons finalement capturé 24 heures d'une chaîne de télévision tunisienne (TV7 Tunis), qui diffuse beaucoup de feuilletons et de clips tunisiens.
Toutes les solutions retournées par le système, lors du monitorage de ce flux vidéo, sont alors considérées comme des fausses alarmes. Nous avons tout de même vérifié, pour chaque expérimentation, que les 30 résultats ayant la plus forte mesure de similarité était bien des fausses alarmes.

La courbe de la fréquence des fausses alarmes en fonction du seuil de la mesure de similarité a ainsi été calculée pour différentes valeurs de l'espérance de la requête statistique α, différentes tailles de base et différentes valeurs du paramètre du modèle de distorsion σ_g. Les résultats sont détaillés dans la suite de cette section. Ces courbes seront ensuite utilisées pour la construction des courbes opérationnelles présentées dans la suite de ce chapitre.

Fréquence de fausses alarmes pour plusieurs tailles de base

Quatre bases de tailles croissantes ont été utilisées. Il s'agit des bases $REEL_{10000}$, $REEL_{2500}$, $REEL_{500}$ et $REEL_{100}$ (cf. section 5.1.4, page 124).
Les paramètres de la requête statistique sont :

$$\sigma_g = 18 \quad \alpha = 80\ \%$$

La figure 6.8 représente la fréquence de fausses alarmes en fonction du seuil de la mesure de similarité pour les quatre bases. La figure de gauche représente les courbes entières, tandis que la figure de droite représente un zoom de ces mêmes courbes sur les faibles valeurs de la fréquence de fausses alarmes.

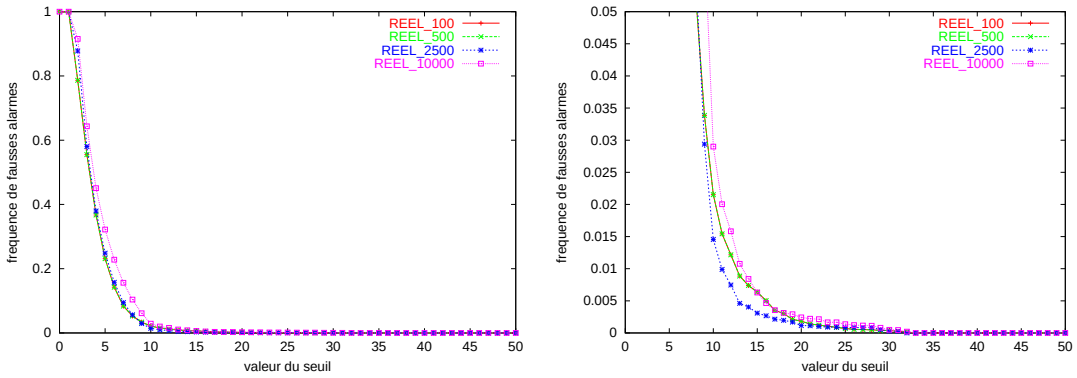


FIG. 6.8 – Fréquence de fausses alarmes en fonction du seuil de la mesure de similarité pour différentes tailles de base

Fréquence de fausses alarmes pour plusieurs valeurs de l'espérance de la requête statistique α

La base utilisée est $REEL_{500}$ (cf. section 5.1.4, page 124).
Les paramètres de la requête statistique sont :

$$\sigma_g = 18$$

$$\alpha \in \{\ 50\ \%\ ,\ 70\ \%\ ,\ 80\ \%\ ,\ 90\ \%\ ,\ 95\ \%\ \}$$

La figure 6.9 représente la fréquence de fausses alarmes en fonction du seuil de la mesure de similarité pour les différentes valeurs de α. La figure de gauche représente les courbes entières, tandis que la figure de droite représente un zoom de ces mêmes courbes sur les faibles valeurs du taux de fausses alarmes.

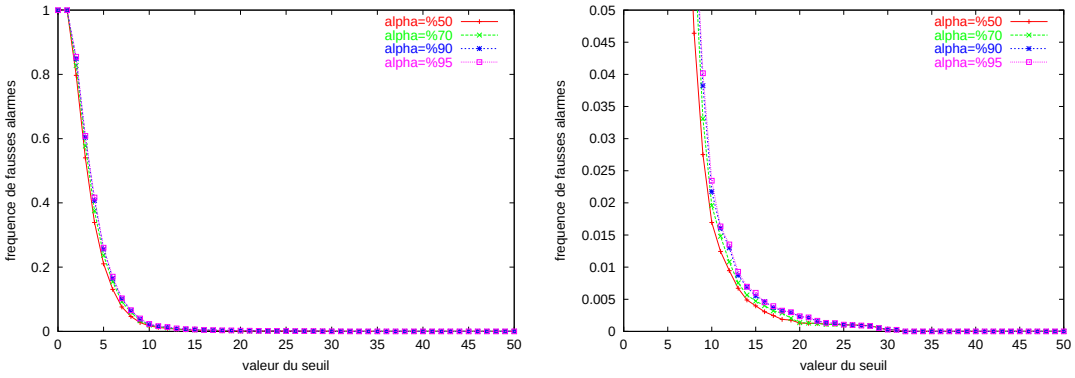


FIG. 6.9 – Fréquence de fausses alarmes en fonction du seuil de la mesure de similarité pour différentes valeurs de α

Fréquence de fausses alarmes pour plusieurs valeurs du paramètre du modèle de distorsion σ_g

La base utilisée est $REEL_{500}$ (cf. section 5.1.4, page 124).
Les paramètres de la requête statistique sont :

$$\sigma_g = \in \{\ 10\ ,\ 15\ ,\ 18\ ,\ 20\ ,\ 25\ \}$$

$$\alpha = 80\ \%$$

La figure 6.10 représente la fréquence de fausses alarmes en fonction du seuil de la mesure de similarité pour les différentes valeurs de σ. La figure de gauche représente les courbes entières, tandis que la figure de droite représente un zoom de ces mêmes courbes sur les faibles valeurs du taux de fausses alarmes.

Analyse

On constate que l'évolution de la fréquence des fausses alarmes en fonction du seuil de la mesure de similarité dépend peu des paramètres de la requête statistique et de la taille de la base

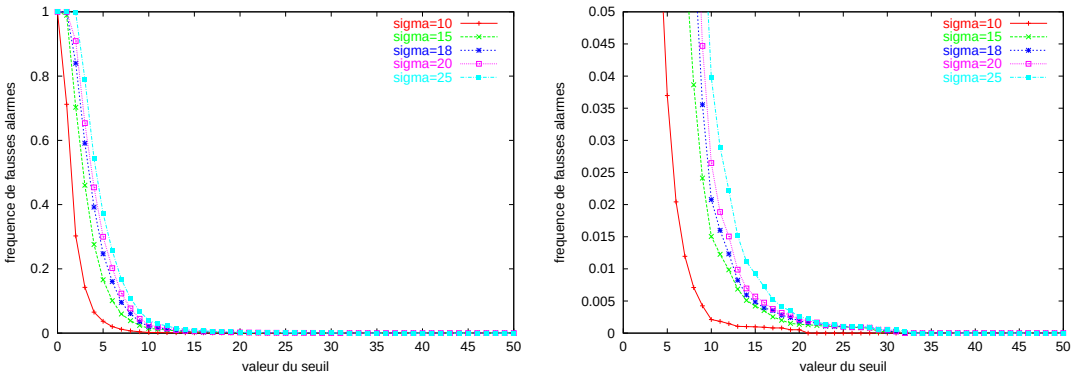


FIG. 6.10 – Fréquence de fausses alarmes en fonction du seuil de la mesure de similarité pour différentes valeurs de σ_g

de signatures. Seule la valeur la plus faible de σ_g conduit à des fréquences de fausses alarmes vraiment plus faibles. Ceci peut être expliqué par la nature même de l'estimation des solutions et de la mesure de similarité. L'identification d'un extrait est en effet uniquement basée sur la cohérence temporelle des signatures candidates et des signatures de référence. La recherche dans la base ne constitue donc qu'une étape de filtrage et l'augmentation même forte du nombre de signatures similaires retrouvées pour chaque signature candidate n'influence que faiblement la probabilité de retrouver les signatures d'un même extrait pour un nombre important de points d'intérêt.

Les fausses alarmes sont généralement dues à la redondance des signatures au sein d'un même extrait qui contribue à corréler les signatures avec leur position temporelle. Pour s'en convaincre, on peut s'intéresser à la fausse alarme ayant la mesure de similarité la plus élevée parmi les 24 heures de capture de TV7 Tunis. Celle-ci possède une mesure de similarité égale à 31, ce qui signifie que 31 points d'intérêt ont été appariés parmi tous les points d'intérêt contenus dans le segment clé (contenant au maximum 180 points d'intérêt). Cette fausse alarme est illustrée par la figure 6.11, où nous avons dessiné la position des points d'intérêt détectés sur une des images clé afin de mieux comprendre l'origine de la fausse alarme.

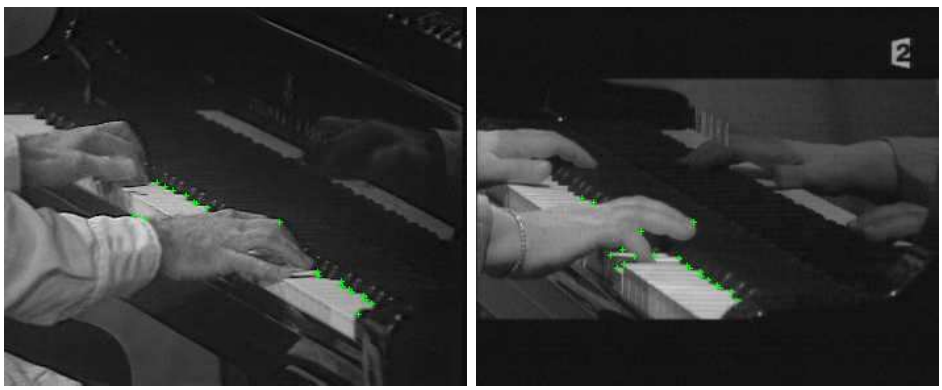


FIG. 6.11 – Points d'intérêt d'une fausse alarme - à gauche : image clé du segment candidat - à droite : image clé du segment retrouvé - $n_{sol} = 31$

Cette fausse alarme est principalement due au fait que beaucoup de points d'intérêt au sein du même extrait possèdent la même signature. Toutes les signatures des points d'intérêt situés sur les touches du piano de la séquence candidate sont ainsi similaires à toutes les signatures des points d'intérêt situés sur les touches du piano de la séquence de référence ce qui augmente fortement la probabilité qu'il y ait une solution respectant la cohérence temporelle. Cette constatation peut être faite pour la plupart des fausses alarmes ayant une forte mesure de similarité. On retrouvera ainsi fréquemment parmi les objets présents dans les fausses alarmes, des pianos, des thermomètres, des murs de brique, des barrières, etc. La suppression de ce genre de cas est un axe de recherche qui peut s'avérer très prolifique mais que nous n'avons pas eu le temps d'explorer. Une première solution consisterait à intégrer la position des points d'intérêt dans le modèle de transformation permettant d'estimer les solutions, comme nous l'avons déjà proposé dans la description générale de notre méthode d'estimation des solutions. On pourrait également s'attacher à développer un détecteur de points d'intérêt abandonnant les points trop présents dans une même image. Une détection de points d'intérêt spatio-temporels pourrait par exemple éviter ce genre de cas.

6.3.4 Influence de l'espérance de la requête statistique α sur la courbe opérationnelle du système

Le principe des expérimentations est de sélectionner aléatoirement un fichier MPEG1 d'environ 20 minutes dans le catalogue de référence, de lui appliquer une transformation (après décodage) puis de rechercher tous les segments clé dans la base.
Le taux de reconnaissance, t_{rn}, est défini par le pourcentage de segments clé correctement identifiés. Un segment clé est correctement identifié si la solution retournée possède bien l'identifiant du fichier MPEG1 transformé, si le code temporel estimé est correct à 2 images prés et si la valeur de la mesure de similarité est supérieure au seuil de décision.
La base de signature utilisée est $REEL_{500}$ (environ 500 heures de vidéo, cf. section 5.1.4, page 124).
Les paramètres de la requête statistique sont :

$$\sigma_g = 18$$

$$\alpha \in \{\ 50\ \%\ ,\ 70\ \%\ ,\ 80\ \%\ ,\ 90\ \%\ ,\ 95\ \%\ \}$$

La transformation appliquée à l'extrait de référence est une combinaison de plusieurs transformation de paramètres :

$$w_{scale} = 0,865 \quad w_{gamma} = 1,4 \quad w_{noise} = 8,0 \quad w_{contrast} = 1,4 \quad w_{shift} = 5\ \%$$

La figure 6.12 représente les courbes opérationnelles de notre système pour les différentes valeurs de α. Les valeurs de la fréquence des fausses alarmes t_{fa} utilisées pour leur construction sont celles qui ont été déterminées pour les mêmes paramètres et la même base, lors des expérimentations présentées dans la section 6.3.3. La table 6.6 récapitule certaines données issues de ces mêmes expérimentations. On y trouvera ainsi le temps moyen d'une recherche dans la base, la valeur du taux de reconnaissance pour trois valeurs de la fréquence de fausses alarmes, dont les deux valeurs extrêmes 0 % et 100 %. La valeur du seuil de la mesure de similarité est indiquée entre parenthèses (sauf pour $t_{fa} = 100$ % où la valeur du seuil est toujours $n_{sol} = 1$). La dernière colonne, enfin, donne la valeur moyenne de la mesure de similarité sur tous les segments clé de l'extrait candidat.

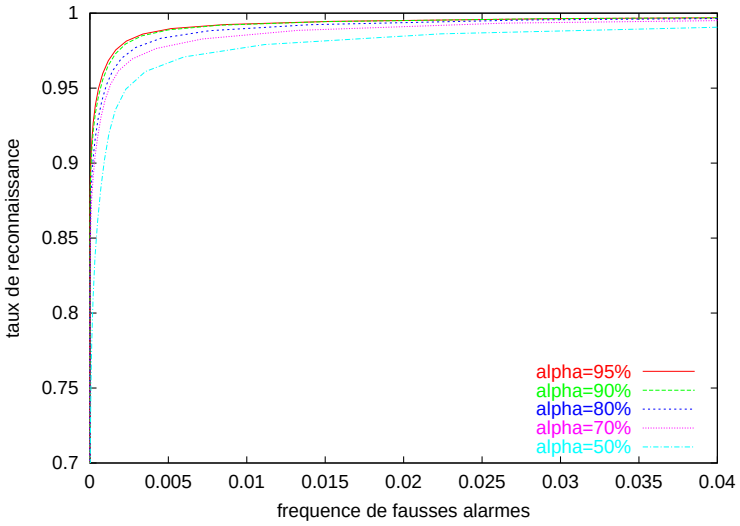


FIG. 6.12 – Courbes opérationnelles du système de détection de copies pour différentes valeurs de α

α	temps (ms)	t_{rn} pour $t_{fa} = 0$ %	t_{rn} pour $t_{fa} = 1$ %	t_{rn} pour $t_{fa} = 100$ %	$\overline{n_{sol}}$
50	0,27	85,33 % (n_{sol}=32)	98,55 % (n_{sol}=12)	100,00 %	51
70	0,55	89,90 % (n_{sol}=32)	99,27 % (n_{sol}=13)	100,00 %	60
80	0,88	92,33 % (n_{sol}=32)	99,51 % (n_{sol}=13)	100,00 %	65
90	1,47	93,75 % (n_{sol}=32)	99,51 % (n_{sol}=13)	100,00 %	68
95	2,21	94,23 % (n_{sol}=32)	99,51 % (n_{sol}=13)	100,00 %	70

TAB. 6.6 – Influence de l'espérance de la requête statistique α sur la détection de copies

Les courbes opérationnelles montrent clairement qu'une forte diminution de l'espérance de la requête statistique et donc du nombre de signatures correctement retrouvées, ne dégrade que peu les performances globales du système. Seule une espérance faible, égale à 50 %, déstabilise les performances. La dégradation du taux de reconnaissance est toutefois d'autant plus forte que la fréquence des fausses alarmes est faible.
Regardons maintenant les données de la table 6.6. On peut tout d'abord remarquer que le taux de reconnaissance pour une fréquence de fausses alarmes de 100 % est toujours égal à 100 %. Cela signifie que l'algorithme d'estimation des solutions retrouve toujours la bonne solution quelle que soit la valeur de *alpha*. Seul le seuil de décision sur la mesure de similarité élimine certaines des solutions dans les résultats définitifs.
Pour le cas le plus contraignant, c'est-à-dire $t_{fa} = 0$ % (0 fausse alarme dans les 24 heures de capture de *TV7 Tunis*), on passe d'un taux de reconnaissance de $94,23$ % pour $alpha = 95$ % à un taux de $89,90$ % pour $\alpha = 70$ %. On a donc perdu uniquement $5,1$ % des détections tandis que le temps moyen d'une recherche a été divisé par un facteur 4. Si l'on se permet une fréquence plus importante de fausses alarmes de l'ordre de 1 %, on peut divisé le temps de recherche moyen par 8 en perdant moins de 1 % des bonnes détections. Cette expérimentation montre donc clairement que la recherche statistique des signatures dans la base ne perturbe que faiblement les étapes d'estimation et de décision de notre méthode de détection de copies. Il est donc possible de trouver des compromis tout à fait profitables entre performances et efficacité du système.

Il est, dans un premier temps, étonnant de remarquer que la valeur moyenne de la mesure de similarité n'est pas proportionnelle à l'espérance de la requête statistique. n_{sol} représentant en effet le nombre de points d'intérêt appariés, on s'attendrait à ce que sa valeur soit proportionnelle au pourcentage de signatures correctement retrouvées dans la base. Ceci n'est en fait pas vrai, puisqu'il faut prendre en compte la (non-)répétabilité du détecteur de points d'intérêt. Le fait que la valeur de n_{sol} semble converger vers une valeur proche de 70 lorsque l'on augmente α, signifie que le nombre de points répétés par le détecteur de points d'intérêt est proche de cette valeur. Pour s'en convaincre, nous avons calculé la $(1,5)$-répétabilité moyenne du détecteur de Harris couplé au détecteur d'images clé, sur le même extrait et pour la même transformation. Nous avons obtenu une répétabilité de $45,6$ % pour un nombre moyen de points d'intérêt par segment clé égal à 171. Le nombre moyen de points répétés correspondant est alors égal à $77,9$.

6.3.5 Influence du nombre de signatures dans la base sur la courbe opérationnelle du système

Le protocole expérimental est le même que pour l'étude de l'influence de α.
Les bases de signatures utilisées sont les bases $REEL_{100}$, $REEL_{500}$, $REEL_{2500}$ et $REEL_{10000}$ (cf. section 5.1.4, page 124).
Les paramètres de la requête statistique sont :

$$\sigma_g = 18 \quad \alpha = 80\%$$

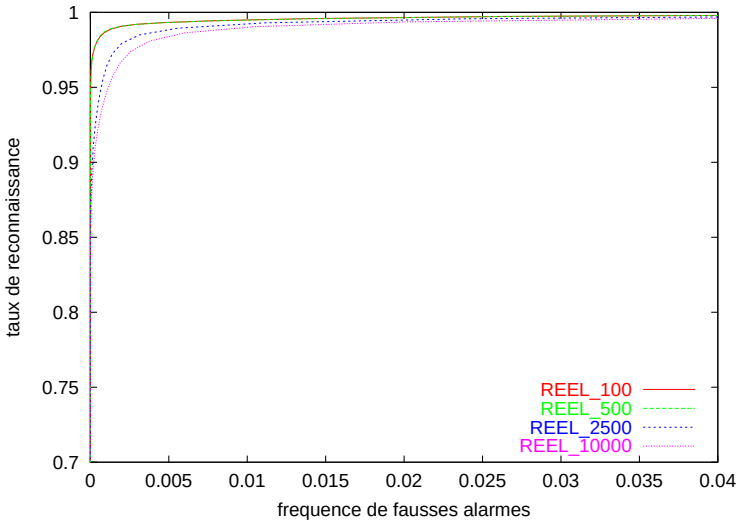


FIG. 6.13 – Courbes opérationnelles du système de détection de copies pour plusieurs tailles de base

Les courbes opérationnelles de la figure 6.13 montrent qu'une augmentation très forte de la taille du catalogue de référence ne dégrade que peu l'efficacité de notre système de détection de copies. Cela signifie que notre algorithme d'estimation des solutions et de décision reste efficace malgré la forte augmentation du nombre de signatures similaires retournées pour chaque signature candidate. Si l'on regarde les données de la table 6.6, on voit ainsi que l'algorithme d'estimation retourne la bonne solution dans 100 % des cas, quelle que soit la taille de la base. Dans le cas le plus contraignant, c'est-à-dire $t_{fa} = 0$ %, on ne perd que 6 % des détections alors

Base	temps (ms)	t_{rn} pour $t_{fa} = 0\,\%$	t_{rn} pour $t_{fa} = 1\,\%$	t_{rn} pour $t_{fa} = 100\,\%$	$\overline{n_{sol}}$
Base $REEL_{100}$	0,30	97,59 % (n_{sol}=32)	99,52 % (n_{sol}=12)	100,00 %	85
Base $REEL_{500}$	0,88	97,59 % (n_{sol}=32)	99,52 % (n_{sol}=13)	100,00 %	84
Base $REEL_{2500}$	2,42	93,02 % (n_{sol}=32)	99,52 % (n_{sol}=13)	100,00 %	69
Base $REEL_{10000}$	8,50	91,60 % (n_{sol}=33)	99,51 % (n_{sol}=13)	100,00 %	66

TAB. 6.7 – Influence du nombre de signatures dans la base sur l'efficacité de la détection de copies

que la taille de la base a été multipliée par 100.
On vérifie également que l'évolution du temps de recherche moyen est sous-linéaire en fonction de la taille de la base, puisque celui-ci est multiplié par environ 30 lorsque la taille de la base est multipliée par 100.
Si l'on observe les valeurs moyennes de la mesure de similarité pour les différentes bases, on constate que celle-ci décroît lorsque la taille de la base augmente, ce qui explique la diminution du taux de reconnaissance. Cela signifie que le nombre de points d'intérêt appariés diminue avec la taille de la base, alors que les paramètres de la requête statistique reste les mêmes. Cela est probablement lié à la limitation logicielle à 500 du nombre de signatures similaires pris en compte pour chaque point d'intérêt. Nous verrons en effet dans la section 6.5, relative à l'analyse des bases de signatures, que pour des bases de signatures très volumineuses, le nombre moyen de signatures contenues dans un rayon autour d'une requête devient linéaire en fonction de la taille de la base. La limitation du nombre de signatures similaires engendre donc une perte de certaines signatures dans la solution finale pour les très grandes bases.
L'augmentation de ce nombre de signatures similaires est pour l'instant problématique pour notre algorithme d'estimation pour des raisons de coût de calcul et de stockage. Cette remarque est importante, puisqu'elle signifie que le goulet d'étranglement pour des bases encore plus volumineuses (100 000 heures par exemple) ne se situera pas au niveau de la recherche des signatures dans la base mais au niveau de l'algorithme d'estimation des solutions. L'optimisation de cet algorithme pour un très grand nombre de signatures similaires par observation est donc un axe de recherche pertinent pour l'avenir de la technologie.

6.3.6 Influence du paramètre σ_g du modèle de distorsion sur la courbe opérationnelle du système

Le protocole expérimental est ici encore le même que pour l'étude de l'influence de α.
Les paramètres de la requête statistique sont :

$$\sigma_g = \in \{\ 10\ ,\ 15\ ,\ 18\ ,\ 20\ ,\ 25\ \}$$

$$\alpha = 80\%$$

Les courbes opérationnelles de la figure 6.14 montrent que les faibles valeurs de σ_g entraînent une perte d'efficacité du système, tandis que les fortes valeurs tendent à stabiliser l'efficacité du système. Comme nous l'avons vu précédemment la valeur de σ_g permet en effet de régler la sévérité des transformations auxquelles on souhaite se rendre robuste.
Nous avons estimé indépendemment de cette expérimentation l'écart-type des composantes de la distorsion pour la transformation expérimentée. Celui-ci est égal à :

$$\overline{\sigma_g} = 18,32$$

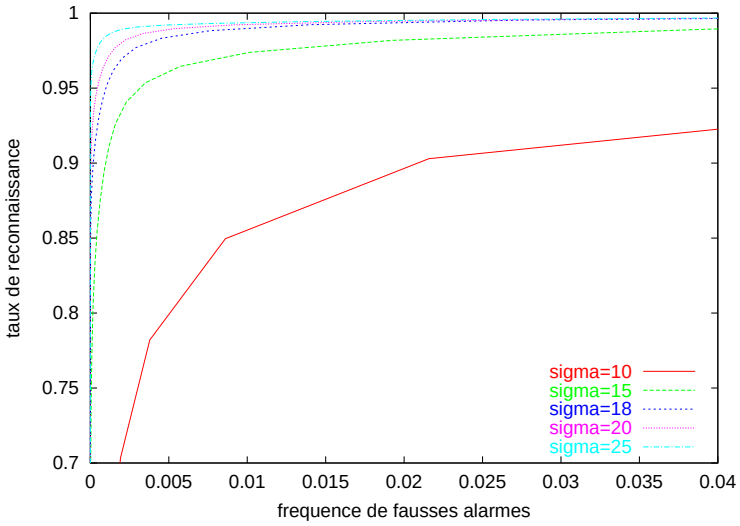


FIG. 6.14 – Courbes opérationnelles du système de détection de copies pour plusieurs valeurs de σ_g

α	temps (ms)	t_{rn} pour $t_{fa} = 0$ %	t_{rn} pour $t_{fa} = 1$ %	t_{rn} pour $t_{fa} = 100$ %	$\overline{n_{sol}}$
10	0,34	39,66 % (n_{sol}=21)	92,54 % (n_{sol}=8)	100,00 %	18
15	0,62	85,53 % (n_{sol}=32)	97,83 % (n_{sol}=12)	100,00 %	50
18	0,88	92,33 % (n_{sol}=32)	99,51 % (n_{sol}=13)	100,00 %	65
20	1,06	95,19 % (n_{sol}=32)	99,51 % (n_{sol}=13)	100,00 %	72
25	1,69	97,59 % (n_{sol}=33)	99,51 % (n_{sol}=15)	100,00 %	84

TAB. 6.8 – Influence de σ_g sur l'efficacité de la détection de copies

Si on paramètre le modèle de distorsion avec des valeurs de σ_g inférieures à cette valeur le nombre de signatures non retrouvées est important et l'efficacité du système est mauvaise. Si l'on observe la valeur moyenne de la mesure de similarité dans la table, on voit ainsi que le nombre de points d'intérêt appariés est de 18 pour $\sigma_g = 10$. Le taux de reconnaissance dans le cas le plus contraignant, c'est-à-dire $t_{fa} = 0$ %, est alors seulement de $39,7$ %.
Si on paramètre le modèle de distorsion avec des valeurs de σ_g supérieures à $\overline{\sigma_g} = 18,32$, l'efficacité du système pour la transformation expérimentée se stabilisera.

6.4 Evaluation de la détection de copies d'extraits vidéo

L'objectif des expérimentations de cette section est d'évaluer l'efficacité de notre système de détection de copies dans des conditions plus proches du fonctionnement réel que lors des analyses précédentes.
La longueur des extraits candidats a une importance capitale dans l'efficacité du système. Lors des expérimentations de la section précédente, la séquence candidate était un extrait d'environ une vingtaine de minutes dont l'intégralité appartenait à la base. Les segments clé candidats étaient donc intégralement constitués d'images appartenant au catalogue de référence. Dans la réalité, les extraits diffusés peuvent être plus courts que les segments clé candidats. Cela a pour effet de diminuer la mesure de similarité puisque le nombre de points appariés est systématique-

ment plus faible.
Dans les expérimentations de cette section, le taux de reconnaissance sera défini par le pourcentage d'extraits candidats correctement identifiés et non plus par le pourcentage de segments clé. Un extrait candidat diffère d'un segment clé candidat en ce qu'il a une durée fixe et qu'il n'y a pas de recouvrement entre deux extraits candidats successifs. Un extrait est considéré comme correctement identifié si un segment candidat l'a lui même été pour au moins une image clé appartenant à l'extrait.
Dans ces expérimentations, nous ne nous intéressons plus aux courbes opérationnelles du système, mais seulement au taux de reconnaissance pour un seuil de décision fixe. Dans tous les cas, le seuil de décision est choisi de manière à ce que la fréquence de fausses alarmes soit égale à

$$t_{fa} = 0,0002$$

soit environ 12 fausses alarmes par jour en moyenne (cf. équation 6.1, page 158).

