N^o Ordre : 2909

THÈSE

présentée

DEVANT L'UNIVERSITÉ DE RENNES 1

pour obtenir

le grade de : *DOCTEUR DE L'UNIVERSITÉ DE RENNES 1*

Mention : *Informatique*

PAR

Marc CHAUMONT

Équipe d'accueil : TEMICS/IRISA/INRIA

École Doctorale : Mathématiques, Informatique, Signal, Électronique, Télécommunications (MATISSE)

Composante universitaire : Institut de Formation Supérieur en Informatique et Communication (IFSIC)

TITRE DE LA THÈSE :

Représentation en objets vidéo pour un codage progressif et concurrentiel des séquences d'images

Soutenue le 13 novembre 2003 devant la commission d'Examen

COMPOSITION DU JURY

M.	Claude	LABIT	Président
M.	Michel	BARLAUD	Rapporteur
M.	Jean-Marc	CHASSERY	Rapporteur
M.	Atilla	BASKURT	Examinateur
Mme.	Cécile	DUFOUR	Examinateur
M.	Henri	NICOLAS	Examinateur
M.	Stéphane	PATEUX	Invité

À Boubou

Remerciements

Je tiens à exprimer mes remerciements aux membres du jury :

- à M. Claude Labit pour avoir accepté de présider le jury de thèse,
- à M. Michel Barlaud et M. Jean-Marc Chassery pour avoir accepté d'être les rapporteurs de cette thèse, et pour leurs remarques constructives.
- à M. Atilla Baskurt, MMe. Cecile Dufour pour avoir bien voulu juger ce travail.

Je remercie aussi les personnes qui m'ont aidé à réaliser cette thèse :

- Christine Guillemot pour m'avoir proposé une solution lorsque j'ai décidé de stopper l'étude de mon premier sujet de thèse,
- Stéphane Pateux pour les nombreuses discussions que j'ai pu avoir avec lui. Je le remercie particulièrement pour m'avoir fait confiance et avoir accepté de me proposer un sujet de thèse ;
- Henri Nicolas, qui a dirigé ma thèse,
- les membres (et ex-membres) du projet Temics.

Je remercie tous mes amis et les personnes que j'ai côtoyés pendant ces 3 années et qui m'ont permis de passer des moments agréables. Enfin, je remercie ma famille.

Table des matières

Table des figures

Glossaire

C3D	*Clustering 3D* Algorithme de segmentation spatio-temporel en objet vidéo.
CIF	*Common Intermediate Format* Format d'images pour la vidéo 352 × 288.
CAE	*Context-based Arithmetic Coding* Codage entropique
CABAC	*Context-based Adaptive Binary Arithmetic Coding* Codage entropique avec prédiction des symboles à coder par utilisation d'un contexte adaptatif.
DC	*Dynamic Coding* mise en concurence de technique de codage
DCT	*Discrete Cosine Transform*
DFD	*Displaced Frame difference*
DPCM	*Differential Pulse Coding Modulation* Mode de codage différentiel. On parle de MICD en français.
DWT	*Discrete Wavelet Transform*
EBCOT	*Embedded Block Coding with Optimized Truncation [Taubman 00]* Algorithme de compression d'images fixes (utilisé dans JPEG-2000).
EQ	*Erreur Quadratique*
EQM	*Erreur Quadratique Moyenne*
EZW	*Embedded Zerotree of Wavelet [Shapiro 93]* Algorithme de compression d'images fixes.
FGS	*Fine Granular Scalability*
GOP	*Group Of Picture*
H.26X	*H.261, H.262, H.263X, H.26L* Ce sont des recommandations de compression vidéo numérique bas débit issues de l'ITU. Notons que H.264/AVC est issue du groupe de travail commun ISO et ITU et prolonge la norme H.26L.

INTRA (I)	*Mode de codage sans utilisation de prédiction temporelle*
INTER (P,B)	*Mode de codage avec utilisation de prédiction(s) temporelle(s)* Le mode INTER regroupe le codage par Prédiction simple (P) et par prédiction Bidirectionnel (B)
IPB	*Codage en Intra, en Prédiction Simple,* *et en Prédiction Bidirectionnelle pour un GOP*
ISO	*Organisation Internationale de normalisation*
ITU	*International Telecommunications Union* Institut de normalisation des Télécommunications.
JPEG	*Joint Picture Expert Group* JPEG et JPEG 2000 sont des normes de compression d'images fixes.
LZW	*Lempel Ziv Welsh* Codage par substitution. Il y a prise en compte de motifs répétitifs
MDL	*Minimum Descrition Lenght [Rissanen 78]* Formalisme permettant d'intoduire le coût de codage par exemple dans la modélisation du problème de segmentation.
MICD	*Modulation par Implusion Codées Différentielles* Mode de codage différentiel. On parle de DPCM en anglais.
MPEG	*Motion Picture Expert Group* MPEG-1, MPEG-2 et MPEG-4 sont des normes de compression numérique. MPEG-7 est une norme de description d'outils audio-visuels.
OBASC	*Object Based Analysis-Synthesis Coder* Codeur basé objet par analyse-synthèse.
PSNR	*Peak Signal to Noise Ratio* Mesure de la qualité visuelle d'un signal après distortion. Pour une image en niveaux de gris codée sur 8 bits on a : $PSNR = -10 \times log_{10}\left(\frac{EQM}{255^2}\right)$
QoS	*Quality of Service*
SPIHT	*Set Partitioning In Hierarchical Tree [Said et al. 96]* Algorithme de compression d'images fixes.
SVH	*Système de Vision Humain* Suite de traitements réalisés par l'Homme pour analyser les informations visuelles.
SA	*Shape Adapted* Un codage adapté a la forme On parle souvent de SA-DCT ou SA-WLT.
VO	*Video Object* Objet vidéo (texture,mouvement,forme) défini pour un intervalle de temps dans MPEG-4
VOP	*Video Object Plan* Objet vidéo (texture,mouvement,forme) défini à un instant donné
WLT	*Wavelet* Transformation ondelette
WLT 3D	*Wavelet 3D* Schéma de codage spatio-temporel par ondelette.

Introduction

Transmettre des flux vidéo sur un réseau ou bien les stocker sur disque nécessite une compression préalable, du fait de l'importance du volume d'information impliqué. Par exemple, le débit sans compression pour un film transmis à la télévision est de 237 Mb/s et pour une heure et demi de film la taille de fichier est de 1,22 térabits. Un tel débit et un tel volume sont trop importants.

La réduction de la taille des flux vidéo passe par des techniques de compression avec perte. Des standards de compression et des recommandations ont été proposés : MPEG1, MPEG2, MPEG4, H261, H262, H263, H263+, H263++, H264/AVC. La plupart de ces approches fonctionne sur une technique qui partitionne l'image en blocs pour ensuite effectuer des traitements sur chacun des blocs.

Parallèlement aux standards, d'autres techniques de codage ont été proposées prenant en compte de manière plus prononcée le contenu des images. Ces approches recherchent la présence de modèles (mouvement ou 3D) dans l'image pour ensuite coder la séquence en utilisant le modèle obtenu. Ainsi, le codage de la vidéo nécessite deux étapes : l'analyse (recherche d'un ou plusieurs modèles) puis la synthèse (le codage).

Par ailleurs, la compression vidéo doit s'adapter aux nombreuses applications et aux nombreux supports de diffusion. En effet, chaque application et chaque support ont leur propres besoins. Il y a par exemple des besoins en gamme de débit, en qualité requise, en robustesse aux pertes, d'adaptation aux variations de la bande passante, de synchronisation, de résolution de la séquence etc. Ainsi, pour les applications de télévisions numérique avec diffusion par satellite, on souhaite du haut débit à grande qualité. On utilise pour cela un codage MPEG2. H264/AVC est d'ailleurs étudié pour remplacer ce standard car il donne des performances bien meilleures. De la même façon, on peut émettre des vidéos sur des réseaux comme l'Internet ou le GSM, où on a besoin de faibles débits avec une adaptation aux variations de la bande passante et une robustesse aux pertes. Les applications de consultations de bases de donnée nécessitent aussi une compression adaptée permettant l'accès à différentes résolutions temporelles et spatiales.

L'étude présentée dans ce manuscrit s'inscrit dans la recherche de techniques de codage par modèle objet. Notre objectif est de mesurer l'intérêt de techniques basées objet. Plus précisément, nous nous intéressons au problème de segmentation automatique et semi-automatique (extraction d'objets vidéo) et aux problèmes de codage des objets vidéo.

Dans cette étude, nous nous posons 2 questions. La première est de savoir si la segmentation en objets permet de gagner en efficacité de compression; la deuxième est de savoir s'il est envisageable d'obtenir des objets de manière automatique utilisable pour le codage. Des réponses positives permettront d'envisager des solutions alternatives au codage défini par les standards, mais aussi de proposer un codage plus proche de la physique de la scène.

Pour répondre à ces questions, trois points seront abordés : la segmentation en objets vidéo, le codage hiérarchique d'un objet vidéo, et le codage dynamique d'objets vidéo.

La segmentation en objets vidéo est intéressante pour le codage car elle permet, grâce à l'obtention d'objets vidéo, une répartition des débits entre objets. Elle est d'ailleurs aussi utilisée pour des domaines comme l'indexation vidéo (indexation par objet) et pour le montage vidéo (modification, suppression, ajout d'objets vidéo).

Le codage hiérarchique vidéo est une fonctionnalité qui est utile pour les applications réseaux et pour le stockage de données. En effet, l'ordonnancement des informations du flux vidéo selon un critère d'ordre permet d'avoir une flexibilité sur la quantité d'information souhaitée.

Enfin, le codage dynamique permet de mettre en concurrence plusieurs codeurs pour coder un objet. Cette approche permet ainsi de valoriser l'approche objet en adaptant le codage aux propriétés de chaque objet.

Ce mémoire est composé de 3 parties : la segmentation, le codage hiérarchique et le codage dynamique. Au début de chacune de ces parties un état de l'art du domaine est présenté. Ensuite, nous présentons nos contributions. Enfin, les troisièmes chapitres exposent les résultats et en donnent une analyse critique.

Avant d'aborder ces trois parties que sont la segmentation en objet vidéo, le codage d'objet vidéo hiérarchique et le codage dynamique d'objet vidéo, nous proposons un chapitre de préambule permettant de positionner le manuscrit.

Préambule

Ce préambule évoque les différentes tendances du codage vidéo. Il s'agit d'une réflexion permettant d'introduire le plan du manuscrit et de cibler l'étude. En effet, deux grands domaines du traitement du signal sont abordés : la segmentation de contenu vidéo et le codage bas débit. Nous donnons également ici un aperçu quant aux évolutions propre à chacun de ces deux domaines.

Ainsi, ce préambule est composé de 4 parties. La première partie introduit la notion de segmentation sous forme chronologique en montrant l'évolution vers des représentations sémantiquement riches. La deuxième partie aborde les fonctionnalités de hiérarchisation [1] proposées maintenant par les codeurs vidéos et adaptées aux besoins des applications. La troisième partie présente le codage, à proprement parler, avec la mise en évidence d'une adaptation au signal à différentes échelles : blocs, macroblocs, régions, images. Enfin, la quatrième partie introduit le codage vidéo dynamique basé objet qui consiste à choisir la représentation et le codeur les plus adaptés pour le codage des objets.

2D, 2D 1/2, 3D : vers une représentation compacte et sémantique

Le domaine du codage vidéo nécessite une représentation compacte des informations. L'histoire du codage vidéo montre que les représentations ont tendance à aller vers une structuration sémantique des données. Ainsi, en une quarantaine d'années, les études ont évoluées de la notion de signal à la notion de modèle 3D.

Les premières approches de codage de flux vidéos étaient uniquement basées sur la compression d'une suite de symboles avec des outils de compression entropique (Huffman, LZW, ...). Les approches suivantes ont raisonné sur le signal vidéo lui-même en le distordant, c'est-à-dire en autorisant des pertes par rapport au signal original. La distorsion introduite s'appuie sur des critères fréquentiels, spatiaux et psychovisuels [ISO/IEC 93]. En permettant le codage vidéo avec perte, les taux de compression ont énormément augmenté, permettant ainsi le stockage et la transmission des flux vidéo.

Dans les années 85, l'analyse du signal s'est enrichie en exploitant les caractéristiques du Système de Vision Humain (SVH) et en considérant une image comme un ensemble de régions. On passait alors d'une représentation pixel à une représentation région. On a ainsi parlé de codage de deuxième génération [Kunt et al. 85].

1. hiérarchisation: scalability en anglais, « Le terme scalabilité est un calque inutile de l'anglais », Le grand dictionnaire terminologique, *http://www.granddictionnaire.com*.

Durant les années 90, la norme MPEG 4 [ISO/IEC 98] a introduit la notion d'objet vidéo et de codeur basé objet. Cette nouvelle approche structure la vidéo en un ensemble d'objets possédant un ordre de profondeur relatif. La figure 1 illustre cette notion d'objet vidéo. Par rapport à l'approche région, l'approche objet est plus riche car elle propose une indépendance de la carte de segmentation et permet de donner à un objet sa forme réelle. Ces deux notions sont illustrées par la figure 2(b) et 2(c) qui représentent respectivement une approche région et une approche idéale en objet vidéo.

FIG. 1 – *Illustration de l'ordre de profondeur pour les deux objets vidéo de la séquence Foreman*

Il faut bien remarquer que la notion d'objet vidéo reste évasive et qu'en aucun cas la norme MPEG 4 ne la définit. En effet, la norme MPEG4 se contente de définir la syntaxe de codage et de décodage. Ainsi, le codeur MPEG 4 prend en paramètres:

- un masque par image et par objet vidéo,
- une texture par image et par objet vidéo,
- un ordre de profondeur par objet vidéo.

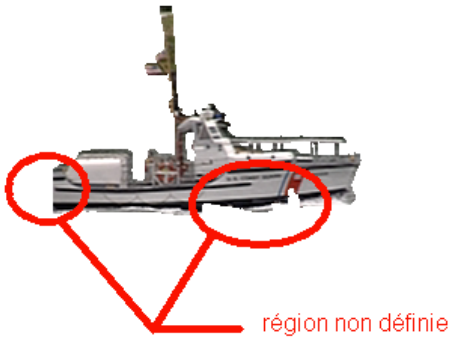
Au décodage, une composition de la scène est effectuée en respectant l'ordre de profondeur relatif associé à chaque objet de sorte que, dans les zones de superposition, l'objet le plus éloigné soit recouvert. Ainsi, la définition et l'extraction automatique de ces informations restent problématiques.

Entre le 2D 1/2 et la 3D, l'approche par *mosaïque* pour un objet de type arrière plan, ou pour tout type d'objet [Pateux et al. 01], [Cammas et al. 03a] permet d'obtenir une texture et un mouvement pour un groupe d'images. Cette représentation permet d'apprendre la texture associée à un objet. En effet, les zones se découvrant au cours du temps enrichissent la *mosaïque*. De plus, l'utilisation d'un maillage permet d'estimer des mouvements long terme très fins. La figure 3 montre la *mosaïque* associée au visage de la séquence vidéo nommée Armel.

Cette représentation est très intéressante puisqu'elle permet d'avoir une modélisation de l'objet par une seule texture ou une texture à faible variation dans le temps, et un

(a) image 50 coastguard

(b) frontières plaquées



(c) région pour l'image 50

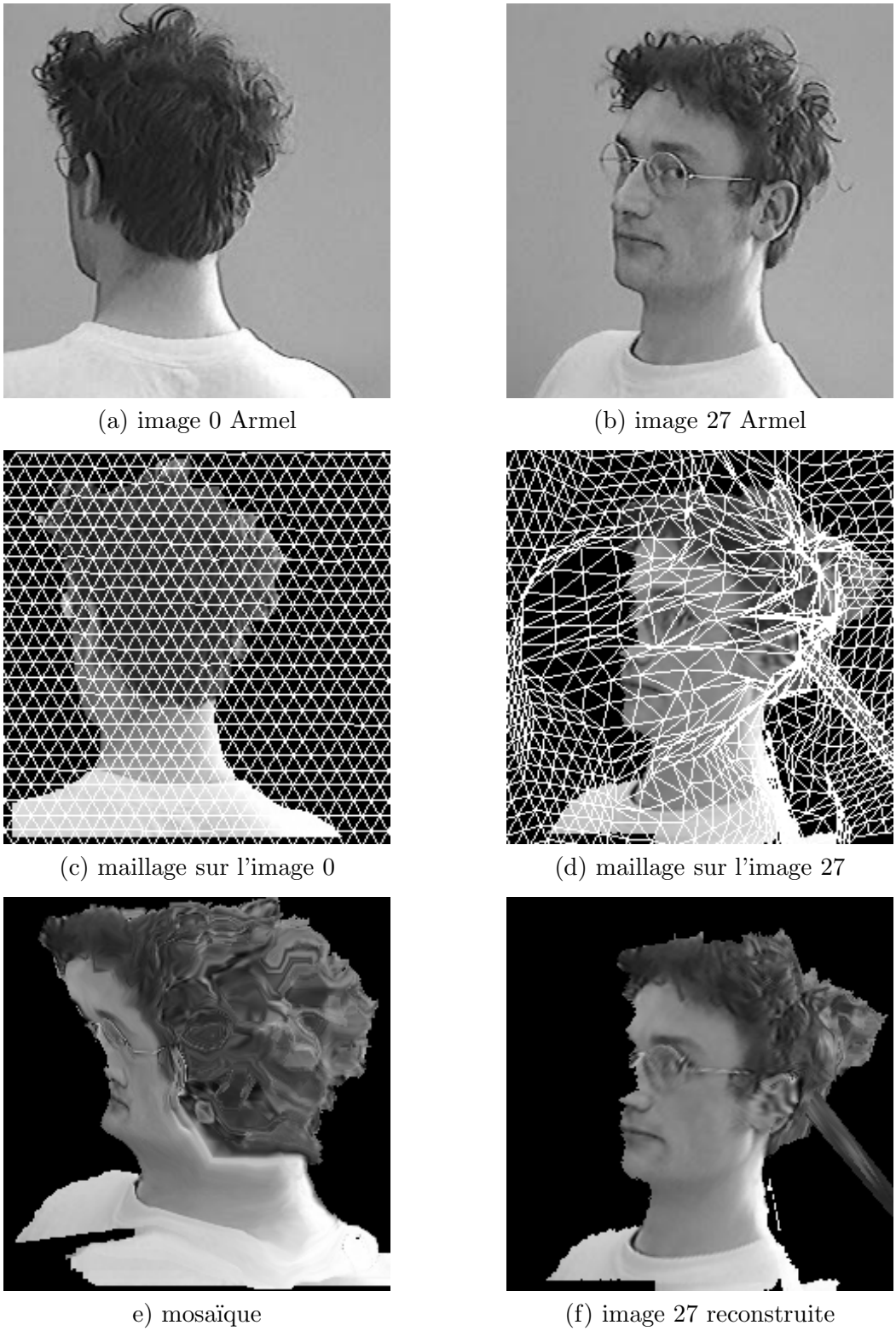
(d) objet pour l'image 50

FIG. 2 – *Illustration de la notion de région et d'objet vidéo. L'image (a) représente l'image 50 de la séquence Coastguard. L'image (b) représente les frontières de régions plaquées sur l'image 50. L'image (c) représente la région grand bateau. L'image (d) représente l'objet vidéo grand bateau*

mouvement 2D adapté. Ainsi, la texture d'un objet vidéo représente plusieurs images à la fois et non plus une seule. Il est évident que l'on gagne en compacité ainsi qu'en sémantique avec l'approche *mosaïque*.

L'approche par modélisation 3D permet en théorie d'obtenir la position 3D de tous les points 2D d'un groupe d'images et les positions de la caméra. Ainsi, un objet est représenté par la géométrie 3D du modèle, les positions de la caméra et une texture. Cette approche est encore moins coûteuse qu'une approche *mosaïque* puisqu'il suffit de décrire le mouvement de la caméra pour passer d'une image à la suivante.

Cette approche basée modèle 3D est à priori plus compacte et plus riche sémantiquement mais c'est aussi la plus complexe en temps de traitement. En pratique l'identification d'un tel modèle est souvent très difficile ce qui nécessite l'introduction d'hypothèses restrictives sur le contenu des scènes. Par exemple [Galpin et al. 01] ne fait pas d'a priori sur le



(a) image 0 Armel (b) image 27 Armel

(c) maillage sur l'image 0 (d) maillage sur l'image 27

e) mosaïque (f) image 27 reconstruite

FIG. 3 – *Illustration de la notion de mosaïque d'objet non rigide. Le mouvement est estimé par un maillage actif, figure (c) et (d), et l'on obtient la mosaïque de la figure (e). La figure (f) représente l'image 27 regénérée grâce à la mosaïque et au mouvement*

modèle 3D mais considère que la scène doit être rigide. Le codage de visage par modèle 3D [Eisert et al. 99] suppose quant à lui un modèle paramétrique de visage 3D réservé à ce type de scène.

L'évolution du codage vidéo montre l'importance de déterminer la bonne représentation de l'information, c'est-à-dire de choisir le modèle 2D, 2D 1/2 ou 3D le plus adapté aux informations à coder. Cependant, ces approches sont bien souvent non génériques et se heurtent au probleme de l'analyse de la séquence avant son codage. Dans cette optique d'analyse, la première partie de ce manuscrit étudie le thème de la segmentation et donne des pistes pour renforcer l'apport de l'analyse du signal vidéo avant le codage.

Une fonctionnalité intéressante : la hiérarchisation

La hiérarchisation vidéo consiste à avoir un flux d'information dont les informations sont réparties dans des sous-ensembles hiérarchisés, de telle sorte qu'elles puissent être utilisées par ordre d'importance, au moment de reconstituer les images. On définit alors plusieurs types de hiérarchisations c'est-à-dire différents ordonnancements : spatial, temporel, qualité, train binaire[2] et objet ([Marquant 00] p.39-44).

On peut noter que toutes les hiérarchisations ne sont pas proposées par les codeurs. H264/AVC [Schwarz et al. 02] par exemple, ne propose que la hiérarchisation de type temporelle (schéma de codage IPB). MPEG4 propose la hiérarchisation temporelle, spatiale (différentes résolutions), qualité et objet.

Sachant que les applications utilisatrices de flux vidéos sont fortement demandeuses de hiérarchisations, il y a un réel effort à faire dans ce sens. En effet, pour les applications sur réseaux, un fichier unique peut être émis avec un débit pouvant varier tant à l'émission (adaptation du débit à la bande passante disponible), qu'à l'intérieur du réseau. Ce même fichier peut aussi être émis sur plusieurs canaux (s'il existe une système de différentiation de service avec qualité de service « QoS »). À la réception du flux, le décodeur, de par les propriétés du flux, peut n'en décoder qu'une partie. Pour les applications de type consultation de base de données vidéo ou bien de montage vidéo, la hiérarchisation permet d'avoir un aperçu rapide de la vidéo sans la visualiser en pleine résolution (spatiale et temporelle) ni à qualité maximale. Il est évident que pour des raisons de stockage, la hiérarchisation train de bits permet de n'avoir qu'un seul fichier au lieu d'en avoir plusieurs, correspondant à différents débits.

Une hiérarchisation particulièrement intéressante est la hiérarchisation du train de bits. Pour le codage d'image, cette hiérarchisation permet la troncature, au bit près ou par pallier, du fichier binaire. Les données du train binaire sont donc rangées dans un ordre décroissant d'importance. Pour ce faire, on applique une transformée sur l'image puis on ordonne avec une priorité spatiale, fréquentielle ou débit-distorsion les coefficients issus de la transformée. Ces techniques ont entre autre été mises en valeur avec l'utilisation d'une transformée ondelette dans EZW (1993) [Shapiro 93] , SPIHT (1996) [Said et al. 96] puis EBCOT (2000) [Taubman 00].

Pour le codage vidéo, plusieurs hiérarchisations ont été proposées pour obtenir les mêmes possibilités qu'en image fixe. Cependant, assez souvent la hiérarchisation vidéo

2. Train binaire : traduction du mot anglais « bitstream »

pose des problèmes de redondance d'information. À titre d'exemple, nous présentons ici la hiérarchisation temporelle et la hiérarchisation en couches :

- La hiérarchisation temporelle: schéma IPB, « Intra, Prédite, Bidirectionnelle ». Elle est à la base de nombreux codeurs (typiquement les codeurs de type MPEG) puisque le schéma IPB exploite la redondance et permet ainsi de réduire fortement les coûts de codage. Cette hiérarchisation permet d'ordonner le flux vidéo en donnant plus d'importance aux blocs I qu'aux blocs P et qu'aux blocs B.
- La hiérarchisation en couches. Une couche basse est réservée pour les informations essentielles. Les couches hautes permettent un raffinement des informations de la couche basse. Par exemple, la hiérarchisation MPEG4 FGS, « Fine Granularity Scalability » permet de coder la couche basse à débit constant et ensuite de réguler le débit global par une couche haute améliorant chaque image séparément. Cette approche ne permet cependant pas de profiter de la redondance apportée lors de l'amélioration d'une image passée.

Pour les images fixes, la transformée en ondelette a permis d'améliorer les performances des schémas progressifs. Il semble intéressant d'essayer d'étendre à la vidéo l'utilisation de cette transformée pour obtenir une hiérarchisation du train binaire. Ceci est abordé dans la deuxième partie de ce manuscrit.

Les codeurs actuels : vers un codage adapté au signal

Tous les types de codeurs (MPEG, H264/AVC, codage par mosaïque, codage par ondelette 3D, codage par modèle) s'adaptent au signal vidéo ou en sont fortement dépendants.

L'évolution du codage peut être vue comme un enrichissement des approches actuelles par une complexification des modèles. En effet, la tendance est à l'augmentation du nombre de paramètres grâce à l'utilisation d'optimisation débit-distorsion.

L'évolution du codage peut aussi être vue en terme d'échelle d'analyse. Les codeurs s'adaptent de manière très locale (CABAC, quantification), de manière locale (blocs, macroblocs) ou de manière plus globale (segmentation, mosaïque, objets, modèle 3D).

À une échelle locale, MPEG et H264/AVC ont évolué vers une diminution de la taille des blocs, une augmentation de la précision du mouvement, un grand choix de modes de codage de bloc (Intra, Inter ...). H264/AVC permet même de segmenter les blocs. Le choix est donc effectué bloc par bloc. Ce choix est dicté par le signal via une optimisation débit-distorsion et via une analyse locale.

À une échelle moins locale, les codeurs par maillages et ondelettes ainsi que les codeurs par modèles 3D sont parfaitement adaptés au signal lorsque leurs modèles représentent bien le contenu de la séquence.

On peut remarquer qu'il y a deux évolutions parallèles dans le codage vidéo. La première consiste à mettre en œuvre des techniques d'optimisation débit-distorsion et à enrichir les codeurs, c'est-à-dire à complexifier les modèles. La deuxième consiste à introduire l'analyse, par exemple la segmentation, pour trouver des régions homogènes en texture ou en mouvement, ce qui permet de réduire les coût de codage. La tendance de tous les codeurs est donc de se rapprocher d'une modélisation plus physique du signal que ce soit par une approche d'optimisation-débit distorsion ou par une analyse à différentes échelles (blocs, region/objet, modèle(sprite, 3D, ...) .

Le codage objet dynamique : représentation et codage adaptés

Les applications utilisant le codage vidéo sont de plus en plus demandeuses de fonctionnalités de hiérarchisation telles que la hiérarchisation spatiale, temporelle, PSNR, train binaire etc. Ainsi, l'introduction de la notion de codage objet couplé avec une approche dynamique du codage, c'est-à-dire avec le choix du codage adapté à chaque objet, semble attractive [Reusens et al. 97].

En effet, chaque objet a souvent une propriété sémantique qui induit un codage adapté. Par exemple, on a tout intérêt à coder un objet vidéo de type *mosaïque* en image fixe ou dynamique. Le codage d'un objet de type *scène rigide* permettant le calcul d'un modèle 3D, tire profit d'une approche basée modèle. Ainsi, dans un codeur basé objets avec codage dynamique, au sein d'une même image plusieurs objets peuvent cohabiter, avec pour chacun un codage qui leur est propre.

On peut comparer l'approche H264/AVC et l'approche par codage dynamique en terme d'échelle d'analyse du signal vidéo et en terme de possibilités. Dans le cas de H264/AVC, l'échelle d'analyse est locale au bloc. On optimise sur des paramètres simples comme sur le choix du type de codage, le choix de la taille du bloc, le calcul du vecteur mouvement. Dans le cas de l'approche par codage dynamique, l'échelle d'analyse est beaucoup moins locale puisqu'on raisonne sur des objets issus de la segmentation. L'optimisation est alors d'un niveau supérieur et beaucoup plus large puisqu'il faut faire un choix sur le type de codage de l'objet (DCT, Sprite, ondelette ...).

Le codage dynamique d'objet nécessite donc le choix de la représentation de l'objet lors de l'analyse de la séquence ; la première partie de ce manuscrit aborde cette question. Ensuite, chaque objet doit être codé. On peut alors faire appel aux codeurs d'objets vidéo de type MPEG4, ou bien pour plus de hiérarchisation on peut utiliser le schéma ondelette 3D comme présenté en deuxième partie du manuscrit. Enfin, si l'on désire choisir pour chaque objet le codage le plus adapté, il faut mettre en concurrence sur chaque objet l'ensemble des codeurs à disposition ; ceci est expliqué dans la dernière partie du manuscrit.

Orientation et justification de l'étude

L'évolution du codage en une quarantaine d'année fait ressortir deux grandes tendances : l'enrichissement des modèles (augmentation du nombre de paramètres et optimisation débit-distorsion) et la recherche de modèles de plus en plus proches de la physique de la scène (2D, 2D1/2, 3D). En plus de cette évolution, les applications sont de plus en plus demandeuses de fonctionnalités telles que la hiérarchisation.

Ainsi, le codage dynamique d'objets vidéo s'inscrit parfaitement dans l'évolution actuelle du codage. En effet, y sont présents : la recherche de modèle proche de la physique de la scène, la hiérarchisation objet et l'adaptation au signal via l'optimisation débit-distorsion.

Plus généralement, ce manuscrit essaye d'analyser les apports du codage dynamique d'objets vidéo et la faisabilité de la segmentation automatique pour le codage d'objets vidéo. La première partie de ce manuscrit aborde la segmentation en objets vidéo, la deuxième partie aborde la hiérarchisation d'un objet, et la troisième partie aborde le codage dynamique.

première partie

La segmentation

Chapitre 1

État de l'art : les approches de segmentation basées régions v.s. basées contours

Que cela soit dans le domaine spatial, temporel ou spatio-temporel, la segmentation a pour objectif de partitionner une image ou un groupe d'images de sorte que chacune des régions ait une caractéristique qui la distingue des autres. Dans le cas des approches basées régions, on recherche une homogénéité spatiale, temporelle ou spatio-temporelle de la région. Dans le cas des approches contour, on cherche des contours prononcés spatialement, temporellement ou spatio-temporellement.

De nombreuses techniques de segmentation ont été proposées. Ces techniques ont toutes en commun la minimisation d'une fonctionnelle d'énergie. Les auteurs de [Zhu et al. 96] montrent que quelque soit la technique utilisée (contours actifs, croissance de région ou approches Bayes/MDL), le problème posé est équivalent. Ainsi quelque soit l'algorithme utilisé, on peut s'attendre à obtenir sensiblement les mêmes résultats. Le choix d'un algorithme est alors fonction de ses avantages et de ses inconvénients (sensibilité par rapport à l'initialisation, vitesse d'exécution, complexité, convergence ...). L'objectif de l'état de l'art de ce chapitre est de présenter les fonctionnelles énergétiques ainsi que les bénéfices et les défauts des différentes modélisations.

Les deux premières sections abordent donc les deux grandes familles d'approches : régions et contours, sans faire de distinction entre l'utilisation de critères spatiaux, temporels ou spatio-temporels. La troisième section porte sur le suivi (« tracking ») et illustre l'approche classique par projection qui permet d'étendre le résultat d'une segmentation d'une image aux images suivantes. Enfin, la dernière section sur les tubes spatio-temporels aborde les nouvelles techniques de segmentation vidéo prenant en compte une dimension temporelle plus importante.

1.1 Les approches basées régions

Les approches de segmentation basées régions cherchent des régions homogènes selon des critères spatiaux, temporels ou spatio-temporels. Les sections suivantes présentent différentes approches pour obtenir les homogénéités. Il y a les approches par croissance de

régions, les approches par division-fusion et les approches par classification. Ces différentes approches utilisent bien souvent les mêmes formalismes de régularisation des solutions (champs de Markov, MDL, optimisation débit-distorsion ...).

1.1.1 Les approches par croissance de régions

Les approches par croissance de région sont des techniques de recherche de zone homogène mais par contre ne sont pas optimales dans le sens d'une minimisation de fonctionnelle. Ces approches sont cependant intéressantes pour leur rapidité. Ici, nous présentons deux techniques : la croissance de région classique et l'approche par ligne de partage des eaux.

L'approche classique par croissance de régions

La croissance de régions consiste à faire grandir progressivement un ensemble de germes choisis dans l'image. La croissance des régions est effectuée de sorte que l'on respecte un critère d'homogénéité (luminance, couleur, texture...). Lorsque deux régions se rencontrent, on obtient la frontière [Kunt et al. 85].

L'inconvénient majeur est que les frontières des régions ne correspondent pas toujours à une limite nette dans l'image, ce qui rend difficile le réglage de la phase de croissance. On utilise le plus souvent une technique avec file d'attente, avec ajout itératif du point le plus proche d'un germe. On obtient ainsi des régions homogènes mais ce n'est pas la solution qui permet d'obtenir la partition optimale. Si on modélise le problème par la fonctionnelle 1.1 le résultat n'est pas équivalent à la solution minimisant la fonctionnelle. Le résultat est une solution homogène mais pas la solution la plus homogène.

$$E = \sum_{k=1}^{K} \sum_{i=1}^{N} P_{i,k} d_{i,k}^2, \tag{1.1}$$

$$\sum_{k=1}^{K} P_{i,k} = 1,$$

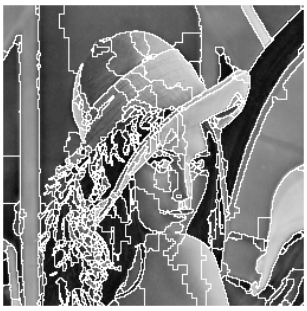
avec $d_{i,k}$ une distance définissant la similarité entre le représentant de la région $\mathcal{R}_k$ et le pixel x_i, et $P_{i,k}$ valant 1 ou 0 selon que l'individu x_i appartienne ou n'appartienne pas à la région $\mathcal{R}_k$.

Certaines approches essayent de prendre en compte de manière plus forte les contraintes de contour [Benois et al. 92]. Ainsi, on obtient des régions plus homogènes (cf. figure 1.1). Cependant l'approche par croissance de régions reste assez empirique en terme de positionnement des régions et de nombre de régions. De plus, étant donné que la croissance des régions se fait de manière simultanée, les frontières obtenues ne sont pas nécessairement les frontières de texture.

L'approche par ligne de partage des eaux

Les approches morphologiques comme l'algorithme de ligne de partage des eaux (« watershed ») [Meyer et al. 90] permettent de faire grossir des régions en partant de germes

(a) Image «lenna»



(b) Frontières plaqués

FIG. 1.1 – *Illustration de l'approche de segmentation par croissance de régions. Figures extraites de [Benois et al. 92]*

calculés lors d'une première étape. Les germes sont les zones de gradient local minimum. On considère l'image de gradient comme une surface topographique dont les valeurs des pixels représentent l'altitude. On immerge d'eau la surface en partant des germes. Les zones où il y a rencontre d'eau provenant de bassins différents constituent les frontières (barrages) de la carte de segmentation.

Un algorithme possible pour mettre en œuvre cette méthode considère successivement des altitudes h croissantes. Ainsi pour une altitude h, on a immergé toutes les zones d'altitude inférieure et on réalise alors des dilatations conditionnelles des bassins d'altitude h vers l'altitude $h+1$. Les zones d'altitude $h+1$ n'ayant pas été atteintes constituent de nouveaux bassins. On itère ainsi jusqu'à immersion totale de la surface.

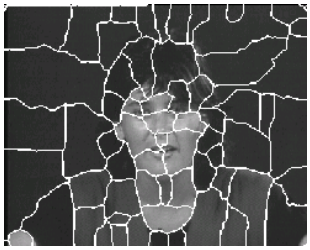
Cet algorithme fait croître une région autour d'un germe. On constate que le nombre de régions est difficilement maîtrisable puisqu'il est égal au nombre de minima locaux de l'image. On utilise couramment un filtrage de l'image des gradients (filtre de fermeture morphologique) pour réduire ce nombre de minima locaux. On peut, comme cela est fait dans [Perez et al. 99], ajouter des critères pour prendre en compte de manière plus prononcée les contours.

L'approche par croissance de régions est généralement très rapide et permet d'obtenir des régions de tailles à peu près similaires et de formes régulières (fig. 1.2). Par contre le nombre de régions est généralement très élevé et la réduction de ce nombre de régions se fait au détriment des détails de l'image. De la même façon que pour l'approche par croissance de régions, l'approche par ligne de partage des eaux ne repose pas sur la minimisation d'une fonctionnelle.

1.1.2 Les approches par division-fusion

Les approches de division-fusion (« split and merge ») consistent soit à diviser de manière très fine une image et ensuite à fusionner les régions voisines grâce à des critères d'homogénéité et de cohérence soit à diviser itérativement tant que les régions ne sont pas assez homogènes et pas assez cohérentes. On utilise couramment une structure en quad-arbre (« quad-tree ») (cf. figure 1.3) ou bien en graphe (Region Adjancy Graph - RAG) pour résoudre le problème. Les régions sont alors représentées par les nœuds du graphe ou bien les feuilles de l'arbre. Les algorithmes réalisent alors des fusions-divisions de régions

(a) Image 0, séquence «Miss America»



(b) Frontières plaquées

FIG. 1.2 – *Illustration de l'approche par ligne de partage des eaux. Figures extraites de [Bonnaud 98]*

(feuilles ou nœuds) voisines.

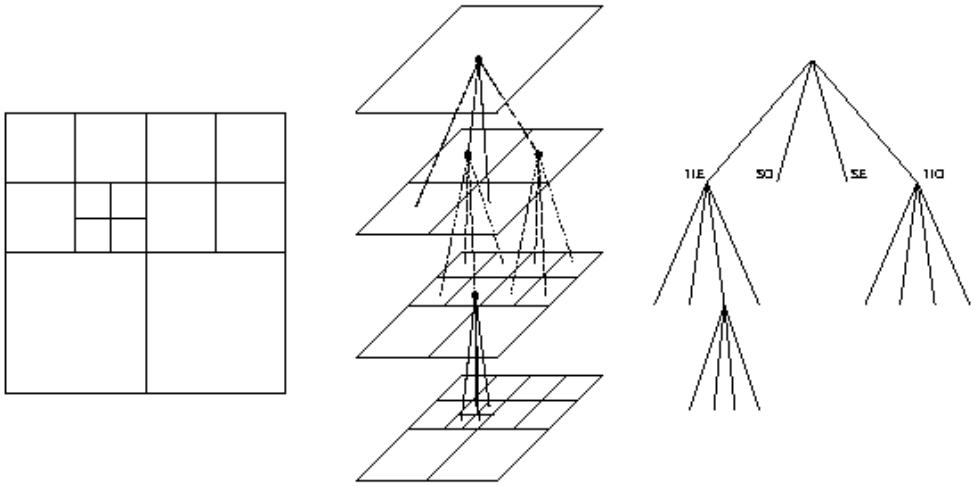


FIG. 1.3 – *Illustration du quad-arbre d'une image, et sa représentation sous forme d'arbre. Figures extraites de [Pateux 98]. Les symboles NO, NE, SO, SE indiquent la position des fils d'un nœud par une localisation géographique*

La suite du document traite des critères de fusions/divisions. Ensuite, on introduit les formalismes Champs de Markov, MDL et débit-distorsion.

Le critère de fusion

Que ce soit dans les approches de fusion-division ou pour d'autres techniques, il est nécessaire de définir une mesure de similarité ou de dissimilarité entre deux régions. Cela revient à modéliser le problème par une fonctionnelle d'énergie qu'il faut minimiser.