6.4.1 Influence de la longueur des extraits diffusés

Nous étudions ici l'influence de la longueur des extraits candidats sur le taux de reconnaissance de notre système de détection de copies. Le protocole expérimental est le suivant : 100 extraits d'une durée l_{sec} sont sélectionnés aléatoirement dans le catalogue de référence. Chaque extrait est obtenu en sélectionnant aléatoirement un des fichiers MPEG1 du catalogue, puis en ne décodant qu'un extrait d'une durée l_{sec} sélectionné aléatoirement à l'intérieur du fichier.
Les 100 extraits sont ensuite accolés les uns aux autres pour constituer une séquence candidate unique dont les segments clé sont recherchés dans la base. L'opération est renouvelée pour différentes valeurs de l_{sec}.
La base de référence est la base $REEL_{2500}$ (cf. section 5.1.4, page 124).
La transformation appliquée aux extraits candidats est une combinaison de plusieurs transformations de paramètres :

$$w_{scale} = 0,9 \;\; w_{gamma} = 1,4 \;\; w_{noise} = 8,0 \;\; w_{contrast} = 1,4 \;\; w_{shift} = 5\%$$

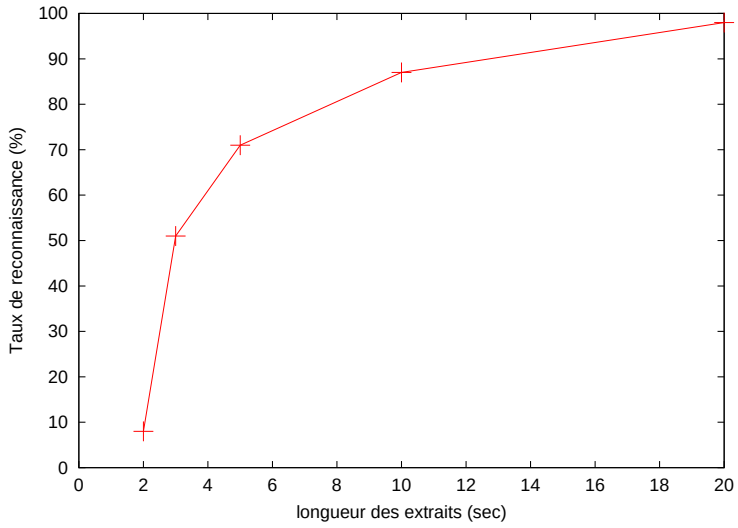


FIG. 6.15 – Taux de reconnaissance en fonction de la longueur des extraits candidats

l_{sec} (sec)	taux de reconnaissance (%)
2	8,2
3	51,2
5	71,2
10	87,4
20	97,6

TAB. 6.9 – Taux de reconnaissance en fonction de la longueur des extraits candidats

Nous rappelons que la durée moyenne d'un segment clé est de $11,7$ secondes. On constate que pour des extraits candidats d'une longueur de 5 secondes, le taux de reconnaissance est encore de $71,2$ % bien qu'en moyenne, les extraits de cette longueur ne contiennent que $3,85$ images clé.
L'efficacité du système s'écroule pour des extraits d'une durée inférieure à 3 secondes.
Pour augmenter la durée minimale des extraits détectables par notre système, il faudrait augmenter la fréquence des images clé ou augmenter le nombre de points d'intérêt maximum par image. Cela se traduirait nécessairement par une augmentation de la taille de la base et donc des coûts de stockage et de recherche. Il s'agit donc encore une fois d'un compromis entre efficacité et performances du système.
En pratique, la longueur des extraits d'archives diffusés à la télévision est très variable. On trouve ainsi des diffusions d'une durée supérieure à 1 heure (films, documentaires, pièces de théâtre filmées, etc.), des diffusions d'une durée moyenne, d'une dizaine de secondes à quelques minutes (rediffusion d'extraits de variété comme des chansons ou des sketchs, interviews, etc.) ou bien des diffisions de courte durée comprise entre 3 et 10 secondes (journaux télévisés, émissions sportives, etc.). Il existe enfin des documents où le remontage temporel des extraits est tellement resserré que la durée de certains extraits peut descendre en dessous des 3 secondes. C'est par exemple le cas de documentaires à base de photographies ou d'archives anciennes sans bande sonore.

6.4.2 Evaluation de la robustesse aux transformations

L'objectif des expérimentations de cette section est d'évaluer le taux de reconnaissance de notre système pour l'ensemble des transformations étudiées dans ce chapitre, pour différentes tailles de base et pour différentes valeurs des paramètres α et σ_g. Le protocole expérimental est le même que pour les expérimentations de la section précédente. La longueur des extraits candidats sélectionnés aléatoirement dans le catalogue de référence est cependant fixe et égale à :

$$l_{sec} = 10\ sec$$

Les bases de signatures utilisées sont exactement du même type que la série $REEL_H$ décrite dans la section 5.1.4 (page 124). Cependant, le nombre d'heures qu'elles contiennent est légèrement différent. Nous les nommons $REEL_{110}$ (environ 110 heures de vidéo), $REEL_{875}$, $REEL_{3500}$ et $REEL_{10000}$.
Les paramètres de la requête statistique sont :

$$\alpha \in \{\ 50\%\ ,\ 70\%\ ,\ 80\%\ ,\ 90\%\ ,\ 95\%\ \}$$

$$\sigma_g \in \{\ 10\ ,\ 15\ ,\ 20\ ,\ 25\ \}$$

Lorsque la variation de la taille de la base n'est pas étudiée, la base par défaut est :

$$REEL_{3500}$$

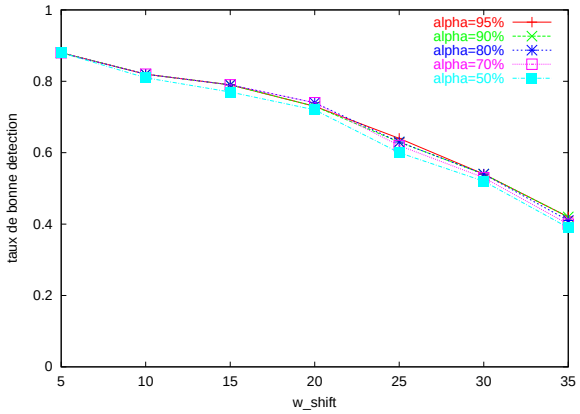
Lorsque les variations de l'espérance de la requête statistique α ne sont pas étudiées, sa valeur par défaut est :

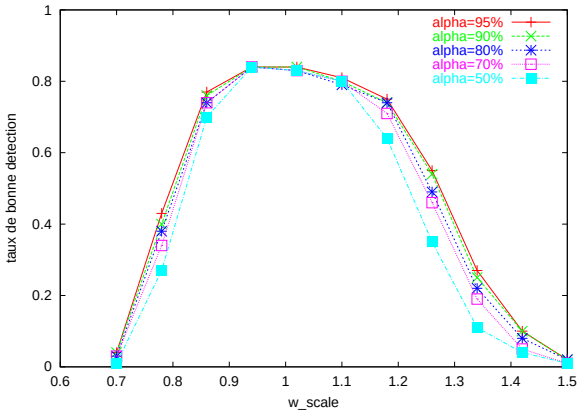
$$\alpha = 80\ \%$$

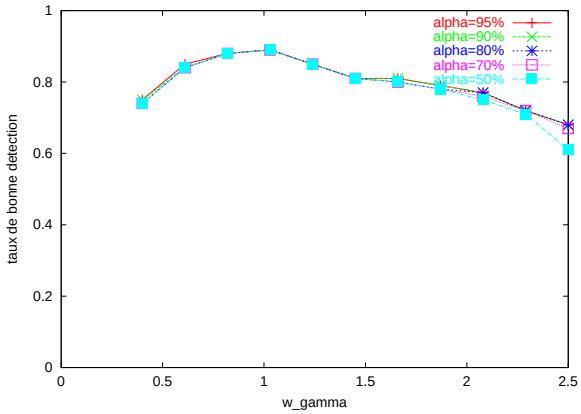
Lorsque les variations du paramètre σ_g du modèle de distorsion ne sont pas étudiées, sa valeur par défaut est :

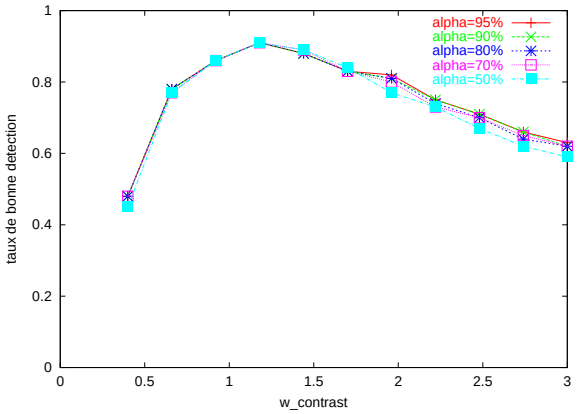
$$\sigma_g = 20$$

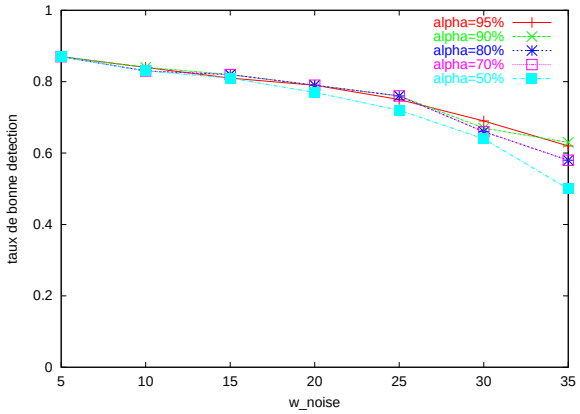
La suite de cette section est constituée de 3 pages de résultats contenant chacune 5 figures et une table. Les 5 figures de chaque page correspondent aux cinq types de transformations étudiées : décalage vertical, redimensionnement, modification du gamma, modification du contraste et ajout de bruit gaussien (cf. section 6.1.3, page 148). Les 3 pages correspondent aux trois paramètres dont on étudie l'influence sur l'efficacité du système à savoir : l'espérance de la requête statistique α, la taille de la base et le paramètre σ_g du modèle de distorsion. La table présentée sur chaque page présente l'évolution du temps de recherche moyen d'une requête en fonction de la valeur du paramètre étudié.





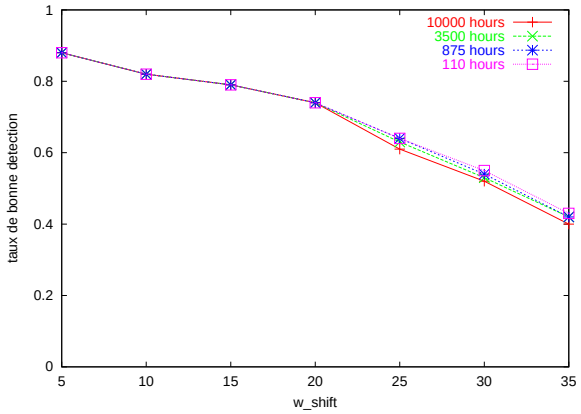


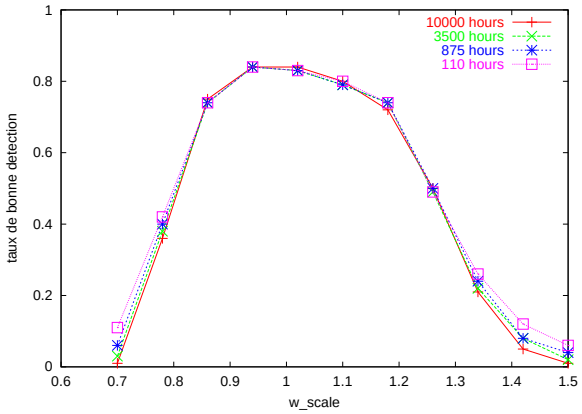


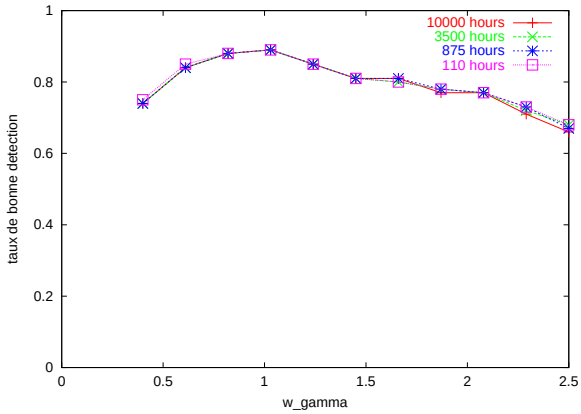


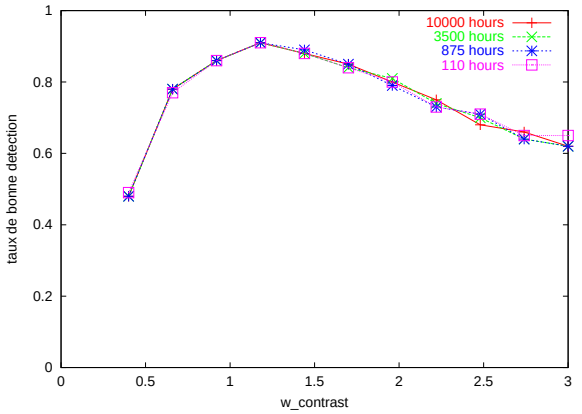
α (%)	temps de recherche (ms)
50	1,05
70	2,15
80	3,45
90	5,76
95	8,64

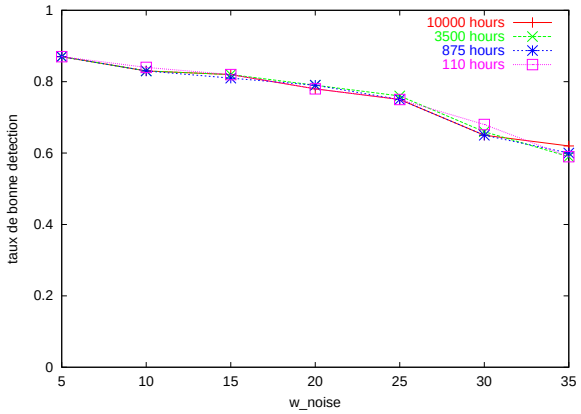
TAB. 6.10 – Efficacité de la détection d'extraits de 10 secondes pour plusieurs valeurs de l'espérance de la requête statistique α





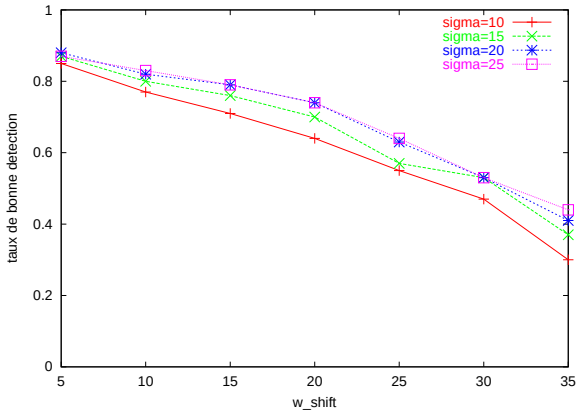


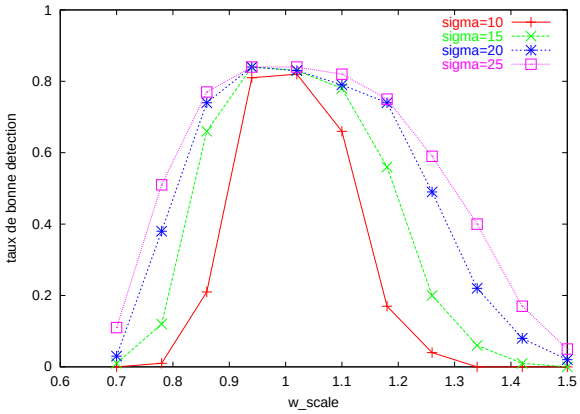


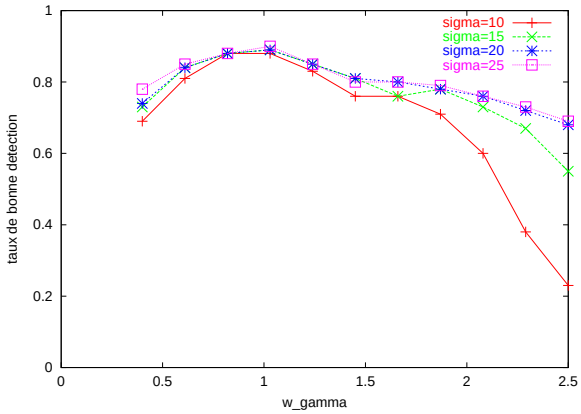


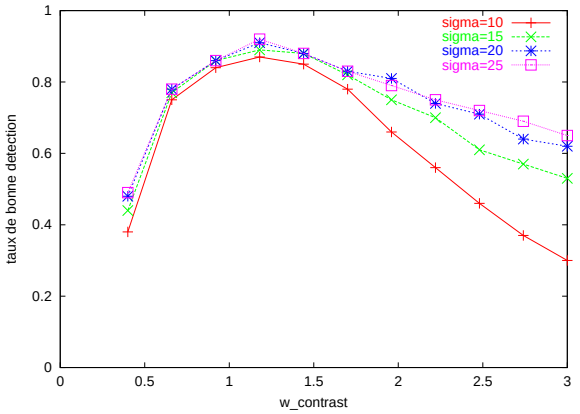
nombre d'heures	N	temps (ms)
110	6 013 876	0,69
875	49 107 362	1,81
3500	195 103 572	3,45
10 000	555 238 372	9,11

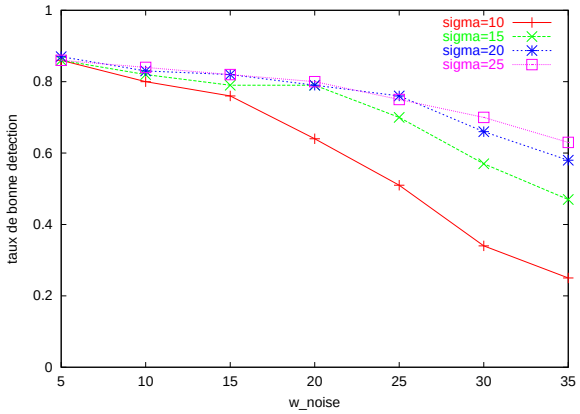
TAB. 6.11 – Efficacité de la détection d'extraits de 10 secondes pour plusieurs bases de signatures de taille croissante











σ	temps de recherche (ms)
10	1,11
15	2,01
20	3,45
25	5,50

TAB. 6.12 – Efficacité de la détection d'extraits de 10 secondes pour plusieurs valeurs de σ_g

D'une manière générale, on constate que notre système de détection de copies présente une très bonne robustesse aux transformations étudiées, compte tenu de la faible fréquence des fausses alarmes ($t_{fa} = 0,0002$ %) et de la très grande taille des catalogues de référence.
Le fait que le taux de reconnaissance ne dépasse pas les 91 %, même pour des transformations très peu sévères est principalement dû au fait que la longueur des extraits candidats est de 10 secondes. Avec des extraits d'une longueur de 20 secondes ou plus, on aurait des taux de reconnaissance maximum proche de 100 % comme cela l'a déjà été montré dans les expérimentations précédentes.
Un autre phénomène rend difficile la détection de certains extraits. Nous avons en effet constaté que de 2 à 5 % du contenu vidéo de la base SNC de l'INA d'où sont extraites nos vidéos, contiennent en fait des séquences très peu pertinentes mais très difficiles à discriminer entre elles. Il s'agit principalement de plans noirs ou bruités entre les séquences, mais également de mires fixes ou encore de compte à rebours qui précèdent généralement les journaux télévisés et qui peuvent être présents plus de 1 000 fois dans la base, perturbant alors notre algorithme d'estimation des résultats. Les extraits utilisés pour nos expérimentations étant tirés aléatoirement parmi l'intégralité de la base, le même pourcentage de ce type d'extraits se retrouvent dans notre corpus de test.

Les résultats relatifs à l'influence de l'espérance de la requête statistique α (figure 6.10, page 170), montrent clairement qu'une réduction de la qualité de la recherche permet d'obtenir des gains de performance importants en ne dégradant que très peu le taux de reconnaissance du système. On peut ainsi diviser le coût moyen d'une recherche par 4 sans dépasser quelques pourcents de perte et ce, quelle que soit la transformation. Même lorsque seulement 50 % des signatures sont censées être retrouvées, la pourcentage de séquences détectées ne diminue que d'une quinzaine de points pour les transformations les plus sévères.

Les résultats relatifs à l'influence de la taille de la base (figure 6.11, page 171), montrent que le taux de reconnaissance reste quasiment inchangé pour de très fortes augmentations de la taille de la base de référence. La base la plus volumineuse expérimentée ici contient 10 000 heures de vidéo et il n'existe à notre connaissance aucun système permettant de rechercher des copies vidéo dans une base aussi volumineuse, avec une telle robustesse et des temps de recherche aussi faibles. Ces résultats mettent particulièrement en lumière l'avantage de notre approche utilisant conjointement des signatures locales et une mesure de similarité globale basée sur la position des points d'intérêt. Avec une signature globale unique, l'augmentation de la densité des signatures lorsque la taille de la base augmente, se traduirait forcément par une perte de discriminance de la signature et donc par une dégradation de l'efficacité globale du système. Dans le cas de notre approche l'augmentation de la densité des signatures locales se traduit par une augmentation du nombre de signatures retournées par le système de recherche mais celle-ci n'a que très peu d'influence sur la mesure de similarité globale basée sur le comptage des signatures temporellement pertinentes.

Les résultats relatifs à l'influence du paramètre σ_g du modèle de distorsion (figure 6.11, page 172), montre que ce paramètre permet de régler la sévérité des transformations auxquelles on souhaite se rendre robuste. Lorsque la valeur de σ_g augmente, le *décrochage* de la courbe du taux de reconnaissance se fait pour des transformations de plus en plus sévères. Cela est particulièrement visible pour l'ajout d'un bruit gaussien dans l'image. On peut vérifier, grâce à la table 6.3 (page 154), que la valeur du paramètre d'une transformation à partir duquel une courbe décroche des autres courbes correspond à une valeur de $\overline{\sigma_g}$ très proche de la valeur de σ_g.

6.4.3 Détection des duplicata

L'objectif de l'expérimentation de cette section est de vérifier que notre système détecte bien les duplicata, c'est-à-dire les extraits présents plusieurs fois dans la base sous des identifiants différents. Nous avons pour cela inséré dans la base $REEL_{2500}$ un extrait vidéo d'environ 30 secondes en 50 exemplaires. Nous avons ensuite transformé ce même extrait par une composition de plusieurs transformations légères de paramètres :

$$w_{scale} = 0,95 \;\; w_{gamma} = 1,3 \;\; w_{noise} = 3,0 \;\; w_{contrast} = 1,3 \;\; w_{shift} = 5\%$$

avant de le rechercher dans la base.
Les paramètres de la requête statistique sont :

$$\sigma_g = 18 \quad \alpha = 80\%$$

Nous avons ensuite simplement vérifié que les 50 résultats étaient bien présents dans le fichier résultat édité par notre système.

6.4.4 Vérités terrain

Afin de vérifier l'efficacité de notre système de détection de copies dans le cas de séquences vidéo réellement diffusées à la télévision et ayant subit des transformations non simulées, nous avons mené deux expérimentations relatives à deux vérités terrain différentes que nous avons appelé *Coluche* et *Libération*, en rapport avec le thème abordé dans ces documents.

Coluche

Le document initial est un enregistrement sur cassette *VHS* d'une émission de variété diffusée en 2001 sur une chaîne de télévision française. Ce document était en possession du service chargé du suivi des droits à l'INA, suite à une réclamation d'ayant-droits se plaignant de la diffusion de certaines archives sans leur consentement. Dans cette émission d'une durée totale de $2H15$, 21 extraits de sketchs du comique Coluche ont été diffusés. Le reste de l'émission est constitué uniquement de scènes de plateaux qui ne sont a priori pas archivées à l'INA. Les 21 extraits ont été recherchés dans la base d'archives de l'INA par un documentaliste et les archives originales ont toutes pu être identifiées. Cette recherche s'est faite sans aucun système automatique de recherche par le contenu mais uniquement par des requêtes textuelles dans la base des notices accompagnant les documents archivés à l'INA.
Ces 21 archives étant réparties dans 16 fichiers $MPEG1$ de la base SNC de l'INA, nous nous sommes procurés ces 16 fichiers (d'environ 20 minutes chacun) et nous les avons insérés dans la base $REEL_{1000}$. D'un autre côté un ingénieur de l'INA a construit la vérité terrain exacte en ajustant manuellement les codes temporels de début et de fin des 21 extraits diffusés. Il en résulte que les 21 extraits ont une durée moyenne de 59 secondes, l'extrait le plus long ayant une durée de 350 secondes et l'extrait le plus court une durée de 6 secondes.

Nous avons ensuite numérisé le contenu de la cassette *VHS* et nous l'avons recherchée dans

la base $REEL_{1000}$ complétée par les fichiers de référence de la vérité terrain. Les paramètres de la requête statistique ont été fixé à :

$$\sigma_g = 20 \quad \alpha = 90\%$$

Le résultat de l'expérimentation est que les 21 extraits ont été correctement identifiés et qu'il n'y a eu aucune fausse alarme pour la totalité de l'émission (un extrait est considéré comme correctement identifié si un segment clé a été correctement identifié pour au moins une image clé).
Un exemple de détection de cette vérité terrain est présenté sur la figure 6.16.

FIG. 6.16 – Une détection de la vérité terrain *Coluche*

Il est difficile d'estimer précisément toutes les transformations qu'ont subies les extraits diffusés, mais les images montrent clairement la présence systématique de modifications colorimétriques souvent couplées à un redimensionnement (comme illustré sur la figure 6.16). Nous avons également constaté des effets de saturation du signal des niveaux de gris qui n'existaient pas dans les archives d'origine.
L'expérimentation montre également que notre système est robuste à la chaîne de traitements qu'ont subie les séquences. Le premier traitement est le processus de post-production et de diffusion en lui-même. Le deuxième est la conversion analogique et le stockage de la vidéo sur *VHS* réalisés par le service juridique de l'INA, ainsi que la renumérisation réalisée par nos soins. Enfin, il faut savoir que les extraits vidéo fournis par l'INA aux chaînes de télévision ne sont pas les mêmes que les fichiers MPEG1 que nous utilisons pour construire notre base de signatures, qui eux ne servent en principe qu'à la visualisation des archives. Les vidéos fournies aux exploitants sont en général au format MPEG2 et stockées sur cassette *Betacam Numérique*. Il est donc fort probable que les extraits candidats n'aient pas subi le même processus de compression/décompression que les extraits que nous avons insérés dans notre base de signatures.

Libération

La vérité terrain *Libération* est moins rigoureuse que la vérité terrain *Coluche* dans le sens où elle a été construite en partie par notre propre système de détection de copies avec des paramètres de réglage très lâches.
Il s'agit d'un documentaire sur le thème de la libération de la France à la fin de la seconde guerre mondiale, qui a été édité et commercialisé sur support DVD. Le service juridique de l'INA ayant

des doutes sur la provenance de certaines archives réutilisées dans ce documentaire, nous a proposé de rechercher dans le documentaire les extraits qui sont effectivement présents dans la base de l'INA. La quantité d'archives contenues dans le documentaire étant très importante, il n'a pas été possible de les rechercher dans la base des notices de l'INA comme cela l'a été fait pour la vérité *Coluche*. Une grande partie des archives réutilisées dans ce DVD n'appartiennent sûrement pas à l'INA mais il est impossible de le garantir avec certitude puisqu'il faudrait connaître toutes les archives de la base. Étant donné qu'il n'est pas possible de bâtir manuellement une vérité terrain à partir de ce document, nous nous sommes contentés d'essayer d'en retrouver un maximum avec notre système de détection de copies.
Un ingénieur de l'INA s'est chargé d'identifier un ensemble de fichiers de la base de l'INA susceptible de contenir des extraits réutilisés dans le DVD. Il a en fait simplement interrogé la base des notices de l'INA par une requête sur la date de production des archives. Il nous a ainsi fourni une liste de **tous** les fichiers MPEG1 de la base SNC, contenant des archives de l'INA dont la date de production se situe **entre le 01/01/1940 et le 31/12/1945**. Cela représente un total de 356 matériels (fichiers MPEG1) cumulant au total 126 heures de vidéo. Nous avons ensuite simplement construit une base de signatures à partir de tous ces fichiers.
D'un autre côté, le documentaire du DVD a été reconverti au format MPEG1 par l'algorithme *ffmpeg* avant d'être recherché dans la base. En réglant au plus bas les seuils de détection de notre système et en essayant plusieurs jeux de paramètres, nous avons pu identifier 138 extraits présents dans la base de référence.

L'objectif de l'expérimentation présentée dans cette section est d'évaluer la capacité de notre système à retrouver ces 138 extraits avec des paramètres fixes similaires à ceux utilisés lors des expérimentations précédentes (même nombre d'images clé, même nombre de points d'intérêt, etc.). Les paramètres de la requête statistique ont ainsi été réglés à :

$$\sigma_g = 20 \quad \alpha = 80\%$$

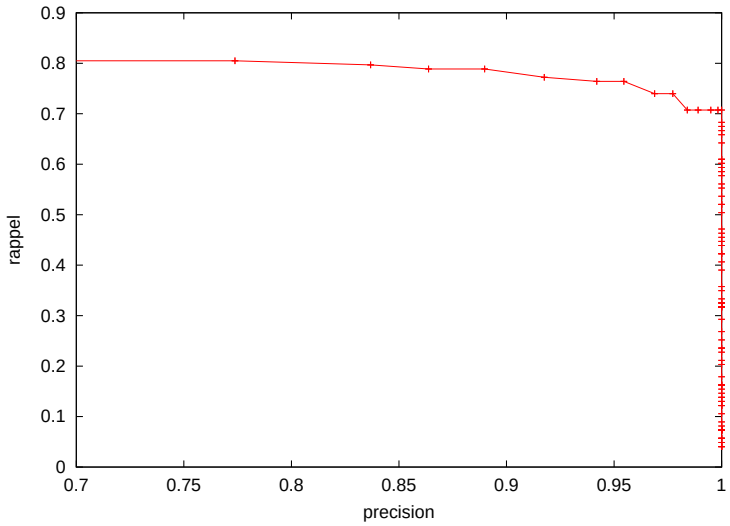


FIG. 6.17 – Courbe *ROC* de notre système sur la vérité terrain *Libération*

La figure 6.17 présente la courbe *ROC* de cette expérimentation. Il s'agit d'une courbe *ROC* classique, dans le sens où elle est exprimée en termes de rappel et de précision. S'agissant ici d'une

évaluation du système sur un document unique et non d'une évaluation d'un fonctionnement général de contrôle d'un flux TV, la précision est en effet plus pertinente que la fréquence des fausses alarmes. Le rappel est défini comme le pourcentage des 138 extraits de la vérité qui ont été correctement identifiés. La précision est définie comme le pourcentage des résultats retournés par le système qui sont des bonnes détections.