Le choix de fusionner deux régions peut alors être fonction de la variation d'énergie avant - après fusion ou fonction de la similarité entre deux régions. Il y a donc une notion de seuil qui intervient pour stopper les fusions. Différents seuils peuvent être utilisés comme le seuil sur la variation d'énergie, celui sur la similarité ou celui sur le nombre total de fusions.

Pour une fusion-division dans le domaine spatial, on utilise couramment une distance de type erreur quadratique moyenne entre la description M_A d'une première région A et la description M_B d'une seconde région B [Déforges et al. 99] :

$$d^2_{a,b} = \|M_a - M_b\|^2. \tag{1.2}$$

Les prédictions M_A et M_B peuvent par exemple être deux vecteurs caractéristiques composés de l'intensité moyenne, du gradient local, de la variance locale.

Pour une fusion-division dans le domaine temporel, on peut utiliser le résultat d'une estimation dense du mouvement. On peut définir alors une distance de type erreur quadratique des prédictions issues des deux modèles de mouvement. La distance est donnée par l'équation 1.2. La prédiction M_A (resp. M_B) peut par exemple être le vecteur translationnel moyen de la région A (resp. B). La prédiction M_A peut aussi être le résultat de la transformation des positions des régions A et B par un modèle de mouvement affine ou quadratique [Wang et al. 94].

Pour une fusion-division dans le domaine spatio-temporel, on peut citer par exemple [Bouthemy et al. 87] qui utilise un test de maximum de vraisemblance basé sur la variation d'erreur quadratique moyenne. Soient R_A, R_B et $R_{A,B}$ trois régions; $R_{A,B}$ la région résultant de la fusion de R_A et R_B, N le nombre de points d'une région et EQM l'erreur quadratique moyenne. On définit alors le critère de vraisemblance par :

$$(N_{R_A} + N_{R_B}) lnEQM_{R_{AB}} - [N_{R_A} lnEQM_{R_A} + N_{R_B} lnEQM_{R_B}].$$

Finalement, que ce soit dans le domaine spatial, temporel ou spatio-temporel, l'objectif est d'obtenir des régions homogènes. Une approche couramment utilisée pour ajouter plus de régularité dans la segmentation est d'imposer des contraintes sur le voisinage d'un pixel. On utilise, par exemple, les champs de Markov présentés dans la sous section suivante.

Le problème majeur de l'approche par fusion est le réglage de seuil, qui, de manière indirecte, revient à choisir le nombre de régions. Pour rendre automatique le réglage de ce seuil mais aussi pour adapter la segmentation au codage, deux formalismes sont présentés dans les sous-sections suivantes : le formalisme MDL et le formalisme débit-distorsion.

Les champs de Markov

Les champs de Markov [Besag 86] [Geman et al. 84] [Azencott 87] permettent d'introduire la notion de voisinage et de contrainte de régularisation d'un voisinage dans la formulation énergétique du problème de segmentation. Ainsi, chaque étiquette de la segmentation est un site relié aux pixels proches appartenant à un certain voisinage (en général un 4- ou 8- voisinage). Pour chaque pixel, on est capable de calculer la probabilité d'appartenance à une région. Il s'agit d'une probabilité a posteriori sachant l'étiquette de la région. Pour cela, il faut avoir un modèle statistique de la région. On peut par exemple utiliser un modèle de niveau de gris constant plus un bruit aléatoire, ou un modèle de texture etc.

Le champ de Markov est l'expression statistique de la probabilité a priori que deux étiquettes voisines soient égales ou différentes. Il sert à la régularisation statistique de la solution. On se ramène alors à la minimisation d'une fonction d'énergie globale définie sur toutes les étiquettes. Elle se compose de deux termes :

– une énergie d'attache aux données E^d qui exprime l'adéquation de chaque pixel à la région qui lui est attribuée,

– une énergie de régularisation E^r qui favorise les situations où 2 pixels voisins ont la même étiquette.

Nous donnons ici, pour exemple, une fonctionnelle d'énergie (équation 1.3) utilisant deux termes d'énergies E^d et E^r [Fablet et al. 99]. L'objectif est de déterminer l'étiquetage e_i pour chaque pixel x_i. L'énergie d'attache aux données E^d est choisie utilisant un modèle gaussien de moyenne μ_i et de matrice de covariance Λ_i. L'énergie de régularisation E^r favorise l'homogénéité dans un voisinage $\mathcal{V}(i)$ d'un pixel x_i. La résolution est effectuée par ICM (Iterated Conditional Modes) [Chou et al. 90] multi-échelle en effectuant des basculements d'étiquettes et des fusions.

$$E = \sum_{i=1}^{N} \left[E^d(e_i, x_i) \right] + \alpha \times \sum_{j \in \mathcal{V}(i)} \left[E^r(e_i, e_j) \right], \tag{1.3}$$

avec :

$$\begin{aligned} E^d(e_i, x_i) &= (x_i - \mu_i) \times \Lambda_i^{-1} \times (x_i - \mu_i)^t, \\ E^r(e_i, e_j) &= 1 - \delta(e_i, e_j), \\ \delta(x, y) &= \begin{cases} 1 & si\ x = y, \\ 0 & si\ x \neq y. \end{cases} \end{aligned}$$

Les champs de Markov sont souvent utilisés pour segmenter car il permettent d'introduire les termes de régularisation. On trouve alors des segmentations spatial [Fablet et al. 99]. On trouve aussi les approches par détection, où l'on détecte les points en mouvement par rapport à un fond fixe [Diehl 91] [Caplier et al. 01] ou par rapport à un mouvement global [Odobez 94]. Dans tous les cas, le terme d'attache aux données et le terme de régularisation sont présents.

Le formalisme MDL

Le formalisme MDL (Minimum Description Length) introduit par Rissanen [Rissanen 78] permet de modéliser le problème de segmentation en prenant en compte le coût de codage de cette segmentation. On considère le problème de segmentation comme la recherche du modèle le plus compacte de représentation en région d'une image. L'objectif est donc d'avoir un compromis entre l'erreur de prédiction spatiale, temporelle ou spatio-temporelle résultant de la segmentation et le coût de codage de cette segmentation et du modèle.

De manière générale, des schémas de codage prédictifs sont considérés. Pour coder une image I, on utilise l'erreur de prédiction $\epsilon = I - \widehat{I}$ sachant que l'on dispose de l'image prédite $\widehat{I}$. Dans le cas où $\widehat{I}$ n'est pas connue, ϵ vaut zéro.

La modélisation du problème fait donc apparaître l'erreur de prédiction ϵ, la carte de segmentation S avec ses paramètre globaux θ_G et l'ensemble des paramètres de chaque région θ_r. La fonctionnelle MDL comporte alors deux termes [Pateux 98] [Leclerc 89] :

– un terme de coût de codage des paramètres décrivant la segmentation : $DL(S, \theta_G, \theta_r)$,

– et un terme de coût de codage de l'erreur résiduelle : $DL(\epsilon/(S, \theta_G, \theta_r))$.

On doit donc effectuer la minimisation suivante :

$$\min_{S,\theta_G,\theta_r} DL(S,\theta_G,\theta_r) + DL(\epsilon/(S,\theta_G,\theta_r)),$$

c'est-à-dire :

$$\min_{S,\theta_G,\theta_r} DL(\theta_G) + DL(S/\theta_G) + \sum_r DL(\theta_r/(S,\theta_G)) + DL(\epsilon/(S,\theta_G,\theta_r)).$$

$DL(S/\theta_G)$ correspond au coût de codage de la carte de segmentation. Dans l'approche division-fusion, c'est le nombre de bits (débit) utilisés pour décrire l'arbre ou bien le graphe de segmentation. $DL(\theta_r/(S,\theta_G))$ correspond au coût de description d'une région. Ce coût est fonction du nombre de paramètres décrivant une région et du pas de quantification utilisé pour coder ces paramètres. $DL(\epsilon/(S,\theta_G,\theta_r)$ correspond au coût de codage de l'erreur de prédiction ϵ. Ce coût est fonction de la technique de codage retenue (quantification scalaire, DCT, codage morphologique, ...). $DL(\theta_G)$ correspond au coût de codage des paramètres globaux.

Par exemple, [Wareham et al. 96] donnent pour une segmentation par fusion-division grâce à un quad-arbre avec un codage arithmétique de l'erreur quantifiée, la formule suivante à minimiser (avec omission du coût de la carte de segmentation $DL(S/\theta_G)$) :

$$E = \underbrace{(\sum_{p\in I} -log_2 P^c[\overline{\epsilon}(p)])}_{\sum_r DL(\epsilon/(S,\theta_G,\theta_r))} + \underbrace{(\sum_r \sum_{f=1}^{F} -log_2 \frac{M_f}{\delta_f})}_{\sum_r DL(\theta_r/(S,\theta_G))}.$$

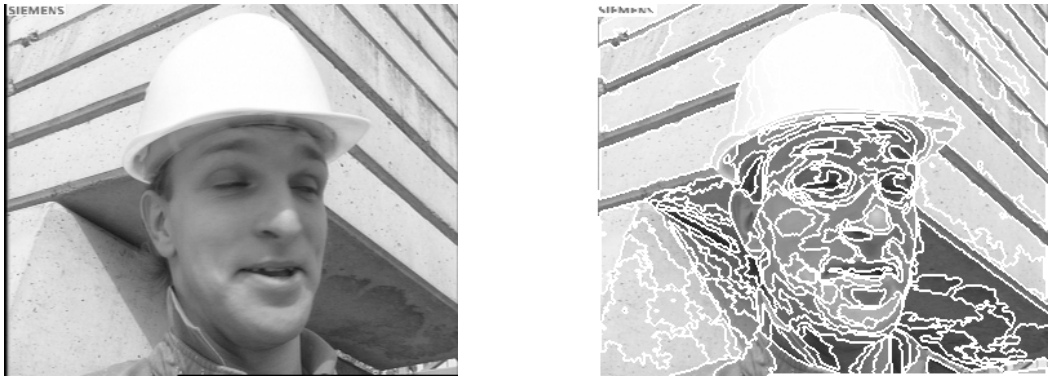
P^c est une loi de probabilité paramétrique (Gaussienne, Laplacienne, Gaussienne Généralisée) associée au codage. M_f est l'intervalle de valeurs du paramètre f et δ_f le pas de quantification appliqué au paramètre f. Le terme $DL(\theta_G)$ est omis car il est de coût fixe. On retrouve ainsi dans cette formule un terme d'attache aux données ($\sum_r DL(\epsilon)$) qui a tendance à faire augmenter le nombre de régions et un terme de régularisation ($\sum_r DL(\theta_r)$) qui a tendance à faire diminuer le nombre de régions.

Le principe de résolution est le suivant : on utilise une approche par fusion (approche montante des feuilles vers la racine), par division (approche descendante) ou bien par fusion-division. Le choix de fusionner (resp. diviser) est effectué si la fusion (resp. division) permet d'obtenir un coût de codage moindre [Ndili et al. 01]. Les résultats obtenus par l'approche MDL sont en général assez bons et assez rapides par rapports aux autres approches. La figure 1.4 illustre le résultat d'une segmentation MDL sur l'image 1 de la séquence foreman.

Le formalisme débit-distorsion

Le formalisme débit-distorsion permet de définir une partition en fonction d'une contrainte de débit ou bien de distorsion [Pardàs et al. 96]. Cette approche est particulièrement bien adaptée au domaine du codage vidéo. Sachant la mesure de la distorsion et du débit fonction d'un jeu de paramètres Θ, on modélise le problème par la minimisation d'une fonctionnelle sans contrainte grâce à l'utilisation du lagrangien :

$$\Theta^* = arg \min_{\Theta} R(\Theta) + \lambda D(\Theta).$$



(a) Image séquence «Foreman» (b) Régions obtenues

FIG. 1.4 – *Illustration de l'approche par fusion de régions et modélisation MDL*

Ici, le jeu de paramètres Θ^* représente la structure de l'arbre ainsi que les paramètres associés à chaque feuille de l'arbre. On recherche donc l'arbre minimisant la fonctionnelle $R + \lambda D$. Pour résoudre ce problème, dans un premier temps on construit un arbre de partition qui a pour racine l'image entière et qui a pour feuilles de petites régions de l'image (cf. figure1.5). Ainsi, un nœud de l'arbre de partition représente une région et ses nœuds fils représentent la partition de cette région. Chaque nœud se voit alors attribuer un débit et une distorsion.

L'algorithme de recherche de la meilleure partition est alors itératif. On fixe le paramètre λ, on détermine ainsi une partition et le débit associé. Si le débit R_{max} est dépassé, on réduit la valeur de λ sinon on augmente la valeur de λ. On ré-estime alors une nouvelle partition sachant la nouvelle valeur de λ. On itère ainsi jusqu'à approcher suffisamment la contrainte de débit R_{max}. La partition obtenue est alors optimale en terme de distorsion sachant le débit imposé.

La modélisation par débit-distorsion est très utilisée dans le domaine du codage car elle permet d'optimiser n'importe quel jeu de paramètres sous une contrainte de débit ou de distorsion. Dans le cas de la segmentation par division-fusion, elle automatise le choix des seuils permettant de décider si l'on fusionne ou non. De plus, par rapport aux modélisations basées similarité (1.1.2), l'équation énergétique est similaire puisque l'on retrouve deux termes :

- un terme de distorsion qui est le terme d'attache aux données (distance, similarité, erreur...),
- et un terme de débit qui s'apparente aux termes de régularisation portant sur le nombre de régions ou la taille des frontières.

1.1.3 Les approches par classification

Les approches par classification raisonnent sur un nuage de points (ou de régions). Ce nuage de points est aussi couramment appelé : population, individus, données. Dans le cadre de la segmentation, l'individu est le pixel et son vecteur caractéristique est composé d'attributs tels que le niveau de gris, la variance, le mouvement, ... L'objectif est

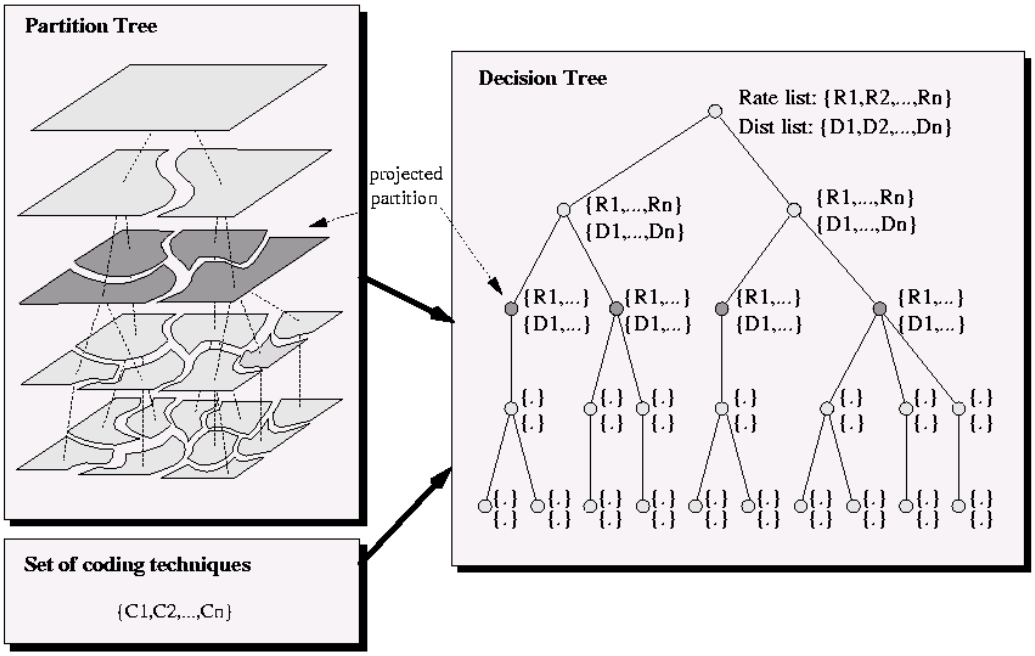


FIG. 1.5 – *Illustration de la notion d'arbre de partition. Un débit et une distorsion sont attribués à chaque nœud. Figure extraite de [Salembier et al. 97]*

d'agglomérer en nuages de points la population, c'est-à-dire de déterminer :

- les K représentants (centroïdes). On cherche donc K groupes[1] (classes) d'individus,
- et l'appartenance ou le degré d'appartenance de chaque individu à un des K groupes.

Dans le cadre de la technique par « clustering », le problème est modélisé par la minimisation de la somme des erreurs d'attribution de chaque individu à un groupe. Dans le cadre de la technique par maximum de vraisemblance, le problème est modélisé par la maximisation des similarités entre chaque individu et chaque groupe. La résolution des deux problèmes est très similaire. Le schéma est souvent itératif avec le calcul des représentants et le calcul des probabilités de chaque individu. La section suivante présente les caractérisations multiples d'un individu, le clustering et l'approche par maximum de vraisemblance.

Les caractérisations multiples d'un individu

La caractérisation multiple d'un pixel est utilisée dans de nombreuse approches. En effet, une caractérisation possible d'un pixel peut être un vecteur multi-composantes. On utilise alors couramment au moins six dimensions : trois pour la couleur et trois pour la position. La position est souvent introduite pour tenter d'assurer une cohérence spatiale. D'autres dimensions peuvent s'ajouter telles que le mouvement translationnel ou bien la variance locale comme mesure de texture. On cherche alors à regrouper les vecteurs similaires c'est-à-dire à trouver les meilleurs représentants de chaque groupe. Le calcul des

1. groupe - ensemble : en anglais «cluster»

représentants ainsi que de l'appartenance d'un pixel à l'un des groupes peut être obtenu par exemple par mélange de paramètres [Greenspan et al. 02], clustering [DeMenthon 02], ou calcul de valeurs propres [Shi et al. 98].

On peut penser que l'utilisation de plusieurs caractéristiques pour définir un individu apporte plus de robustesse aux résultats. Cependant, un problème majeur est la définition d'une distance entre des caractéristiques qui ne sont pas a priori comparables. On utilise alors couramment la distance de Mahalanobis [Mahalanobis 30] :

$$d^2_{i,k} = (\mu_k - x_i)\Lambda^{-1}(\mu_k - x_i)^t,$$

avec μ_k le représentant de la classe, x_i un individu, et Λ la matrice de covariance. Cette distance permet ainsi d'avoir une indépendance d'échelle et de comparer des caractéristiques différentes.

Cependant, dans certains cas, il est préférable de favoriser certaines caractéristiques plus discriminantes que d'autres. Il y a donc un système de pondérations variables à introduire pour moduler la distance. Même s'il est possible de trouver certaines pondérations automatiquement par logique flou [Castagno et al. 98], il reste cependant difficile de justifier la comparaison de caractéristiques de natures différentes.

Technique de clustering

L'algorithme de clustering dur (« hard ») [Ball et al. 66] et [Ball et al. 67], appelé aussi algorithme des k-moyens (« k-means ») ou c-moyens (« c-means ») consiste à classer les individus de la population dans un des K groupes. Soient $X = \{x_1, ...x_N\}$ la population de N vecteurs et $M = \{\mu_1, ..., \mu_K\}$ les K représentants (centroïdes). L'algorithme consiste alors à calculer itérativement l'appartenance $P_{i,k}$ de chaque pixel à un des K groupes et les représentants μ_k des K groupes.

On définit alors l'appartenance $P_{i,k}$ d'un individu x_i à un groupe k :

$$\forall i, P_{i,k} = \begin{cases} 1 & si\ k = \arg_{k'} \min_{k' \in [0,K]} \{d_{i,k'}\} \\ 0 & sinon \end{cases}$$

avec :

$$d_{i,k} = \|\mu_k - x_i\|.$$

$\|.\|$ est une mesure de distance dans l'espace des caractéristiques. Les K représentants sont alors mis à jour via la formule suivante :

$$\mu_k = \frac{\sum_{i=1}^{N} P_{i,k} \times x_i}{\sum_{i=1}^{N} P_{i,k}}.$$

Le clustering flou (« fuzzy ») [Dunn 74] [Bezdek 81] est une extension qui permet d'avoir une classification en probabilité d'appartenance (pourcentage d'appartenance) à un des K groupes. On définit alors non plus une appartenance mais une probabilité d'appartenance $P_{i,k}$ d'un individu x_i à un groupe k :

$$\forall i, \forall k, P_{i,k} \in [0, 1],$$

avec :

$$\forall i, \sum_{k=1}^{K} P_{i,k} = 1.$$

Le problème est alors modélisé par la minimisation d'une fonctionnelle :

$$\min_{P_{i,k},\mu_k} \sum_{k=1}^{K} \sum_{i=1}^{N} P_{i,k}^m d_{i,k}^2,$$

avec :

$$d_{i,k} = \|\mu_k - x_i\|.$$

m correspond au coefficient flou et appartient à l'intervalle $[1, \infty[$. On obtient par annulation des dérivées partielles les formules suivantes :

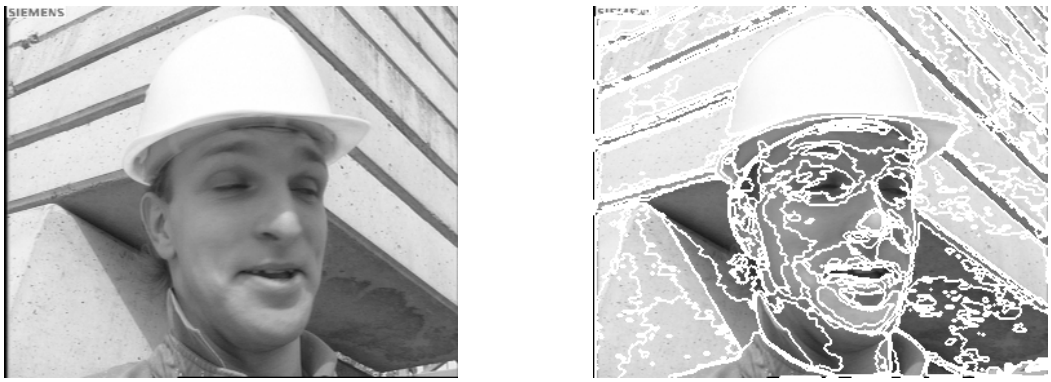
$$P_{i,k} = \frac{1}{\sum_{s=1}^{K} (\frac{d_{i,k}}{d_{i,s}})^{\frac{2}{m-1}}} \; si \; \forall k, d_{i,k} \neq 0, \tag{1.4}$$

$$\mu_k = \frac{\sum_{i=1}^{N} P_{i,k}^m \times x_i}{\sum_{i=1}^{N} P_{i,k}^m}. \tag{1.5}$$

L'algorithme consiste à calculer successivement jusqu'à stabilisation des probabilités :

- les probabilités $P_{i,k}$ (équation 1.4),
- puis les centroïdes μ_k (équation 1.5).

On peut constater que les approches par clustering ne prennent pas suffisamment en compte l'adjacence des pixels. La figure 1.6 illustre la dispersion des classes et donc le grand nombre de petites zones.



(a) Image séquence «foreman» (b) Régions obtenus

FIG. 1.6 – *Illustration de l'approche par clustering*

Cependant, l'utilisation de termes de régularisation dans la fonctionnelle permettent d'assurer une plus grande connexité spatiale dans les regroupements d'individus. Par

contre, le problème majeur reste le choix du nombre de groupes. En fonction du problème, on peut cependant injecter de nouvelles contraintes pour déterminer le nombre de groupes le plus adapté. Ces contraintes peuvent être introduites par l'utilisation d'un formalisme AIC (A-Information Criterion) [Akaike 74], d'un terme de compacité des nuages de points et de la distance entre eux [Nguyen et al. 93], du formalisme MDL (Maximum Description Length) [Rissanen 78] [Leclerc 89] ou du formalisme débit-distorsion [Pardàs et al. 96]... Par exemple, [Lorette 99] modélise le problème par l'ajout d'un terme entropique $\sum_{k=1}^{K} P_k log P_k$ pénalisant les centroïdes non représentatifs. La formule énergétique est alors :

$$J = \sum_{k=1}^{K} \sum_{i=1}^{N} P_{i,k}^{m} d_{i,k}^{2} - \alpha \sum_{k=1}^{K} P_k log P_k,$$

avec :

$$\forall k, P_k = \frac{1}{N} \sum_{i=1}^{N} P_{i,k}.$$

L'algorithme consiste à calculer successivement les probabilités $P_{i,k}$ puis les centroïdes μ_k. On diminue alors le nombre de classes à chaque itération si la probabilité P_k d'une classe k est inférieure à un certain seuil. De plus, on stabilise le problème en faisant tendre le terme α vers la valeur zéro, à chaque itération, via une décroissance géométrique.

L'approche par clustering est donc très flexible, facile à mettre en œuvre et permet d'avoir une mesure de confiance sur l'affectation des pixels. De plus la stabilisation des paramètres est rapide. Par contre, l'initialisation des clusters reste problématique.

Le Maximum de vraisemblance

L'approche par maximum de vraisemblance (« Maximum Likelihood ») cherche à estimer un mélange de paramètres pour représenter une population de vecteurs. Le mélange représente de manière similaire au clustering un ensemble de centroïdes. On a ainsi pour un centroïde M_k, un vecteur moyen μ_k, une matrice de covariance Λ_k et une densité de probabilité P_k. Le vecteur moyen et la matrice de covariance permettent de définir une mesure de similarité entre un individu x_i et le centroïde M_k. On peut prendre par exemple une fonction de distribution de probabilité gaussienne : $P((x_i/k); \Theta)$, de moyenne μ_k et de covariance Λ_k :

$$S_{i,k} = (2\pi)^{-\frac{F}{2}} |\Lambda_k|^{-\frac{1}{2}} exp[-\frac{1}{2}(x_i - \mu_k)^t \Lambda_k^{-1}(x_i - \mu_k)], \tag{1.6}$$

où F représente le nombre de caractéristiques des vecteurs individus. Ainsi, il faut déterminer le mélange Θ :

$$\Theta = (P_1, ..., P_k, \mu_1, ..., \mu_k, \Lambda_1, ..., \Lambda_k),$$

de sorte que la fonction de vraisemblance L_Θ soit maximisée :

$$L_\Theta = \prod_{i=1}^{N} \sum_{k=1}^{K} P_k \times S_{i,k}.$$

Cela peut être exprimé pour des commodités de calcul par la maximisation du logarithme de L_Θ :

$$arg\max_{\Theta} logL_\Theta = arg\max_{\Theta} L_\Theta.$$

On peut alors définir et observer que l'on a à l'optimum [Schroeter 96], la probabilité d'appartenance à la classe k d'un individus x_i :

$$P_{i,k} = \frac{P_k \times S_{i,k}}{\sum_{s=1}^{K} P_s \times S_{i,s}}, \tag{1.7}$$

et les paramètres Θ :

$$P_k = \frac{1}{N}\sum_{i=1}^{N} P_{i,k}, \tag{1.8}$$

$$\mu_k = \frac{\sum_{i=1}^{N} P_{i,k} \times x_i}{\sum_{i=1}^{N} P_{i,k}}, \tag{1.9}$$

$$\Lambda_k = \frac{\sum_{i=1}^{N} P_{i,k}(x_i - \mu_k)(x_i - \mu_k)^t}{\sum_{i=1}^{N} P_{i,k}}. \tag{1.10}$$

L'algorithme de maximum de vraisemblance est itératif et consiste à calculer successivement jusqu'à stabilisation des paramètres Θ :

- les paramètres Θ sachant les équations 1.8, 1.9, 1.10. C'est l'étape de maximisation (« Maximisation-step »),
- les probabilités conditionnelles $P_{i,k} = P((k/x_i);\Theta)$ d'appartenance à la classe k pour un individu x_i et un jeu de paramètres Θ avec l'équation 1.7. C'est l'étape de vraisemblance (« Expectation-step »). On peut remarquer que les probabilités conditionnelles $P_{i,k}$ (formule 1.7) découlent de la loi de Bayes :

$$P_{i,k} = P((k/x_i);\Theta) = \frac{P((x_i/k);\Theta) \times P(k)}{P(x_i;\Theta)}.$$

L'approche par maximisation de vraisemblance reste assez complexe en calcul et peut être simplifiée en supposant les caractéristiques indépendantes. On utilise alors un vecteur de variances σ_k à la place de la matrice de covariance Λ_k. La mesure de similarité 1.6 devient alors :

$$S_{i,k} = (2\pi)^{-\frac{F}{2}}(\prod_{f=1}^{F}\sigma_{k,f})^{-\frac{1}{2}} exp[-\frac{1}{2}(\sum_{f=1}^{F}\frac{(x_{i,f}-\mu_{k,f})^2}{\sigma_{k,f}})]. \tag{1.11}$$

Finalement, l'approche par maximum de vraisemblance est intéressante dans sa formulation mais reste insuffisante pour modéliser correctement les distributions de données. Comme pour le clustering, la notion de voisinage entre pixels n'est pas présente et le nombre de classes K doit être connu.

1.2 Les approches basées contours

Les approches basées contours recherchent la rupture d'homogénéité. C'est donc en quelque sorte l'approche duale des approches de segmentation par région qui, elles, recherchent les homogénéités. Nous présentons ici la détection de contour par recherche de rupture d'homogénéité via l'utilisation de filtres de détection de contours et via l'utilisation de techniques de minimisation de l'énergie associée à un contour.

La première sous-section décrit principalement les outils de filtrage. Dans la deuxième sous-section, nous retrouvons la notion de minimisation de fonctionnelle avec la présence d'un terme d'attache aux données et de termes de régularisation.

1.2.1 Les détecteurs de contours

Les approches d'extraction de contours utilisent le filtrage d'images pour extraire les pixels frontières. Ainsi, on utilise des filtres dérivateurs tels que les filtres de Roberts [Roberts 65], Prewitt [Prewitt 70], Sobel [Sobel 70], Canny [Canny 83] ou Deriche [Deriche 87]. Par exemple le filtre de Prewitt est défini par deux masques ∇_x et ∇_y comme ceci :

$1/6$

1	0	-1
1	0	-1
1	0	-1

∇_x $\quad 1/6$

-1	-1	-1
0	0	0
1	1	1

∇_y

La norme du gradient de l'image I en un point (x, y) est alors donnée par :

$$\|\Delta I(x,y)\| = \sqrt{(\nabla_x * I(x,y))^2 + (\nabla_y * I(x,y))^2}.$$

Une seconde classe d'opérateurs est basée sur l'évaluation du Laplacien :

$$\Delta I(x,y) = \left(\frac{\partial^2 I(x,y)}{\partial x^2} + \frac{\partial^2 I(x,y)}{\partial x^2} \right).$$

Le changement rapide de signe du Laplacien et son passage par zéro indique la présence d'une frontière de région. En effet, lorsque les valeurs de $I(x, y)$ changent plus fortement que linéairement, on assiste à un changement de signe du Laplacien au point d'inflection de $I(x, y)$. On arrive alors à définir plus facilement des courbes fermées ainsi que les maxima locaux.

Ces techniques donnent des résultats peu exploitables directement. En effet, on se retrouve le plus souvent avec des contours non fermés, des faux contours, des contours non détectés (cf. figure 1.7). Des heuristiques sur la taille, la forme, et la proximité sont alors mise en œuvre pour fermer les contours et trouver ceux qui sont intéressants [Moulet et al. 88] [Cocquerez et al. 95]. Cependant, ces algorithmes génèrent souvent des contours artificiels et bruités.

1.2.2 Les contours actifs

Les contours actifs

L'approche par contours actifs (« snakes ») consiste à trouver un contour fermé qui va délimiter une région [Kass et al. 88] [Terzopoulos et al. 87]. Le calcul de ce contour est

FIG. 1.7 – *Illustration de l'approche par extraction du passage par zéro du Laplacien puis par seuillage*

itératif (le contour actif se déforme – se déplace – à chaque itération (cf. figure 1.8), de sorte que l'on minimise l'erreur composée d'une énergie d'attache aux données E^d (qui permet d'attirer le contour vers des caractéristiques de l'image) d'une énergie de régularisation E^r (qui permet de contraindre la forme et l'évolution du contour) et d'une énergie E^c permettant d'ajouter des contraintes sur la solution (équation 1.12).

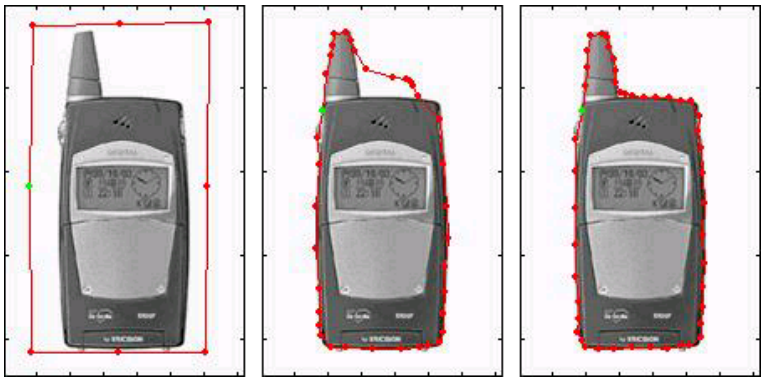


FIG. 1.8 – *Illustration de l'évolution d'un contour actif*

On définit le contour actif par une courbe paramétrique C (par exemple une B-Spline), telle que :

$$C : [0,1] \rightarrow \mathbb{R}^2, s \rightarrow C(s).$$

L'énergie du contour actif à minimiser est alors [Paragios 00] :

$$E = \alpha \int_0^1 E^d(C(s))ds + \beta \int_0^1 E^r(C(s))ds + \gamma \int_0^1 E^c(C(s))ds. \qquad (1.12)$$

L'énergie d'attache aux données E^d utilise les données de l'image I. Cette énergie peut être composée de termes qui permettent d'attirer le contour vers les frontières de texture

$E_{contour}$ et les angles E_{angle} :

$$E^d(C(s)) = w_{contour} \underbrace{|\nabla I(C(s))|^2}_{E_{contour}} + w_{angles} \underbrace{\frac{\partial \theta(C(s))}{\partial n_t(C(s))}}_{E_{angle}}.$$

Les valeurs $w_{contour}$ et w_{angles} sont des constantes de pondération. $\theta(C(s))$ correspond à la direction du gradient au point $C(s)$ et $n_t(C(s))$ est le vecteur unité perpendiculaire à la direction du gradient.

L'énergie de régularisation E^r permet d'imposer des contraintes sur la courbure du contour $E_{courbure}$, sur son élasticité $E_{tension}$:

$$E^r(C(s)) = w_{tension} \underbrace{|\frac{\partial C}{\partial s}(s)|^2}_{E_{tension}} + w_{courbure} \underbrace{|\frac{\partial^2 C}{\partial^2 s}(s)|^2}_{E_{courbure}}$$

Enfin l'énergie E^c permet d'imposer des contraintes sur la solution. On peut citer par exemple l'énergie ballon qui permet d'introduire un potentiel de dilatation sur le contour. Cette énergie force le contour à se dilater ou bien à se contracter.

$$w_{ballon} \underbrace{\frac{\partial C}{\partial s}(s) \times C(s)}_{E_{ballon}} \tag{1.13}$$

L'approche par contour actif est très intéressante dans de nombreuses applications. Ce types d'approches présente néamoins de nombreux problèmes :

- la nécessité d'avoir une initialisation proche de la solution,
- la nécessité de choisir entre une dilatation et une contraction lorsque l'on utilise la force ballon,
- le grand nombre de paramètres et leur sensibilité,
- la dépendance de la solution par rapport aux paramètres de la courbe,
- la difficulté de changer la topologie du contour (il est difficile de faire varier le nombre de contours).

contour actif géodésique

Les contours actifs géodésiques [Caselles et al. 95] [Caselles et al. 97] [Kichenassamy et al. 95] ont été introduits comme une alternative géométrique aux contours actifs et peuvent être vus comme une extension.

Sans perte de généralité, soit le modèle de snake simplifié suivant (avec $w_{courbure} = 0$) :

$$E = \alpha \int_0^1 |\frac{\partial C}{\partial s}(s)|^2 ds + (1-\alpha) \int_0^1 g(|\nabla I(C(s))|)^2 ds \tag{1.14}$$

avec :

$$\begin{array}{l} g : [0, \infty[\rightarrow \mathbb{R}^+ \\ g(0) = 1 \\ g(x) \rightarrow 0 \ quand \ x \rightarrow \infty, \end{array}$$

tel que :

$$g(x) = w_{contour} \times g_{contour}(x)$$

La fonction d'énergie 1.14 dépend de la paramétrisation de la courbe ce qui est contraignant. Pour remédier à ce problème, il a été montré [Caselles et al. 95] [Aubert et al. 99] que la minimisation de l'énergie 1.14 est équivalente à la minimisation de l'énergie de la courbe géodésique donnée par :

$$E' = \int_0^1 \underbrace{g(|\nabla I(C(s))|)}_{\text{attraction}} \underbrace{|\frac{\partial C}{\partial s}(s)|}_{\text{régularisation}} ds, \tag{1.15}$$

La minimisation de l'équation énergétique E' (équation 1.15) est effectuée par descente de gradient. On obtient l'équation du mouvement du contour actif géodésique :

$$\frac{\partial C}{\partial t} = \underbrace{g(|\nabla I|)\mathcal{K}\mathcal{N}}_{\text{force de frontière}} - \underbrace{(\nabla g(|\nabla I|)\mathcal{N})\mathcal{N}}_{\text{force de rafinement}}, \tag{1.16}$$

avec t représentant les itérations d'évolution du contour, $\mathcal{N}$ le vecteur Euclidien normal à la courbe C orienté vers l'intérieur, et $\mathcal{K}$ la courbure Euclidienne ($\mathcal{K} = div(\frac{\nabla I}{|\nabla I|})$).

On constate que la formulation du déplacement du contour actif fait apparaître en chaque point de la courbe, deux forces le long de la normal à la courbe :

- la première force (force de frontière) déplace la courbe vers les frontières d'objets et ceci de manière régulière, grâce à la courbure. Cette force tend à s'amenuiser lorsque les frontières d'objets sont atteintes ($g(|\nabla I(C(s,t)| \rightarrow 0$ lorsque $\nabla I(C(s,t)$ tend vers une grande valeur) ou lorsque la courbure $\mathcal{K}$ tend vers zéro,
- la deuxième force (force de raffinement) prend effet lorsque la courbe est proche des vrais frontières d'objet ($\nabla g(|\nabla I|) \neq 0$). Elle a alors le rôle d'attirer la courbe vers les vrais frontières en contrecarrant les effets de la force de frontière et de raffiner la courbe autour des vrais frontières.

Le contour actif géodésique apporte en plus du contour actif classique l'indépendance de la solution par rapport à la paramétrisation de la courbe. Cependant, il est très sensible aux informations locales. Ainsi, beaucoup d'approches actuelles ajoutent à la formulation énergétique des termes de régions. On parle alors d'approche par régions actives géodésiques.

La méthodologie par régions actives géodésiques

Par rapport à l'approche de contour actif géodésique, on ajoute des contraintes sur les régions et non plus seulement sur le contour [Paragios et al. 98] [JehanBesson et al. 01]. Cela permet de mieux prendre en compte l'homogénéité interne des régions. De plus, le système énergétique présenté permet d'être moins sensible à la phase d'initialisation.