On constate sur la figure, que la totalité des 138 extraits n'est pas détectée même pour des seuils de similarité très bas. Ainsi, avec un seuil de similarité égal à $n_{sol} = 5$, seuls $82,9$ % des extraits sont détectés. Cette perte est en fait due à la très faible durée de certains extraits. Nous avons en effet vérifié que tous les extraits manqués avaient une durée inférieure à 2 secondes. Lorsque les extraits sont très courts, il est en effet possible qu'il n'y ait aucune image clé dans l'extrait, rendant toute détection impossible quel que soit le seuil de la mesure de similarité.
Le rappel de notre système pour une précision de 100 % (aucune fausse alarme dans tous les résultats) est de $70,75$ %, soit $12,9$ % de moins que le rappel maximal du système sur ce test.

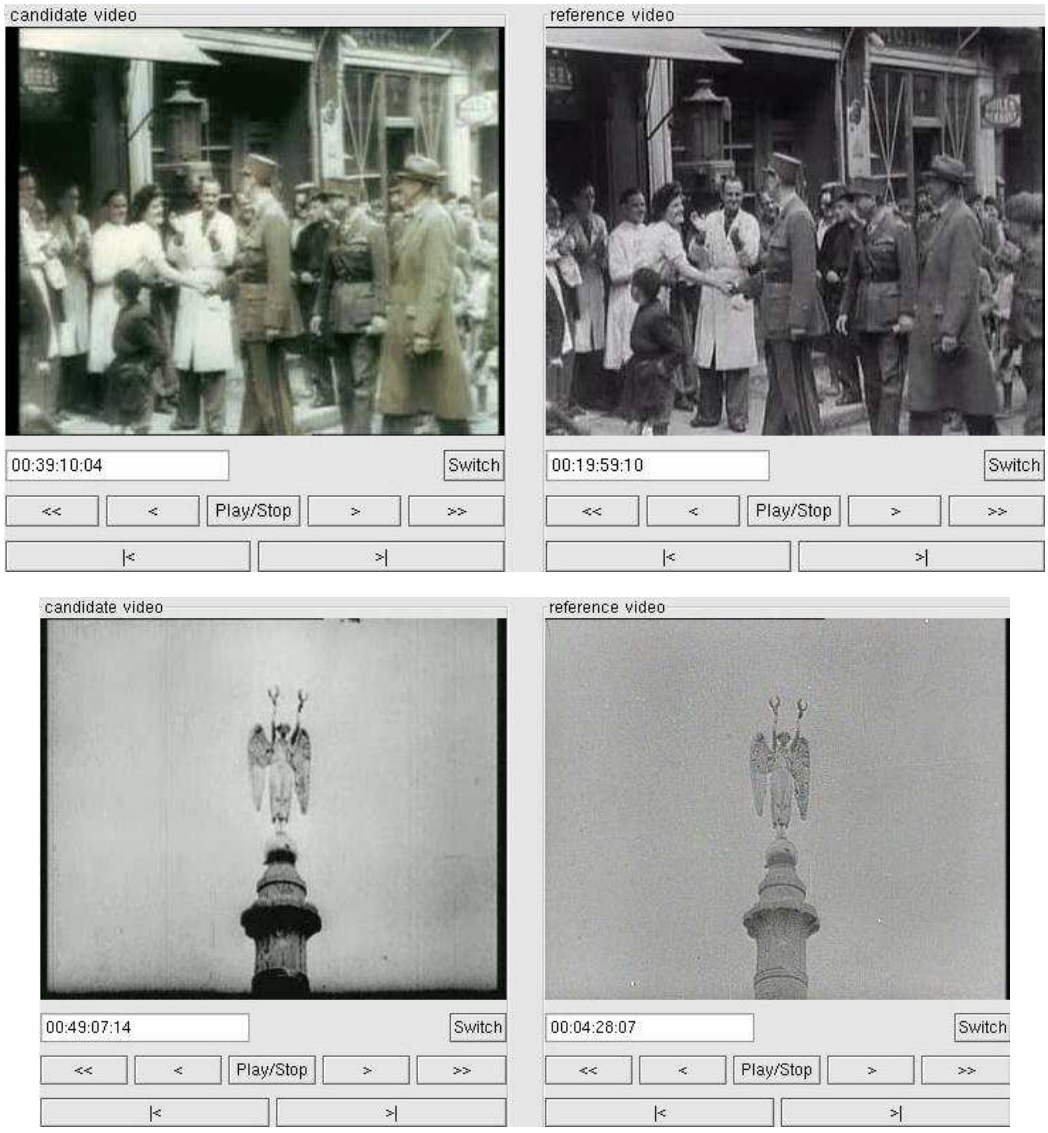


FIG. 6.18 – Deux détection de la vérité terrain *Libération*

La figure 6.18 présente deux détections de la vérité terrain *Libération.* parmi les transformations visuellement identifiables, on constate sur le premier résultat que certains extraits du DVD ont été recolorisés alors que les images d'origine étaient en noir et blanc. Notons que cette recolorisation n'a pas forcément été faite lors de l'édition du DVD et qu'elle peut être elle aussi plus ancienne. Notre système semble en tout cas robuste à ce type de transformation. On peut également constater que la séquence du DVD a été légèrement redimensionnée et décalée horizontalement.
Le deuxième résultat est très intéressant puisque l'extrait du DVD semble beaucoup plus altéré par le vieillissement que l'extrait d'origine conservé à l'INA. Lorsque l'on visionne ces deux séquences simultanément et dans le détail, il ne fait pourtant aucun doute qu'il s'agit de deux copies d'un même document d'origine. Cela signifie que ces deux documents ont un historique tout à fait différent depuis un certain nombre d'années. Le document de l'INA aura seulement été mieux conservé.
Cette expérimentation montre également la robustesse de notre système de détection au changement de codeur/décodeur $MPEG$ puisque le documentaire du DVD n'a pas été compressé avec le même encodage que les fichiers $MPEG1$ de la base SNC.

6.5 Analyse des bases de signatures

L'objectif des expérimentations de cette section est d'explorer le contenu des bases de signatures afin de faire une analyse critique de leur mode d'extraction et de proposer des améliorations potentielles.
La première expérimentation consiste à évaluer le nombre moyen de signatures se situant dans une hyper-sphère de rayon croissant pour plusieurs tailles de base différentes. Le principe est de sélectionner aléatoirement 10 000 signatures appartenant à la base, puis de compter le nombre moyen de signatures se situant dans une hyper-sphère de rayon ϵ centrée en ces points (sans recompter la requête elle même). La figure 6.19 représente l'évolution de ce nombre moyen de signatures en fonction de ϵ pour les trois bases $REEL_{500}$, $REEL_{3500}$ et $REEL_{18000}$. La courbe est tracée en coordonnées logarithmiques car elle présente ainsi beaucoup plus d'intérêt que la simple croissance exponentielle observée en coordonnées cartésiennes.

On peut d'abord vérifier, comme nous l'avions déjà remarqué dans la section 4.2.2 relative à l'étude du comportement asymptotique d'une recherche à ϵ-près, que le nombre moyen de signatures contenues dans une hyper-sphère est linéaire en fonction du nombre de signatures contenues dans la base. Sur cette courbe en coordonnées logarithmiques, cela se traduit par la constance de l'intervalle vertical entre chacune des courbes et l'on voit que celle-ci est respectée pour des rayon supérieurs à $\epsilon = 80$. Pour vérifier de manière plus approfondie la linéarité du nombre moyen de signatures contenues dans une hyper-sphère en fonction du nombre d'éléments dans la base, nous invitons le lecteur à revoir les courbes de la figure 4.1 (page 84).
La deuxième remarque que l'on peut faire sur ces courbes, est qu'elles présentent deux parties distinctes selon que l'on se trouve dans les faible valeurs du rayon de l'hyper-sphère (signatures très similaires) ou dans les fortes valeurs.
La linéarité de la courbe logarithmique pour les fortes valeurs du rayon, signifie que l'on peut exprimer le nombre moyen de signatures contenues dans une hyper-sphère de rayon ϵ sous la forme :

$$\overline{n_{HS}}(\epsilon) = k(\epsilon)^a \tag{6.2}$$

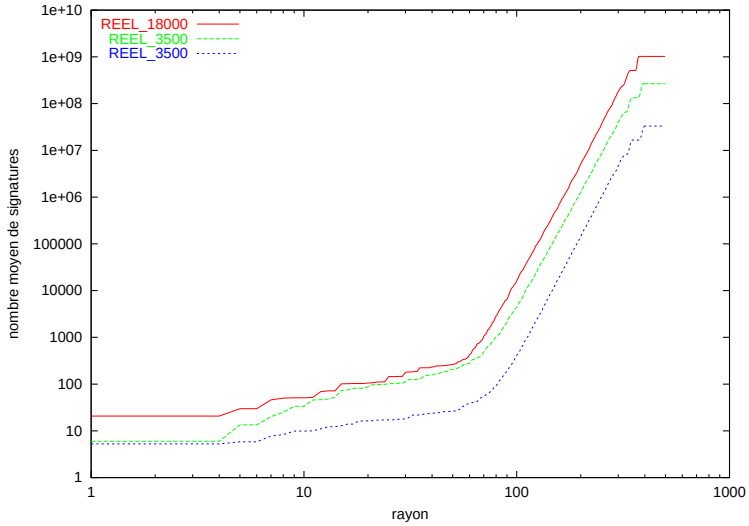


FIG. 6.19 – Nombre moyen de signatures contenues dans une hyper-sphère de rayon croissant

où a représente le coefficient directeur de la droite en coordonnées logarithmiques. Il est quasiment identique pour les trois bases et nous l'avons estimé à environ

$$a = 10,2$$

Si les signatures de la base étaient uniformément réparties dans l'espace, le nombre moyen de signatures serait proportionnel au volume de l'hyper-sphère et on aurait :

$$\overline{n_{HS}}(\epsilon) = k(\epsilon)^D$$

La valeur de a représente une estimation de la dimension intrinsèque des signatures. Notons que cette estimation est bien plus faible que la dimension intrinsèque que nous avions estimée à l'aide des partitions hiérarchiques de l'espace (cf. section 4.4.7). Nous avions alors trouvé $D' = 15,88$. La différence s'explique par le fait qu'il ne s'agit pas du même estimateur de la dimension intrinsèque. La première était basée sur un partitionnement hiérarchique de l'espace en blocs hyper-rectangulaires très volumineux tandis qu'il s'agit ici d'hyper-sphères centrées sur des signatures représentatives de la base. La résolution d'observation n'est donc pas la même. La première estimation était adaptée à la modélisaton de notre technique de recherche par requêtes statistiques tandis que cette deuxième estimation donne une valeur plus juste de la dimension intrinsèque réelle des données.

Pour les faibles valeurs du rayon de l'hyper-sphère on constate que le nombre moyen de signatures évoluent beaucoup moins rapidement et qu'il n'est pas nul lorsque ϵ est égal à zéro. Pour mieux comprendre pourquoi le nombre moyen de signatures similaires est aussi important dans le cas des faibles rayons, nous avons calculé, pour plusieurs valeur de ϵ, l'histogramme de $n_{HS}(\epsilon)$ (nous n'avions estimé pour l'instant que la valeur moyenne de $n_{HS}(\epsilon)$ sur les 10 000 signatures requêtes). Ces histogrammes n'étant pas vraiment analysables sous forme graphique, nous les présentons dans la table 6.13 pour quatre valeurs de ϵ et pour la base $REEL_{3500}$. Les valeurs sont données en pourcentage par rapport aux 10 000 signatures requêtes. La valeur de chaque cellule représente donc une estimation de la proportion des signatures de la base ayant $n_{HS}(\epsilon)$ signatures situées dans une hyper-sphère de rayon ϵ autour de celles-ci.

ϵ	proportion des signatures pour lesquelles					$\overline{n_{HS}}(\epsilon)$
	$n_{HS}(\epsilon) = 0$	$1 \leq n_{HS}(\epsilon) < 4$	$4 \leq n_{HS}(\epsilon) < 32$	$32 \leq n_{HS}(\epsilon) < 256$	$256 \leq n_{HS}(\epsilon)$	
0	96,8 %	0,9 %	1,4 %	0,3 %	0,6 %	5,97
30	82,9 %	9,2 %	4,1 %	2,0 %	1,8 %	113,17
50	72,3 %	12,5 %	8,4 %	3,5 %	3,3 %	207,58
100	3,9 %	14,2 %	27,3 %	22,4 %	32,2 %	4193,40

TAB. 6.13 – Répartition des signatures de la base en fonction du nombre de signatures contenues dans une hyper-sphère de rayon ϵ autour de celles-ci

Si l'on s'intéresse à la première ligne du tableau, on constate que $96,8$ % des signatures de la base n'ont en fait aucune signature identique, alors que le nombre moyen de signatures identiques dans la base est égal à $5,97$. La valeur élevée du nombre moyen de signatures identiques dans la base est donc uniquement due à un très faible pourcentage de signatures de la base qui sont présentes un très grand nombre de fois (jusqu'à 738 fois parmi les 10 000 signatures représentatives de la base $REEL_{3500}$). Nous avons visualisé une cinquantaine des séquences correspondant à ces signatures identiques et nous avons constaté qu'il s'agit à chaque fois de duplicata présents un très grand nombre de fois dans la base comme celui présentés sur la figure 6.20.
On vérifie donc bien que les signatures sont discriminantes puisque dans $96,8$ % des cas il n'existe aucune signature identique et dans $82,9$ % des cas il n'y a aucune signature similaire dans un rayon de $\epsilon = 30$. Presque toutes les signatures sont donc bien uniques. Les signatures qui ne sont pas uniques, sont en général présentes un très grand nombre de fois dans la base et correspondent à des duplicata dans le catalogue des séquences vidéo de référence.

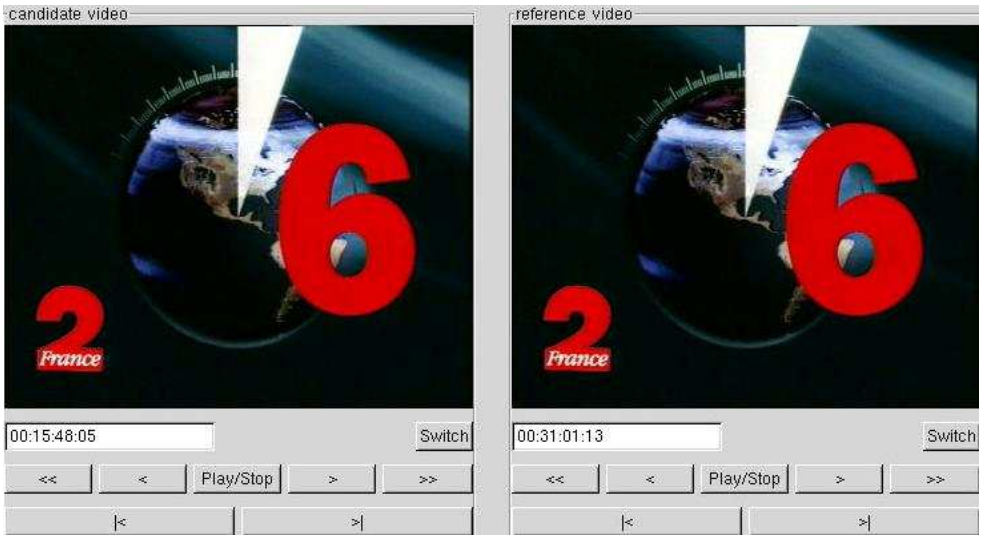


FIG. 6.20 – Exemple de duplicata présent un très grand nombre de fois dans la base

Pour les rayons supérieurs, nous avons vérifié en visionnant les séquences, que les signatures ayant un grand nombre de signatures dans l'hyper-sphère avoisinante ne correspondent pas forcément à des duplicata dans la base. Nous avons en revanche observé, en analysant les codes temporels des signatures comprises dans ces hyper-sphères très denses, que la redondance de ces signatures était due à la fois à une redondance intra-séquence et à une redondance inter-séquence.

Il s'agit donc de points d'intérêt qui sont présents de nombreuses fois à l'intérieur de la même séquence (points du décors par exemple) ou bien de points qui sont présents dans de nombreuses séquences (motifs textuels par exemple). Or, ce sont précisément ce genre de points d'intérêt qui contribuent à l'apparition des fausses alarmes de notre système (voir l'exemple du piano de la figure 6.11 ou les exemples de l'annexe H).

Pour étudier l'importance et l'influence de ces points redondants sur le nombre moyen de signatures contenues dans un rayon, nous avons construit les courbes présentées sur la figure 6.21. Elles représentent le nombre moyen de signatures contenues dans une hyper-sphère pour des valeurs croissantes du rayon, comme dans le cas de la figure 6.19. Sur les 10 000 signatures de la base utilisées pour calculer la valeur moyenne, nous avons cependant retiré celles qui avaient le plus grand nombre de signatures voisines à l'intérieur de l'hyper-sphère. Le pourcentage de signatures conservées est noté w_{keep}. En d'autres termes, nous avons estimé le nombre moyen de signatures contenues dans une hyper-sphère avoisinante que l'on aurait si l'on avait supprimé de la base les $(100 - w_{keep})$ % de signatures les moins rares de la base. La figure 6.21 montrent que l'élimination de ces signatures les moins rares, permet de diminuer considérablement le nombre moyen de signatures contenues dans une hyper-sphère avoisinante. Observons la valeur des courbes pour un rayon de $70,0$ qui est un bon ordre de grandeur de la norme moyenne du vecteur de distorsion pour une transformation moyennement sévère. On constate que le nombre moyen de signatures contenues dans un tel rayon pour la base entière vaut environ 630. En éliminant les 3 % de signatures les moins rares, il est égal à 100, puis à 12 en éliminant les 10 % les moins rares, puis à 2 en éliminant les 20 % les moins rares. On peut ainsi diviser très fortement la densité locale moyenne des signatures en n'éliminant qu'un faible pourcentage de signatures les moins rares.

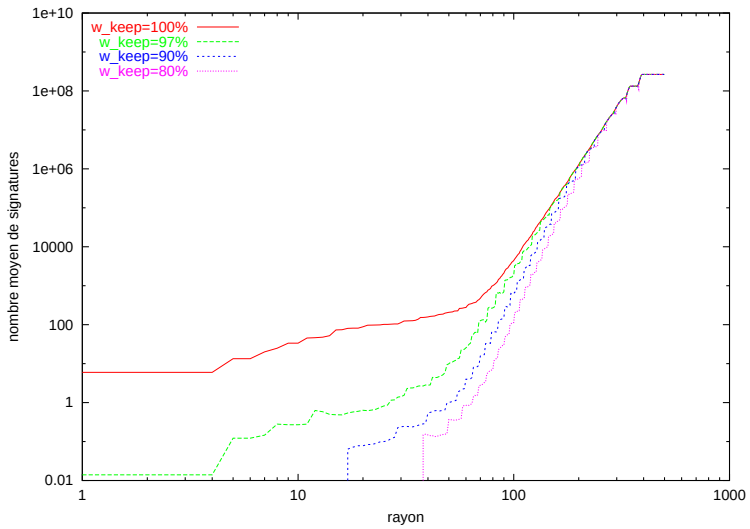


FIG. 6.21 – Nombre moyen de signatures dans une hyper-sphère avoisinante pour les w_{keep} % signatures de la base les moins redondantes

Cette expérimentation est très intéressante puisqu'elle montre que les signatures très redondantes sont en fait assez peu nombreuses bien qu'elles contribuent fortement à l'augmentation du nombre moyen de signatures contenues dans une hyper-sphère avoisinante. Il serait donc très judicieux de les éliminer soit en amont au niveau du détecteur de points d'intérêt, soit en aval en

les sélectionnant comme dans cette expérimentation. Nous n'avons pas eu le temps d'approfondir cette approche et de montrer son gain au niveau de l'efficacité globale de la détection mais il s'agit là d'un axe de recherche qui nous semble très pertinent.

6.6 Synthèse

Ce chapitre était dédié à l'évaluation expérimentale de notre méthode de détection de copies vidéo. Nous avons dans un premier temps montré l'efficacité de notre méthode de recherche par requêtes statistiques pour retrouver les signatures dans le cas de distorsions réelles. L'analyse des courbes opérationnelles de notre système sous l'influence des paramètres de la requête statistique a pour sa part démontré que la qualité de la recherche dans la base pouvait être sensiblement réduite au bénéfice du temps de recherche sans pour autant dégrader l'efficacité de la détection de copies. Les expérimentations de ce chapitre ont clairement montré que notre système de détection de copies restait robuste à des transformations assez sévères même pour des tailles de base très importantes contenant jusqu'à 10 000 heures de vidéo. Le double objectif d'un système robuste et rapide pour des très grandes tailles de base est ainsi pleinement atteint. Nous avons enfin validé l'applicabilité de notre système dans des cas réels en l'éprouvant avec deux vérités terrain représentatives des problématiques juridiques de l'INA. Dans le chapitre suivant, nous verrons que l'efficacité de notre méthode de détection de copies en terme de robustesse et de rapidité permet de réaliser le monitorage continu d'une chaîne de télévision avec des catalogues de référence pouvant atteindre jusqu'à 40 000 heures de vidéo.

Chapitre 7

Applications

Ce chapitre est consacré à la description de trois applications de notre méthode complète de détection de copies vidéo. La première application était l'objectif principal de la convention CIFRE dans le cadre de laquelle cette thèse s'est déroulée. Il s'agit du monitorage continu d'une chaîne de télévision afin de surveiller la diffusion d'extraits vidéo issus d'un catalogue d'archives télévisées de l'INA (section 7.1). Les deux autres applications sont apparues au cours de la thèse et n'ont pas été développées de manière aussi aboutie que l'application principale. Il s'agit de la recherche de liens dans une base d'archives (section 7.2) et de la recherche de liens entre plusieurs flux de télévision (section 7.3).

7.1 Application principale : monitorage de flux télévisés

7.1.1 Architecture du système de monitorage

L'architecture générale du système de monitorage est schématisée par la figure 7.1. On y distingue deux processus distincts : l'indexation et le monitorage, ainsi qu'une station de visionnage permettant de contrôler les résultats et éventuellement de sauvegarder les images correspondantes.

Le processus d'indexation consiste à construire des bases de signatures à partir des séquences vidéo MPEG1 de la base SNC de l'INA. Une batterie de PC chargent en continu les fichiers MPEG1 à travers le réseau et en extraient les signatures qui sont simplement stockées dans des fichiers de signatures. Le débit actuel d'extraction des signatures est d'environ 130 heures de vidéo par jour et par machine, lorsqu'aucun problème technique n'apparaît (coupures de courant, défaillance ou maintenance du réseau, défaillance ou maintenance des serveurs SNC, fichiers MPEG1 endommagés, défaillance d'un disque, etc.). Actuellement, 105 000 heures de vidéo ont d'ores et déjà été signées en une année et demi, avec en moyenne trois machines de calcul.
La construction d'une base indexée de signatures est effectuée par une machine distincte. Dans un premier temps, celle-ci fusionne le nombre souhaité de fichiers de signatures, soit de manière aléatoire, soit selon certains critères sur les séquences vidéo (séquences commandées par des diffuseurs, année de la première diffusion, type d'archives : parallèles antenne, cassettes montées, productions internes). Une fois la fusion effectuée, une base de signature indexée est créée par notre structure de recherche de signatures (tri des signatures selon leur position sur la courbe de Hilbert et création de la table d'index).

Le processus de monitorage schématisé sur la figure 7.1 (uniquement le cadre MONITORAGE)

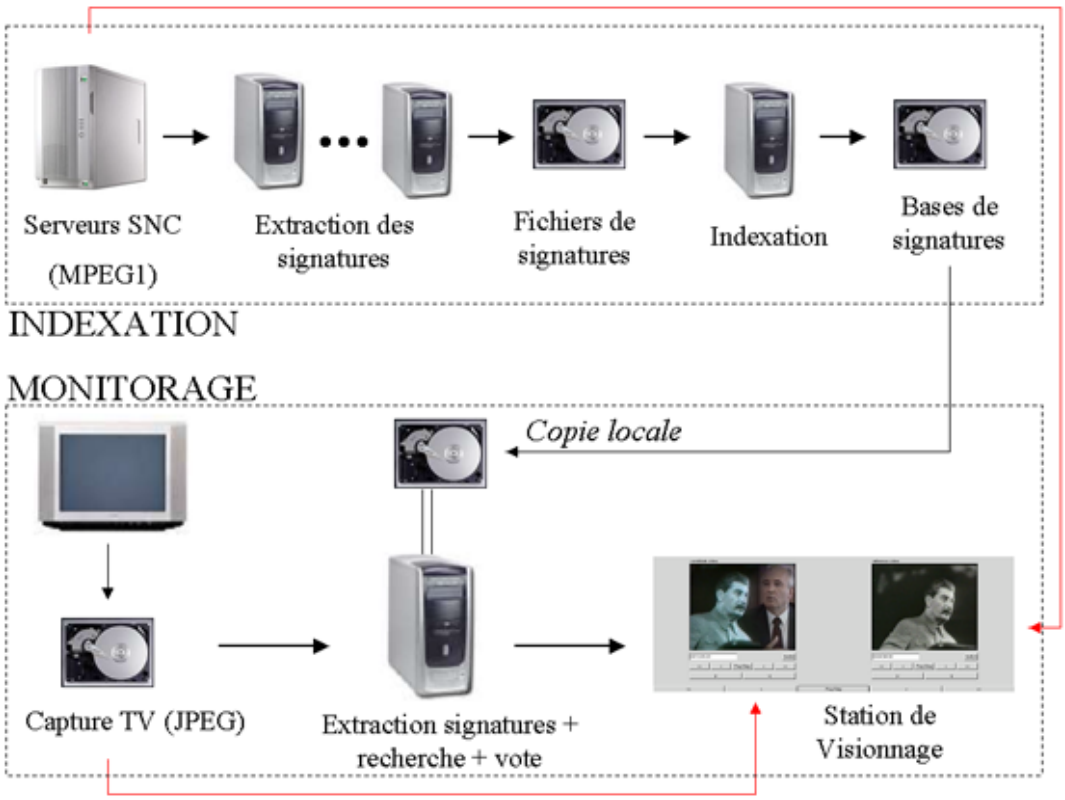


FIG. 7.1 – Architecture générale du système de monitorage

doit être répliqué pour chaque chaîne de télévision monitorée. La capture des flux vidéo télévisés est pour l'instant réalisée par une carte de capture montée sur un PC. A terme, notre système devrait accéder directement aux serveurs de l'INAthèque, qui est le service de l'INA chargé du dépôt légal de la télévision française et qui à ce titre, enregistre 24h/24 plus d'une cinquantaine de chaînes de télévision. Le calcul des signatures, la recherche des signatures dans la base et l'étape de décision globale sont réalisées par une seule machine qui fournit en sortie une liste de détections. La base indexée est copiée sur un disque locale de cette machine. Une interface graphique permet enfin de visionner les listes de détections en décodant les fichiers SNC et les images capturées à travers le réseaux (cf. annexe I pour une capture d'écran de cette interface).

7.1.2 Système de *pré-production* : monitorage temps réel des extraits commandés

Ce système est le premier mis en place et il est actuellement utilisé par le service juridique de l'INA. Il ne fonctionne qu'en mode par requête unique et ne peut donc traiter que des bases de signatures de taille limitée (jusqu'à environ 1 000 heures de vidéo). Les résultats sont en revanche fournis instantanément. Le choix des séquences référencées s'est porté sur tous les extraits qui ont été commandés par un diffuseur au cours des cinq dernières années. La base a ensuite été complétée par des extraits, dits *AGPE*, constitués par la numérisation des supports analogiques commandés au cours des dix dernières années.

La base correspondante contient environ 50 millions de signatures correspondant à 880 heures de vidéo.

Trois chaînes de télévision sont ainsi continuellement monitorées (TF1, France2, Histoire). Le nombre moyen de bonnes détections est égal à 303 par an et par chaîne. Le nombre moyen de fausses alarmes est égal à 28 par an et par chaîne. La précision du système est donc de l'ordre de 91,5 %.

7.1.3 Système expérimental : monitorage avec des catalogues de référence très volumineux

Fonctionnement

Ce système est pour l'instant maintenu par nos soins à titre expérimental sur une seule chaîne de télévision. Il est cependant en cours d'intégration dans un système de production. Il fonctionne en mode par requêtes groupées et permet d'utiliser des bases de signatures très volumineuses pouvant contenir jusqu'à 40 000 heures de vidéo. La limite exacte du nombre de signatures est pour l'instant égale à 2^{31} et est uniquement due à un problème logiciel.
Le mode de fonctionnement par requêtes groupées introduit un différé entre la capture temps-réel des images et la disponibilité des résultats de la détection. Nous avons vu dans la section 5.4, que plus la longueur de ce différé est importante et plus le coût moyen dû au chargement de la base en mémoire (lecture disque) est faible. Celui-ci peut même être maintenu à une faible fraction du temps de recherche moyen total, à condition de fixer judicieusement le nombre de signatures cumulées et donc la longueur du différé.
Dans le cas de ce système expérimental de monitorage, seule une chaîne de télévision est traitée par la machine. On est donc libre de ralentir la vitesse moyenne totale de la détection jusqu'au temps réel. On peut ainsi minimiser la longueur du différé de manière à ce que le coût moyen dû au chargement de la base soit égal à la fraction du temps réel encore disponible (les autres coûts étant le calcul des signatures, la recherche des signatures en mémoire et le calcul de la mesure de similarité globale).