Les auteurs de [JehanBesson et al. 03] proposent un schéma général de résolution du problème de contour actif utilisant des descripteurs de régions pouvant varier. L'image est partitionnée en deux ensembles de régions Ω_{in} et Ω_{out}. L'ensemble des frontières entre les

deux ensembles de régions est noté Γ. Les deux ensembles de régions sont chacuns décrits par $k^{(in)}$ et $k^{(out)}$. Ces descripteurs peuvent par exemple être constants ou fonction de l'intensité moyenne, de la variance, du déterminant de la matrice de covariance etc. Le descripteur de frontières est noté $k^{(b)}$ et peut par exemple être égal à $g(|\nabla I|)$ [Caselles et al. 97]. En prenant τ le paramètre d'évolution, la formulation général du problème de contour actif basé région se note :

$$E = \iint_{\Omega_{out}(\tau)} k^{(out)}(x,y,\Omega_{out}(\tau))dxdy + \iint_{\Omega_{in}(\tau)} k^{(in)}(x,y,\Omega_{in}(\tau))dxdy + \int_{\Gamma(\tau)} k^{b}(x,y)ds$$

Le vecteur vitesse est alors calculé par dérivation (avec utilisation d'un outil mathématique d'optimisation de forme). On constate alors qu'avec l'utilisation de descripteurs de régions variant lors des itérations, des termes additionnel apparaissent dans l'équation d'évolution. Ces termes additionnel proviennent de l'expression des descripteurs sous forme de combinaisons de caractéristiques globalement attachés à l'évolution des régions. (les détails de ces termes additionnels sont donnés dans l'article [JehanBesson et al. 03]) :

$$\begin{aligned}\frac{\partial\Gamma(\tau)}{\partial(\tau)} &= \Big[k^{(in)} - k^{(out)} + k^{(b)}.\mathcal{K} - \nabla k^{(b)}.\mathcal{N} \\ &\underbrace{+\sum_{j=1}^{p} A_j^{(in)} H_j^{(in)} - \sum_{j=1}^{m} A_j^{(out)} H_j^{(out)}}_{termes\quad additionnels} \\ &\underbrace{+\sum_{j=1}^{p} A_j^{(in)} \sum_{i=1}^{lj} B_{ji}^{(in)} L_{ji}^{(in)}}_{termes\quad additionnels} \\ &\underbrace{-\sum_{j=1}^{p} A_j^{(out)} \sum_{i=1}^{kj} B_{ji}^{(out)} L_{ji}^{(out)}}_{termes\quad additionnels} \Big]\mathcal{N}\end{aligned}$$

Si les descripteurs $k^{(in)}$ et $k^{(out)}$ ne dépendent pas de τ (descripteurs indépendants des régions), l'équation se réduit à l'expression suivante :

$$\frac{\partial\Gamma(\tau)}{\partial(\tau)} = \Big[k^{(in)} - k^{(out)} + k^{(b)}.\mathcal{K} - \nabla k^{(b)}.\mathcal{N}\Big]\mathcal{N}$$

On retrouve le même terme de force de frontière que pour l'équation 1.16 qui attire le contour vers des arêtes présentes dans l'image. Le terme de force de région déplace le contour de sorte que les pixels appartiennent après déplacement aux régions les décrivant le mieux.

Finalement, il subsiste le problème d'impossibilité de changer la topologie du contour par rapport à l'initialisation. Ce problème peut être résolu par la méthodologie par ensembles de niveaux (« level set »).

La méthodologie par ensemble de niveaux

Osher et Sethian on introduit [Osher et al. 88] pour palier ce genre de problèmes de changement de topologie, une résolution par ensembles de niveaux (« level-set »). Cela consiste tout simplement à considérer non plus un contour mais une grille sur laquelle chaque point connaît sa distance au contour actif. Ainsi, la résolution du problème ne se fait plus en déplaçant le contour actif mais en modifiant les valeurs des points de la grille. Les points ayant une distance nulle représentent la courbe de niveau zéro c'est-à-dire le contour solution. Il faut noter que cette courbe de niveau zéro est topologiquement indépendante puisque le contour peut changer de topologie lors de l'évolution dans le temps des courbes de niveaux. Ainsi, la solution peut être composée d'un ensemble de contours fermés.

De manière plus formelle, on définit une fonction φ que l'on appelle la surface d'ensemble de niveau, telle que :

$$\varphi(x, y, t) : \mathbb{R}^2 \times [0, T[\rightarrow \mathbb{R},$$

avec la propriété que les points (x, y, t) dont la valeur $\varphi(x, y, t)$ vaut zéro définissent la courbe $C(s, t)$ (on parle de l'ensemble de points de niveau zéro) :

$$\begin{cases} C(s, 0) = \{(x, y) | \varphi(x, y, 0) = 0\} \\ C(s, t) = \{(x, y) | \varphi(x, y, t) = 0\}. \end{cases}$$

La fonction φ est souvent la distance Euclidienne avec un signe négatif pour indiquer les points à l'intérieur de la courbe et positif pour indiquer les points à l'extérieur de la courbe.

En posant que l'équation du mouvement de la courbe C est fonction de la courbure $\mathcal{K}$, on a :

$$\frac{\partial C}{\partial t} = F(\mathcal{K})\mathcal{N}.$$

On trouve alors la formulation de l'équation de mouvement de φ :

$$\frac{\partial \varphi}{\partial t} = F(\mathcal{K})|\nabla \varphi|.$$

Cette équation de mouvement permet de modifier à chaque itération l'ensemble des valeurs de la grille représentant la fonction φ. Ainsi, la méthodologie par ensembles de niveaux permet de résoudre le changement de topologie d'un contour de manière élégante (cf. figure 1.9).

1.3 Le suivi de segmentation : le «tracking»

La segmentation vidéo peut être vue en deux grandes étapes :

- la segmentation d'une image ou d'un groupe d'images (segmentation 3D) par une approche basée contour ou région,
- puis le suivi de cette segmentation sur les images suivantes.

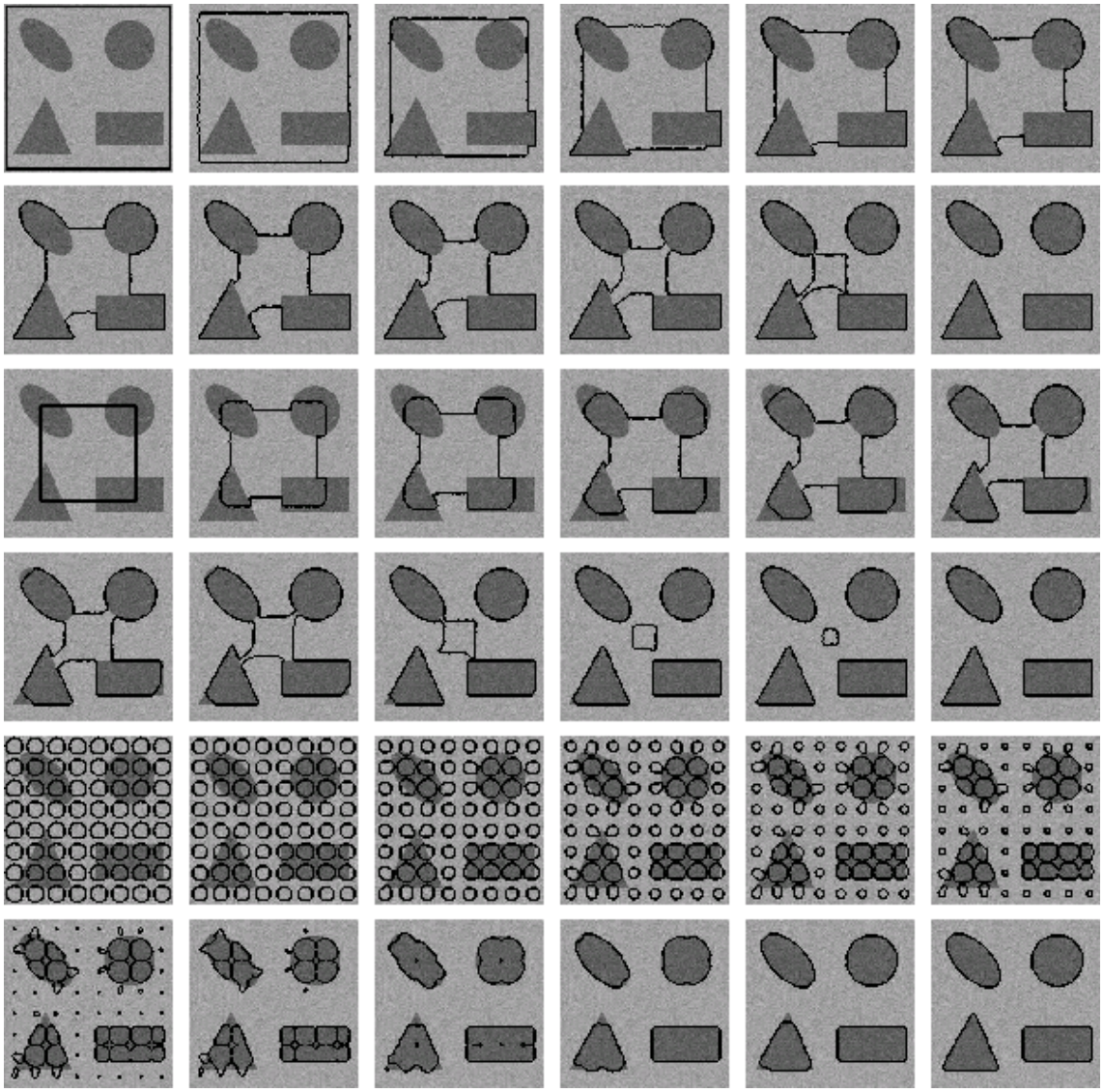


FIG. 1.9 – *Illustration de l'évolution de l'ensemble de niveau zéro pour une modélisation en régions géodésiques actives. Figures extraites de [Paragios 00]. Trois initialisations différentes mènent à la même solution. Le changement de topologie est obtenu automatiquement grâce à la résolution par ensemble de niveaux*

Le suivi permet donc d'étendre sur plusieurs images la segmentation. Bien entendu, le suivi doit être robuste, c'est-à-dire permettre de suivre une zone même si elle change de forme, de texture ou bien est occultée. Deux approches sont possibles :

- une approche par mise en correspondance de deux cartes de segmentation consécutives. La mise en correspondance de l'image au temps t et de celle au temps $t+1$ se fait en identifiant les régions présentes au temps t dans la carte de segmentation au temps $t+1$. On utilise alors des critères de recouvrement;
- une approche par projection suivie d'une re-segmentation avec utilisation de la carte projetée comme initialisation ou bien comme référence. Ce suivi revient à calculer de nouvelles segmentations avec l'a priori de connaître approximativement la partition solution. En effet, on utilise la projection de la carte de segmentation du temps t vers le temps $t+1$ comme initialisation ou référence du problème de segmentation de l'image I_{t+1}.

1.3.1 L'approche par mise en correspondance

Beaucoup d'approches essayent de faire de la mise en correspondance de régions issues de deux segmentations, l'une au temps t et l'autre au temps $t+1$. Dans [Marqués et al. 98] par exemple, l'algorithme tente d'affecter l'ensemble des régions définies à l'instant $t+1$ à un des objets connus à l'instant t. Ce processus est ici réalisé à l'aide de trois phases algorithmiques :

- la première phase apparie toutes les régions de $t+1$ qui recouvrent totalement une région de t projetée au temps $t+1$,
- la deuxième phase étiquette toutes les régions de $t+1$ recouvertes à plus de cinquante pour-cent par une région de t projetée au temps $t+1$ sous la contrainte que la distance de couleur entre les deux régions n'est pas trop grande,
- la troisième phase affecte toutes les régions incertaines restantes à un des objets via un algorithme de ligne de partage des eaux en prenant pour germes les régions précédemment affectées.

De manière assez proche, [Foret et al. 02] apparie les régions similaires via un « bloc matching » en avant. Les zones non appareillés sont alors segmentées en régions (par un algorithme de croissance de régions prenant en compte les régions déjà étiquetées), puis une projection arrière permet d'attribuer ces régions.

Dans [Alatan et al. 98], il y a calcul d'une carte de segmentation en objets vidéo au temps t et au temps $t+1$, par détection des régions spatiales en mouvement. Ensuite, la mise en correspondance des objets du temps t avec ceux du temps $t+1$ est effectuée en prenant en compte trois règles :

- une règle qui permet de suivre les objets précédemment détectés,
- une règle permettant de détecter de nouveaux objets (ceux se détachant du fond),
- une règle permettant de repérer les objets se scindant en plusieurs objets.

Les règles sont basées sur l'intersection des régions du temps t projetées vers $t+1$ avec les régions du temps $t+1$. L'intérêt de ces règles est que le suivi est évolutif. En effet, le nombre d'objets peut varier.

On constate que ces approches sont empiriques, avec l'utilisation de seuils très sensibles. Les difficultés rencontrées viennent de :

- la superposition d'une région projetée sur plusieurs régions,
- l'apparition de régions lors de découvrements,
- la disparition de régions lors d'occultations,
- la sensibilité des segmentations spatiales,
- la variation du nombre de régions d'une segmentation à l'autre.

1.3.2 L'approche par projection initialisation

Beaucoup d'approches utilisent la projection comme initialisation ou référence pour une nouvelle segmentation. En effet, la carte de segmentation projetée du temps t vers $t+1$ n'est pas très loin de la segmentation solution du temps $t+1$. Ainsi, on atteint rapidement la solution et l'on peut contraindre la segmentation à être peu éloignée de l'initialisation.

Par exemple, [Castagno et al. 98] utilisent la projection des cartes de probabilité d'affectation comme initialisation au problème de clustering flou.

De manière similaire, [McKenna et al. 98] utilisent un mélange de gaussiennes qui évoluent dans le temps. Ainsi, les paramètres des gaussiennes varient au cours du suivi pour s'adapter aux changements d'illuminations, aux changements de forme et aux occultations.

Les auteurs de [Mazière et al. 00] projettent le contour actif et le dilatent pour initialiser une nouvelle segmentation par contour actif.

[Mansouri et al. 00] utilisent la région géodésique projetée au temps $t+1$ et imposent que la solution soit proche de cette projection. Les critères sont :

- une géométrie similaire entre la solution au temps t et la solution au temps $t+1$,
- des régions à intensités similaires entre la solution au temps t et la solution au temps $t+1$,
- et une attraction de la région géodésique vers les zones de gradient fort.

Les approches par projections sont intéressantes car on dispose d'une certaine stabilité dans le temps des cartes de segmentation. Cependant, les zones de découvrement et de recouvrement posent encore problème. Ainsi, même en proposant des techniques de calcul d'ordre de profondeur basée DFD [BenoisPineau et al. 02], des techniques de cohérence temporelle ou de cohérence spatiale [Odobez 94], le fait de restreindre à deux images la segmentation limite l'apport d'information dans ces zones. La section suivante aborde l'extension de la segmentation sur plusieurs images.

1.4 Les nouvelles approches basées long terme : la notion de tube spatio-temporel

Les approches couramment utilisées en segmentation vidéo se limitent à l'utilisation de deux images successives. Certaines ambiguïtés sont alors impossible à résoudre. Ainsi, les zones d'occultations sont difficilement attribuables à une région ou bien à un des objets vidéo. De plus, la distinction de régions ou d'objets par le mouvement est impossible lorsque les mouvements sont faibles ou similaires.

L'utilisation d'un contexte temporel long terme permet de réduire la sensibilité des algorithmes à l'occultation et aux mouvements proches. En effet, en utilisant un contexte temporel de plusieurs images, on améliore la stabilité, la robustesse et la cohérence des résultats.

Toutes les techniques basées long terme cherchent à obtenir des tubes spatio-temporels, c'est-à-dire des régions ou des objets qui ont un mouvement et une texture homogènes et stables sur plusieurs images. La figure 1.10 illustre la notion de tube. Bien entendu, l'analyse d'un groupe d'images doit être faite de manière conjointe spatialement et temporellement. Megret et al. [Megret et al. 02] proposent une revue détaillée de ces approches de segmentation spatio-temporelle long terme avec priorité spatiale, priorité temporelle ou bien de manière conjointe.

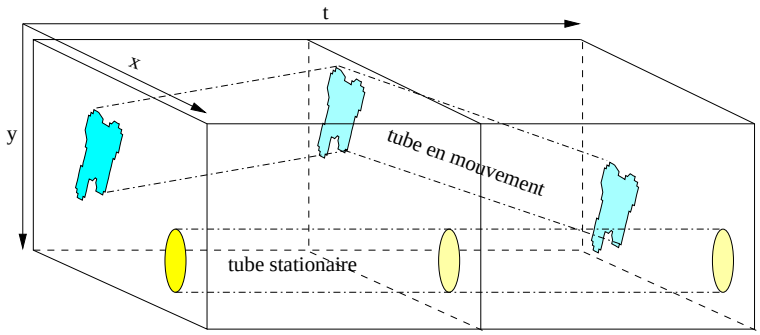


FIG. 1.10 – *Illustration de la notion de tube spatio-temporel*

Dans le cas des approches avec priorité spatiale, on retrouve les approches de type segmentation sur deux images puis un suivi de la segmentation. On ne bénéficie alors pas totalement des avantages du long terme.

Dans le cas des approches avec priorité temporelle, c'est-à-dire d'analyse de mouvement, il est nécessaire de connaître le nombre de trajectoires, sans quoi on obtient des frontières spatiales aberrantes. Encore une fois, on perd alors les avantages du long terme.

Dans la suite de cette section sur les tubes spatio-temporels, nous évoquerons uniquement les techniques de segmentation long terme conjointes spatio-temporellement. Nous expliquerons ainsi comment obtenir des régions spatio-temporelles ou bien des ensembles de régions spatio-temporelles par deux exemples récents.

1.4.1 L'approche basée graphe de Parker et al.

L'approche de [Parker et al. 01] permet d'obtenir une carte de segmentation spatiale 2D+t robuste et stable dans le temps. La technique mise en œuvre est basée sur l'approche division-fusion à partir d'un graphe tridimensionel. On retrouve une technique 2D utilisant un graphe (Region Adjancy Graph) étendu à la 3D.

À l'initialisation, chaque nœud représente un pixel appartenant au groupe d'images traité. Les nœuds sont alors fusionnés spatialement ou temporellement en choisissant la fusion de distance minimale entre tous les nœuds. Un critère supplémentaire est ajouté pour ne pas fusionner les tubes en contact de manière temporaire. L'arrêt des fusions des nœuds est gérée automatiquement.

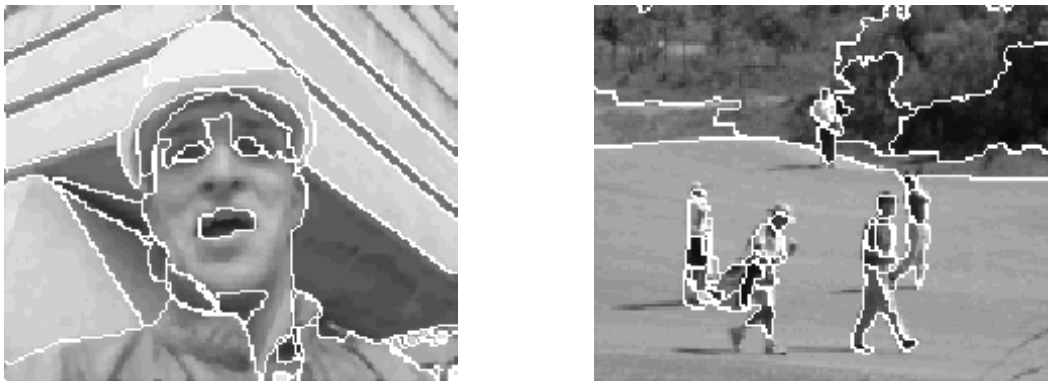
On modélise le problème par la minimisation d'une énergie E composée d'un terme d'attache aux données $E^d_{r,t}$ et d'un terme de régularisation $l(B)$ portant sur la longueur des frontières de région :

$$E = \sum_{t=1}^{T} \sum_{r=1}^{R} \left\{ E^d_{r,t} + \lambda . l(B) \right\}, \tag{1.17}$$

avec :

$$E^d_{r,t} = \sum_{i \in R_r} dist^2(I(i,t), \mu_r).$$

L'approche permet donc d'avoir une fusion des régions simultanément sur les trois dimensions x, y, t. La carte de segmentation spatiale 2D+t définit les tubes spatio-temporels. La figure 1.11 illustre la segmentation obtenue sur la séquence (« Foreman ») et (« golfers »). On peut constater que la segmentation est encore composée de nombreuses régions mais il faut noter que l'approche ne cherche pas à obtenir des objets vidéos. En effet, aucune notion de mouvement n'est utilisée. Par contre les régions sont représentatives des zones d'intérêt de l'image et la notion de tube permet d'avoir une segmentation stable sur le groupe d'images.



(a) séquence «Foreman» (b) séquence «golfers»

FIG. 1.11 – *Segmentations résultantes d'une approche par tube spatio-temporel. Figure extraites de [Parker et al. 01]*

1.4.2 L'approche croissance de région et fusion de tubes de Porikli et al.

De la même façon que Parker et al, [Porikli et al. 01] cherchent à obtenir dans un premier temps des tubes spatio-temporels. Ils utilisent une technique de croissance de régions à partir de marqueurs (germes). Le marqueur est choisi parmi le groupe d'images comme une zone de plus faible gradient. On fait alors grandir la région spatialement et temporellement autour du marqueur. La fusion d'un pixel avec un tube est effectuée si la distance de texture entre le pixel et le tube est faible. On peut remarquer que la technique d'obtention de tube spatio-temporel peut être modélisée de façon très similaire à celle de

Parker. En effet, on minimise une somme composée de la même énergie d'attache aux données et d'une énergie de régularisation favorisant l'homogénéité de texture du tube.

Une fois que les tubes spatio-temporels sont obtenus, une deuxième étape est mise en œuvre pour fusionner les tubes entre eux. Trois critères sont retenus :

- un critère de similarité de mouvement entre les deux tubes,
- un critère sur le ratio de compacité avant et après fusion des tubes,
- et un critère sur le ratio de la taille des frontières avant et après fusion des tubes.

Un seuil automatique est utilisé pour stopper les fusions.

Cette approche est très intéressante car elle propose d'obtenir des objets vidéo et ceci en raisonnant non plus sur des pixels ou des régions mais directement sur des tubes spatio-temporels. Or, les tubes spatio-temporels ont des propriétés de stabilité et de cohérence bien plus fortes. La figure 1.12 illustre le processus de segmentation utilisant une succession d'étapes avec des critères différents en fonction du niveau où l'on se trouve (niveau pixels/nano, niveau tubes/micro, niveau groupe de tubes/macro). On constate que la dernière étape, au niveau groupe de tubes, permet d'obtenir des objets par une distinction entre objets de type fond, et objets en avant plan. L'utilisation de différentes étapes dans la segmentation est justifiée car les caractéristiques ne sont discriminantes que pour un niveau donné.

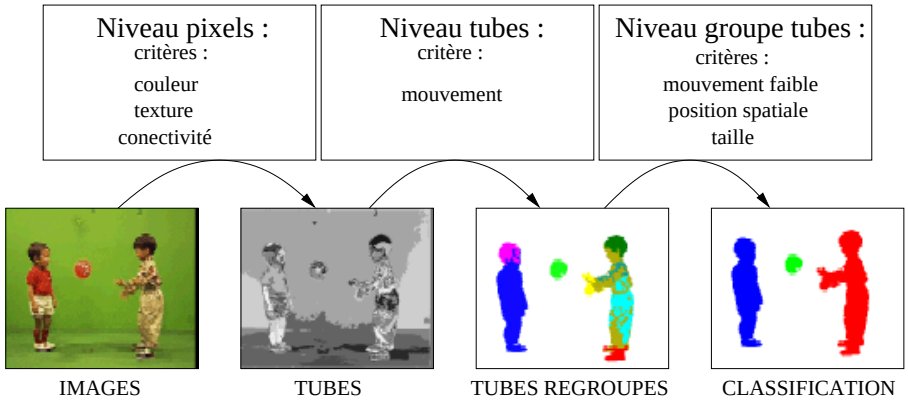


FIG. 1.12 – *Illustration des différentes étapes utilisées pour la segmentation en objets vidéos de [Porikli et al. 01]*

Les tubes spatio-temporels obtenus permettent de partir sur de bonnes bases pour l'obtention d'objets vidéo. On peut cependant remarquer que le critère de fusion basé mouvement est un peu trop simple. En effet, la similarité de mouvement est calculée sur un mouvement translationnel. De plus, la première étape qui consiste à obtenir les tubes peut échouer dans le cas où il y a un fort mouvement sur le groupe d'images.

Tant dans l'approche long terme de Parker que dans celle de Porikli, on essaye dans un premier temps de trouver des tubes spatio-temporels. Dans un deuxième temps, la fusion de ces tubes permet d'obtenir des objets vidéo. La notion d'objet vidéo n'est pourtant pas bien définie. Les critères de fusion de tubes sont assez empiriques. Le chapitre suivant propose une formalisation de la notion d'objets vidéos et donne une approche pour la segmentation automatique en objets vidéos.

1.5 Résumé du chapitre

Ce chapitre présente les grandes classes de technique de segmentation. Chacune des techniques recherche un ensemble de régions homogène ou bien un ensemble de ruptures d'homogénéité. Cette recherche revient à minimiser une fonctionnelle d'énergie représentant l'homogénéité ou la rupture d'homogénéité. Quelque soit la technique utilisée (contours actifs, croissance de région ou approches Bayes/MDL), le problème posé est équivalent [Zhu et al. 96]. Ainsi, le choix d'un algorithme est fonction de ses avantages et de ses inconvénients.

De manière générale, la formulation énergétique du problème fait intervenir deux termes : un terme d'attache aux données E^d et un terme de régularisation E^r. Le terme d'attache aux données E^d permet de prendre en compte les caractéristiques propres aux images c'est à dire l'intensité, la variance, la position, le mouvement de chaque pixel. Le terme de régularisation E^r permet d'imposer une certaine homogénéité aux résultats ainsi que de maîtriser le nombre final de régions.

Une des évolutions actuelles de la segmentation vidéo est de prendre en compte l'aspect temporel long terme. On recherche à segmenter directement sur un groupe d'images et ainsi à obtenir des tubes. Cette approche se justifie par le fait qu'elle permet une meilleure gestion des zones à problème : zones d'occultation et de découvrement. La modélisation énergétique du problème se voit alors enrichie d'une dimension temporelle.

Le chapitre suivant formule la notion d'objet vidéo par un modèle de mouvement long terme et une texture. Puis, sachant ce modèle, la formulation énergétique met en jeu les deux termes E^d et E^r. La résolution du problème de minimisation peut alors être effectuée en utilisant une technique quelconque de segmentation.

Chapitre 2

Vers la notion d'objet vidéo

2.1 Définition d'un objet vidéo

Définir un objet vidéo est chose difficile. Intuitivement, un objet vidéo est défini par une forme, une texture et un mouvement (rigide ou non rigide). Cependant, la notion d'objet vidéo est beaucoup plus descriptive qu'une simple région. Comme expliqué dans le préambule, et dans la deuxième partie du manuscrit, un objet vidéo peut par exemple être un modèle 3D.

La notion d'objet ne fait pas forcément référence à un objet du monde réel. En effet, dans le domaine de la vidéo, un objet n'est pas nécessairement un objet d'une scène 3D mais plutôt le résultat de l'analyse de la projection d'un monde 3D sur un plan. Ainsi, un objet vidéo est défini comme une région de la vidéo conforme à un modèle. On peut par exemple avoir pour modèle :

- un modèle de mouvement (e.q. affine, quadratique, etc)
- un modèle d'objet physique (e.q. modèle de visage, modèle de corps humain etc)

La notion d'objets est alors subjectivement définie par rapport aux modèles utilisés.

On constate que pour pouvoir trouver de manière automatique des objets vidéo, il est nécessaire de proposer des modèles auxquels ces objets répondront. Ainsi, la segmentation automatique en objet vidéo nécessite de préciser ce que l'on recherche, c'est-à-dire de préciser quel est le ou les modèles que l'on souhaite trouver. Les modèles proposés définissent alors la sémantique de la scène. La richesse de cette sémantique dépend alors directement de la richesse des modèles proposés et peut donc différer d'une sémantique "humaine".

La plupart des méthodes de segmentation existantes aujourd'hui essayent de regrouper des régions ayant des caractéristiques de texture et de mouvement communes. Comme on l'a vu précédemment, on peut par exemple obtenir des tubes spatio-temporels [Porikli et al. 01]. Cependant, une fois les tubes obtenus, leur fusion dans l'objectif d'obtenir des objets vidéo ayant un contenu plus significatif n'est pas encore très bien formalisée. Ainsi, dans la suite de ce chapitre, nous proposons un modèle pour formaliser la notion d'objet vidéo. De plus, nous présentons une technique de segmentation via cette modélisation.

2.2 Modélisation spatio-temporelle d'un objet vidéo

Le modèle de représentation d'un objet vidéo que nous proposons suppose qu'un objet vidéo est défini par son mouvement, qui le distingue des autres objets. Ainsi, notre approche est basée sur l'utilisation d'un mouvement long terme par objet et sur la stabilité de la texture de l'objet sachant le mouvement de celui-ci. Notre modèle définit un objet k par son mouvement Θ_k et sa mosaïque M_k (voir l'illustration d'une mosaïque sur la figure 2.1). On recompose ainsi chacune des images de sorte qu'un pixel i de l'image I_t à un instant t soit définit par le pixel $\Theta_k^{t \to t_{ref}}(i)$ de la mosaïque M_k de l'objet k :

$$\forall t, \forall i, \exists k, I_t(i) = M_k(\Theta_k^{t \to t_{ref}}(i)) + \eta \tag{2.1}$$

où η correspond à un bruit de modèle (supposé ici être un bruit blanc gaussien).

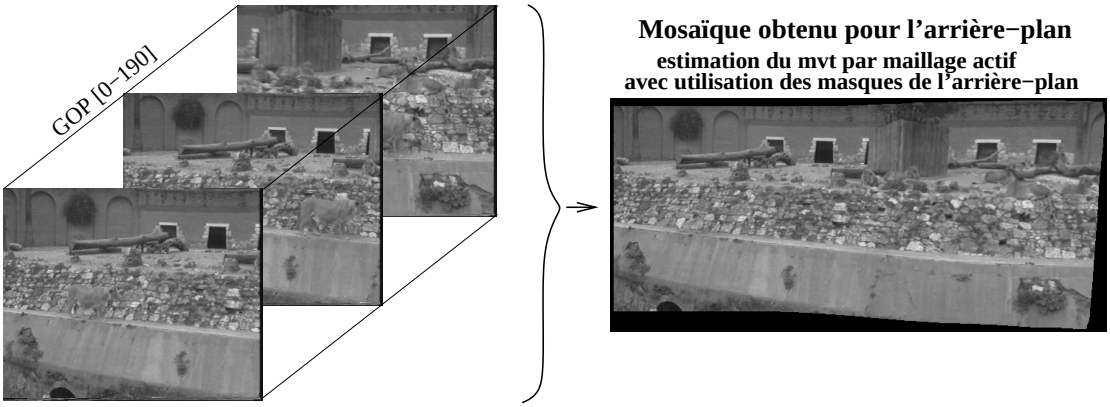


FIG. 2.1 – *Mosaïque de la séquence Lion sur le GOP [0-190]*

Ainsi, le modèle définissant un objet repose sur la stabilité temporelle de la texture d'un objet. On travaille sur un groupe d'images ce qui permet de régler plus aisément les affectations dans les zones d'occultation et de découvrement. De plus, nous utilisons un mouvement par objet estimé par un maillage actif [Marquant et al. 00], ce qui permet d'avoir une description fine. Enfin, nous choisissons une texture mosaïque non dynamique en faisant l'hypothèse que la texture d'un objet varie peu sur un groupe d'images si l'on possède un mouvement fin de cette texture.

Le principe de la segmentation est alors de rechercher les différents objets vérifiant le modèle que nous proposons. Nous devons donc déterminer le nombre d'objets K, le mouvement long terme de chacun des objets Θ_k et la texture de chacun des objets M_k. L'algorithme de segmentation met alors en concurrence les mouvements de chaque objet et ensuite affecte les pixels aux objets les plus probables. Nous découpons l'approche en deux étapes (voir schéma 2.2):

– une étape d'initialisation (section 2.4) qui consiste à trouver une localisation grossière pour chaque objet (on parlera de germes). Les germes sont utilisés comme support pour l'estimation du mouvement des objets. On obtient ainsi l'ensemble des mouvements Θ_k ;

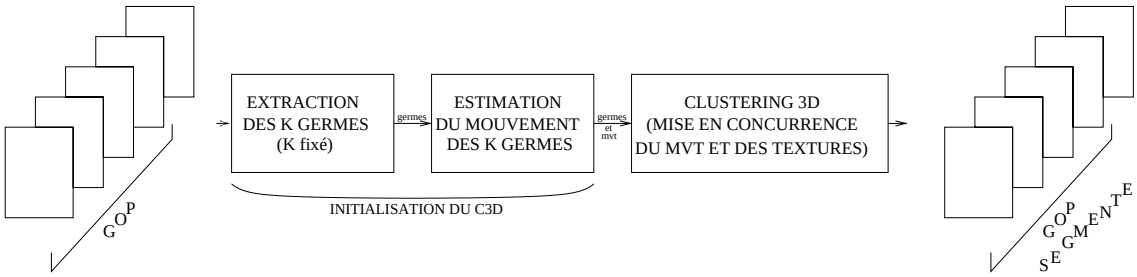


FIG. 2.2 – *Illustration du principe de segmentation par clustering 3D (C3D)*

– une deuxième étape (section 2.5) qui consiste à mettre en concurrence les mouvements Θ_k et les textures M_k associés pour obtenir la segmentation finale selon le modèle d'objet de l'équation 2.1.

Par rapport à l'approche de [Wang et al. 94] qui effectue des mises en concurrence de mouvements affines, nous utilisons un mouvement beaucoup plus complexe grâce à l'utilisation d'un maillage. De plus, notre modélisation est basée long terme. Enfin, nous proposons un modèle pour définir ce qu'est un objet vidéo.

La section suivante aborde la formulation énergétique que nous avons retenue pour le calcul de la segmentation en fonction de notre modèle d'objet vidéo.

2.3 Formulation énergétique

La recherche de modèle d'objet vidéo consiste à affecter chaque pixel du groupe d'images à un des objets. Nous formulons ce problème de segmentation comme la minimisation d'une fonctionnelle. Or, les auteurs de [Zhu et al. 96] montrent que quelque soit l'algorithme utilisé, le résultat de la minimisation d'une fonctionnelle est équivalent. Nous avons donc décidé le résoudre ce problème par une approche par clustering 3D flou en s'inspirant des travaux de [Castagno 98] sur le clustering flou. Ce choix est justifié par le fait que le clustering est une technique facile à implémenter, et rapide en temps d'exécution.

Ainsi, nous allons rechercher la probabilité d'appartenance de chacun des pixels à chacun des objets. Le fait d'utiliser des probabilités plutôt que des étiquettes permet d'obtenir une plus grande souplesse lors de la résolution et permet d'évaluer le résultat de chaque affectation.

Ainsi, nous formulons le problème de segmentation comme la minimisation d'une fonctionnelle E faisant intervenir les probabilités d'appartenance. La minimisation de la fonctionnelle d'énergie E (équation 2.2) est une triple somme sur le temps t, les objets k et les pixels i, composée d'un terme d'attache aux données E^d (équation 2.3), d'un terme de régularisation spatial E^{rs} (équation 2.4) et d'un terme de régularisation temporelle E^{rt} (équation 2.5). Les termes d'attache aux données et de régularisations sont expliqués dans les deux sous-sections suivantes.

$$E = \sum_{t=1}^{T}\sum_{k=1}^{K}\sum_{i=1}^{N}\left\{E^d_{i,k,t} + E^{rs}_{i,k,t} + E^{rt}_{i,k,t}\right\}, \tag{2.2}$$

$$E^d_{i,k,t} = P^2_{i,k,t} \times dist(I_t(i), M_k(\Theta_k^{t\to t_{ref}}(i)))^2, \tag{2.3}$$

$$E^{rs}_{i,k,t} = \alpha \sum_{j\in\mathcal{V}(i)} dist(P_{i,k,t}, P_{j,k,t})^2, \tag{2.4}$$

$$E^{rt}_{i,k,t} = \beta P^2_{i,k,t} \sum_{l=1}^{K} \left[\begin{array}{c} dist(P_{i,l,t}, P_{\Theta_k^{t\to t-1}(i),l,t-1})^2 \\ +dist(P_{i,l,t}, P_{\Theta_k^{t\to t+1}(i),l,t+1})^2 \end{array} \right]. \tag{2.5}$$

Deux contraintes supplémentaires sont à ajouter à la formulation énergétique :

$$\forall(i,k,t), P_{i,k,t} \in [0,1],$$

et :

$$\forall(i,t), \sum_{k=1}^{K} P_{i,k,t} = 1.$$

2.3.1 Le terme d'attache aux données $E^d_{i,k,t}$

Le terme d'attache aux données $E^d_{i,k,t}$ (équation 2.3) sert à donner une probabilité d'affectation à chaque pixel. En effet, chaque pixel i du temps t a une probabilité $P_{i,k,t}$ d'appartenir à un objet k. Le terme d'attache aux données permet de donner les plus fortes probabilités aux modèles de mouvements les plus adaptés.

On entend par modèle de mouvement adapté le modèle de mouvement qui produit la plus faible distance *dist* entre l'intensité $I_t(i)$ du pixel i au temps t et l'intensité de la mosaïque M_k du pixel $\Theta_k^{t\to t_{ref}}(i)$ pour l'objet k. On note $\Theta_k^{t\to t_{ref}}$ le mouvement associé à un objet k du temps t vers le temps t_{ref}. La fonction de distance *dist* sera la distance Euclidienne dans notre cas (puisque l'on fait l'hypothèse d'un bruit gaussien sur le modèle). De plus, la valeur de la mosaïque M_k associée à l'objet k est considérée constante le long du temps, c'est-à-dire qu'on ne prend pas en compte les variations de texture au cours du temps.

On constate que le terme d'attache aux données nécessite :

- la connaissance du nombre d'objets,
- la connaissance du mouvement de chaque objet sur un groupe d'images et ceci sur toute la surface de chaque image. On utilise pour ceci un maillage recouvrant une surface bien plus grande que l'objet. Les déformations du maillage sont suivies sur le groupe d'images [Pateux et al. 01] ;
- le calcul d'une mosaïque par objet.

C'est l'étape d'initialisation (section 2.4) qui permet d'extraire les mouvements d'un nombre fixé de germes et de calculer chaque mosaïque sachant les mouvements extraits.

2.3.2 Les deux termes de régularisation $E^{rs}_{i,k,t}$ et $E^{rt}_{i,k,t}$

Les termes de régularisation (équations 2.4 et 2.5) sont introduits pour lisser spatialement et temporellement les résultats du processus de classification en pénalisant la différence de probabilité par objet pour des pixels voisins. Ces termes peuvent être interprétés comme des termes d'énergie de Gibbs utilisés dans les champs de Markov.

Le terme de régularisation spatiale $E^{rs}_{i,k,t}$ est une mesure de la distance des probabilités entre le pixel i et son voisinage spatial $\mathcal{V}(i)$. Puisque le terme d'énergie E est à minimiser, la minimisation du terme $E^{rs}_{i,k,t}$ implique une homogénéisation des probabilités dans un voisinage spatial $\mathcal{V}(i)$. Le terme énergétique $E^{rs}_{i,k,t}$ est pondéré par α.

Le terme de régularisation temporelle $E^{rt}_{i,k,t}$ est une mesure de la stabilité des probabilités le long d'un mouvement. On utilise donc la notion de voisinage temporel avant et arrière. Les distances de probabilité sont pondérées par la probabilité $P_{i,k,t}$ d'appartenir à la classe k parce que la cohérence temporelle dépend de la classe à laquelle le pixel appartient. Ainsi, le terme $E^{rt}_{i,k,t}$ sera peu significatif si la probabilité $P_{i,k,t}$ est faible. Cela signifie qu'il y a lissage préférentiellement si la probabilité $P_{i,k,t}$ est forte. On introduit en plus un terme de pondération β.

2.4 Initialisation du problème

Le modèle d'objets que nous proposons (équation 2.2) nécessite la connaissance du nombre d'objets ainsi que du mouvement de chaque objet. Pour estimer chaque mouvement, il est donc nécessaire d'obtenir une localisation grossière des objets (germes) (voir schéma 2.2). Comme nous l'avons présenté dans la section 1.4 sur la segmentation spatio-temporelle long terme, une solution à l'obtention de germes pourrait être de travailler sur les tubes pour voir quels sont ceux qui formeraient au mieux un objet vidéo.

Cependant, notre objectif dans cette section n'est pas de donner un algorithme qui trouverait des germes de manière totalement automatique. Notre objectif est plutôt d'obtenir des germes qui ressembleraient à ce qu'un algorithme entièrement automatique pourrait donner. Il faut bien noter que la technique présentée ici n'est qu'une initialisation au clustering 3D (section 2.5) et qu'elle peut être substituée par une autre approche.

Une fois les germes obtenus, il est possible d'estimer le mouvement de ceux-ci sur un groupe d'images suffisamment grand. L'estimateur de mouvement par maillage est suffisamment robuste pour effectuer cette tâche. On dispose alors de germes sur toutes les images ainsi que le mouvement associé à chaque germe. Le clustering 3D de la section 2.5 peut alors être mis en œuvre pour résoudre le problème de segmentation en objets vidéo sachant la modélisation de l'équation 2.2.