Cela peut se faire automatiquement, de manière très simple, en déclenchant la recherche groupée des signatures dans la base dès que l'algorithme d'extraction et de cumul des signatures a rattrapé la capture temps réel des images. Le système va calibrer lui même le différé optimal au bout de quelques cycles de fonctionnement. Ce mode de fonctionnement est illustré par la figure 7.2, qui présente les premiers cycles de fonctionnement pour une base contenant environ 20 000 heures. A partir du 4^{eme} cycle, le nombre de signatures cumulées (et donc la longueur du différé) devient constant et le système converge vers une vitesse moyenne égale au temps réel. Sur cet exemple, la longueur du différé converge vers environ 50 minutes.
Notons que lorsque la vitesse moyenne de la détection est plus lente que le temps réel même sans compter le coût dû au chargement de la base, le système sera divergent et la longueur du différé augmentera indéfiniment.

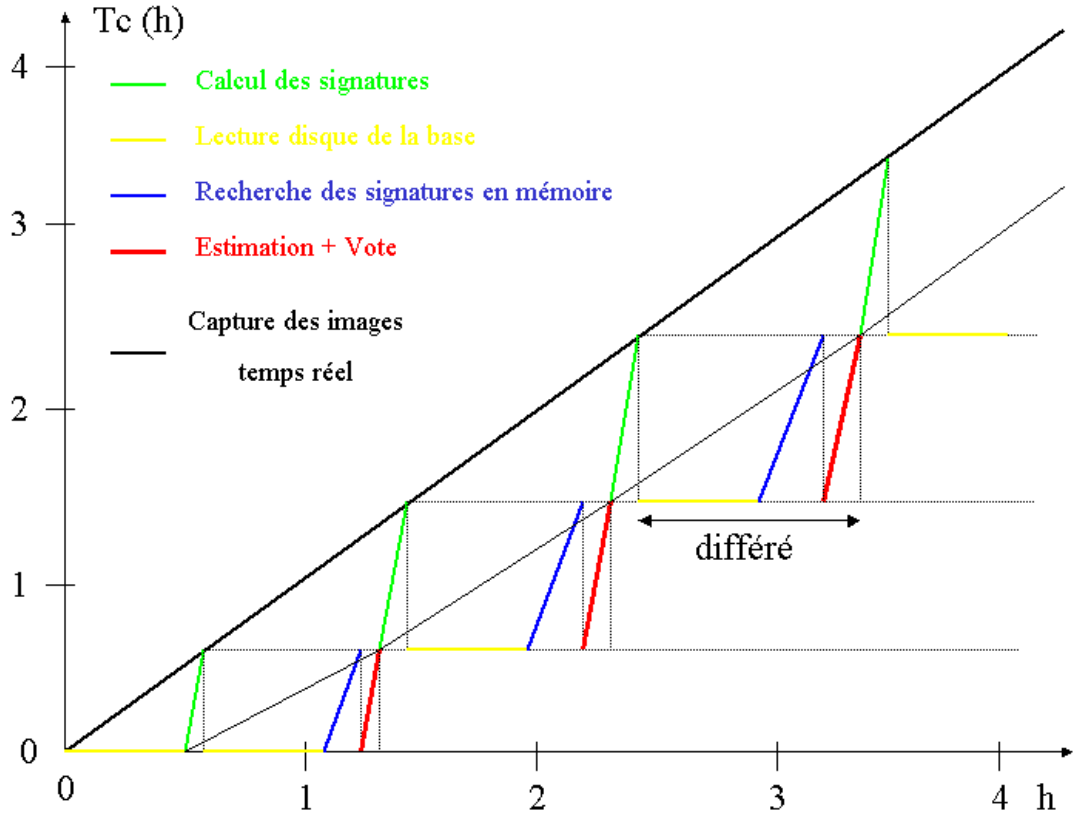


FIG. 7.2 – Convergence de la longueur du différé après quelques cycles de fonctionnement

Résultats

Le nombre de détections quotidiennes du système dépend fortement de la nature des séquences vidéo référencées.
Notre première approche consistait simplement à sélectionner aléatoirement les fichiers $MPEG1$ des serveurs SNC. Le nombre de bonnes détections sur une chaîne publique française et pour une base de référence de 30 000 heures est alors colossal (plusieurs milliers par jour). Il s'agit principalement de publicités, d'habillage de chaîne (clips de début et de fin de temps publicitaire, bande-annonces d'émission, etc.), d'habillages d'émissions (génériques, jingles, annonces de chroniques, etc.). Le nombre de détections est fortement accru par le fait que ce genre de séquence peut être présent un grand nombre de fois dans le catalogue de référence. Elles sont particulièrement fréquentes dans un type particulier d'archives que sont les parallèles antenne. Il s'agit d'enregistrements d'émissions réalisés lors de leur diffusion et qui inclut systématiquement quelques minutes de capture avant et après l'émission, collectant ainsi toutes sortes de séquences diffusées entre les émissions. Un tel nombre de détections rend les résultats difficilement exploitables et ne présente pas un grand intérêt immédiat tant qu'aucun post-traitement ne permettra de les classifier automatiquement. Cette constatation est fondamentale puisqu'elle montre que les résultats bruts obtenus par un système de détections de copie fonctionnant avec des catalogues de référence très volumineux peuvent être très bruités d'un point de vue applicatif. Il s'agit d'une matière première d'une grande valeur mais qui doit nécessairement être remaniée afin d'en tirer des informations pertinentes. Cela ouvre un large champs de recherche de type *data mining* que l'on peut justifier par quelques exemples intuitifs simples :

- une séquence détectée tous les jours à la même heure a de fortes chances d'être un générique.
- une séquence détectée entre 20H00 et 20H45 sur les trois principales chaînes françaises est probablement une image d'archive illustrant un évènement d'actualité.
- une séquence détectée de plus de 10 minutes est certainement la rediffusion intégrale d'une archive (un film ou un documentaire par exemple).

L'exploration des données pourrait également être enrichie par des informations externes comme les grilles de programme, les notices des séquences référencées, les données textuelles présentes dans les images, etc.

Pour réduire le nombre de détections quotidiennes, nous avons dans un deuxième temps construit des bases de signatures en éliminant les parallèles antenne du catalogue de référence. Les archives référencées sont alors majoritairement des cassettes montées d'émission, fournies par les diffuseurs avant leur diffusion et ne contenant que le contenu même de l'émission. Il ne s'agit là que d'une approche provisoire puisque les parallèles antenne peuvent contenir des séquences intéressantes du point de vue applicatif et que les archives restantes contiennent encore un certain nombre de séquences non pertinentes (génériques d'émission, cartes météo, habillage de journaux télévisés, etc.). Le nombre de détections quotidiennes est alors de l'ordre de quelques centaines si l'on compte toutes les séquences identifiées. Si l'on regroupe les détections d'une même séquence représentée plusieurs fois dans le catalogue de référence en une seule détection, le nombre de détections est alors de l'ordre de quelques dizaines dont certaines sont des rediffusions de la même séquence au cours de la journée. On peut considérer que le nombre de détections réellement pertinentes pour une application juridique est inférieure à 10 détections par jour. Des exemples de détections intéressantes du point de vue de la robustesse du système sont présentées en annexe G.

Le nombre de fausses alarmes quotidiennes est de l'ordre de quelques dizaines dont beaucoup correspondent à une même séquence candidate ayant été appariée avec plusieurs séquences du catalogue de référence. Le nombre quotidien de séquences candidates conduisant à des fausses alarmes est plutôt compris entre 1 et 5. Il s'agit principalement d'images contenant beaucoup de textes (textes défilant à la fin des films, photographies de journaux) et d'images contenant plusieurs fois un même motif très contrasté (pianos, chemises à carreaux, etc.). Des exemples de fausses alarmes sont présentées en annexe H.

7.2 Recherche de liens dans une base d'archives

La recherche de liens dans une base d'archives constitue une autre application de notre système de détection de copies vidéo. Le principe consiste simplement à considérer les séquences du catalogue de référence comme séquences candidates et à en rechercher toutes les éventuelles copies par l'intermédiaire de notre système. Il n'est cependant pas nécessaire de recalculer les signatures des séquences candidates puisque celles-ci ont déjà été calculées. Plusieurs objectifs majeurs sont visés par une telle application :

- **Suppression des dupplicata :** Certaines archives sont involontairement indexées plusieurs fois dans la base de l'INA. Outre le surcoût de stockage qu'elles représentent, elles peuvent également avoir des notices différentes (complémentaires ou conflictuelles) qu'il serait intéressant de fusionner.
- **Débruitage de la base :** Certains extraits contenus dans la base un grand nombre de fois n'ont pas un contenu informatif pertinent. Il peut s'agir de plans inter-séquences complète-

ment noir, de mires, de publicités, d'habillages de chaîne, de compte à rebours, etc. Outre le surcoût de stockage qu'ils représentent, ils peuvent également polluer tout traitement automatique basé sur le contenu (en particulier le monitorage d'une chaîne de télévision, comme nous l'avons vu auparavant). Un exemple de détection d'un compte-à-rebours est présenté sur la figure 7.3.

FIG. 7.3 – Exemple de détection d'un compte-à-rebours

– **Indexation de liens inter-émissions :** Certaines séquences vidéo de la base sont présentes dans plusieurs archives distinctes. Leur détection permet d'établir un lien qui peut avoir une signification de haut-niveau sémantique. La figure 7.4 montre l'exemple d'un tel lien. Il s'agit d'une courte séquence représentant une éruption volcanique qui a été ré-exploitée dans deux émissions différentes ayant pour thème la vulcanologie. Ce type de liens pourrait s'avérer très utile à la navigation dans la base. Il permet par exemple de lier une séquence d'actualité originale à toutes ses ré-exploitations successives au cours des années. Notons également que l'existence de plusieurs ré-exploitations est en elle même une information pertinente pour la valeur de l'extrait. On pourrait par exemple imaginer des requêtes visant à retrouver les images les plus rediffusées au cours du siècle dernier.

7.3 Recherche de liens intra-chaîne et inter-chaîne

Le principe de cette application est de détecter la rediffusion d'extraits vidéo par une même chaîne de télévision (liens intra-chaîne) ou de détecter la diffusion d'un même extrait vidéo sur plusieurs chaînes de télévision (liens inter-chaîne). Notons qu'une telle application pose en réalité une problématique distincte de celle de la détection de copies dans un catalogue de référence. Toutes les données sont en effet des flux de signatures tandis que notre structure ne permet de rechercher des flux que dans des bases de signatures statiques. On ne peut donc rechercher des liens que de manière statique en construisant une base de signatures à partir d'une capture de plusieurs chaînes limitée dans le temps.
Le principe de notre système expérimental est simplement de calculer les signatures d'une semaine de télévision sur plusieurs chaînes (une machine par chaîne) et d'en construire une base indexée. Les fichiers de signatures de chaque chaîne peuvent ensuite être recherchés dans la base

FIG. 7.4 – Exemple d'un lien entre deux émissions archivées

afin de trouver tous les liens existant dans cette semaine de capture. Notre système de détection de copies étant beaucoup plus rapide que le temps réel sur des petites bases, une seule machine permet d'effectuer en une semaine, l'indexation de la base et la recherche de plusieurs chaînes (jusqu'à environ 5 chaînes). Cette opération peut ensuite être réitérée toutes les semaines. On peut faire croître la base de référence sur plusieurs semaines si l'on souhaite détecter des liens à cheval sur plusieurs semaines.

La détection de liens intra-chaîne et inter-chaîne ouvre des perspectives intéressantes en termes d'indexation automatique ou semi-automatique. L'objectif est d'extraire des informations relatives au contexte de la ré-exploitation des documents. Ces informations peuvent s'avérer d'un niveau sémantique plus élevé qu'une analyse automatique du contenu de l'image. On peut donner à titre illustratif quelques exemples de concepts calculables automatiquement :

- **Persistance d'un événement :** La persistance dans le temps de la rediffusion d'une image d'actualité peut donner une idée de son impact médiatique.
- **La dispersion géographique :** Une séquence diffusée sur plusieurs chaînes de nationalités

différentes possède certainement un caractère international.

- **Analyse fréquentielle des instants de rediffusion :** Les publicités, les génériques d'émissions, les habillages d'émission, les événements d'actualité sont autant de séquences vidéo rediffusées plusieurs fois au cours d'une même semaine (ou d'une même journée) mais dont les instants de rediffusion n'ont pas forcément le même comportement fréquentiel.

Les caractéristiques que l'on peut extraire de l'ensemble des liens de rediffusion sont nombreuses et potentiellement très pertinentes. Nous pensons qu'une approche de type *data mining* serait tout à fait utile à l'exploration de ces données. Les quelques premiers tests que nous avons fait sur deux chaînes de télévision (France2 et TF1) ont montré que le nombre d'extraits rediffusés était très important. En moyenne, près de 850 extraits sont diffusés plus d'une fois en quatre jours sur une même chaîne (avec un nombre moyen de diffusions égal à $4,35$). Près de 110 extraits en moyenne, sont diffusés sur les deux chaînes sur une durée de 4 jours (Presque exclusivement des publicités). La classification automatique de ces détections est donc primordiale pour exploiter pleinement ces résultats bruts.

La figure 7.5 présente un exemple de détection inter-chaîne intéressant. Il s'agit d'une séquence dans laquelle le visage du personnage de la scène a été flouté sur l'une des chaînes et pas sur l'autre.

FIG. 7.5 – Exemple d'une détection inter-chaîne

La figure 7.6 présente un exemple de détection intra-chaîne. Il s'agit d'un extrait d'une émission, qui a été utilisé quelques minutes avant l'émission comme bande-annonce. Nous avons choisi cette détection pour montrer qu'une rediffusion n'est pas forcément une copie exacte.

7.4 Synthèse

Trois applications de notre technique de détection de copies vidéo ont été présentées dans ce chapitre. L'application principale, le monitorage continu d'une chaîne de télévision, est déjà pleinement utilisée au sein de l'INA dans un contexte de surveillance juridique. Des catalogues d'archives très volumineux, contenant jusqu'à 40 000 heures de vidéo, peuvent ainsi être surveillés continuellement sur une chaîne de télévision. Les expérimentations des trois applications

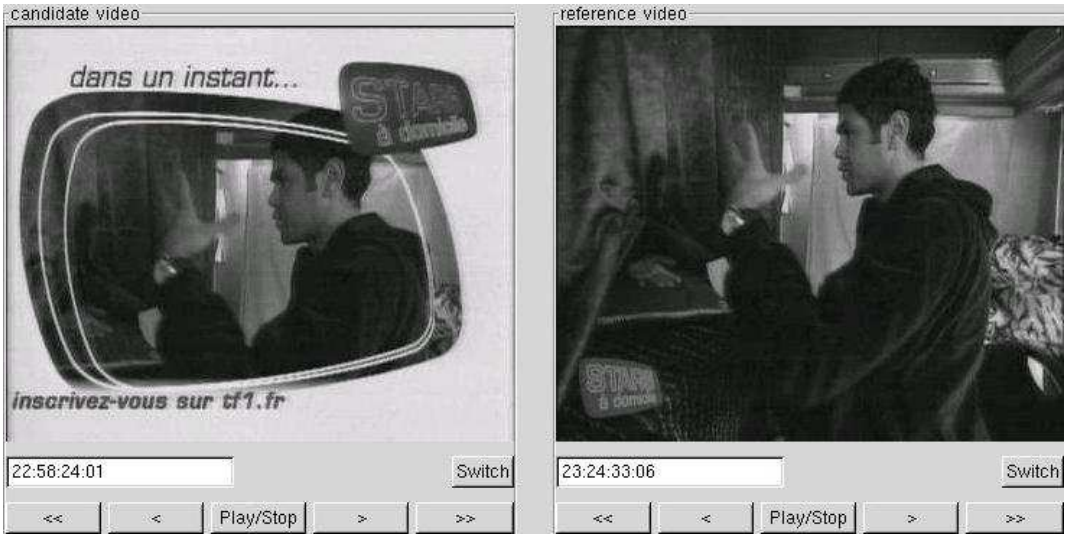


FIG. 7.6 – Exemple d'une détection intra-chaîne

présentées dans ce chapitre ont montré que la ré-exploitation des documents télévisés n'était pas le fait de quelques événements isolés mais bien une propriété omniprésente du mode de diffusion télévisé. La réutilisation des images a même tendance à s'accentuer au cours des années.
Les trois applications envisagées dans ce chapitre permettent de détecter les rediffusions de documents dans trois contextes de natures différentes et complémentaires :

- Détection de rediffusions archivées (recherche de liens dans une base d'archives)
- Détection de la diffusion d'archives dans un flux télévisé (monitorage)
- Détection de rediffusions entre plusieurs flux télévisés (détections inter-chaîne et intra-chaîne).

L'ensemble de ces ré-exploitations constitue une mine d'information très riche ouvrant un large champs d'applications de type exploration de données (*data mining*).

Conclusion générale

Les travaux de cette thèse traitent du problème de la détection de copies basée sur le contenu. Pour résoudre conjointement les problèmes de qualité et de rapidité de la détection, liés à l'augmentation de la taille du catalogue de référence, nous avons proposé une méthode complète originale et efficace. Celle-ci tient compte à la fois des aspects traitement de l'image, des aspects base de données et de leurs interactions. L'approche proposée a été complètement intégrée et validée dans un système applicatif réel et a permis de soulever des questionnements nouveaux relatifs à l'utilisation de catalogues de référence aussi volumineux et à l'essence même des différents processus de ré-exploitation d'un document. Dans cette conclusion, nous faisons d'abord une synthèse des travaux présentés dans ce mémoire de thèse (section 1). Nous récapitulons ensuite les conclusions majeures de nos travaux (section 2) afin d'en tirer des perspectives pour les recherches futures (section 3).

1 Synthèse des travaux

Nous avons d'abord situé la problématique de la détection de copies par le contenu dans le cadre plus général de la recherche d'images et de vidéos par le contenu. A partir d'un panorama de ce domaine, nous avons ensuite précisé les spécificités de la détection de copies. Les descripteurs utilisés dans le cas général ne sont pas directement applicables à la détection de copies et l'utilisation de métriques complexes rend leur usage inadapté au contexte temps réel de la traçabilité d'un catalogue de séquences vidéo. Nous avons enfin soulevé le problème de la croissance du catalogue de référence, souvent oublié, et pourtant capital pour l'évolution des performances. La majorité des méthodes dédiées à la détection de copies n'abordent également pas cet aspect. Beaucoup sont inadaptées aux problèmes de redécoupage temporel, de décalage des images et d'incrustations, qui sont très fréquents dans le cas des images diffusées à la télévision. L'identification d'extraits multiples, présents plusieurs fois dans le catalogue de référence est également rarement résolu.

L'originalité de notre approche est qu'elle est basée sur l'utilisation conjointe de signatures locales et d'une mesure de similarité globale, calculée après la recherche des signatures similaires dans la base. Cette mesure globale repose uniquement sur la position des signatures locales et non sur les signatures elles mêmes. Son calcul a été traité comme un problème d'estimation robuste du recalage temporel entre la séquence candidate et les séquences retrouvées par la recherche dans la base de signatures. Une équation de minimisation originale prenant en compte la nature particulière des observations a été proposée. Chaque signature candidate possède en effet un ensemble de signatures similaires qui ne peuvent être considérées comme un ensemble d'observations de contributions égales.

Dans le domaine de la recherche rapide de descripteurs par similarité, les structures d'indexation multidimensionnelle les plus utilisées voient leurs performances se dégrader rapidement lorsque la dimension des données dépassent $D = 16$. En dimension très élevée, nous avons vu que certaines techniques récentes permettaient d'être toujours plus rapide qu'une recherche séquentielle dans la base mais malgré l'utilisation de ce type de structures, le temps moyen d'une recherche reste trop coûteux pour beaucoup des applications multimédia émergentes. Le besoin de rapidité supplémentaire a conduit au paradigme de recherche approximative dont le principe est d'échanger une forte accélération contre une faible perte dans la qualité des résultats. Ce type d'approches est actuellement exclusivement dédiées à des requêtes de type K-plus proche voisins.
Après avoir justifié la non-pertinence de ce type de requêtes dans le cadre de notre recherche de signatures locales similaires, nous avons introduit notre nouveau paradigme de recherche approximative : la recherche par requêtes statistiques. Celle-ci peut être vue comme une extension du paradigme de recherche approximative aux requêtes géométriques à ϵ-près. L'idée est de s'abstenir de toute contrainte géométrique de la requête en recherchant directement les signatures indexées les plus pertinentes au sens d'un modèle probabiliste. Nous avons montré que ce type de requêtes permettait de conserver la même qualité de recherche qu'une requête géométrique exacte à ϵ-près, tout en réduisant très fortement les temps de recherche (jusqu'à 150 fois plus rapide).

La méthode proposée pour traiter ce type de requête profite du contexte statique de notre application. Les signatures sont ordonnées physiquement selon une courbe de Hilbert ce qui évite de parcourir une structure d'indexation pour accéder à la base. La recherche elle-même n'est en fait qu'un parcours séquentiel de certaines parties de cette base triée de signatures. Ces intervalles sont déterminés par un ensemble de blocs de l'espace partitionné selon leurs probabilités de contenir des signatures pertinentes au sens du modèle de distorsion de la requête statistique. Nous avons proposé une modélisation du coût de recherche de notre technique qui nous a permis de définir une valeur optimale de son principal paramètre, la profondeur de la partition, et également de prévoir l'influence de l'augmentation de la taille de la base.
Deux modes de fonctionnements ont été proposés. Le premier mode ne fonctionne que si la base peut être entièrement stockée en mémoire principale. Nous avons vu que dans ce cas la complexité de la recherche évoluait en $O(N^\gamma)$ et était donc sous-linéaire en fonction de la taille de la base. Un mode de fonctionnement par requêtes groupées a été proposé pour repousser la limite de la taille des bases beaucoup plus loin que la taille mémoire. La recherche des signatures dans la base n'est déclenché que lorsque suffisamment de signatures candidates ont été cumulées pour rentabiliser un chargement complet de la base en mémoire. Celle-ci est chargée par pages successives et la totalité des signatures cumulées est recherchée dans chacune des pages. Cette technique permet d'utiliser des bases jusqu'à 32 fois plus grandes que la taille mémoire disponible sans dégrader significativement les temps de recherche obtenus en mode mémoire par requêtes uniques. Sur des bases très volumineuses contenant plus d'un milliard de descripteurs, la méthode que nous proposons est en moyenne jusqu'à 2500 fois plus rapide qu'une recherche séquentielle dans la base.

Des expérimentations variées et à grande échelle nous ont permis de valider l'efficacité de notre méthode et d'analyser l'interaction entre la recherche statistique des signatures et la robustesse globale du système. Notre système est robuste aux transformations les plus fréquemment observées pour une durée minimale de diffusion d'environ 4 secondes et un taux de fausses alarmes faible (moins d'une dizaine par jour lors du monitorage d'une chaîne de télévision). Le taux de compression est proche de 6600 par rapport au flux vidéo non compressé. Notre système répond à la contrainte de l'augmentation du catalogue de référence à la fois en termes de rapidité et en

termes de qualité de la détection. La recherche statistique des signatures est très rentable puisqu'elle permet de retrouver une grande majorité des signatures en réduisant considérablement le temps de recherche. La perte des signatures non retrouvées n'a de plus qu'un impact limité sur l'efficacité globale de la détection de copies.

Enfin, des applications concrètes ont été présentées dont la principale, le monitorage continu d'une chaîne de télévision, est déjà pleinement utilisée au sein de l'INA dans un contexte de surveillance juridique. Des catalogues d'archives très volumineux, contenant jusqu'à 40 000 heures de vidéo peuvent ainsi être surveillés continuellement sur une chaîne de télévision. Les résultats de l'application de notre technique dans des contextes réels ont montré que la ré-exploitation des documents télévisés n'était pas le fait de quelques événements isolés mais bien une propriété omniprésente du mode de diffusion télévisée.

2 Conclusions majeures

Nous donnons ici une liste des conclusions de nos travaux qui nous semblent les plus pertinentes :

1. Dans le cadre de la recherche par similarité dans une base de signatures locales,
   - le nombre de signatures pertinentes par requête peut être très variable et une requête de type K-plus proches voisins n'est pas appropriée.
   - la contrainte géométrique d'une requête à ϵ-près nuit aux performances de la recherche sans apporter d'amélioration pour la qualité des résultats.
   - l'utilisation de requêtes statistiques permet d'adapter la recherche aux formes englobantes des structures d'indexation et de réduire considérablement le temps de recherche.
   - un modèle probabiliste simple permet d'avoir un contrôle suffisant de la qualité de la recherche (et en tous cas meilleur qu'une mesure géométrique de l'erreur basée sur le volume).
2. L'utilisation conjointe de signatures locales et d'une mesure de similarité globale calculée après la recherche dans la base permet
   - de calculer une mesure de similarité robuste et algorithmiquement complexe sans altérer les performances de la recherche dans la base.
   - d'effectuer une recherche approximative dans la base des signatures sans dégrader directement l'efficacité globale de la détection.
   - de limiter fortement l'influence de la taille de la base sur l'efficacité de la détection.
   - d'être robuste aux transformations les plus fréquentes et en particulier aux incrustations et aux décalages d'images.
3. Dans le cadre de l'utilisation d'une mesure de similarité globale basée sur la position des signatures locales,
   - la répétabilité du détecteur de points d'intérêt est fondamentale pour l'efficacité de la détection tandis que l'invariance des descripteurs locaux est pour sa part décisive pour le coût de la recherche des signatures dans la base.
   - les fausses alarmes et les détections imprécises sont principalement dues à la présence de nombreux points d'intérêt très similaires dans une même image ou une même séquence.
4. L'analyse de notre base de signatures locales a montré que la majorité des signatures sont très discriminantes et que seule un faible pourcentage de signatures sont très redondantes.

On peut ainsi diviser la densité locale moyenne des signatures par 200 en éliminant seulement les 20 % de signatures les plus redondantes. Ces signatures redondantes sont nuisibles à la fois pour l'efficacité de la détection (fausses alarmes, divergence de l'estimation) et pour la rapidité des algorithmes de recherche et d'estimation des solutions.

5. Les résultats des applications réelles de notre système de détection de copies ont montré que,
   - la ré-exploitation et la rediffusion d'images sont des processus omniprésent de la diffusion télévisée.
   - la traçabilité des ré-exploitations d'un document est une information à part entière dont l'exploration a un fort potentiel d'analyse sémantique.
   - l'utilisation de catalogues de référence très volumineux engendre des problématiques nouvelles pour l'exploitation des résultats. Les détections sont en particulier très nombreuses et la plupart d'entre elles ne sont pas pertinentes pour une application juridique.

3 Perspectives

Les travaux de cette thèse ont mis en lumière plusieurs perspectives de recherche pour de futures investigations :

1. **Détecteur de points d'intérêt spatio-temporels :** Les points d'intérêt spatio-temporels possède une information temporelle locale qui leur est propre et non globale comme dans le cas des images clé. Il s'agit d'évènements temporels locaux rares et précis en position. Une signature locale extraite autours d'un tel point possédera un fort contenu informatif dans les trois dimensions de la vidéo. L'utilisation d'un détecteur de points d'intérêt spatio-temporels devrait permettre de limiter la redondance des signatures au sein d'une même séquence et d'améliorer le taux de compression. Une amélioration éventuelle de la répétabilité grâce à la composante temporelle est également une piste très prometteuse.
2. **Élimination des signatures locales les moins rares :** En post-traitant l'ensemble des signatures locales extraites dans un même extrait vidéo, on peut envisager d'éliminer le faible pourcentage des signatures les moins rares qui perturbent l'efficacité et la rapidité de la détection. On pourra également étudier la possibilité de post-traiter directement la base de signatures.
3. **Sur-indexation :** Lorsque le temps de recherche des signatures est sous-linéaire en fonction de la taille de la base, une sur-indexation des signatures peut s'avérer très rentable. L'idée est de calculer les signatures sur plusieurs versions transformées des séquences de référence. Outre le fait de se rendre robuste à des transformations plus sévères, cela pourrait permettre de dimimuer encore plus la qualité de la recherche dans la base afin d'accélérer globalement le processus de détection.
4. **Généralisation des requêtes statistiques à d'autres structures :** Il serait intéressant d'évaluer les requêtes statistiques dans d'autres structures d'indexation multidimensionnelle. La principale difficulté est de trouver des algorithmes de filtrage efficaces pour des régions englobantes de formes différentes et de gérer le problème des recouvrements de région.
5. **Généralisation des requêtes statistiques à des modèles plus complexes :** Pour enrichir la modélisation de la distorsion des signatures, il faudra d'abord trouver des algorithmes de filtrage efficaces ne nécessitant pas l'indépendance des composantes.