Cette section aborde le problème d'obtention de germes localisant grossièrement chaque objet.

2.4.1 Méthode de calcul des germes : le clustering affine flou

La solution que nous proposons [Chaumont et al. 02] pour obtenir des germes est de faire un clustering flou affine sur une grande fenêtre temporelle. Ce clustering flou affine permet d'extraire un nombre fixé de modèles affines présents dans le groupe d'images. Le clustering affine permet de donner une probabilité d'appartenance aux pixels de la

première image. On peut alors facilement qualifier la qualité de l'appartenance de chaque pixel et donc définir des germes.

Il faut remarquer que l'on fait deux hypothèses : La première est de connaître le nombre d'objets. La deuxième hypothèse forte qui est faite dans cet algorithme est qu'il recherche des mouvements affines entre images succesives. Or, les objets articulés possèdent un mouvement plus complexe. Cependant, le fait que cela soit un mouvement affine entre images successives permet d'obtenir une description de mouvement suffisamment riche pour définir des germes utilisables par le clustering 3D (voir section 3.1).

Le clustering que l'on met en place est effectué sur un ensemble de T images et donc est de type long terme. Les positions $Pos(i,t)$ de chacun des pixels i de l'image 0 pour chacun des temps t sont connus et obtenus par une estimation long terme par maillage [Pateux et al. 01]. L'estimation des différents mouvements affines est calculée entre le temps t_i et le temps $t_i + \Delta t$ (Δt valant 2 dans nos expériences pour les segmentations traitées).

Le modèle proposé est alors un mouvement affine par objet k pour un temps t. On doit alors avoir pour chaque position $Pos(i,t)$ un modèle $(A_{k,t}, T_{k,t})$ associé :

$$\forall t, \forall i, \exists k, Pos(i, t+\Delta t) = A_{k,t}.Pos(i,t) + T_{k,t} + \eta, \tag{2.6}$$

η étant un bruit de modèle (supposé ici être un bruit blanc gaussien).

On souhaite au final affecter chacun des pixels à un des objets k ayant un modèle composé d'une succession de mouvements affines. On met pour cela en place un clustering flou pour obtenir la probabilité d'appartenance de chaque pixel de l'image 0. On passe alors par la minimisation d'une fonctionnelle d'énergie E :

$$E = \left(\sum_{t=1}^{T-\Delta t} \sum_{k=1}^{K} \sum_{i=1}^{N} P_{i,k}^{m} \times d_{i,k,t}^{2} \right), \tag{2.7}$$

avec :

$$d_{i,k,t} = \| Pos(i, t+\Delta t) - (A_{k,t}.Pos(i,t) + T_{k,t}) \|,$$

où $A_{k,t}, T_{k,t}$ représentent les paramètres affines du mouvement de l'objet k sur l'intervalle $[t,\ t+\Delta t]$ et m le coefficient flou. Ce coefficient est positionné à $1,6$ comme proposé dans [Castagno 98].

La minimisation est réalisée en deux étapes itérées :

- Dans la première étape, les centroïdes représentés par les paramètres de mouvement affines $A_{k,t}, T_{k,t}$ sont mis à jour sachant les valeurs des probabilités d'appartenance $P_{i,k}$ (les probabilités étant initialisées de manière aléatoire). Ce problème est une régression linéaire pondérée par les probabilités. Les équations sont données en annexe (équations A.2, A.3, A.4, A.5, A.6 et A.7).
- Dans la seconde étape, les probabilités $P_{i,k}$ sont mises à jours sachant les valeurs des centroïdes. L'équation 2.10 permet de calculer la valeur des probabilités.

2.4.2 Phase de mise à jour des paramètres affines : $A_{k,t}$ et $T_{k,t}$ et des probabilités $P_{i,k}$

Le calcul de $A_{k,t}$ et $T_{k,t}$ met en jeu la notion de distance $d_{i,k,t}$. Cette distance utilise les positions des pixels i de l'image 0 au temps t que nous appelons position de référence et notons :

$$\begin{pmatrix} X_{ref}(i) \\ Y_{ref}(i) \end{pmatrix} = Pos(i,t),$$

et les positions des pixels i de l'image 0 au temps $t + \Delta t$ que nous appelons positions observées et notons :

$$\begin{pmatrix} X_{obs}(i) \\ Y_{obs}(i) \end{pmatrix} = Pos(i,t+\Delta t).$$

La distance $d_{i,k,t}$ représente donc l'erreur entre la prédiction par le modèle de mouvement affine ($A_{k,t}$,$T_{k,t}$) d'un point de référence $\binom{X_{ref}(i)}{Y_{ref}(i)}$ et la position observée $\binom{X_{obs}(i)}{Y_{obs}(i)}$:

$$d_{i,k,t} = \left\| \begin{pmatrix} X_{obs}(i) \\ Y_{obs}(i) \end{pmatrix} - \left(A_{k,t} \begin{pmatrix} X_{ref}(i) \\ Y_{ref}(i) \end{pmatrix} + T_{k,t} \right) \right\|.$$

On note la transformation affine ($A_{k,t}$,$T_{k,t}$) appliquée en un point $\binom{X_{ref}(i)}{Y_{ref}(i)}$ par :

$$A_{k,t} \begin{pmatrix} X_{ref}(i) \\ Y_{ref}(i) \end{pmatrix} + T_{k,t} = \begin{pmatrix} ax + a00(X_{ref}(i) - \overline{X_{ref}}) + a01(Y_{ref}(i) - \overline{Y_{ref}}) \\ ay + a10(X_{ref}(i) - \overline{X_{ref}}) + a11(Y_{ref}(i) - \overline{Y_{ref}}) \end{pmatrix}, \tag{2.8}$$

avec :

$$\overline{X_{ref}} = \frac{\sum_{i=1}^{i=N} P_i^m X_{ref}(i)}{\sum_{i=1}^{i=N} P_i^m},$$

$$\overline{Y_{ref}} = \frac{\sum_{i=1}^{i=N} P_i^m Y_{ref}(i)}{\sum_{i=1}^{i=N} P_i^m}.$$

$\overline{X_{ref}}$ et $\overline{Y_{ref}}$ sont introduits pour simplifier le calcul des paramètres.

Il nous reste donc maintenant à estimer les paramètres du mouvement affine : ax, ay, $a00$, $a01$, $a10$ et $a11$ pour un temps t et un objet k. Pour cela, il nous faut minimiser la fonctionnelle E (équation 2.7). En annulant la dérivée partielle de E, pour un temps t et un germe k, en fonction de chaque paramètre, on obtient directement ax et ay (équation en annexe A.2 et A.3) et un système linéaire (équations en annexe A.4, A.5 A.6, A.7) à résoudre pour obtenir $a00$, $a01$, $a10$ et $a11$.

Pour calculer les probabilités $P_{i,k}$ il faut ajouter la contrainte suivante : $\forall i, \sum_{l=1}^{l=K} P_{i,l} = 1$. Ainsi, l'équation énergétique E (équation 2.7) se réécrit avec intégration de cette contrainte via l'utilisation des multiplicateurs de Lagrange en l'équation 2.9. L'ajout de la contrainte Lagrangienne permet d'obtenir une expression pour les probabilités $P_{i,k}$ par annulation de la dérivée partielle de E. On obtient la formule des probabilités $P_{i,k}$ de l'équation 2.10. Le détail des calculs est fourni en annexe A.

$$E = \sum_{t=1}^{T-\Delta t} \sum_{k=1}^{K} \sum_{i=1}^{N} (P_{i,k}^{m} \times d_{i,k,t}^{2}) - \sum_{t=1}^{T-\Delta t} \sum_{i=1}^{N} \lambda_{i,t} \sum_{k=1}^{K} (P_{i,k} - 1). \tag{2.9}$$

$$P_{i,k} = \frac{1}{\sum_{l=1}^{K} \left(\frac{\sum_{t=1}^{T-\Delta t} d_{i,k,t}^{2}}{\sum_{t=1}^{T-\Delta t} d_{i,l,t}^{2}} \right)^{\frac{1}{m-1}}}. \tag{2.10}$$

2.4.3 Extraction des germes

Une fois que les probabilités d'appartenance $P_{i,k}$ sont calculées pour chaque pixel i, il est possible d'obtenir une première segmentation en affectant chaque pixel de l'image 0 à l'objet le plus probable. Cependant, c'est une solution qui ne donne pas toujours des résultats cohérents spatialement. Ainsi, il est préférable d'affecter des régions plutôt que des pixels. Par ailleurs, on dispose d'informations de probabilités et donc on peut profiter de la notion de confiance pour affecter uniquement les régions sûres.

Puisque l'on désire obtenir uniquement des germes, il importe peu d'affecter toutes les régions du groupe d'images. On affecte ainsi uniquement les régions r issues d'une première segmentation spatiale (segmentation MDL [Pateux 00]) qui ont une mesure de confiance suffisamment forte (voir le troisième chapitre pour les résultats).

Soit $P(k/r)$ la probabilité d'appartenir à l'objet k pour une région r. On définit la probabilité $P(k/r)$ comme étant le produit des probabilités $p(k/x_i)$ des individus x_i appartenant à la région r :

$$\begin{aligned} P_{r,k} = P(k/r) &= Z \times \prod_{x_i \in r} P(k/x_i) \\ &= Z \times \prod_{x_i \in r} P_{i,k}, \end{aligned}$$

sachant que l'on doit respecter la contrainte suivante :

$$\forall r \sum_{k} P_{r,k} = 1.$$

On obtient pour Z la valeur suivante :

$$Z = \frac{1}{\sum_k \prod_{x_i \in r} P_{i,k}},$$

d'où :

$$P_{r,k} = \frac{\prod_{x_i \in r} P_{i,k}}{\sum_k \prod_{x_i \in r} P_{i,k}}.$$

Les probabilités $P_{r,k}$ nous permettent de définir la probabilité d'appartenance à un objet k pour une région r. On désire alors affecter la région r à l'objet k le plus probable.

Afin de valider cette affectation nous utilisons alors l'entropie $\overline{H}(r)$ (équation 2.11). En effet l'entropie $\overline{H}(r)$ est un nombre strictement positif qui mesure l'étalement des probabilités $P_{i,k}$ pour une région r. Ainsi, si la plupart des probabilités $P_{i,k}$ sont fortes en faveur de l'objet k, l'entropie $\overline{H}(r)$ sera faible, indiquant que l'on peut avoir confiance dans l'affectation de la région r à l'objet k. Inversement, si l'entropie $\overline{H}(r)$ est supérieure à un seuil fixé, alors on n'affecte pas la région. Il est à noter que si toutes les probabilités $P_{i,k}$ valent $1/K$ alors $\overline{H}(r) = log_2(K)$. On décide donc de fixer un seuil sur l'entropie ayant une valeur plus faible que $log_2(K)$. On fixe expérimentalement la valeur seuil à $0.32 \times log_2(K)$. Ainsi,

- dans un premier temps, on détermine l'objet k ayant la plus forte probabilité sur la région r :

$$\begin{aligned} k &= arg\max_k(P_{r,k}) \\ &= arg\max_k(\prod_{x_i \in r} P_{i,k}) \\ &= arg\max_k(\sum_{x_i \in r} logP_{i,k}). \end{aligned}$$

- dans un second temps, on vérifie que le choix d'affecter la région est sûr c'est-à-dire que l'entropie $\overline{H}(r)$ (équation 2.11) sur la région r est suffisamment faible. L'entropie d'un pixel est définie par $h(i) = \sum_{k=1}^{k=K} -P_{i,k} \times log_2 P_{i,k}$ d'où :

$$\overline{H}(r) = \frac{\sum_{x_i \in r} \sum_{k=1}^{k=K} -P_{i,k} \times log_2 P_{i,k}}{\sum_{i \in r} 1}. \tag{2.11}$$

2.5 Segmentation en objet par Clustering 3D

Dans cette section, nous présentons une technique pour résoudre l'équation énergétique E (équation 2.2) qui modélise la notion d'objet vidéo. Nous mettons en œuvre un algorithme de clustering 3D qui vient après la phase d'initialisation (voir schéma 2.2). Nous disposons comme données d'entrée de cet algorithme :

- des germes pour chaque objet sur tout le groupe d'images estimés lors de la phase d'initialisation.
- du mouvement estimé sur chaque germe. Ce mouvement fait office de mouvement pour l'objet sous la forme d'un maillage qui recouvre complètement le support image. Le mouvement de l'objet est donc défini même en dehors du germe objet grâce à une interpolation.

Le clustering 3D est un algorithme itératif comportant trois étapes qui consistent à calculer (détail des calculs en annexe B) :

1. la mosaïque $M_k(j)$ associée à chaque objet k en chaque pixel j de la mosaïque :

$$M_k(j) = \frac{\sum_{t=1}^{T} P^2_{\Theta_k^{t_{ref} \to t}(j),k,t} I_t(\Theta_k^{t_{ref} \to t}(j))}{\sum_{t=1}^{T} P^2_{\Theta_k^{t_{ref} \to t}(j),k,t}}. \tag{2.12}$$

2. la probabilité $P_{i,k,t}$ pour chaque pixel i, pour chaque objet k et pour chaque temps t :

$$P_{i,k,t} = \frac{\sum_{l=1}^{K} \frac{\alpha' \widehat{P}_{i,k,t} + dI^2_{i,l,t} \widehat{P}_{i,l,t}}{\alpha' + dI^2_{i,l,t}}}{\sum_{l=1}^{K} \frac{\alpha' + dI^2_{i,k,t}}{\alpha' + dI^2_{i,l,t}}}, \tag{2.13}$$

avec :

$$\left\{ \begin{array}{lcl} dI^2_{i,k,t} & = & [I_t(i) - M_k(\Theta_k^{t \rightarrow t_{ref}}(i))]^2, \\ \alpha' & = & \alpha \sum_{j \in \mathcal{V}(i)} 1 + \gamma + 2\beta \sum_{l=1}^{K} Q^2_{i,l,t}, \\ \alpha' \widehat{P}_{i,k,t} & = & \alpha \sum_{j \in \mathcal{V}(i)} P_{j,k,t} + \gamma Q_{i,k,t} \\ & & +\beta \sum_{l=1}^{K} Q^2_{i,l,t} \times \left[\begin{array}{c} P_{\Theta_l^{t \rightarrow t-1}(i),k,t-1} \\ + \\ P_{\Theta_l^{t \rightarrow t+1}(i),k,t+1} \end{array} \right]. \end{array} \right. \tag{2.14}$$

3. la probabilité $Q_{i,k,t}$ et le terme γ ont été introduits pour conserver une équation du second degré lors de la résolution de l'équation E :

$$Q_{i,k,t} = \frac{\sum_{l=1}^{K} \frac{\gamma P_{i,k,t} + \beta dP^2_{i,l,t} P_{i,l,t}}{\gamma + \beta dP^2_{i,l,t}}}{\sum_{l=1}^{K} \frac{\gamma + \beta dP^2_{i,k,t}}{\gamma + \beta dP^2_{i,l,t}}}, \tag{2.15}$$

avec :

$$dP^2_{i,k,t} = \sum_{l=1}^{K} \left[\begin{array}{c} [P_{i,l,t} - P_{\Theta_k^{t \rightarrow t-1}(i),l,t-1}]^2 \\ + \\ [P_{i,l,t} - P_{\Theta_k^{t \rightarrow t+1}(i),l,t+1}]^2 \end{array} \right].$$

Comme on l'a vu dans la section précédente, il faut disposer de germes objets sur toute la séquence ainsi que du mouvement de ces germes. Les germes permettent d'initialiser les probabilités $P_{i,k,t}$ et $Q_{i,k,t}$ de sorte que l'on ait une probabilité plus forte dans les zones où le germe est défini. On utilise donc une probabilité de 0,6 qui est supérieure aux probabilités des autres germes sans être trop forte pour que cette probabilité évolue rapidement et aisément.

$$\forall i \in \{germe\ k\ au\ temps\ t\}, P_{i,k,t} = Q_{i,k,t} = 0,6.$$

Les probabilités sont répartie de manière équiprobable entre les autres objets. Dans les zones non recouvertes, on a équiprobabilité entre tous les objets.

Cette initialisation permet de lancer les itérations du clustering 3D. En effet, il est alors possible de calculer toutes les mosaïques $M_k(j)$ (équation 2.12) puis de mettre à jour les probabilités $P_{i,k,t}$ (équation 2.13) et $Q_{i,k,t}$ (équation 2.15). On itère ainsi jusqu'à ce que les cartes des probabilités soit stabilisées.

L'algorithme parcourt, pour tous les temps t, tout les objets k. Ainsi, pour t et k donnés, on met à jour une carte de probabilités sachant la formule des $P_{i,k,t}$. Cependant, pour faire en sorte que la contrainte spatiale et la contrainte temporelle soient bien propagées, on

fait plusieurs passes sur l'image de probabilité avant de passer à l'objet suivant ou bien au temps suivant. De plus, les passes sont telles que l'on met à jour une fois sur l'autre les pixels mutuellement indépendants c'est-à-dire les pixels $i = (x, y)$ dont la valeur $x + y$ est paire et les pixels $i = (x, y)$ dont la valeur $x + y$ est impaire. On garantie ainsi la minimisation de E par l'utilisation des équations 2.12, 2.13, 2.15.

2.5.1 Introduction d'une classe de rejet

En plus des objets présents dans la séquence, on introduit la notion d'objet rejet. C'est une classe qui permet d'écarter les zones qui ne correspondent à aucun des mouvement proposés. On a donc, pour le cluster de rejet $\overline{k}$:

- pour le calcul de la mosaïque $M_{\overline{k}}(j)$ (équation 2.12) :
  - $M_{\overline{k}}(j) = 0$.

  En effet, on ne dispose pas de mouvement et de plus $\overline{k}$ n'a pas de mosaïque associée puisque ce n'est pas un objet mais une classe de rejet ;
- pour le calcul des probabilités $P_{i,\overline{k},t}$ (équation 2.14) :
  - $dI^2_{i,\overline{k},t} = constante$,
  - $\alpha' = \alpha \sum_{j \in \mathcal{V}(i)} 1 + \gamma$;
  - $\alpha' \widehat{P}_{i,\overline{k},t} = \alpha \sum_{j \in \mathcal{V}(i)} P_{j,\overline{k},t} + \gamma Q_{i,\overline{k},t}$.

  On peut constater que les contraintes sur le voisinage temporel ont disparues ce qui suppose que le rejet n'est pas forcément constant dans le temps. De plus le rejet ne possède pas de mouvement associé ;
- pour le calcul des probabilités $Q_{i,\overline{k},t}$ (équation 2.15) :
  - $Q_{i,\overline{k},t} = P_{i,\overline{k},t}$.

2.6 Résumé du chapitre

Ce chapitre introduit un modèle d'objet vidéo basé sur un mouvement long terme par objet et sur une texture mosaïque par objet. À partir de ce modèle, nous avons proposé une formulation énergétique pour le problème de segmentation. Cette formulation permet de mettre en concurrence les mouvements et les textures de chaque objet.

La formulation fait intervenir un terme d'attache aux données ainsi que 2 termes de régularisation. Le terme d'attache aux données met en jeu le mouvement et la mosaïque de chaque objet. Les termes de régularisation permettent d'avoir une stabilité spatiale et temporelle des affectations.

Nous avons alors décidé d'utiliser un algorithme de clustering affine pour obtenir dans un premier temps des germes d'objet et les mouvement de ces germes. Puis, dans un second temps, nous avons résolus cette fonctionnelle via l'utilisation d'un clustering flou 3D. Cette approche permet alors d'obtenir des probabilités d'affectation pour chaque pixel.

La suite du manuscrit présente les résultats obtenus par cette approche.

Chapitre 3

Présentation des résultats de segmentation

Cette section présente les résultats du Clustering 3D ainsi que du clustering affine utilisé comme initialisation. La première section valide l'approche Clustering 3D avec une initialisation manuelle puis semi-automatique. La deuxième section aborde le problème de la segmentation automatique par une approche en plusieurs étapes. Ces deux sections ont pour objectif de valider progressivement notre technique de segmentation.

3.1 Validation expérimental du schéma proposé

Dans cette section, nous évaluons l'approche Clustering 3D en utilisant une initialisation manuelle. Puis, nous présentons le déroulement et les résultats du clustering affine et du clustering 3D avec une initialisation semi-automatique.