6. **Estimation coarse-to-fine du recalage :** Le calcul de la mesure de similarité globale de notre méthode repose sur l'estimation robuste du recalage de la séquence candidate par rapport aux séquences retrouvées par la recherche des signatures locales. Pour sa mise en œuvre nous n'avons pour l'instant estimé que le recalage temporel et non le recalage spatial. L'ajout des coordonnées spatiales contribuera indéniablement à améliorer l'efficacité de la détection mais l'estimation sera plus coûteuse. De plus, nous avons vu que le nombre de signatures locales similaires augmente sensiblement avec la taille de la base et cela contribue également à alourdir le coût de l'estimation. Cette étape va certainement devenir le nouveau goulet d'étranglement de notre méthode et il faudra développer des algorithmes pour l'accélérer. Nous envisageons ainsi de développer une approche *coarse-to-fine* qui raffinera récursivement la granularité (spatiale et temporelle) de l'estimation.
7. **Estimation des transformations :** En plus du recalage temporel et géométrique, on pourrait envisager d'estimer d'autres types de transformation à partir des signatures locales similaires issues de la recherche dans la base. La détection d'incrustations ou l'estimation de certaines transformations colorimétriques sont par exemple envisageables à condition d'avoir plus de points d'intérêt et des signatures locales plus riches que notre système actuel. L'estimation des transformations constituent une caractérisation supplémentaire de la ré-exploitation d'une séquence vidéo. D'une manière générale, la traçabilité d'un document audiovisuel peut ainsi se résumer par la question : où, quand et sous quelle forme un document est-il exploité depuis sa production ?
8. **Exploration des résultats :** Les résultats bruts de la traçabilité d'un ensemble d'extraits vidéo peuvent s'avérer très informatifs pour des applications de type indexation automatique ou macro-segmentation de flux télévisés. Une première piste serait de réaliser une classification des données brutes (fréquence de diffusion, nombre de diffusions, chaîne TV, longueur des extraits diffusés, existence d'archive de l'extrait) et d'analyser les séquences vidéo regroupées dans un même ensemble.

Annexe A

Exemple de trois systèmes de recherche d'images fixes par le contenu

A.1 QBIC (Query By Image Content)

Le système QBIC [132] utilise des caractéristiques de couleur et de forme. Les descripteurs de couleur sont constitués d'histogrammes et du vecteur couleur 3D moyen d'un objet ou d'une image entière dans différents espaces colorimétriques (RGB, YIQ, Lab, Munsell). Les descripteurs de forme consistent en l'aire, la circularité, l'excentricité, les axes d'orientation principale et un ensemble de moments algébriques invariants.
Le système QBIC permet de traiter des requêtes par image-exemple, par construction d'ébauches graphiques et par sélection de couleurs ou de texture parmi un panel proposé à l'utilisateur.
Pour les vecteurs couleur moyens, ainsi que les caractéristiques de forme, la distance utilisée est la distance de Mahalanobis. Dans le cas des histogrammes, deux distances sont utilisées, une de dimension faible, facile à calculer (distance moyenne) et une plus complexe (distance quadratique entre histogrammes). La première fonctionne comme un filtre limitant le nombre de candidats avant de calculer la deuxième.
En ce qui concerne l'indexation, QBIC a été l'un des premier systèmes à avoir intégré une indexation multidimensionnelle pour accélérer les performances, en l'occurrence un $R*$-tree (cf. partie II, [10]). Certains descripteurs subissent préalablement une réduction de dimension par une analyse en composantes principales (ACP).

A.2 Ikona

Le système Ikona [28] permet de faire des requêtes par image-exemple ou par région-exemple. Il inclut également un mode de recherche hybride texte-image. Dans le mode région, c'est l'utilisateur qui sélectionne une région. Une méthode de bouclage de pertinence permet également à l'utilisateur de trouver rapidement une certaine catégorie d'images.
Pour les requêtes globales, Ikona utilise des histogrammes pondérés qui caractérisent l'activité locale des divers aspects visuels que l'on veut décrire (en particulier la couleur et la texture).
Lorsque la recherche visuelle porte sur une région de l'image, deux approches locales et complémentaires sont développées : requêtes partielles par régions d'intérêt (zones visuellement homogènes, par exemple champ de lavande, peau, etc.) et requêtes partielles par points d'intérêt couleur présents dans les zones de l'image à forte variation locale.
Ikona comprend également un système de navigation, qui propose un résumé de la base en ras-

semblant les images similaires par catégorie. Pour chaque catégorie, l'image la plus représentative est choisie pour figurer dans le résumé.

A.3 Blobworld

Dans le système BlobWorld [39] les caractéristiques utilisées sont la couleur, la texture, la position et la forme de certaines régions (les *blobs*) et du fond. La couleur est décrite par un histogramme de grande dimension et les coordonnées des vecteurs couleur dans l'espace Lab. La texture est représentée par le contraste moyen et l'anisotropie dans les régions (vecteur 2D). La forme est représentée par la surface (approximative), l'excentricité et l'orientation.
L'utilisateur sélectionne une catégorie d'images qui limite la recherche. Dans une image initiale, il sélectionne ensuite une région et indique son importance (un peu, beaucoup, etc.). Enfin, il indique l'importance relative de la couleur, de la forme, de la texture et de la position.
Après une réduction de la dimension, les histogrammes sont comparés par une distance quadratique :

$$d_q(\boldsymbol{H_1}, \boldsymbol{H_2}) = (\boldsymbol{H_1} - \boldsymbol{H_2})^t \boldsymbol{U} (\boldsymbol{H_1} - \boldsymbol{H_2})$$

où $\boldsymbol{U}$ est une matrice symétrique définie positive.
La distance utilisée pour comparer les caractéristiques de texture et les centroïdes est la distance euclidienne. La structure d'indexation utilisée est un $R*$-tree (cf. partie II, [10]).

Annexe B

Evaluation du détecteur de Harris

Les transformations habituellement utilisées pour évaluer la répétabilité des détecteurs de points d'intérêt (zoom de la caméra, rotation de la caméra, changements d'illumination de la scène réelle, changements de points de vue) ne correspondent pas tout à fait aux transformations rencontrées dans le cadre de la détection de copies. Dans cette session expérimentale, nous avons testé quatre transformations spécifiques à la détection de copies (cf. section 6.1.3) :

- Redimensionnement de l'image
- Changement du gamma
- Ajout de bruit gaussien
- Changement de contraste

Nous n'avons pas évalué la robustesse à la rotation et aux transformations affines qui sont très rares dans notre contexte applicatif.
Pour mesurer la robustesse du détecteur aux transformations citées, qui sont de la forme $I' = T(I)$, on utilise un critère de répétabilité correspondant au pourcentage de points détectés dans l'image d'origine I que l'on retrouve dans l'image transformée I' à une erreur de position près. On parle alors d'ε-répétabilité, pour une erreur de position tolérée de ε pixels. Cette erreur n'est pas forcément entière puisque dans le cas de transformations géométriques de l'image comme le redimensionnement, les positions réelles doivent être reconverties dans le système de coordonnées de l'image d'origine, via le facteur de redimensionnement, pour pouvoir comparer les positions. Dans toutes les expérimentations suivantes, nous avons fixé $\varepsilon = 1,5$, ce qui dans le cas des transformations non géométriques correspond à tolérer tous les pixels voisins du pixel d'origine. Les résultats qui suivent ont été obtenu en moyennant les résultats sur 106 images non consécutives issues de 5 séquences différentes initialement codées en MPEG1. La taille des images est de 352×288 et les niveaux de gris sont codés sur 8 bits, c'est-à-dire $I(x, y) \in [0, 255], \forall (x, y)$. Le nombre maximum de points retenus est fixé à 20. Le seuil sur la réponse du détecteur dépend de la valeur de σ_h et a été estimé en prenant la réponse maximum sur une image contenant uniquement un bruit gaussien d'écart type 30.

1. *Répétabilité au redimensionnement*
   La figure B.1 représente le critère de répétabilité au redimensionnement de la version stabilisée du détecteur de Harris **sans l'algorithme de relocalisation** et pour différentes valeurs de σ_h.

   On constate bien que le répétabilité du détecteur se dégrade lorsque σ_h augmente. Cela est dû au fait que la délocalisation est croissante avec σ_h. Pour les petites échelles d'analyse, on voit que le détecteur de Harris possède une bonne répétabilité pour les redimensionne-

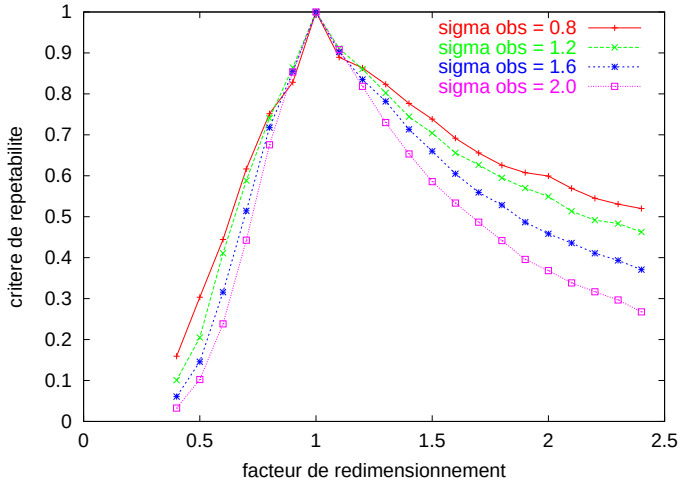


FIG. B.1 – Répétabilité du détecteur de Harris au redimensionnement de l'image

ments de faible amplitude (plus de 70 % des points sont encore présents pour des facteurs de redimensionnement compris entre 75 % et 150 %).

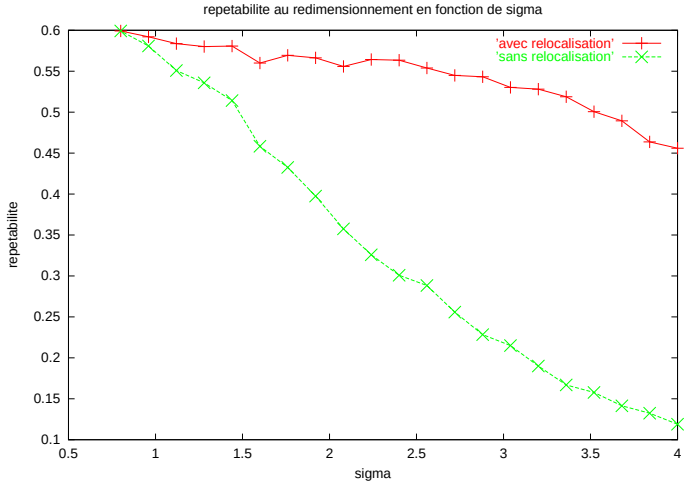


FIG. B.2 – Effet de la relocalisation sur la répétabilité à un redimensionnement de facteur 2,0

La figure B.2 montre l'amélioration de la répétabilité grâce à la relocalisation des points d'intérêt, pour un redimensionnement fixe de facteur $2,0$, en fonction de σ_0 pour la courbe avec recentrage, et de σ_h pour la courbe sans recentrage. On s'aperçoit que l'effet attendu est atteint puisque la répétabilité décroît beaucoup moins rapidement lorsque σ_0 augmente.

2. *Répétabilité aux transformations non géométriques*
   Les figures B.3 et B.4 représentent la répétabilité du détecteur de Harris stabilisé avec relocalisation, à un changement du gamma et à un ajout de bruit gaussien, pour plusieurs valeurs de σ_0 (échelle d'analyse de départ). On vérifie bien que la répétabilité est moins

bonne lorsque l'échelle d'analyse de départ est plus faible. On constate également que dans le cas où l'échelle d'analyse de départ est suffisamment grande, la répétabilité à ce genre de transformations est très bonne. On conserve par exemple 70 % des points pour un bruit gaussien d'écart type $23,0$ ou un changement de gamma de facteur $\gamma = 0,3$.

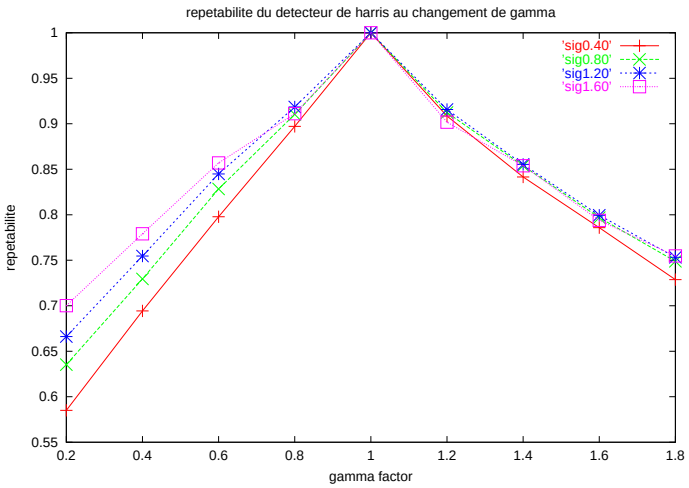


FIG. B.3 – Répétabilité à un changement du gamma

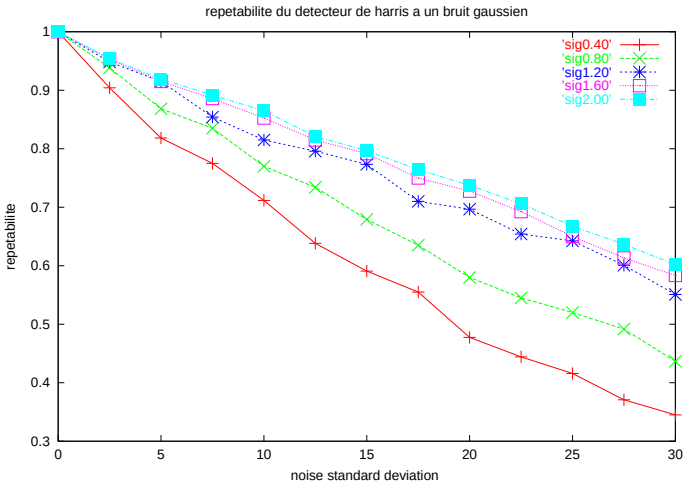


FIG. B.4 – Répétabilité à un ajout de bruit gaussien

La figure B.5 représente la répétabilité du détecteur de Harris stabilisé avec relocalisation, à un changement de contraste. La répétabilité étant très peu dépendante du paramètre σ_0, seule la courbe pour $\sigma_0 = 1,4$ est représentée. Pour un facteur de contraste inférieur à $1,0$, on constate que la répétabilité est excellente (98 % des points sont encore présents lorsqu'on a deux fois moins de contraste). Pour les facteurs suppérieurs à $1,0$, l'effet déstabilisant n'est pas dû au changement de contraste lui-même, mais à la saturation des niveaux de gris qui se produit dans le cas d'un rehaussement de contraste. De tels phénomènes de saturation ne sont pas rares dans les processus de post-production et il est intéressant de connaître leur influence. Ils sont difficiles à modéliser et très peu de descripteurs y sont invariants (ils affectent la forme, la texture et les couleurs). On voit que le détecteur de Harris y est assez

robuste puisque 70 % des points sont encore présents pour un rehaussement de contraste d'un facteur $1,8$.

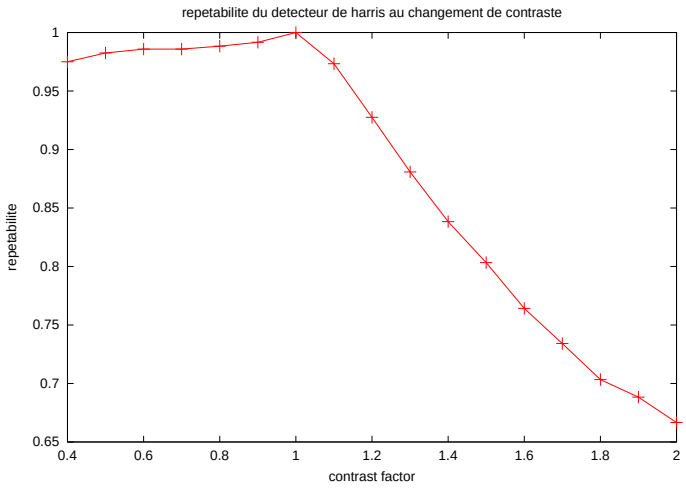


FIG. B.5 – Répétabilité à un changement de contraste

Annexe C

Comparaison qualitative des structures d'indexation vectorielle

Name	problems in high-D	supported query types	locality of node splits	storage utilization	fanout / size of index entries
R-tree	poor split algorithm leads to deteriorated directories	NN, region, range	yes	poor	poor, linearly dimension dependent
R*-tree	dto.	NN, region, range	yes	medium	poor, linearly dimension dependent
X-tree	high probability of queries overlapping MBR's leads to poor performance	NN, region, range	yes	medium	poor, linearly dimension dependent
LSD^h-tree	changing data distribution deteriorates directory	NN, region, range	no	medium	very good, dimension independent
SS-tree	high overlap in directory	NN	yes	medium	very good, dimension independent
TV-tree	only useful for specific data	NN	yes	medium	poor, somehow dimension dependent
SR-tree	very largedirectory sizes	NN	yes	medium	very poor, linearly dimension dependent
space filling curves	poor space partitioning	NN, region, range	yes	medium	as good as B-Tree, dimension independent
Pyramid-tree	problems with asymetric queries	region, range	yes	medium	as good as B-Tree, dimension independent

FIG. C.1 – Comparaison qualitative des structures d'indexation vectorielle

Annexe D

Démonstrations

D.1 Loi sphérique uniforme : convergence de la densité de probabilité d'une composante vers une gaussienne

Démonstration de l'équation 4.4 (page 88)

Pour $|x| \leq \epsilon$, la densité de probabilité de la distorsion selon une composante $f_{HS}^{j}(x)$, peut être développée de la manière suivante :

$$f_{HS}^{j}(x) = \frac{\pi^{\frac{D-1}{2}}(\epsilon^2 - x^2)^{\frac{D-1}{2}}}{\Gamma(\frac{D-1}{2}+1)} \frac{\Gamma(\frac{D}{2}+1)}{\pi^{\frac{D}{2}}\epsilon^D}$$

$$f_{HS}^{j}(x) = \frac{\Gamma(\frac{D}{2}+1)}{\Gamma(\frac{D-1}{2}+1)\sqrt{\pi}\epsilon}\left(1 - \frac{x^2}{\epsilon^2}\right)^{\frac{D-1}{2}}$$

Soit lorsque D est grand devant 1 :

$$f_{HS}^{j}(x) \approx \frac{\sqrt{D}}{\sqrt{2\pi}\epsilon}\left(1 - \frac{x^2}{\epsilon^2}\right)^{\frac{D-1}{2}}$$

dont le développement limité à l'ordre n est :

$$f_{HS}^{j}(x) \approx \frac{\sqrt{D}}{\sqrt{2\pi}\epsilon}\left(1 - \frac{(D-1)x^2}{2\epsilon^2} + ... + \frac{(\frac{D-1}{2})(\frac{D-1}{2}-1)...(\frac{D-1}{2}-n+1)\ x^{2n}}{n!\epsilon^{2n}} + o(x)\right)$$

Lorsque D tend vers l'infini on peut écrire :

$$f_{HS}^{j}(x) \approx \frac{\sqrt{D}}{\sqrt{2\pi}\epsilon}\left(1 - \frac{Dx^2}{2\epsilon^2} + ... + \left(\frac{D}{2\epsilon^2}\right)^n \frac{1}{n!}x^{2n} + o(x)\right)$$

$$f_{HS}^{j}(x) \approx \frac{\sqrt{D}}{\sqrt{2\pi}\epsilon} e^{\left(-\frac{1}{2}\frac{x^2 D}{\epsilon^2}\right)}$$

$$f_{HS}^{j}(x) \approx f_{\mathcal{N}(0,\sigma_{Hx})}(x) = f_{\mathcal{N}(0,\frac{\epsilon}{\sqrt{D}})}(x)$$

D.2 Optimisation de la profondeur

Démonstration de l'équation 4.20 (page 112)

Par récurrence sur l'expression du nombre moyen de p-blocs interceptés $n(p) = 2^{\gamma_q} n(p-1)$, on peut déterminer une expression de $n(p)$ par morceau :

$$n(p) = \begin{cases} 1 & \text{si } p = 0 \\ n_q(p) & \text{si } p \in [(q-1)D, qD] \end{cases} \tag{D.1}$$

$$n_q(p) = 2^{\gamma_q p}\ 2^{-(q-1)D\,\gamma_q} \prod_{r=1}^{q-1} 2^{D\,\gamma_r} \tag{D.2}$$

En substituant cette expression dans l'équation 4.19 du modèle de coût, on obtient une expression par morceau du temps de recherche moyen du système :

$$T(p) = \begin{cases} t_s\ N + t_a & \text{si } p = 0 \\ T_q(p) & \text{si } p \in [(q-1)D, qD] \end{cases}$$

où

$$T_q(p) = 2^{(1-q)D\,\gamma_q}\ \Pi_{q-1}\ \left(t_a\ 2^{\gamma_q p} + t_s\ N\ 2^{(\gamma_q - 1)p}\right) \tag{D.3}$$

et

$$\Pi_{q-1} = \prod_{r=1}^{q-1} 2^{D\,\gamma_r}$$

En résolvant l'équation $\frac{dT_q(p)}{dp} = 0$ et en gardant à l'esprit que $0 < \gamma_q \leq 1$, il est facile de montrer que :

– $T_q(p)$ a un unique minimum situé en

$$p_q^{min} = log_2\left(\frac{t_s}{t_a}\right) + log_2\,(N) + log_2\left(\frac{1-\gamma_q}{\gamma_q}\right) \tag{D.4}$$

– $T_q(p)$ est décroissante lorsque $p < p_q^{min}$ et croissante lorsque $p > p_q^{min}$

$T(p)$ étant défini par morceau, p_q^{min} sera un minimum de $T(p)$ si et seulement s'il est inclus dans l'intervalle $[(q-1)D, qD]$. Cette condition implique que :

$$2^{(q-1)D}\ \frac{\gamma_q}{1-\gamma_q} \leq N\ \frac{t_s}{t_a} \leq 2^{qD}\ \frac{\gamma_q}{1-\gamma_q} \tag{D.5}$$

Puisque la fonction $\frac{\gamma_q}{1-\gamma_q}$ est strictement croissante en fonction de q, l'inégalité D.5 ne peut pas être vérifiée pour deux valeurs différentes de q. Cela prouve qu'un minimum local ne peut exister que sur un seul intervalle $[(q-1)D, qD]$. Comme $T(p)$ est décroissante sur tous les intervalles précédents et croissante sur tous les intervalles suivant, et que $T(p)$ est continue (parce que $n(p)$ est continue), $T(p)$ possède un unique minimum global.

En pratique, on pourrait déterminer ce minimum en calculant les valeurs de la série p_q^{min} et en ne conservant que celui qui sera effectivement dans son intervalle de définition à savoir $[(q-1)D, qD]$.

Cependant, comme nous ne connaissons pas systématiquement la série γ_q, nous prenons une valeur approchée pour p_{min} :

$$p_{min} = log_2\left(\frac{t_s}{t_a}\right) + log_2\left(N\right) \tag{D.6}$$

Tant que γ_q n'atteint pas des valeurs très proches de $\gamma_q = 1$ ou de $\gamma_q = 0$, la suppression du terme $log_2\left(\frac{1-\gamma_q}{\gamma_q}\right)$ n'engendrera pas une erreur trop importante ($< 1,58$ pour $0,25 < \gamma_q < 0,75$).

D.3 Modélisation du temps de recherche

Démonstration de l'équation 4.21 (page 112)

En remplaçant la valeur de p par p_{min} dans l'expression du temps de recherche moyen (cf. equation D.3), on obtient l'expression du temps de recherche optimal en fonction de la taille de la base :

$$T_{min}(N) = T_q(p_{min}(N)) = A_q\ N^{\gamma_q} \tag{D.7}$$

$$N \in \left[2^{(q-1)D}\ \frac{t_a}{t_s}, 2^{qD}\ \frac{t_a}{t_s}\right]$$

où A_q est une constante ne dépendant pas de N sur un intervalle, et qui est égale à :

$$A_q = 2 \times 2^{(1-q)D\ \gamma_q} \times \Pi_{q-1} \times t_a\ \left(\frac{t_s}{t_a}\right)^{\gamma_q}$$

Démonstration de l'équation 4.22 (page 113)

En faisant l'hypothèse que t_s et t_a sont linéaires en fonction de D, i.e $t_s = s\ D$ et $t_a = a\ D$ et en remplaçant la valeur de p par p_{min} dans l'expression du temps de recherche moyen (cf. equation D.3), on obtient la fonction $T_{min}(D)$, définie par morceau :

$$T_{min}(D) = T_q(p_{min}, D) = C_q\ D\ 2^{(q-1)\ D\ \gamma_q} \prod_{r=1}^{q-1} 2^{D\ \gamma_r}$$

si

$$D \in \left[\frac{log_2(N\ \frac{s}{a})}{q-1}, \frac{log_2(N\ \frac{s}{a})}{q}\right]$$

où C_q est une constante ne dépendant pas de D sur un intervalle et égale à :

$$C_q = 2a\ N^{\gamma_q}\ \left(\frac{s}{a}\right)^{\gamma_q}$$

Démonstration de l'équation 4.25 (page 115)

Le temps de recherche total du mode de fonctionnement par requêtes groupées s'exprime par :

$$T_{tot}(p, \theta) = T(p) + 2^{\theta} t_{page} + \frac{N\ t_{load}}{N_{sig}} \tag{D.8}$$

Si l'on reprend la valeur usuelle de θ définie par l'équation 4.16 (page 109) et que l'on considère que

$$2^{\lceil log_2(\frac{N}{N_{mem}})\rceil} \approx \frac{N}{N_{mem}}$$

on aura alors :

$$T_{tot}(p) \approx T(p) + \frac{N}{N_{mem}} t_{page} + \frac{N\ t_{load}}{N_{sig}} \tag{D.9}$$

Le second terme étant indépendant de la profondeur p, on a :

$$\frac{dT_{tot}(p)}{dp} \approx \frac{dT(p)}{dp}$$

La valeur p_{min} pour laquelle $T_{tot}(p)$ sera minimum est donc la même que pour $T(p)$. On peut donc directement utiliser la valeur de p_{min} définie par l'équation 4.20. Finalement, on a :

$$T_{tot} \approx T_{min} + \frac{N}{N_{mem}} t_{page} + \frac{N\ t_{load}}{N_{sig}} \tag{D.10}$$

où T_{min} est la modélisation du temps de recherche en mémoire (étudiée dans la section précédente).

Démonstration de l'équation 4.27 (page 118)

Lorsque le nombre moyen de signatures contenues dans un p-bloc est donnée par

$$\bar{n_b}(p) = \frac{N}{2^{p\frac{D'}{D}}}$$

l'expression du temps de recherche par morceau (équation D.3, page 208) devient :

$$T_q(p) = 2^{(1-q)D\ \gamma_q}\ \Pi_{q-1}\ \left(t_a\ 2^{\gamma_q p} + t_s\ N\ 2^{\left(\gamma_q - \frac{D'}{D}\right)p}\right) \tag{D.11}$$

et le minimum correspondant est :

$$p_q^{min} = \frac{D}{D'}\left(log_2\left(\frac{t_s}{t_a}\right) + log_2\left(N\right) + log_2\left(\frac{\frac{D'}{D} - \gamma_q}{\gamma_q}\right)\right) \tag{D.12}$$

Si l'on conserve tout de même la valeur approximative définie dans le cas uniforme, c'est-à-dire :

$$p^{min} = log_2\left(\frac{t_s}{t_a}\right) + log_2\left(N\right)$$

on obtient alors le temps de recherche suivant, en fonction de la taille de la base :

$$T_{min}(N) = A'_q\ N^{\gamma_q}$$

quand

$$N \in \left[2^{(q-1)D}\ \frac{t_a}{t_s}, 2^{qD}\ \frac{t_a}{t_s}\right]$$

où A'_q est une constante supérieure à A_q ne dépendant pas de N sur un intervalle, et qui est égale à :

$$A'_q = \frac{A_q}{2}\ \left(1 + \left(\frac{t_a}{t_s}\right)^{(1-\frac{D'}{D})}\right)$$

Annexe E

Courbe de Hilbert remplissant l'espace

E.1 Construction

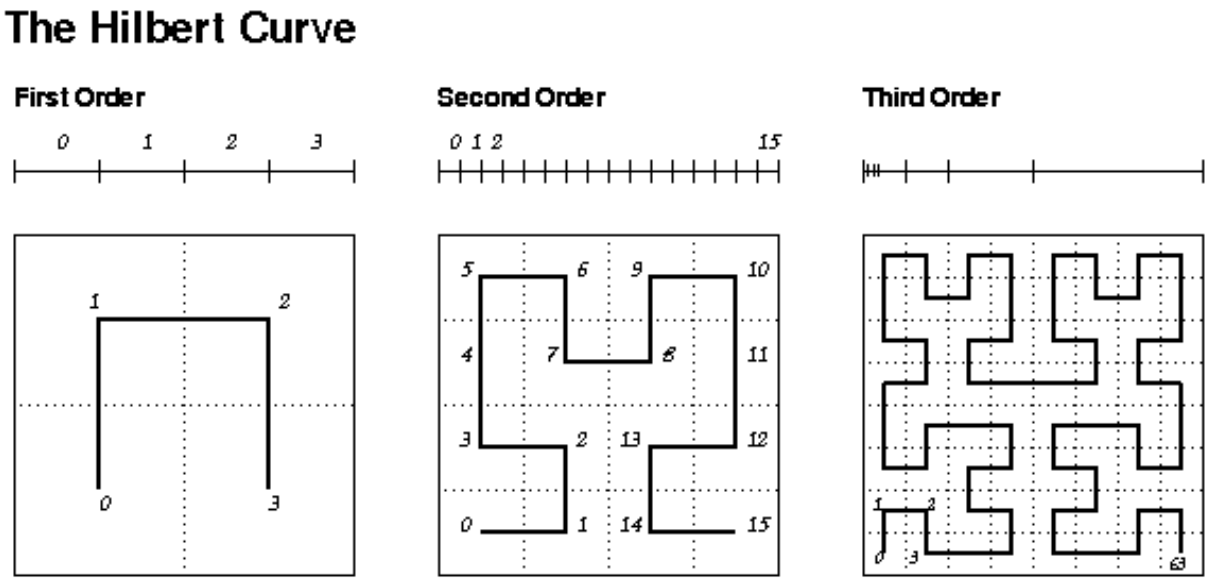


FIG. E.1 – Construction de la courbe de Hilbert de dimension 2

Dans son papier publié en 1891, Hilbert a illustré le concept d'une courbe remplissant l'espace de dimension 2, en donnant une méthode pour la construction par étapes d'une séquence infinie de courbes finies, dont chacune est définie par un arrangement linéaire des sous-espaces résultant du procédé de construction. Hilbert a alors prouvé que ce processus définit dans la limite une courbe passant par tous les points de l'espace, continue en tout point et différentiable en aucun point [146]. Une illustration de la construction de cette courbe est donnée sur la figure 3.3. La courbe d'ordre 2 est constituée de quatre courbes d'ordre 1 ayant chacune une orientation dépendant de sa position. La courbe d'ordre 3 est constituée de quatre courbes d'ordre 2, la courbe d'ordre 4 de quatre courbes d'ordre 3, et ainsi de suite.

Pour décrire le processus de construction de la courbe de Hilbert dans le cas de dimensions plus élevées, nous reprenons quelques unes des notations utilisées dans [131]. Soit H_Q^D l'approxi-

mation d'ordre Q d'une courbe de Hilbert remplissant l'espace de dimension D. Pour $Q \geq 1$ et $D \geq 2$, H_Q^D est une application bijective de l'espace discret multidimensionnel $\left[0, 2^Q - 1\right]^D$ dans l'espace monodimensionnel discret $\left[0, 2^{QD} - 1\right]$.
Dans l'espace discret multidimensionnel $\left[0, 2^Q - 1\right]^D$ chaque point $\boldsymbol{X} = (x_1, x_2, ..., x_D) \in \left[0, 2^Q - 1\right]^D$ possède $2D$ voisins dont les coordonnées diffèrent de celles de $\boldsymbol{X}$ seulement d'une unité sur une des composantes. Cette espace discret correspond en fait à une partition de l'espace continu en 2^{QD} cellules hyper-cubiques régulières.
Dans cet espace, H_2^D se déduit de l'approximation de la courbe de Hilbert d'ordre 1, H_1^D, en remplissant chacune des 2^D cellules de l'espace discret $[0, 1]^D$ par H_1^D, après une éventuelle rotation autour d'un axe et/ou une symétrie par rapport à un des hyper-plans médians.
L'approximation d'ordre Q, H_Q^D se déduit de H_{Q-1}^D en appliquant cette même règle à toutes les courbes H_1^D constituant H_{Q-1}^D, en tenant compte de leur orientation.

Les courbes de Hilbert dans un espace de dimension supérieure à 2 diffèrent en ce qu'il existe beaucoup plus de courbes possibles gardant la propriété de continuité. Chaque courbe est entièrement définie par sa courbe au premier ordre et une carte d'orientation permettant de passer du premier au deuxième ordre. L'existence de plusieurs courbes alternatives empêche d'exprimer arithmétiquement la courbe de Hilbert de manière unique. Un cadre théorique d'étude de ces courbes alternatives a été proposé par Alber et Niedermeier qui ont formalisé mathématiquement l'étude combinatoire de la courbe de Hilbert dans des dimensions plus élevées [3]. Les conditions nécessaires à la continuité de la courbe ainsi que le nombre de courbes possibles y sont en particulier détaillés.

E.2 Encodage et Décodage

Il existe principalement trois techniques pour calculer la coordonnée d'un point de l'espace multidimensionnel sur la courbe de Hilbert (encodage) ou inversement pour calculer les coordonnées d'un point dans l'espace multidimensionnel en partant de sa coordonnée sur la courbe de Hilbert [112]. Ces techniques ne permettent pas de calculer la coordonnée dans l'espace monodimensionnel continu mais seulement une approximation de celle-ci, dite d'ordre Q, correspondant à la coordonnée du point sur l'approximation de la courbe de Hilbert d'ordre Q. Cette coordonnée est en fait exprimée sous forme d'un mot binaire de longueur QD bits. Celui-ci peut ensuite être converti en un réel dans l'intervalle $[0, 1]$ ou bien en un entier dans l'intervalle $[0, 2^{QD}]$.

La première technique consiste à construire un arbre représentant la courbe, dans lequel les 2^{qD} nœuds du q^{eme} niveau contiennent chacun les 2^D correspondances entre la position dans l'espace et la position sur la courbe. La figure E.2 montre un tel arbre pour la courbe de Hilbert en dimension 2 et au 3^{eme} ordre. L'encodage consiste alors en un parcours de cet arbre. L'inconvénient est que l'espace mémoire requis est exponentiel en fonction de l'ordre de la courbe et de la dimension D de l'espace. Pour un fonctionnement en mémoire, cela limite leur utilisation à des dimensions inférieures à 10.

Une deuxième solution consiste à utiliser un diagramme d'état dans lequel chaque état correspond à l'orientation de la courbe à l'intérieur du sous-espace (cf. figure E.3). Chaque état contient 2^D transitions sortantes pour les 2^D sous-espaces de l'approximation à l'ordre supé-

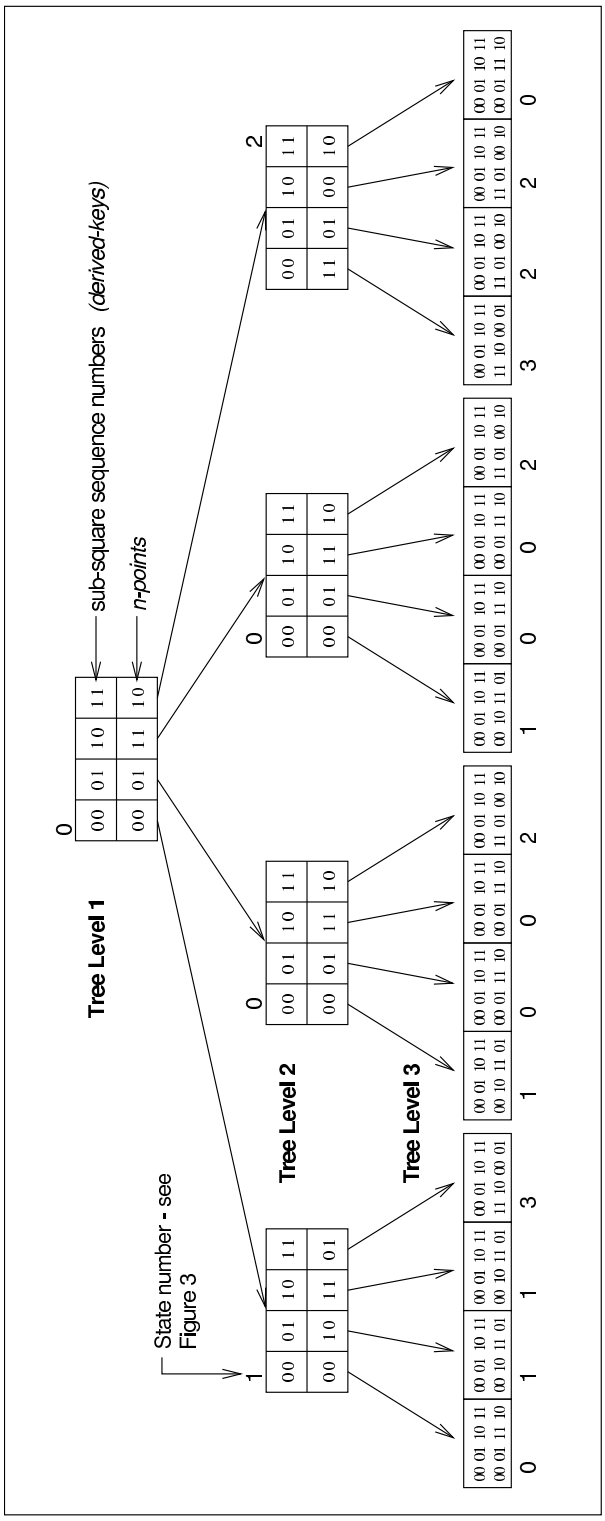


Fig. 2. The tree representation of the third order Hilbert Curve in 2 dimensions

FIG. E.2 – Arbre représentatif d'une courbe de Hilbert de dimension 2 - Image extraite de la thèse de J.K. Lawder [112]

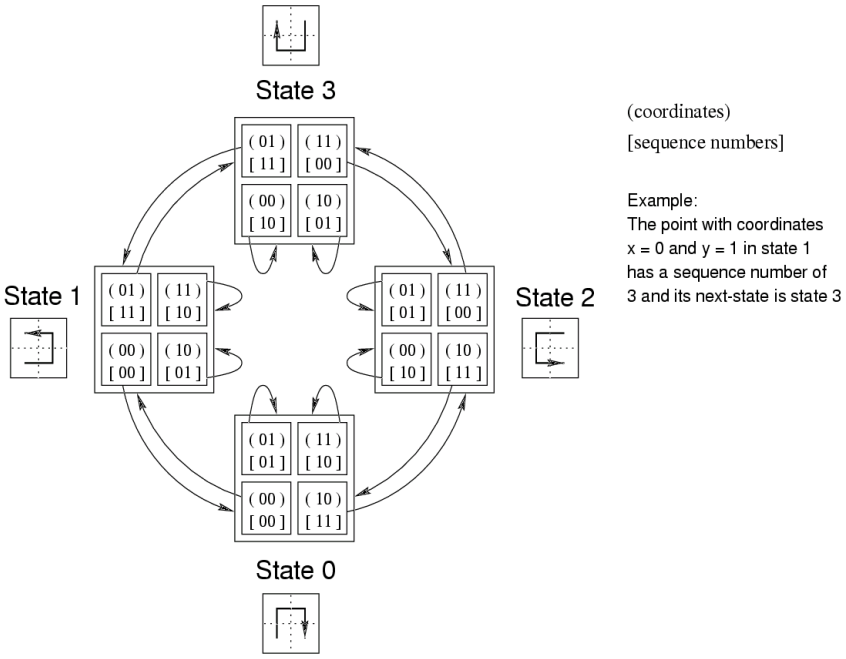


FIG. E.3 – Diagramme d'état d'une courbe de Hilbert de dimension 2 - Image extraite de la thèse de J.K. Lawder [112]

rieur. L'avantage est que ce diagramme reste fixe quel que soit l'ordre de l'approximation de la courbe de Hilbert. Le nombre d'orientations possibles étant limité, le nombre d'états est inférieur à 2^D et n'est pas exponentiel en fonction de la dimension de l'espace. En revanche la taille d'un état est proportionnelle à 2^D et la taille mémoire nécessaire au diagramme entier est encore une fois exponentielle avec la dimension. Cette forte croissance limite l'utilisation des diagrammes d'état en mémoire à une dimension d'environ 14 [112].

La troisième méthode pour calculer les coordonnées sur la courbe de Hilbert est purement calculatoire et ne nécessite aucun stockage volumineux en mémoire. Cette méthode est ainsi efficace quelle que soit la dimension. Le seul algorithme, proposé à l'heure actuel, permettant de faire la conversion entre l'espace multidimensionnel et la courbe de Hilbert est l'algorithme de Butz [36]. Cet algorithme ne génère qu'une seule des nombreuses possibilités de courbe de Hilbert multidimensionnelle. Le détail de l'algorithme de Butz est donné dans la section suivante.

E.3 Algorithme de Butz

Par la suite, nous noterons $\rho^{(i)}$, le i^{eme} bit d'un mot binaire ρ et $\rho_{(10)}$, la valeur de ρ, converti en base-10 :

$$\rho_{(10)} = \rho^{(1)}2^0 + \rho^{(2)}2^1 + ... + \rho^{(D)}2^{D-1}$$

Soit un point $\boldsymbol{X}$ appartenant à l'espace multidimensionnel discret $\left[0, 2^Q - 1\right]^D$. Chacune de ses D composantes peut s'exprimer sous forme d'un mot binaire de Q bit :

$$\boldsymbol{X} = \begin{pmatrix} x_1 \\ \vdots \\ x_d \\ \vdots \\ x_D \end{pmatrix} = \begin{pmatrix} x_1^{(1)} \dots\ x_1^{(q)} \dots\ x_1^{(Q)}{}_{(10)} \\ \vdots \quad\quad \vdots \quad\quad \vdots \\ x_d^{(1)} \dots\ x_d^{(q)} \dots\ x_d^{(Q)}{}_{(10)} \\ \vdots \quad\quad \vdots \quad\quad \vdots \\ x_D^{(1)} \dots\ x_D^{(q)} \dots\ x_D^{(Q)}{}_{(10)} \end{pmatrix}$$

où chaque vecteur colonne $\left(x_1^{(q)} \dots\ x_d^{(q)} \dots\ x_D^{(q)}\right)^t$ représente la position binaire de $\boldsymbol{X}$ à l'intérieur du sous-espace formé par la cellule hyper-cubique d'ordre $(q-1)$, contenant $\boldsymbol{X}$. Nous notons ce mot binaire

$$a_q(\boldsymbol{X}) = x_1^{(q)} \dots\ x_d^{(q)} \dots\ x_D^{(q)}$$

La coordonnée de $\boldsymbol{X}$ sur l'approximation de la courbe de Hilbert d'ordre Q obtenue par l'algorithme de Butz, est un mot binaire de QD bits. Sa valeur en base-10 est :

$$H_Q^D(\boldsymbol{X}) = h_1(\boldsymbol{X})_{(10)} \dots\ h_q(\boldsymbol{X})_{(10)} \dots\ h_Q(\boldsymbol{X})_{(10)}$$

où chaque $h_q(\boldsymbol{X})$ est un mot binaire de D bits :

$$h_q(\boldsymbol{X}) = h_q^{(1)}(\boldsymbol{X}) \dots\ h_q^{(d)}(\boldsymbol{X}) \dots\ h_q^{(D)}(\boldsymbol{X})$$

La procédure de l'algorithme de Butz pour convertir les coordonnées d'un point $\boldsymbol{X}$ de l'espace multidimensionnel en sa coordonnée sur l'approximation d'ordre Q de la courbe de Hilbert, est [36, 113] :

$$h_q(\boldsymbol{X}) = P\left(\psi_{q-1}(a_q(\boldsymbol{X}))\right) \quad \forall q \in [1, Q] \tag{E.1}$$

où P est une fonction binaire définie par une opération de ou exclusif :

$$\beta = P(\theta) \Leftrightarrow \beta^{(d)} = \bigoplus_{d'=1}^{d} \theta^{(d')} \quad \forall d \in [1, D]$$

et ψ_{q-1} est une fonction binaire qui peut se décomposer en :

$$\psi_{q-1}(a_q) = (a_q \oplus \omega_{q-1}) \ll \Sigma_{q-1} \tag{E.2}$$

où $\ll \Sigma_{q-1}$ représente une permutation circulaire vers la gauche de Σ_{q-1} bits. Avant de fournir le détail du calcul récursif de ω_{q-1} et Σ_{q-1}, nous pouvons d'ores et déjà expliquer simplement le principe géométrique de la récursion.
Pour $q = 1$, l'équation (E.1) se réduit à $h_1 = P(a_1)$. On voit donc que la fonction binaire P correspond en fait à la conversion des coordonnées de $\boldsymbol{X}$ dans l'espace multidimensionnel binaire (un seul bit par dimension) en la coordonnée de Hilbert de $\boldsymbol{X}$ avec une précision de D bits. P définit donc l'approximation d'ordre 1 de la courbe de Hilbert, H_1^D, également appelé motif générateur de la courbe. Cette conversion est ensuite appliquée récursivement pour les ordres q supérieurs, à la différence que le motif générateur n'a plus la même orientation et que celle-ci dépend de la position de $\boldsymbol{X}$ à l'ordre inférieur c'est-à-dire de son vecteur de position binaire a_q. Avant d'appliquer la transformation P, a_q est donc transformé par la fonction ψ_{q-1} qui

correspond dans l'espace à des rotations et/ou des symétries. Si l'on regarde plus précisément la décomposition de ψ_{q-1} dans l'équation E.2, on constate que le ou exclusif réalise des symétries par rapport aux hyper-plans médians, tandis que la permutation circulaire réalise des permutations d'axe.
Le détail des calculs d'une récursion est :

(E.3)

1. $\sigma_q = \psi_{q-1}(a_q) = (a_q \oplus \omega_{q-1}) \ll \Sigma_{q-1}$
2. $\begin{cases} \Sigma_0 = 0 \\ \Sigma_{q-1} = (J_1 - 1) + (J_2 - 1) + ... + (J_{q-1} - 1) \end{cases}$
3. J_{q-1} est la position principale du mot binaire $h_{q-1}(\boldsymbol{X})$. La position principale est la position j du bit de poids le plus fort tel que $h_{q-1}^{(j)} \neq h_{q-1}^{D}$. Si tous les bits de $h_{q-1}(\boldsymbol{X})$ sont égaux, la position principale est égale à D.
4. $\begin{cases} \omega_0 = 000...00 \\ \omega_{q-1} = \omega_{q-2} \oplus \widetilde{T}_{q-1} = \widetilde{T}_1 \oplus \widetilde{T}_2 \oplus ... \oplus \widetilde{T}_{q-1} \end{cases}$
5. $\widetilde{T}_{q-1} = T_{q-1} \ll \Sigma_{q-2}$
6. T_{q-1} est obtenu en complémentant le premier bit de σ_{q-1} et, si et seulement si le mot résultant est paire, en complémentant le ${J_{q-1}}^{eme}$ bit de σ_{q-1}

Pour une description détaillée de la procédure inverse de l'algorithme de Butz, convertissant une coordonnée sur la courbe en un vecteur dans l'espace multidimensionnel discret, on pourra se référer à [113].

E.4 Partitionnement hiérarchique de l'espace multidimensionnel

En incrémentant la profondeur p du préfixe binaire des clés de Hilbert, on peut définir une succession hiérarchique de partitions de l'espace. Chaque fois que p augmente d'une unité, tous les p-blocs sont séparés en deux blocs par un hyper-plan perpendiculaire à un des axes de l'espace (cf. figure E.4) :

1. Pour le premier ordre ($q = 1$), c'est-à-dire $1 \leq p \leq D$, l'espace hyper-cubique entier est divisé itérativement selon l'ordonnancement naturel des dimensions : par un hyper-plan perpendiculaire à l'axe x_1 pour le passage de $p = 0$ à $p = 1$, par un hyper-plan perpendiculaire à l'axe x_2 pour le passage de $p = 2$ à $p = 3$, et ainsi de suite jusqu'à la profondeur $p = D$.
2. Pour les ordres q supérieurs à 1, c'est-à-dire

   $$(q-1)D \leq p \leq qD \quad \forall q > 1$$

   le processus est identique sauf que l'ordonnancement des axes de division n'est pas le même pour tous les blocs hyper-cubiques de la partition obtenue à la profondeur $p = (q-1)D$. Le nouvel ordonnancement correspond à une permutation circulaire de l'ordonnancement naturel d'une valeur égale à la variable Σ_{q-1} de l'algorithme de Butz (cf. équation E.3, page 216). Ainsi, chaque bloc est divisé par l'hyper-plan perpendiculaire à l'axe $x_{\Sigma_{q-1}}$ pour le passage de $p = (q-1)D$ à $p = (q-1)D + 1$, par l'hyper-plan perpendiculaire à l'axe $x_{\Sigma_{q-1}+1}$ pour le passage de $p = (q-1)D + 1$ à $p = (q-1)D + 2$, et ainsi de suite jusqu'à $p = qD$.

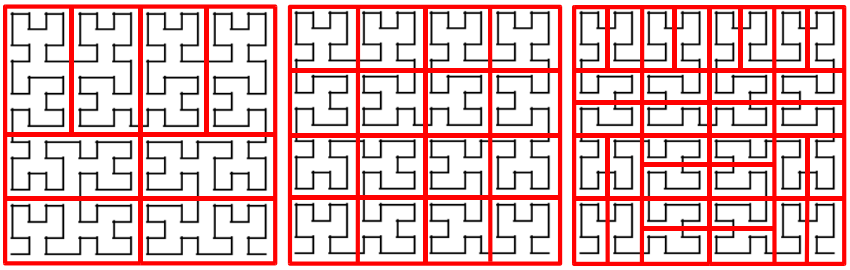


FIG. E.4 – Partition de l'espace pour $D = 2$ et $Q = 4$ à différentes profondeurs - de gauche à droite : p=3, 4, 5

Toute région de l'espace peut être approximée à une profondeur p quelconque, par l'ensemble des p-blocs ayant une intersection avec celle-ci et donc par un ensemble d'intervalles sur la courbe. Il suffit pour cela d'augmenter incrémentalement la profondeur de la partition de 1 jusqu'à p et de déterminer à chaque itération quels sont les blocs fils des blocs sélectionnés à la profondeur précédente, qui ont une intersection avec la région considérée.
Au niveau de l'implémentation, il faut conserver en mémoire une liste des p-blocs sélectionnés et la remettre à jour à chaque itération ($p \leftarrow p + 1$). Chaque p-bloc est caractérisé par ses $2D$ coordonnées caractérisant sa position dans l'espace. Les coordonnées des 2 blocs fils d'un p-bloc père sont déterminées en divisant en deux les coordonnées du bloc père sur l'axe perpendiculaire à l'hyperplan de la division. L'indice ν de cet axe de division est déterminée par :

$$\nu = \Sigma_{q-1} + (p - qD)$$

où q est l'ordre de la partition ($q = \lceil \frac{p}{D} \rceil$) et Σ_{q-1} est une variable de l'algorithme de Butz calculée à partir du préfixe de Hilbert H_p du bloc à l'ordre q (cf. équation E.3, page 216).
Une fois les coordonnées des blocs fils déterminées, ceux qui n'ont pas d'intersection avec la région considérée sont supprimés de la liste.

L'exemple d'un tel algorithme pour une recherche à ϵ-près autour d'une requête Q est présentée sur la page suivante.

Données d'entrée : p (profondeur finale de la partition);
$\boldsymbol{Q}$ (requête);
ϵ (rayon de la requête);
Résultats : B_ϵ (Ensemble des p-blocs sélectionnées);
Données temporaires : p_{loop} (profondeur courante);
B (ensemble courant des blocs sélectionnées);
n (nombre de blocs dans B);
B_f (ensemble des blocs fils sélectionnés);
n_f (nombre de blocs dans B_f);

$p_{loop} = 1$;
$n = 1$;
$B[0] = [0, 255]^D$;
tant que $p_{loop} \leq p$ **faire**
  $n_f = 0$;
  pour $i = 1 : n$ **faire**
1   $\nu = \boldsymbol{calculDirection}(B[i], p_{loop})$;
2   $[Fils1, Fils2] = \boldsymbol{divise}(B[i], \nu)$;
    si $MinDist(Fils1, \boldsymbol{Q}) \leq \epsilon$ **alors**
      $B_f[n_f] = Fils1$;
      $n_f = n_f + 1$;
    fin
    si $MinDist(Fils2, \boldsymbol{Q}) \leq \epsilon$ **alors**
      $B_f[n_f] = Fils2$;
      $n_f = n_f + 1$;
    fin
  fin
  $B = B_f$;
  $n = n_f$;
  $p_{loop} = p_{loop} + 1$;
fin
$B_\epsilon = B$;

Algorithme 1: Algorithme de filtrage pour une requête à ϵ-près

(1) La fonction $\nu = \boldsymbol{calculDirection}(B[i], p)$ est calculée par : $\nu = \Sigma_{q-1} + (p - qD)$ où Σ_{q-1} est une variable de l'algorithme de Butz, appliqué à n'importe quel point de $B[i]$ jusqu'au q^{eme} ordre.
(2) La fonction $[Fils1, Fils2] = \boldsymbol{divise}(B[i], \nu)$ consiste à diviser en deux le bloc $B[i]$ selon le ν^{eme} axe : Si on note $B.u_j$ et $B.w_j$ les bornes supérieures et inférieures d'un bloc quelconque B suivant le j^{eme} axe, alors :

$$\begin{aligned} Fils1.w_\nu &= B[i].w_\nu \\ Fils2.u\nu &= B[i].u_\nu \\ Fils1.u_\nu = Fils2.w_\nu &= \tfrac{B[i].w_\nu + B[i].u_\nu}{2} \end{aligned}$$

toutes les autres coordonnées restent les mêmes :

$$\forall j \neq \nu, \quad \begin{aligned} Fils1.w_j = Fils2.w_j &= B[i].w_j \\ Fils1.u_j = Fils2.u_j &= B[i].u_j \end{aligned}$$

Annexe F

Détermination de $B(\tau)$

Cette annexe décrit l'algorithme de l'étape de filtrage permettant de déterminer l'ensemble $B(\tau)$ des p-blocs ayant une probabilité supérieur à un seuil τ (cf. section 4.3.4). Soit, pour une requête $\boldsymbol{Q} = [q_1, q_2, ..., q_D]$:

$$B(\tau) = \left\{ \{b_i\} : \int_{b_i} p_{\boldsymbol{\Delta S}}(\boldsymbol{X} - \boldsymbol{Q})\, \boldsymbol{dX} > \tau \right\}$$

En respect de l'hypothèse d'indépendance des composantes de la distorsion, la probabilité d'un p-bloc peut être calculée facilement par :

$$P(b_i) = \int_{b_i} p_{\boldsymbol{\Delta S}}(\boldsymbol{X} - \boldsymbol{Q})\, \boldsymbol{dX} = \prod_{j=1}^{D} \left[P_{\Delta S_j}(u_i^j - q_j) - P_{\Delta S_j}(w_i^j - q_j) \right] \tag{F.1}$$

où $P_{\Delta S_j}(x)$ est la fonction de répartition de la j^{eme} composante du vecteur de distorsion tabulée dans une *Look Up Table* ($LUT(\alpha, \sigma_g)$), u_i^j et w_i^j sont les bornes supérieures et inférieures du p-bloc suivant le j^{eme} axe.
La probabilité d'un p-bloc au niveau p est égale à la somme des probabilités des deux blocs fils au niveau $p+1$. Si un p-bloc a une probabilité inférieure à τ, tous ses blocs fils auront donc une probabilité inférieure à τ. L'approche la plus simple pour déterminer $B(\tau)$ est donc un algorithme de type *divide-and-conquer* qui consiste à faire croître la valeur de la profondeur jusqu'à la profondeur finale (de 0 à p). A chaque itération, la probabilité des p-blocs fils est calculée et seuls ceux ayant une probabilité supérieure à τ sont conservés dans une queue de priorité.
La probabilité des deux blocs fils $Fils1$ et $Fils2$ d'un bloc père quelconque b_i peut être déduite de la probabilité du bloc père par :

$$P_{Fils1} = P(b_i) \frac{P_{\Delta S_\nu}(\frac{u_i^\nu + w_i^\nu}{2} - q_\nu) - P_{\Delta S_\nu}(w_i^\nu - q_\nu)}{P_{\Delta S_\nu}(u_i^\nu - q_\nu) - P_{\Delta S_\nu}(w_i^\nu - q_\nu)} \tag{F.2}$$

$$P_{Fils2} = P(b_i) \frac{P_{\Delta S_\nu}(u_i^\nu - q_\nu) - P_{\Delta S_\nu}(\frac{u_i^\nu + w_i^\nu}{2} - q_\nu)}{P_{\Delta S_\nu}(u_i^\nu - q_\nu) - P_{\Delta S_\nu}(w_i^\nu - q_\nu)} \tag{F.3}$$

où ν représente la direction de l'axe de division du bloc père, donnée par

$$\nu = \Sigma_{q-1} + (p - qD)$$

où q est l'ordre de la partition ($q = \lceil \frac{p}{D} \rceil$) et Σ_{q-1} est une variable de l'algorithme de Butz calculée à partir du préfixe de Hilbert H_p du bloc b_i (cf. équation E.3, page 216).
Le principe de l'algorithme est très semblable à celui présenté pour une recherche à ϵ-près (cf. page 218) sauf que la règle de filtrage géométrique des blocs fils ($MinDist(Fils, \boldsymbol{Q}) \leq \epsilon$) est remplacée par une règle de filtrage probabiliste ($P_{Fils} \geq \tau$).

F.1 Mode par requête unique

Données d'entrée : p (profondeur finale de la partition);
$\boldsymbol{Q}$ (requête);
$LUT(\alpha, \sigma_g)$ (Tableau contenant les valeurs de $P_{\Delta S_j}(x)$);
τ (Seuil de probabilité)
Résultats : B_τ (Ensemble des p-blocs sélectionnées);
Données temporaires : p_{loop} (profondeur courante);
B (ensemble courant des blocs sélectionnées);
n (nombre de blocs dans B);
B_f (ensemble des blocs fils sélectionnés);
n_f (nombre de blocs dans B_f);

$p_{loop} = 1$;
$n = 1$;
$B[0] = [0, 255]^D$;
tant que $p_{loop} \leq p$ **faire**
  $n_f = 0$;
  pour $i = 1 : n$ **faire**
1   $\nu = \boldsymbol{calculDirection}(B[i], p_{loop})$;
2   $[Fils1, Fils2] = \boldsymbol{divise}(B[i], \nu)$;
3   $P_{Fils1} = \boldsymbol{probFils1}(B[i], \nu, \boldsymbol{Q}, LUT(\alpha, \sigma_g))$;
4   $P_{Fils2} = \boldsymbol{probFils2}(B[i], \nu, \boldsymbol{Q}, LUT(\alpha, \sigma_g))$;
    si $P_{Fils1} \geq \tau$ **alors**
      $B_f[n_f] = Fils1$;
      $n_f = n_f + 1$;
    fin
    si $P_{Fils2} \geq \tau$ **alors**
      $B_f[n_f] = Fils2$;
      $n_f = n_f + 1$;
    fin
  fin
  $B = B_f$;
  $n = n_f$;
  $p_{loop} = p_{loop} + 1$;
fin
$B_\tau = B$;

Algorithme 2: Algorithme de filtrage d'une requête satistique

(1) cf. page 218
(2) cf. page 218
(3,4) cf. équation F.2 et F.3

F.2 Mode par requêtes groupées

Dans le cas du mode de fonctionnement par requêtes groupées, seuls les blocs appartenant à la page courante doivent être sélectionnés par l'algorithme de filtrage. Il suffit pour cela de commencer directement le partitionnement hiérarchique à la profondeur θ, c'est à dire la profondeur de découpage des 2^θ pages. L'algorithme correspondant est présenté sur la page suivante.

Données d'entrée : p (Profondeur de la partition);
$\boldsymbol{Q}$ (Requête);
i_{page} (Numéro de la page courante);
θ (Profondeur du découpage des pages);
$LUT(\alpha, \sigma_g)$ (Tableau contenant les valeurs de $P_{\Delta S_j}(x)$);
τ (Seuil de probabilité);

Résultats : $B_\tau(i_{page})$ (Ensemble des p-blocs sélectionnées);

Données temporaires : p_{loop} (profondeur courante);
B (ensemble courant des blocs sélectionnées);
n (nombre de blocs dans B);
B_f (ensemble des blocs fils sélectionnés);
n_f (nombre de blocs dans B_f);

$n = 0$;
1 $B[0] = \boldsymbol{thetaBloc}(i_{page})$;
2 **si** $\boldsymbol{probBloc}(B[0], \boldsymbol{Q}, LUT(\alpha, \sigma_g)) \geq \tau$ **alors**
  $n = 1$;
  $p_{loop} = \theta$;
  tant que *($p_{loop} \leq p$) **et** ($n > 0$)* **faire**
    $n_f = 0$;
    pour $i = 1 : n$ **faire**
      $\nu = \boldsymbol{calculDirection}(B[i], p_{loop})$;
      $[Fils1, Fils2] = \boldsymbol{divise}(B[i], \nu)$;
      $P_{Fils1} = \boldsymbol{probFils1}(B[i], \nu, \boldsymbol{Q}, LUT(\alpha, \sigma_g))$;
      $P_{Fils2} = \boldsymbol{probFils2}(B[i], \nu, \boldsymbol{Q}, LUT(\alpha, \sigma_g))$;
      si $P_{Fils1} \geq \tau$ **alors**
        $B_f[n_f] = Fils1$;
        $n_f = n_f + 1$;
      fin
      si $P_{Fils2} \geq \tau$ **alors**
        $B_f[n_f] = Fils2$;
        $n_f = n_f + 1$;
      fin
    fin
    $B = B_f$;
    $n = n_f$;
    $p_{loop} = p_{loop} + 1$;
  fin
  $B_\tau(i_{page}) = B$;
fin

Algorithme 3: Algorithme de filtrage d'une requête satistique dans une page

(1) La fonction $B = \boldsymbol{thetaBloc}(i_{page})$ calcul les coordonnées du θ-bloc correspondant à la ${i_{page}}^{eme}$ page. Si on suppose que $\theta \leq D$ (soit $n_{pages} \leq 1\,048\,576$ pour $D = 20$), le calcul est le suivant :

$$\forall j \in [1,\theta] \qquad \begin{cases} B.w[j] = 0 \; et \; B.u[j] = 127,5 & si \; (H_1^D)^{-1}(i_{page}).x_j = 0 \\ B.w[j] = 127,5 \; et \; B.u[j] = 255 & si \; (H_1^D)^{-1}(i_{page}).x_j = 1 \end{cases}$$

$$\forall j \in [\theta+1, D] \qquad B.w[j] = 0 \; et \; B.u[j] = 255$$

où $(H_1^D)^{-1}$ est la procédure de décodage d'une coordonnée curviligne de Hilbert à l'ordre 1 et $(H_1^D)^{-1}.x_j$ la j^{eme} composante du vecteur binaire ainsi obtenu. Notons que la fonction ***thetaBloc***(i_{page}) peut être calculée une fois pour toutes lors du chargment de la page et qu'il est inutile de la recalculer pour chaque signature candidate.

(2) La fonction ***probBloc*** calcule la probabilité d'un bloc quelconque à partir de ses coordonnées (cf. équation F.1).

Annexe G

Exemples de détection du système de monitorage d'une chaîne de télévision

FIG. G.1 – Exemples de détection du système de monitorage d'une chaîne de télévision - à gauche : séquences diffusées (capturées en noir et blanc) - à droite : séquences identifiées

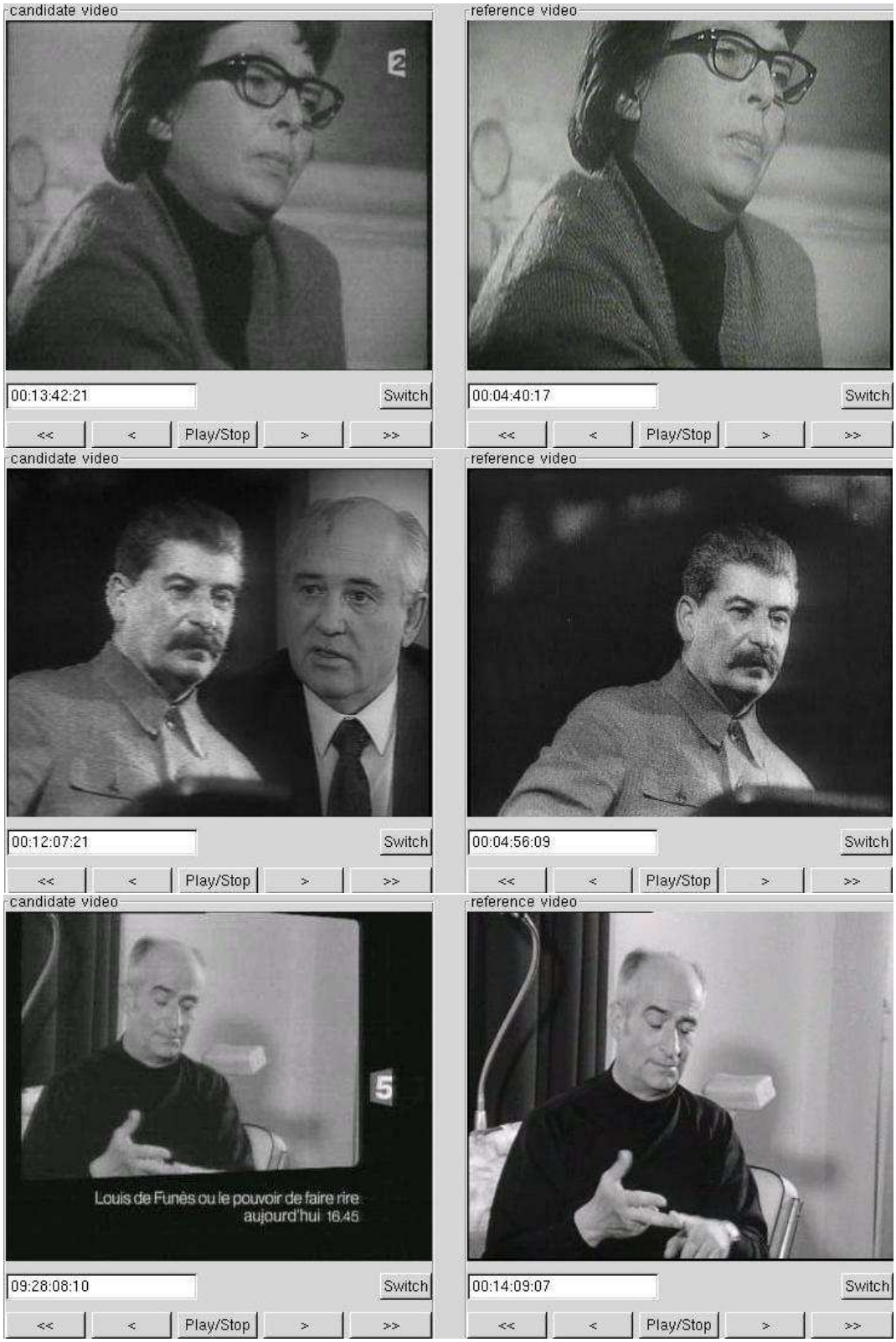


FIG. G.2 – Exemples de détection du système de monitorage d'une chaîne de télévision - à gauche : séquences diffusées (capturées en noir et blanc) - à droite : séquences identifiées

FIG. G.3 – Exemples de détection du système de monitorage d'une chaîne de télévision - à gauche : séquences diffusées (capturées en noir et blanc) - à droite : séquences identifiées

FIG. G.4 – Exemples de détection du système de monitorage d'une chaîne de télévision - à gauche : séquences diffusées (capturées en noir et blanc) - à droite : séquences identifiées

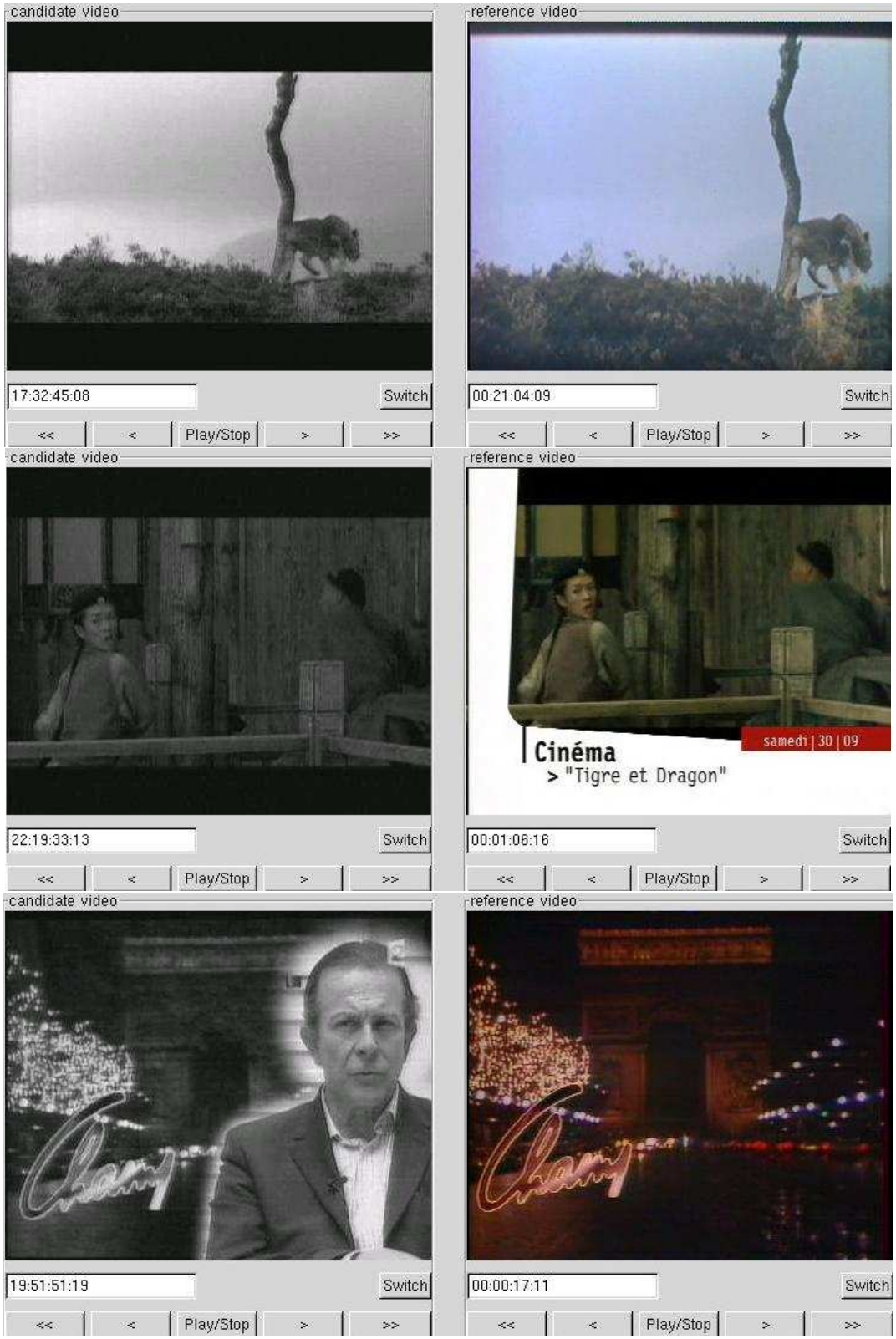


FIG. G.5 – Exemples de détection du système de monitorage d'une chaîne de télévision - à gauche : séquences diffusées (capturées en noir et blanc) - à droite : séquences identifiées

Annexe H

Exemples de fausse alarme du système de monitorage d'une chaîne de télévision

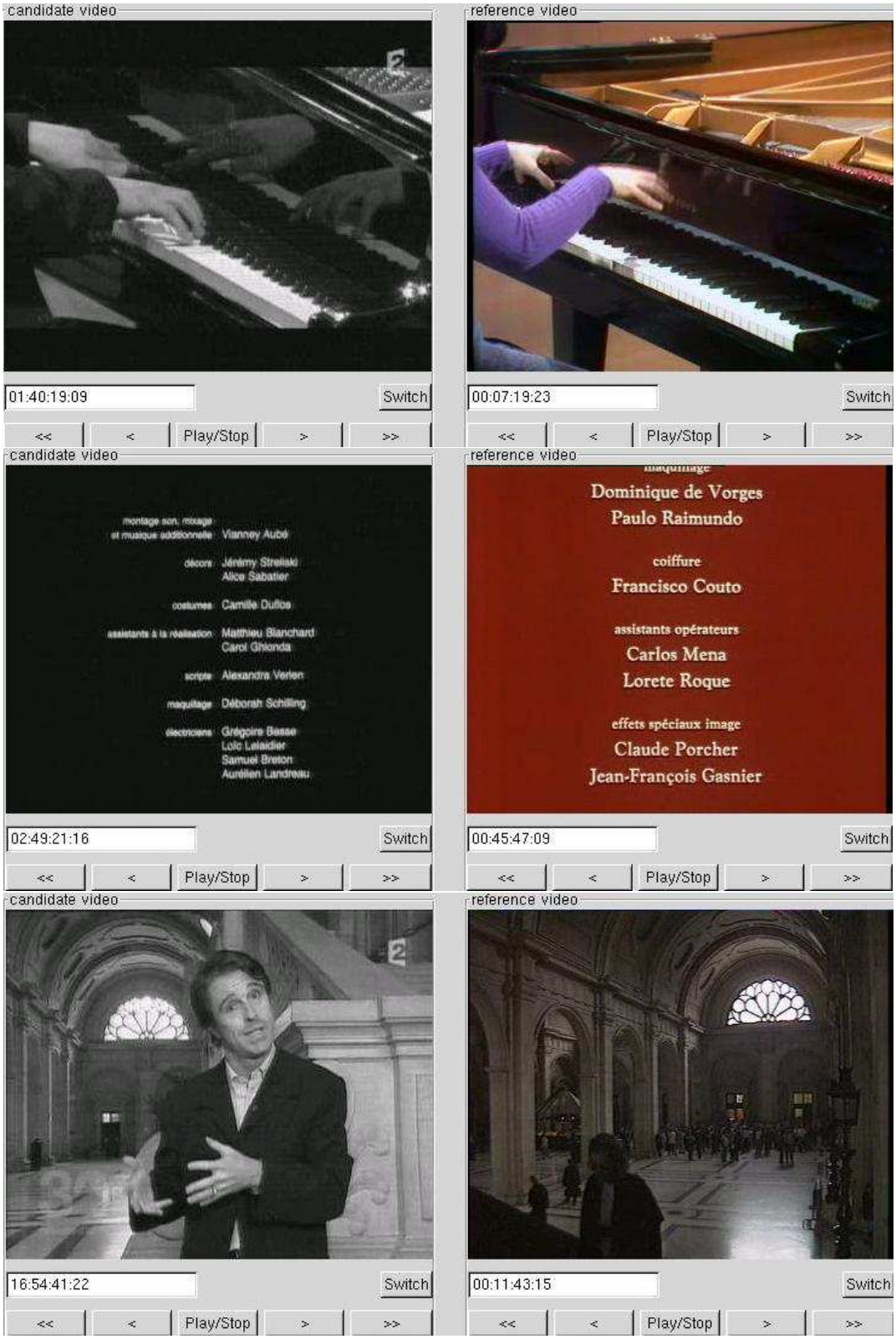


FIG. H.1 – Exemples de fausse alarme - à gauche : séquences diffusées (capturées en noir et blanc) - à droite : séquences identifiées

Annexe I

Capture d'écran de l'interface graphique

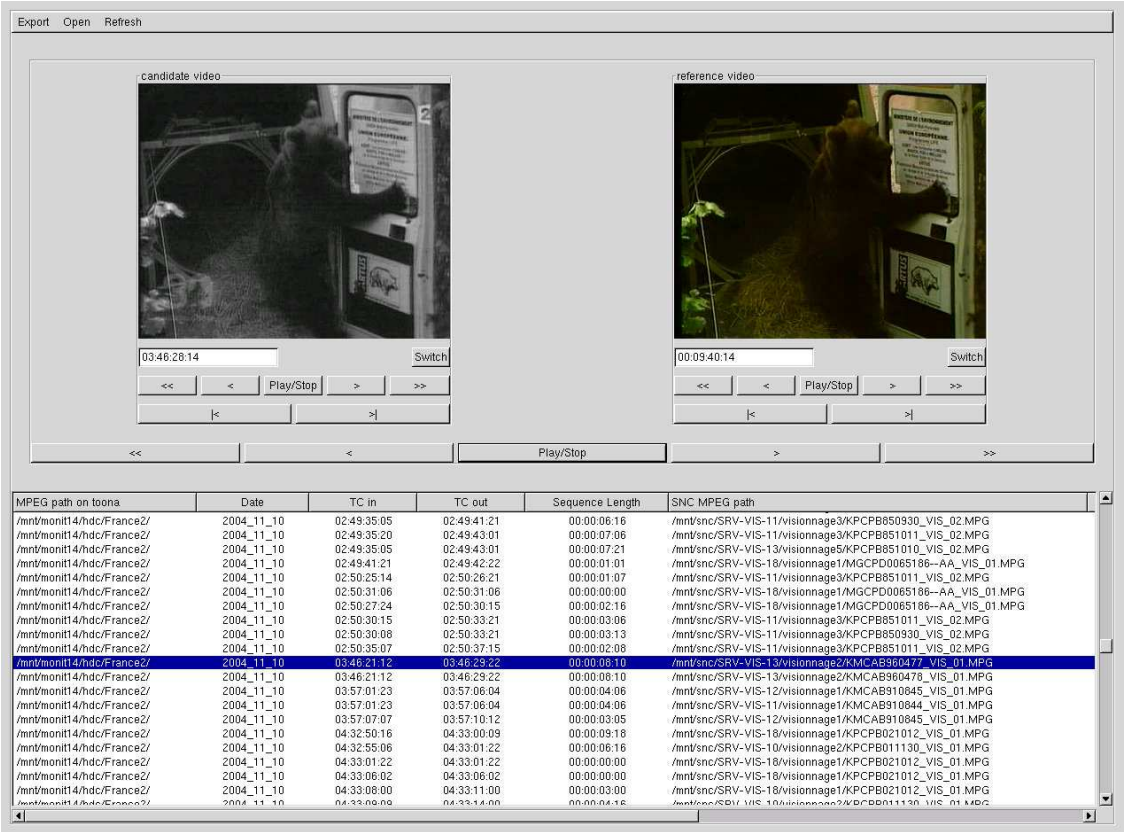


FIG. I.1 – Interface graphique de visionnage des résultats

Bibliographie

[1] A. Aboulnaga and J. F. Naughton. Accurate estimation of the cost of spatial selections. In *Proc. of Int. Conf. on Data Engineering*, pages 123–134, 2000.

[2] D. A. Adjeroh, M. C. Lee, and I. King. A distance measure for video sequence similarity matching. In *Proc. of Int. Workshop on Multimedia Database Management Systems*, pages 72–79, 1998.

[3] J. Alber and R. Niedermeier. On multi-dimensional hilbert indexings. In *Proc. of Int. Conf. on Computing and Combinatorics*, pages 329–338, 1998.

[4] L. Amsaleg and P. Gros. Content-based retrieval using local descriptors : Problems and issues from a database perspective. *Pattern Analysis and Applications*, 4(2) :1–124, 2001.

[5] L. Amsaleg, P. Gros, and S-A. Berrani. Robust object recognition in images and the related database problems. *Special issue of the Journal of Multimedia Tools and Applications*, 23 :221–235, 2003.

[6] S. Arya and D. M. Mount. Approximate range searching. In *Proc. of Symp. on Computational geometry*, pages 172–181, 1995.

[7] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *Journal of the ACM*, 45(6) :891–923, 1998.

[8] S. Ayer and H. S. Sawhney. Layered representation of motion video using robust maximum-likelihood estimation of mixture models and mdl encoding. In *Proc. of Int. Conf. on Computer Vision*, pages 777–784, 1995.

[9] R. Bayer and C. McCreight. Organisation and maintenance of large ordered indexes. *Acta Informatica*, 1(3) :173–179, 1972.

[10] N. Beckmann, H-P. Kriegel, R. Schneider, and B. Seeger. The r*-tree : an efficient and robust access method for points and rectangles. In *Proc. of ACM SIGMOD Int. Conf. on Management of Data*, pages 322–331, 1990.

[11] A. Belussi and C. Faloutsos. Estimating the selectivity of spatial queries using the 'correlation' fractal dimension. In *Proc. of Int. Conf. on Very Large Data Bases*, pages 299–310, 1995.

[12] K. P. Bennett, U. Fayyad, and D. Geiger. Density-based indexing for approximate nearest-neighbor queries. In *Proc. of Conf. on Knowledge Discovery in Data*, pages 233–243, 1999.

[13] J. L. Bentley. Multidimensional binary search trees in database applications. *IEEE Trans. on Software Engineering*, 5(4) :333–349, 1979.

[14] S. Berchtold, C. Böhm, D. Keim, F. Krebs, and H. P. Kriegel. On optimizing nearest neighbor queries in high-dimensional data spaces. In *Proc. of Int. Conf. on Database Theory*, pages 435–449, 2001.

[15] S. Berchtold, C. Böhm, D. A. Keim, H-P. Kriegel, and X. Xu. Optimal multidimensional query processing using tree striping. In *Proc. of Int. Conf. on Data Warehousing and Knowledge Discovery*, pages 244–257, 2000.

[16] S. Berchtold, C. Böhm, and H-P. Kriegel. Improving the query performance of high-dimensional index structures by bulk-load operations. In *Proc. of Int. Conf. on Extending Database Technology*, pages 216–230, 1998.

[17] S. Berchtold, C. Böhm, and H. P. Kriegel. The pyramid-tree : breaking the curse of dimensionality. In *Proc. of ACM SIGMOD Int. Conf. on Management of Data*, pages 142–153, 1998.

[18] S. Berchtold, D. A. Keim, and H-P. Kriegel. The x-tree : An index structure for high-dimensional data. In *Proc. of Int. Conf. on Very Large Data Bases*, pages 28–39, 1996.

[19] S-A. Berrani. Recherche approximative de plus proches voisins avec contrôle probabiliste de la précision ; application à la recherche d'images par le contenu. Thèse de doctorat, Université de Rennes, 2004.

[20] S-A. Berrani, L. Amsaleg, and P. Gros. Robust content-based image searches for copyright protection. In *Proc. of ACM Int. Workshop on Multimedia Databases*, pages 70–77, 2003.

[21] P. J. Besl and N. D. McKay. A method for registration of 3-d shapes. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 14(2) :239–256, 1992.

[22] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft. When is nearest neighbor meaningful ? In *Proc. of Int. Conf. on Database Theory*, pages 217–235, 1999.

[23] C. Böhm, B. Braunmüller, H-P. Kriegel, and M. Schubert. Efficient similarity search in digital libraries. In *Proc. of Int. Conf. on Advances in Digital Libraries*, pages 193–199, 2000.

[24] M. J. Black and P. Anandan. A framework for the robust estimation of optical flow. In *Proc. of Int. Conf. on Computer Vision*, pages 231–236, 1993.

[25] P. Bohannon, P. Mcllroy, and R. Rastogi. Main-memory index structures with fixed-size partial keys. In *Proc. of ACM SIGMOD Int. Conf. on Management of Data*, pages 163–174, 2001.

[26] C. Böhm. A cost model for query processing in high dimensional data spaces. *ACM Trans. on Database Systems*, 25(2) :129–178, 2000.

[27] C. Böhm, S. Berchtold, and D. A. Keim. Searching in high-dimensional spaces : Index structures for improving the performance of multimedia databases. *ACM Computing surveys*, 33(3) :322–373, 2001.

[28] N. Boujemaa, J. Fauqueur, M. Ferecatu, F. Fleuret, V. Gouet, B. Le Saux, and H. Sahbi. Ikona for interactive specific and generic image retrieval. In *Proc. of Workshop on Multi-Media Content Based Indexing and Retrieval*, 2001.

[29] J. Bourgain. On lipschitz embedding of finite metric spaces in hilbert space. *Israel Journal of Mathematics*, 52(1) :46–52, 1985.

[30] T. Bozkaya and M. Ozsoyoglu. Distance-based indexing for high-dimensional metric spaces. In *Proc. of Int. Conf. on Management of Data*, pages 357–368, 1997.

[31] T. Bozkaya and M. Ozsoyoglu. Indexing large metric spaces for similarity search queries. *ACM Trans. on Database Systems*, 24(3) :361–404, 1999.

[32] S. Bres and J. M. Jolion. Detection of interest points for image indexing. In *Proc. of Int. Conf. on Visual Information Systems*, pages 427–434, 1999.

[33] R. Brunelli, O. Mich, and C. M. Modena. A survey on the automatic indexing of video data. *Visual Communication and Image Representation*, 10(2) :78–112, 1999.

[34] W. A. Burkhard and R. M. Keller. Some approaches to best-match file searching. *Communications of the ACM*, 16(4) :230–236, 1973.

[35] B. Bustos, G. Navarro, and E. Chavez. Pivot selection techniques for proximity searching in metric spaces. In *Proc. of Conf. of the Chilean Computer Science Society*, pages 33–40, 2001.

[36] A. R. Butz. Alternative algorithm for hilbert's space-filling curve. *IEEE Trans. on Computers*, C(2) :424–426, 1971.

[37] P. Cano, E. Batle, T. Kalker, and J. Haitsma. A review of algorithms for audio fingerprinting. In *Proc. of IEEE Workshop on Multimedia Signal Processing*, pages 169–173, 2002.

[38] G. Carneiro and A. D. Jepson. Phase-based local features. In *Proc. of European Conf. on Computer Vision*, pages 282–296, 2002.

[39] C. Carson, M. Thomas, S. Belongie, J. M. Hellerstein, and J. Malik. Blobworld : A system for region-based image indexing and retrieval. In *Proc. of Int. Conf. on Visual Information Systems*, pages 509–516, 1999.

[40] G-H. Cha, X. Zhu, P. Petkovic, and C. Chung. An efficient indexing method for nearest neighbor searches in high-dirnensional image databases. *IEEE Trans. on Multimedia*, 4(1) :76–87, 2002.

[41] E. Chang, J. Wang, C. Li, and G. Wilderhold. Rime - a replicated image detector for the world-wide web. In *Proc. of SPIE Symp. of Voice, Video, and Data Communications*, pages 58–67, 1998.

[42] E. Chavez, G. Navarro, R. A. Baeza-Yates, and J. L. Marroquin. Searching in metric spaces. *ACM Computing Surveys*, 33(3) :273–321, 2001.

[43] S. Cheung and A. Zakhor. Estimation of web video multiplicity. In *Proc. of SPIE Conf. on Internet Imaging*, pages 34–46, 2000.

[44] S-C. S. Cheung and A. Zakhor. Fast similarity search on video signatures. In *Proc. of Int. Conf. on Image Processing*, pages 1–4, 2003.

[45] S. Cho and S. Yoo. Image retrieval using topological structure of user sketch. In *Proc. of Int. Conf. on Systems, Man, and Cybernetics*, pages 4584–4588, 1998.

[46] P. Ciaccia and M. Patella. Pac nearest neighbor queries : Approximate and controlled search in high-dimensional and metric spaces. In *Proc. of Int. Conf. on Data Engineering*, pages 244–255, 2000.

[47] P. Ciaccia and M. Patella. Approximate similarity queries : A survey. Tech. report, University of Bologna : MultiMedia DataBase Group, 2001.

[48] P. Ciaccia, M. Patella, and P. Zezula. M-tree : An efficient access method for similarity search in metric spaces. In *Proc. of Int. Conf. on Very Large Data Bases*, pages 426–435, 1997.

[49] K. L. Clarkson. Nearest neighbor queries in metric spaces. In *Proc. of ACM Symp. on Theory of computing*, pages 609–617, 1997.

[50] D. Comer. The ubiquitous b-tree. *ACM Computer Survey*, 11(2) :121–137, 1979.

[51] P. Comon. Independent component analysis, a new concept ? *Signal Processing*, 36(3) :287–314, 1994.

[52] R. Coudray and B. Besserer. Global motion estimation for mpeg-encoded streams. In *Proc. of Int. Conf. on Image Processing*, 2004.

[53] B. Cui, B. C. Ooi, J. Su, and K. Tan. Contorting high dimensional data for efficient main memory knn processing. In *Proc. of ACM SIGMOD Int. Conf. on Management of Data*, pages 479–490, 2003.

[54] M. Davy and S. J. Godsill. Audio information retrieval : A bibliographical study. Tech. report, university of cambridge, 2002.

[55] D. DeMenthon and D. Doermann. Video retrieval using spatio-temporal descriptors. In *Proc. of ACM Int. Conf. on Multimedia*, pages 508–517, 2003.

[56] R. Deriche and G. Giraudon. A computational approach for corner and vertex detection. *Computer Vision*, 10(2) :101–124, 1993.

[57] G. N. DeSouza and A. C. Kak. Vision for mobile robot navigation : A survey. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(2) :237–267, 2002.

[58] N. Dimitrova and M. Abdel-Mottaleb. Content-based video retrieval by example video clip. In *Proc. of SPIE Conf. on Storage and Retrieval for Image and Video Databases*, pages 59–70, 1997.

[59] D. S. Doermann, H. Li, and O. E. Kia. The detection of duplicates in document image databases. In *Proc. of Int. Conf. on Document Analysis and Recognition*, pages 314–318, 1997.

[60] J. Eakins and M. Graham. Content-based image retrieval. Tech. report, University of Northumbria, 1999.

[61] S. Eickeler and S. Müller. Content-based video indexing of tv broadcast news using hidden markov models. In *Proc. of Int. Conf. on Acoustics, Speech, and Signal Processing*, pages 2997–3000, 1999.

[62] R. Fablet and P. Bouthemy. Motion-based feature extraction and ascendant hierarchical classification for video indexing and retrieval. In *Proc. of Int. Conf. on Visual Information and Information Systems*, pages 221–228, 1999.

[63] C. Faloutsos. Gray codes for partial match and range queries. *IEEE Trans. on Software Engineering*, 14(10) :1381–1393, 1988.

[64] C. Faloutsos. *Searching Multimedia Databases by Content*. Kluwer Academic Publishers, 1996.

[65] C. Faloutsos and V. Gaede. Analysis of n-dimensional quadtrees using the Hausdorff fractal dimension. In *Proc. of Int. Conf. on Very Large Data Bases*, pages 40–50, 1996.

[66] C. Faloutsos and I. Kamel. Beyond uniformity and independence : Analysis of r-trees using the concept of fractal dimension. In *Proc. of ACM Symp. on Principles of Database Systems*, pages 4–13, 1994.

[67] C. Faloutsos and K-I. Lin. Fastmap : A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In *Proc. of ACM SIGMOD Int. Conf. on Management of Data*, pages 163–174, 1995.

[68] J. Fauqueur. Contributions pour la recherche d'images par composantes visuelles. Thèse de doctorat, Université de Versailles, 2003.

[69] H. Ferhatosmanoglu, E. Tuncel, D. Agrawal, and A. El Abbadi. Vector approximation based indexing for non-uniform high dimensional data sets. In *Proc. of ACM Int. Conf. on Information and Knowledge Management*, pages 202–209, 2000.

[70] M. A. Fischler and R. C. Bolles. Random sample consensus : a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6) :381–395, 1981.

[71] L. M. J. Florack, B. M. Haar-Romeny, J. J. Koenderink, and M. A. Viergever. General intensity transformations and differential invariants. *Journal of Mathematical Imaging and Vision*, 4(2) :171–187, 1994.

[72] J. Flusser and T. Suk. Degraded image analysis : An invariant approach. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(6) :590–603, 1998.

[73] W. Förstner. A framework for low level feature extraction. In *Proc. of European Conf. on Computer Vision*, pages 383–394, 1994.

[74] J. Fournier. Indexation d'images par le contenu et recherche interactive dans les bases généralistes. Thèse de doctorat, Université de Cergy-Pontoise, 2002.

[75] W. T. Freeman and E. H. Adelson. The design and use of steerable filters. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(9) :891–906, 1991.

[76] V. Gaede and O. Günther. Multidimensional access methods. *ACM Computing Surveys*, 30(2) :170–231, 1998.

[77] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *Proc. of Int. Conf. on Very Large Data Bases*, pages 518–529, 1999.

[78] K-S. Goh, B. Li, and E. Chang. Dyndex : a dynamic and non-metric space indexer. In *Proc. of ACM Int. Conf. on Multimedia*, pages 466–475, 2002.

[79] J. Goldstein and R. Ramakrishnan. Contrast plots and p-sphere trees : Space vs. time in nearest neighbour searches. In *Proc. of Int. Conf. on Very Large Data Bases*, pages 429–440, 2000.

[80] L. J. Van Gool, T. Moons, and D. Ungureanu. Affine photometric invariants for planar intensity patterns. In *Proc. of European Conf. on Computer Vision*, pages 642–651, 1996.

[81] V. Gouet and N. Boujemaa. Object-based queries using color points of interest. In *Proc. of IEEE Workshop on Content-Based Access of Image and Video Libraries*, pages 30–36, 2001.

[82] V. Gouet, P. Montesinos, R. Deriche, and D. Pelé. Evaluation de détecteurs de points d'intérêt pour la couleur. In *Actes de la Conf. Reconnaissance des formes et Intelligence Artificielle*, pages 257–266, 2000.

[83] A. Gray. A statistical approach to algorithmic analysis of high-dimensional nearest-neighbor search. Technical report, School of Computer Science, 2004.

[84] A. Guttman. R-trees : A dynamic index structure for spatial searching. In *Proc. of ACM SIGMOD Conf. of Management of Data*, pages 47–57, 1984.

[85] J. Haitsma, T. Kalker, and J. Oostveen. Robust audio hashing for content identification. In *Proc. of Int. Workshop. on Content-Based Multimedia Indexing*, pages 329–343, 2001.

[86] A. Hampapur and R. Bolle. Feature based indexing for media tracking. In *Proc. of Int. Conf. on Multimedia Computing and Expo*, pages 1709–1712, 2000.

[87] A. Hampapur and R. Bolle. Comparison of sequence matching techniques for video copy detection. In *Proc. of Conf. on Storage and Retrieval for Media Databases*, pages 194–201, 2002.

[88] C. Harris and M. Stephens. A combined corner and edge detector. In *Proc. of Alvey Vision Conf.*, pages 147–151, 1988.

[89] F. Heitger, L. Rosenthaler, R. Heydt, and O. Kubler. Simulation of neural contour mechanisms : From simple to end-stopped cells. *Vision Research*, 32(5) :963–981, 1992.

[90] A. Henrich. The lsd^h-tree : An access structure for feature vectors. In *Proc. of Int. Conf. on Data Engineering*, pages 362–369, 1998.

[91] A. Henrich, H-W. Six, and P. Widmayer. The lsd tree : Spatial access to multidimensional point and nonpoint objects. In *Proc. of Int. Conf. on Very Large Data Bases*, pages 45–53, 1989.

[92] G. R. Hjaltason and H. Samet. Ranking in spatial databases. In *Proc. of Symp. on Large Spatial Databases*, pages 83–95, 1995.

[93] T. C. Hoad and J. Zobel. Fast video matching with signature alignment. In *Proc. of ACM SIGMM Int. Workshop on Multimedia information retrieval*, pages 262–269, 2003.

[94] T. C. Hoad and J. Zobel. Video similarity detection for digital rights management. In *Proc. of Australasian Conf. on research and practice in information technology*, pages 237–245, 2003.

[95] R. Horaud, T. Skordas, and F. Veillon. Finding geometric and relational structures in an image. In *Proc. of European Conf. on Computer Vision*, pages 374–384, 1990.

[96] G. Hristescu and M. Farach-Colton. Cluster-preserving embedding of proteins. Tech. report, Rutgers University, 1999.

[97] J. Illingworth and J. Kittler. A survey of the hough transform. *Graphical Model and Image Processing*, 44(1) :87–116, 1988.

[98] P. Indyk, G. Iyengar, and N. Shivakumar. Finding pirated video sequences on the internet. Tech. report, Stanford University, 1999.

[99] P. Indyk and R. Motwani. Approximate nearest neighbors : towards removing the curse of dimensionality. In *Proc. of ACM Symp. on Theory of Computing*, pages 604–613, 1998.

[100] H. V. Jagadish. Linear clustering of objects with multiple attributes. In *Proc. of ACM SIGMOD Int. Conf. on Management of Data*, pages 332–342, 1990.

[101] A. Jaimes, S-F. Chang, and A. C. Loui. Duplicate detection in consumer photography and news video. In *Proc. of ACM Int. Conf. on Multimedia*, pages 423–424, 2002.

[102] A. K. Jain, R. P. W. Duin, and J. Mao. Statistical pattern recognition : A review. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(1) :4–37, 2000.

[103] H. R. Jaltason. Properties of embedding methods for similarity searching in metric spaces. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 25(5) :530–549, 2003.

[104] A. Joly, C. Frélicot, and O. Buisson. Robust content-based video copy identification in a large reference database. In *Int. Conf. on Image and Video Retrieval*, pages 414–424, 2003.

[105] I. Kamel and C. Faloutsos. Hilbert r-tree : An improved r-tree using fractals. In *Proc. of Int. Conf. on Very Large Data Bases*, pages 500–509, 1994.

[106] N. Katayama and S. Satoh. The sr-tree : An index structure for high-dimensional nearest neighbor queries. In *Proc. of ACM SIGMOD Int. Conf. on Management of Data*, pages 369–380, 1997.

[107] K. Kim, S. K. Cha, and K. Kwon. Optimizing multidimensional index trees for main memory access. *SIGMOD Records*, 30(2) :139–150, 2001.

[108] V. Kobla and D. Doermann. Extraction of features for indexing mpegcompressed video. In *Proc. of Workshop on Multimedia Signal Processing*, pages 337–342, 1997.

[109] B. Kobus, F. Graham, and F. Brian. Colour constancy for scenes with varying illumination. *Computer Vision and Image Understanding*, 65(2) :311–321, 1996.

[110] J. B. Kruskal. Multidimensional scaling and other methods for discovering structure. In *Statistical Methods for Digital Computers*, pages 296–339. John Wiley and Sons, 1977.

[111] I. Laptev and T. Lindeberg. Space-time interest points. In *Proc. of Int. Conf. on Computer Vision*, pages 432–439, 2003.

[112] J. Lawder. The application of space-filling curves to the storage and retrieval of multi-dimensional data. Phd thesis, University of London, 1999.

[113] J. K. Lawder and P. J. H. King. Querying multi-dimensional data indexed using the hilbert space-filling surve. *SIGMOD Record*, 30(1) :19–24, 2001.

[114] C. Li, E. Chang, M. Garcia-Molina, and G. Wiederhold. Clustering for approximate similarity search in high-dimensional spaces. *IEEE Trans. on Knowledge and Data Engineering*, 14(4) :792–808, 2002.

[115] R. Lienhart, W. Effelsberg, and R. Jain. Visualgrep : A systematic method to compare and retrieve video sequences. In *Proc. of SPIE Conf. on Storage and Retrieval for Image and Video Databases*, pages 271–283, 1998.

[116] R. Lienhart, C. Kuhmunch, and W. Effelsberg. On the detection and recognition of television commercials. In *Proc. of Int. Conf. on Multimedia Computing and Systems*, pages 509–516, 1997.

[117] K-I. Lin, H. V. Jagadish, and C. Faloutsos. The tv-tree : An index structure for high-dimensional data. *Very Large DataBase Journal*, 3(4) :517–542, 1994.

[118] T. Lindeberg. Scale-space theory : A basic tool for analysing structures at different scales. *Applied Statistics*, 21(2) :224–270, 1994.

[119] X. Liu, Y. Zhuang, and Y. Pan. A new approach to retrieve video by example video clip. In *Proc. of ACM Int. Conf. on Multimedia*, pages 41–44, 1999.

[120] E. Loupias and N. Sebe. Wavelet-based salient points for image retrieval. Tech. report, Laboratoire Reconnaissance de Formes et Vision, 1999.

[121] D. G. Lowe. Object recognition from local scale-invariant features. In *Proc. of Int. Conf. on Computer Vision*, pages 1150–1157, 1999.

[122] J. Malki, N. Boujemaa, C. Nastar, and A. Winter. Region queries without segmentation for image retrieval by content. In *Proc. of Int. Conf. on Visual Information and Information Systems*, pages 115–122, 1999.

[123] S. Manegold, P. Boncz, and M. Kersten. Generic database cost models for hierarchical memory systems. In *Proc. of Int. Conf. Very Large Data Bases*, pages 191–202, 2002.

[124] J. Martinez and E. Loisant. Browsing image databases with galois lattices. In *Proc. of ACM Symp. on Applied Computing*, pages 791–795, 2002.

[125] K. Mikolajczyk and C. Schmid. Indexing based on scale invariant interest points. In *Proc. of Int. Conf. on Computer Vision*, pages 525–531, 2001.

[126] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, pages 257–263, 2003.

[127] R. Milanese, F. Deguillaume, and A. Jacot-Descombes. Video segmentation and camera motion characterization using compressed data. In *Proc. of SPIE Conf. on Multimedia Storage and Archiving Systems*, pages 79–89, 1997.

[128] S. Mitra, S. K. Pal, and P. Mitra. Data mining in soft computing framework : A survey. *IEEE Trans. on Neural Networks*, 13(1) :3–14, 2002.

[129] B. Moghaddam, H. Biermann, and D. Margaritis. Defining image content with multiple regions-of-interest. In *Proc. of IEEE Workshop on Content-based Access of Image and Video Libraries*, pages 89–98, 1999.

[130] P. Montesinos, V. Gouet, and R. Deriche. Differential invariants for color images. In *Proc. of Int. Conf. on Pattern Recognition*, pages 838–840, 1998.

[131] B. Moon, H. V. Jagadish, C. Faloutsos, and J. H. Saltz. Analysis of the clustering properties of the hilbert space-filling curve. *Trans. on Knowledge and Data Engineering*, 13(1) :124–141, 2001.

[132] W. Niblack, R. Barber, W. Equitz, M. Flickner, E. Glasman, D. Petkovic, P. Yanker, C. Faloutsos, and G. Taubin. The qbic project : Quering images by content using color, texture, and shape. In *Proc. of SPIE Conf. on Storage and Retrieval for Image and Video Databases*, pages 173–181, 1993.

[133] J. Oostveen, T. Kalker, and J. Haitsma. Feature extraction and a database strategy for video fingerprinting. In *Proc. of Int. Conf. on Visual Information and Information Systems*, pages 117–128, 2002.

[134] E. J. Pauwels and G. Frederix. Finding salient regions in images : nonparametric clustering for image segmentation and grouping. *Computer Vision and Image Understanding*, 75(1) :73–85, 1999.

[135] A. Pentland, R. Picard, and S. Sclaroff. Photobook : Content-based manipulation of image databases. In *Proc. of SPIE Conf. on Storage and Retrieval for Image and Video Databases*, pages 34–47, 1994.

[136] E. G. M. Petrakis, C. Faloutsos, and K-I. Lin. Imagemap : An image indexing method based on spatial similarity. *Trans. on Knowledge and Data Engineering*, 14(5) :979–987, 2002.

[137] R. Radhakrishnan, Z. Xiong, and N. D. Memom. Security of visual hash function. In *Proc. of SPIE Conf. on Electronic Imaging, Security and Watermarking of Multimedia Contents*, pages 644–652, 2003.

[138] D. Reisfeld, H. Wolfson, and Y. Yeshurun. Context-free attentional operators : the generalized symmetry transform. *Computer Vision*, 14(2) :119–130, 1995.

[139] J. T. Robinson. The k-d-b-tree : a search structure for large multidimensional dynamic indexes. In *Proc. of ACM SIGMOD Int. Conf. on Management of Data*, pages 10–18, 1981.

[140] N. Roussopoulos, S. Kelley, and F. Vincent. Nearest neighbor queries. In *Proc. of ACM SIGMOD Int. Conf. on Management of Data*, pages 71–79, 1995.

[141] Y. Rubner. Perceptual metrics for image database navigation. Tech. report, Stanford University, 1999.

[142] Y. Rui, T. Huang, and S. Chang. Image retrieval : current techniques, promising directions and open issues. *Visual Communication and Image Representation*, 10(4) :39–62, 1999.

[143] Y. Rui, T. Huang, and S. Mehrotra. Content-based image retrieval with relevance feedback in mars. In *Proc. of Int. Conf. on Image Processing*, pages 815–818, 1997.

[144] Y. Rui, A. She, and T. Huang. Modified fourier descriptors for shape representation - a practical approach. In *Proc. of Int. Workshop on Image Databases and Multi Media Search*, 1996.

[145] M. Rukoz, M. Manouvrier, and G. Jomier. Distances de similarité d'images basées sur les arbres quaternaires. In *Actes des 18 èmes Journées Bases de Données Avancées*, pages 307–326, 2002.

[146] H. Sagan. *Space-filling curves.* Springer-Verlag, 1994.

[147] J. M. Sanchez, X. Binefa, J. Vitria, and P. Radeva. Local color analysis for scene break detection applied to tv commercials recognition. In *Proc. of Int. Conf. on Visual Information and Information Systems*, pages 237–244, 1999.

[148] F. Schaffalitzky and A. Zisserman. Multi-view matching for unordered image sets, or "how do i organize my holiday snaps ?". In *Proc. of European Conf. on Computer Vision*, pages 414–431, 2002.

[149] C. Schmid. Appariement d'images par invariants locaux de niveaux de gris. Thèse de doctorat, Institut National Polytechnique de Grenoble, 1996.

[150] C. Schmid and R. Mohr. Local grayvalue invariants for image retrieval. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(5) :530–535, 1997.

[151] C. Schmid, R. Mohr, and C. Bauckhage. Evaluation of interest point detectors. *Computer Vision*, 37(2) :151–172, 2000.

[152] E. Di Sciascio and M. Mongiello. Query by sketch and relevance feedback for content-based image retrieval over the web. *Journal of Visual Languages and Computing*, 10(6) :565–584, 1999.

[153] N. Sebe and M. S. Lew. Comparing salient point detectors. *Pattern Recognition Letters*, 24(1) :89–96, 2003.

[154] T. K. Sellis, N. Roussopoulos, and C. Faloutsos. The r+-tree : A dynamic index for multi-dimensional objects. In *Proc. of Int. Conf. on Very Large Data Bases*, pages 507–518, 1987.

[155] C. J. Setchell and N. W. Campbell. Using colour gabor texture features for scene understanding. In *Proc. of Int. Conf. on Image Processing and its Applications*, pages 372–376, 1999.

[156] P. Y. Simard, Y. LeCun, J. S. Denker, and B. Victorri. Transformation invariance in pattern recognition - tangent distance and tangent propagation. *Imaging Systems and Technology*, 11(3) :181–194, 2000.

[157] A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain. Content-based image retrieval at the end of the early years. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(12) :1349–1380, 2000.

[158] J. Smith. Integrated spatial and feature image systems : Retrieval. Phd thesis, Columbia University, 1997.

[159] C. G. M. Snoek and M. Worring. A state-of-the-art review on multimodal video indexing. In *Proc. of Conf. of the Advanced School for Computing and Imaging*, pages 194–202, 2002.

[160] C. V. Stewart. Robust parameter estimation in computer vision. *SIAM Review*, 41(3) :513–537, 1999.

[161] M. Stricker and M. Orengo. Similarity of color images. In *Proc. of SPIE Conf. on Storage and Retrieval for Image and Video Databases*, pages 381–392, 1995.

[162] M. J. Swain and D. H. Ballard. Color indexing. *Computer Vision*, 7(1) :11–32, 1991.

[163] J-P. Tarel and S. Boughorbel. On the choice of similarity measures for image retrieval by example. In *Proc. of ACM Int. Conf. on Multimedia*, pages 446–455, 2002.

[164] C. Taskiran, J. Chen, A. Albiol, L. Torres, C. A. Bouman, and E. J. Delp. Vibe : A compressed video database structured for video active browsing and search. *IEEE Trans. on Multimedia*, 6(1) :103–118, 2004.

[165] E. Tuncel, H. Ferhatosmanoglu, and K. Rose. Vq-index : An index structure for similarity searching in multimedia databases. In *Proc. of ACM Int. Conf. on Multimedia*, pages 543–552, 2002.

[166] R. C. Veltkamp and M. Tanase. Content-based image retrieval systems : A survey. Tech. report, Department of Computing Science, Utrecht University, 2000.

[167] R. Venkatesan, S. Koon, M. Jakubowski, and P. Moulin. Robust image hashing. In *Proc. of Int. Conf. on Image Processing*, pages 664–666, 2000.

[168] X. Wang, J. T-L. Wang, K-I. Lin, D. Shasha, B. A. Shapiro, and K. Zhang. An index structure for data mining and clustering. *Knowledge and Information Systems*, 2(2) :161–184, 2000.

[169] R. Weber and K. Böhm. Trading quality for time with nearest neighbor search. In *Proc. of Int. Conf. on Extending Database Technology*, pages 21–35, 2000.

[170] R. Weber, H. J. Schek, and S. Blott. A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In *Proc. of Int. Conf. on Very Large Data Bases*, pages 194–205, 1998.

[171] Y. Weiss. Motion segmentation using em - a short tutorial. Tutorial, Massachusetts Institute of Technology, 1997.

[172] D. A. White and R. Jain. Similarity indexing with the ss-tree. In *Proc. of Int. Conf. on Data Engineering*, pages 516–523, 1996.

[173] A. Winter and C. Nastar. Differential feature distribution maps for image segmentation and region queries in image databases. In *Proc. of IEEE Workshop on Content-based Access of Image and Video Libraries*, pages 13–23, 1999.

[174] H. J. Wolfson and I. Rigoutsos. Geometric hashing : An overview. *Computational Science and Engineering*, 4(4) :10–21, 1997.

[175] Y. Wu, Y. Zhuang, and Y. Pan. Content-based video similarity model. In *Proc. of ACM Int. Conf. on Multimedia*, pages 465–467, 2000.

[176] M-H. Yang, D. J. Kriegman, and N. Ahuja. Detecting faces in images : A survey. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(1) :34–58, 2002.

[177] P. N. Yianilos. Data structures and algorithms for nearest neighbor search in general metric spaces. In *Proc. of ACM Symp. on Discrete algorithms*, pages 311–321, 1993.

[178] P. Zezula, P. Savino, G. Amato, and F. Rabitti. Approximate similarity retrieval with m-trees. *Very Large Data Bases Journal*, 7(4) :275–293, 1998.

[179] T. Zhang, R. Ramakrishnan, and M. Livny. Birch : An efficient data clustering method for very large databases. In *Proc. of ACM SIGMOD Int. Conf. on Management of Data*, pages 103–114, 1996.

[180] Q. Zhou, Q. Li, and H. J. Zhang. A practitioner's approach to multi-dimensional indexing. In *Proc. of Int. Conf. on Distributed Multimedia Systems*, pages 257–265, 2001.

[181] B. Zitova and J. Flusser. Image registration methods : a survey. *Image and Vision Computing*, 21(11) :977–1000, 2003.

Références de l'auteur

Congrès internationaux

A. Joly, C. Frélicot, and O. Buisson. Robust content-based video copy identification in a large reference database. In *Int. Conf. on Image and Video Retrieval*, 2003.

A. Joly, C. Frélicot, and O. Buisson. Feature statistical retrieval applied to content-based copy identification. In *Int. Conf. on Image Processing*, 2004.

A. Joly, C. Frélicot, and O. Buisson. Statistical similarity search applied to content-based video copy detection. In *IEEE Int. Workshop on Managing Data for Emerging Multimedia Applications*, 2005.

Brevet

A. Joly. Search of similar features representing objects in a large reference database. US patent pending, 2004.

Résumé

Le domaine de l'indexation vidéo par le contenu s'intéresse à l'ensemble des techniques utiles pour analyser et exploiter des stocks de vidéos sans passer par des descriptions textuelles extérieures aux documents dont on dispose. Plus particulièrement, les travaux de cette thèse traitent du problème de la détection de copies basée sur le contenu. Pour résoudre conjointement les problèmes de qualité et de rapidité de la détection, liés à l'augmentation de la taille du catalogue de référence, nous avons proposé une méthode complète et efficace. Celle-ci tient compte à la fois des aspects traitement de l'image, des aspects base de données et de leurs interactions.

La première partie du mémoire est consacrée à la présentation du contexte particulier de la détection de copies en vidéo et aux signatures utilisées pour caractériser le contenu des vidéos. L'originalité de notre approche est qu'elle est basée sur l'utilisation conjointe de signatures locales et d'une mesure de similarité globale calculée après la recherche des signatures similaires dans la base. Cette mesure globale n'est pas un simple vote comme dans les approches classiques car elle est précédée d'une étape de recalage originale entre l'objet candidat et les objets retournés par la recherche dans la base.

La deuxième partie présente le coeur théorique du travail. Il s'agit d'une nouvelle méthode d'indexation et de recherche de descripteurs numériques s'intégrant dans le cadre de la recherche par similarité approximative. Il a en effet récemment été montré qu'une faible perte contrôlée dans la qualité des résultats de la recherche pouvait permettre des accélérations importantes du temps de recherche. Le principe de la technique présentée est d'étendre ce paradigme à la recherche à ϵ-près, contrairement aux autres approches qui s'intéressent uniquement à la recherche approximative des K plus proches voisins. L'originalité est de déterminer les régions pertinentes de l'espace selon un modèle théorique des distorsions que subissent les descripteurs, par des requêtes dites statistiques. Seule une portion de l'espace donnant une probabilité forte et contrôlée de trouver la réponse cherchée est visitée. Celle-ci est déterminée par une courbe de Hilbert et la partition qu'elle induit, simplifiant ainsi fortement l'accès à la base de descripteurs. L'évaluation expérimentale de la technique a montré que celle-ci est sous-linéaire avec un comportement asymptotique linéaire (mais que l'on observe que pour des tailles de base énormes) et que les performances en qualité sont stables. Il est également montré que les requêtes statistiques apportent une accélération conséquente par rapport aux requêtes à ϵ-près exactes.

La troisième partie est consacrée à l'évaluation du système dans son ensemble et à la présentation de trois applications. Les expérimentations ont tout d'abord montré que le modèle théorique, bien que simple, permet un contrôle suffisant de la probabilité de retrouver un descripteur dans la pratique. Elles ont ensuite montré que la recherche approximative des descripteurs était particulièrement rentable lorsque l'on utilise des signatures locales puisque la perte de certains descripteurs n'influencent que très peu la qualité globale de la détection tout en accélérant fortement la recherche. Il a enfin été montré que la méthode globale était quasiment invariante à de très fortes augmentations de la quantité de vidéos dans la base (jusqu'à trois ordres de grandeur). L'approche proposée a été complètement intégrée et validée dans un système applicatif réel dont l'ampleur est sans précédent (le catalogue de référence contient jusqu'à 40 000 heures de vidéo, soit 500 fois plus que la moyenne des quantités utilisées dans l'état de l'art). Cela a soulevé des questionnements relatifs à l'utilisation des résultats issus de catalogues de référence aussi volumineux et d'envisager des pistes pour en extraire des informations de nature sémantique.

Abstract

Content-based video indexing deals with techniques used to analyse and to exploit video databases without needs of any additional textual description. The work presented in this report is focused more precisely on content-based video copy detection, which is one of the emerging multimedia applications for which there is a need of a concerted effort from the database community and the computer vision community. To overcome the difficulties due to the use of very large databases, both in terms of robustness and speed, we propose a complete original and efficient strategy. The first part of this report presents the particular context of copy detection and the signatures used to describe the content of the videos. The originality of our method is that it is based both on local signatures and on a global similarity measure computed after the search in the signatures database. This similarity measure is not only a vote like other classical local approaches but it includes a registration step between candidate objects and objects retrieved by the search. The second part presents the main contribution of the thesis : A new indexing and retrieval technique belonging to the approximate similarity search techniques family. Recent works shows that trading quality for time can be widely profitable to speed-up descriptors similarity search. Whereas all other approximate techniques deal with K Nearest Neighbors search, the principle of our method is to extend the approximate paradigm to range queries. The main originality consists in determining relevant regions of the space according a theoritical model for the distortions undergone by the signatures. The method allows to determine the optimal region of the space with a high controlled probability to contain the good answer. This search paradigm is called *statistical query*. In practice, to simplify the access to signatures, the relevant regions are determined by using an Hilbert space filling curve and the space partition that induces. The experiments show that the technique is sublinear in database size with an assymptotically linear behavior (but only for huge databases) and that the quality performances are stable. Furthermore, they highlight that statistical queries provide a very high speed-up compared to classical exact range queries. The third part is focused on the global system assessment and the description of three applications. The experiments show that the simple theoretical distortion model is efficient enough to control the effective probability to retrieve a descriptor. They also point out that approximate similarity search is particularly profitable when using local signatures since the lost of some search results does not affect the global robustness of the detection. Furthermore, the detection results are almost invariant to strong database size growing (three orders of magnitude). The proposed approach was integrated in a difered real-time TV monitoring system which is able to control 40 000 hours of videos. The high quantity and variability of the results of this system open new data mining perspectives.