3.1.1 Clustering 3D avec initialisation manuelle

Cette sous-section présente les résultats du clustering 3D sur la séquence Mobile en supprimant la phase d'initialisation, consistant normalement à extraire des germes objet et à estimer le mouvement de ces germes. On va donc utiliser des masques obtenus manuellement, et estimer le mouvement de chaque objet grâce à ces masques. L'objectif est de valider l'approche par clustering 3D. En effet, idéalement on doit obtenir comme résultats du clustering 3D les masques utilisés lors de l'initialisation.

La figure 3.1 montre les résultats obtenus. On peut constater que les masques résultats sont proches des masques originaux. Cependant, il apparaît quelques trous et quelques mauvaises affectations. On a donc décidé de modifier la distance $dI^2_{i,k,t}$ présente dans l'équation d'attache aux données 2.3. En effet, il parait judicieux de prendre en compte le bruit qu'il peut y avoir sur la texture et sur l'estimation de mouvement. On suppose que le bruit sur la texture est gaussien, centré en 0 et de variance σ^2. On suppose que le bruit sur le mouvement gaussien, centré en 0 et de variance ϵ^2. Le développement limité d'ordre 1 nous donne approximativement une gaussienne centrée en 0 et de variance

$\sigma^2 + \epsilon^2 \times \|\nabla I_t(i)\|$. On obtient alors une nouvelle distance :

$$dInew^2_{i,k,t} = \frac{dI^2_{i,k,t}}{\sigma^2 + \epsilon^2 \times \|\nabla I_t(i)\|}. \tag{3.1}$$

Expérimentalement, nous avons décidé de figer les paramètres suivants : σ^2 à 30 et ϵ^2 à 2 pour toutes les expérimentations. Le gradient $\nabla I_t(i)$ est obtenu par filtrage gaussien puis via le filtre de Prewitt [Prewitt 70]. On lance l'algorithme avec une distance de rejet valant 300 et des valeurs α, β, γ de 60. Le nombre d'itérations est de 40. On obtient le résultat de la figure 3.2[1].

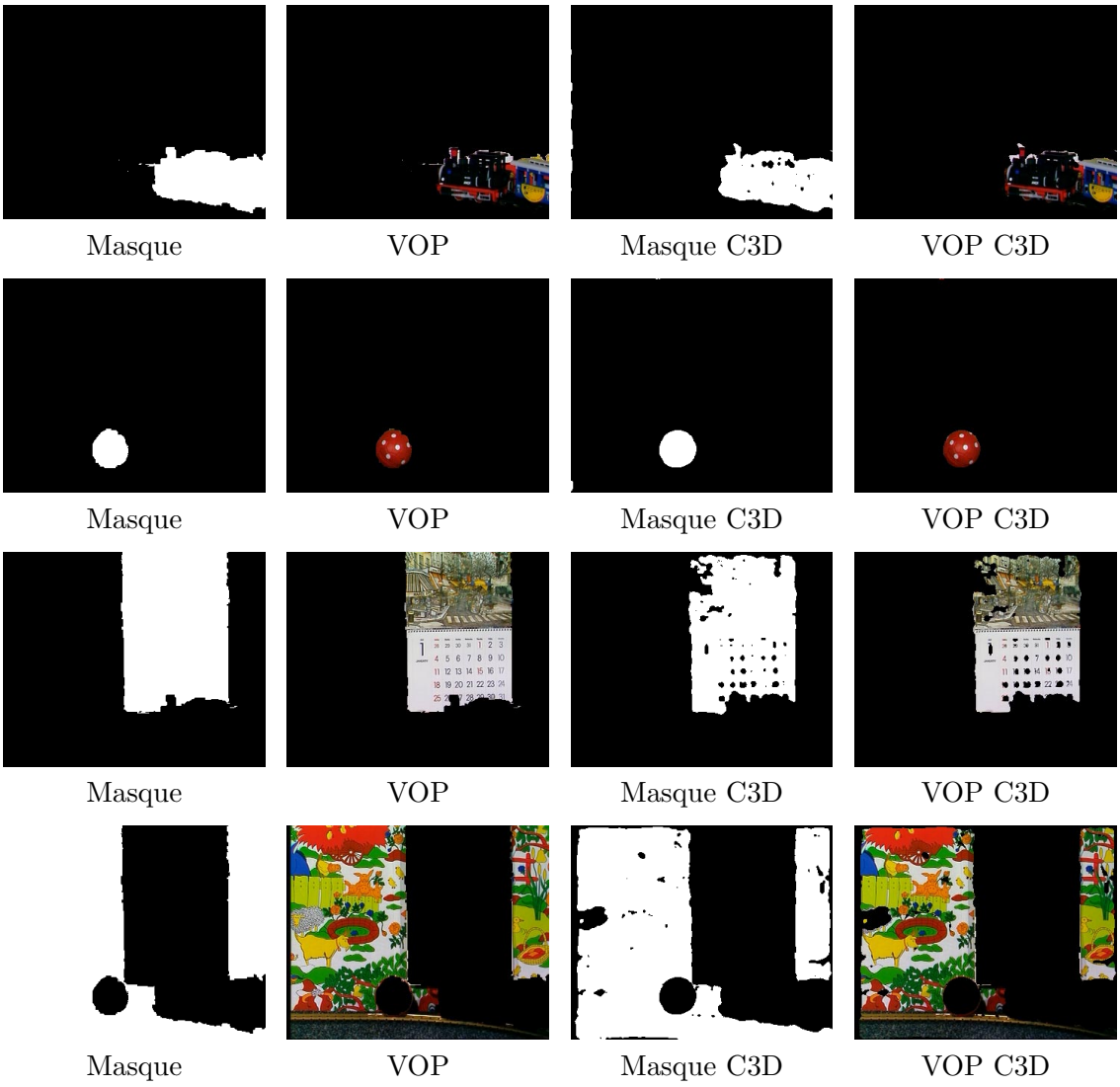


FIG. 3.1 – *Résultat C3D avec une initialisation manuel des masques*

Cette fois ci, les trous dans le calendrier ont disparu et l'on est très proche des masques initiaux. Ces premiers résultats valident l'approche et l'on va maintenant vérifier si avec

1. VOP : «video object plan»

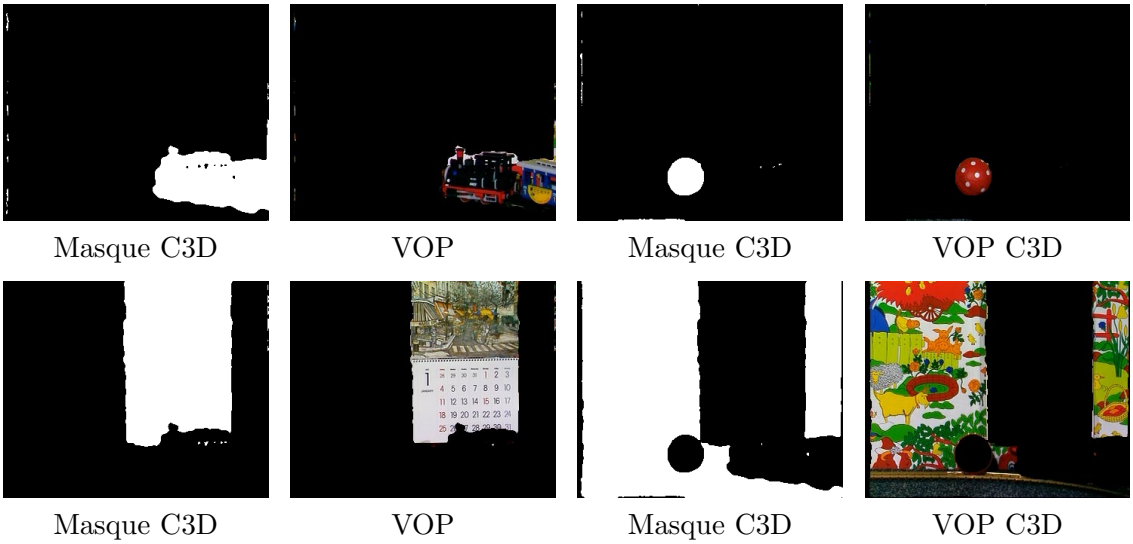


FIG. 3.2 – *Résultat C3D avec une initialisation manuelle des masques et une nouvelle distance*

une initialisation automatique on retrouve des résultats similaires. Trois exemples type de séquence sont présentés : la séquence Mobile qui présente de nombreux mouvements rigides, la séquence Foreman qui présente un mouvement non rigide et la séquence Stefan qui présente un mouvement articulé.

3.1.2 Les séquences Mobile et Foreman

Déroulement algorithmique

Cette sous-section décrit le déroulement du clustering affine puis du clustering 3D sur les séquences Mobile et Foreman. Dans un premier temps, il nous faut extraire les germes et leur mouvement. Nous utilisons l'algorithme de clustering affine flou entre l'image 10 et l'image 20 de la séquence CIF Mobile à 15 Hz et entre l'image 50 et 55 de la séquence CIF Foreman à 30Hz. Le fait de choisir un intervalle d'images suffisamment grand est important pour renforcer l'information temporelle. Par contre, plus on augmente la fenêtre de temps, plus l'estimation peut échouer dans certaines zones. Ainsi, sur la séquence Foreman, on est obligé de prendre une petite fenêtre temporelle à cause du décrochage rapide du maillage. Les choix des intervalles aont été retenus sur une base expérimentale.

On effectue donc une estimation du mouvement par maillage dynamique sur le groupe d'images. La figure 3.3 montre le maillage sur l'image 10 puis sur l'image 20 et la figure 3.13 montre le maillage sur l'image 50 puis sur l'image 55. Le maillage nous permet d'extraire les trajectoires des points de l'image 10 pour la séquence Mobile et les trajectoires des points de l'image 50 pour la séquence Foreman. On lance alors l'algorithme de clustering affine flou avec un nombre de clusters recherché de 4 pour Mobile et de 2 pour Foreman.

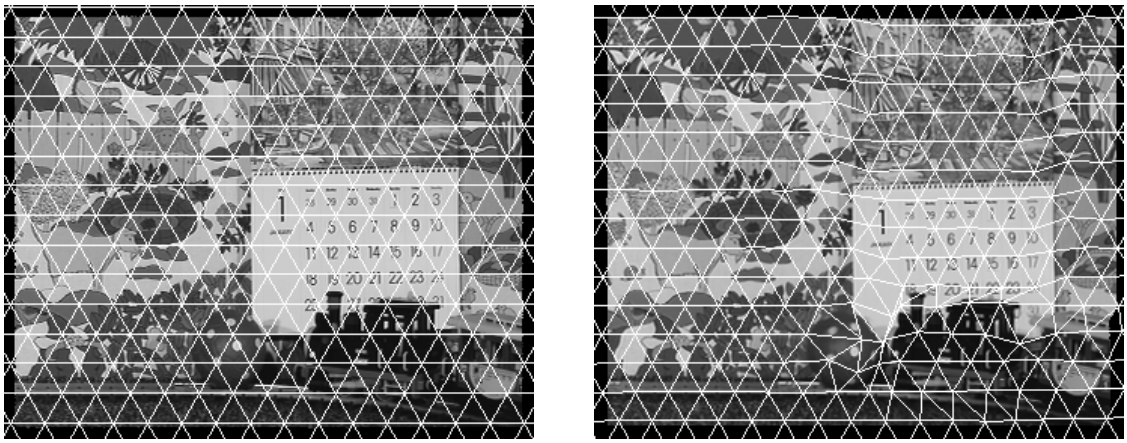
On extrait ainsi les probabilités d'appartenance de chaque pixel de l'image 10 et de l'image 50. Les images 3.4(a) et 3.14(a) montrent l'entropie de chaque pixel ($\forall i, h(i) = \sum_{k=1}^{k=K} -P_{i,k} \times log_2(P_{i,k})$). Les zones claires indiquent les zones à forte entropie c'est-à-dire les zones n'étant pas fiables ou difficilement affectables. Pour obtenir les germes

fiables, on affecte les régions dont l'entropie est faible. Les régions sont obtenues par une segmentation MDL [Pateux 00] (figures 3.4(b) et 3.14(b)), car elle est rapide et donne des régions compactes.

On obtient alors les germes des figures 3.5 et 3.15. Pour le germe du ballon (séquence Mobile), on ne conserve que la plus grande zone. En effet, les autres zones sont suffisamment faibles pour être considérées comme du bruit. Ce choix est fait très simplement par le calcul de l'aire de chaque région puis par la sélection de la région la plus grande.

Une fois les germes sur la première image obtenus, on peut alors estimer le mouvement de chaque germe. Le mouvement est interpolé en dehors du support. Les figures 3.6 et 3.16 illustrent le mouvement estimé sur chaque germe. L'estimation du mouvement doit prendre en compte le problème d'occultation. On affecte donc les zones de superposition au germe ayant la plus faible erreur de prédiction. On peut remarquer que pour la séquence Foreman, le visage subit des déformations que le maillage est capable de suivre.

On dispose donc des germes sur tout le groupe d'images ainsi que des mouvements de chaque germe. Il est alors possible de lancer l'algorithme de clustering 3D. On lance l'algorithme avec une classe rejet, dont la distance de rejet vaut 300, et avec des valeurs α, β, γ de 60. Le nombre d'itérations est de 40. Les résultats après quarante itérations sont illustrés sur les figures 3.7, 3.8, 3.9, 3.10, 3.17, 3.18, 3.19 et les mosaïques sur les figures 3.12 et 3.20.



(a) Maillage sur l'image 10 (b) Maillage sur l'image 20

FIG. 3.3 – *Illustration de l'estimation de mouvement par maillage actif*

L'analyse des résultats

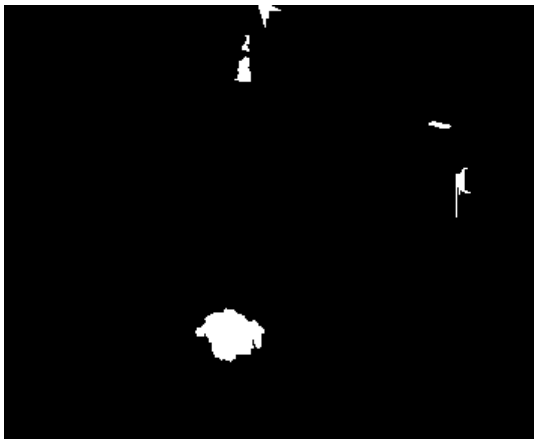
Pour la séquence Mobile, on peut remarquer que les résultats ne sont pas très éloignés des frontières de texture. Par contre, on peut constater que les résultats sont fortement dépendants de la phase d'initialisation. En effet, les masques restent proches de l'initialisation proposée. Dans l'ensemble, la localisation est acceptable bien quel ne soit pas exactement sur les frontières de texture.

Les résultats pour la séquence Foreman sont assez bons. On peut remarquer que le personnage entre l'image 60 et 70 ouvre puis ferme la bouche. Cela produit l'apparition de

(a) Entropie sur l'image 10 ré-échantillonné entre 0 et 255

(b) Segmentation MDL sur l'image 10

FIG. 3.4 – *Avant l'affectation des régions aux objets*



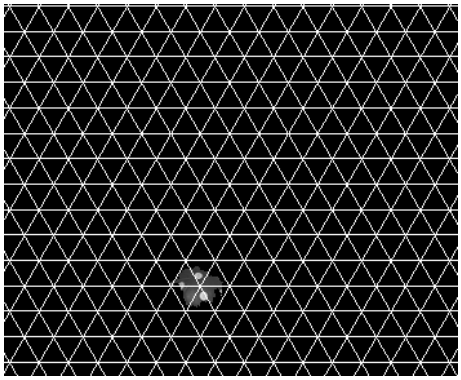
(a) Germe du ballon

(b) Germe du train

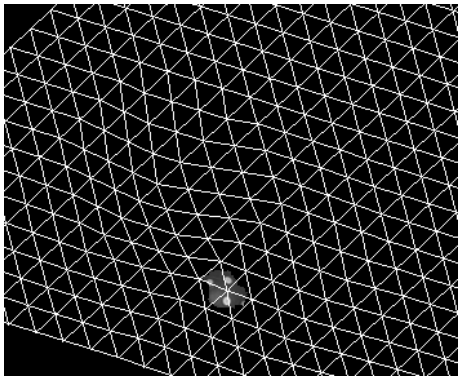
(b) Germe du fond

(b) Germe du calendrier

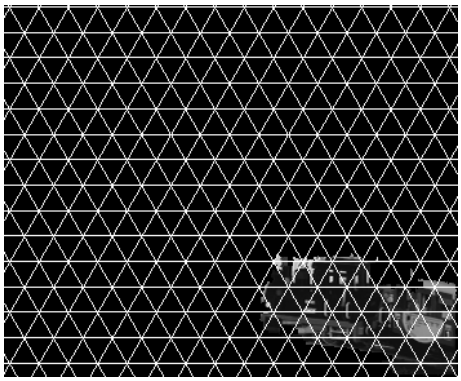
FIG. 3.5 – *Germes issus du clustering affine flou*



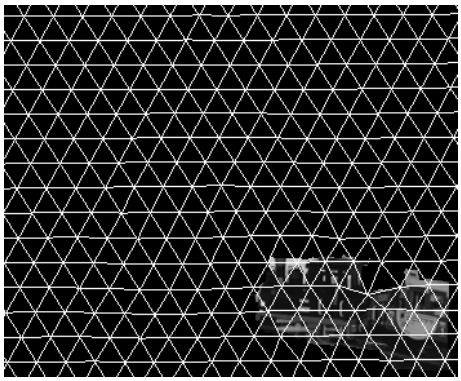
Im 10 : Maillage sur le germe ballon



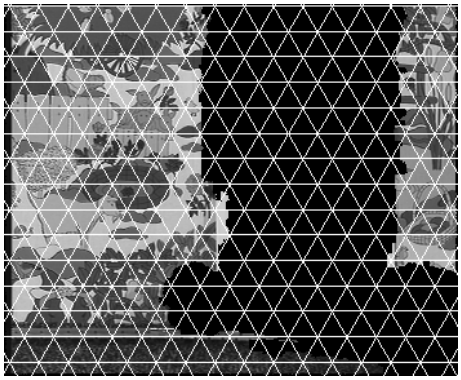
Im 20 : Maillage sur le germe ballon



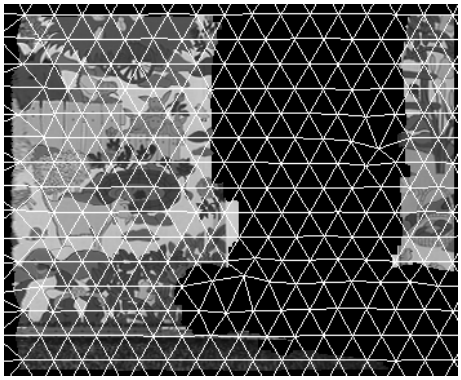
Im 10 : Maillage sur le germe train



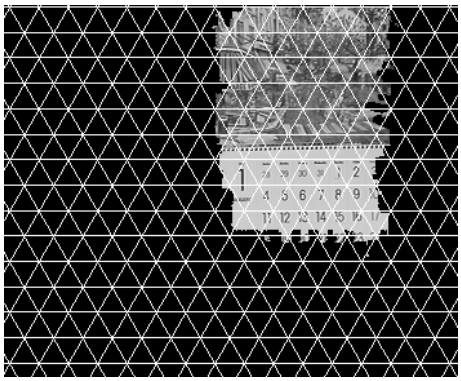
Im 20 : Maillage sur le germe train



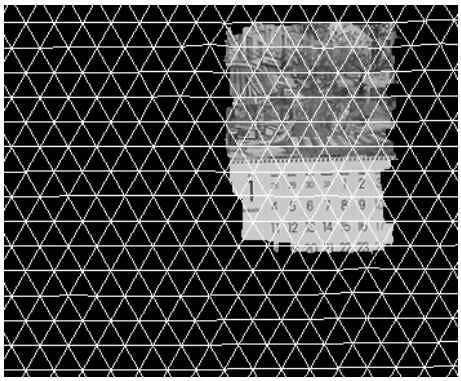
Im 10 : Maillage sur le germe fond



Im 20 : Maillage sur le germe fond



Im 10 : Maillage germe calendrier



Im 20 : Maillage germe calendrier

FIG. 3.6 – *Illustration de l'estimation de mouvement sur chaque germe*

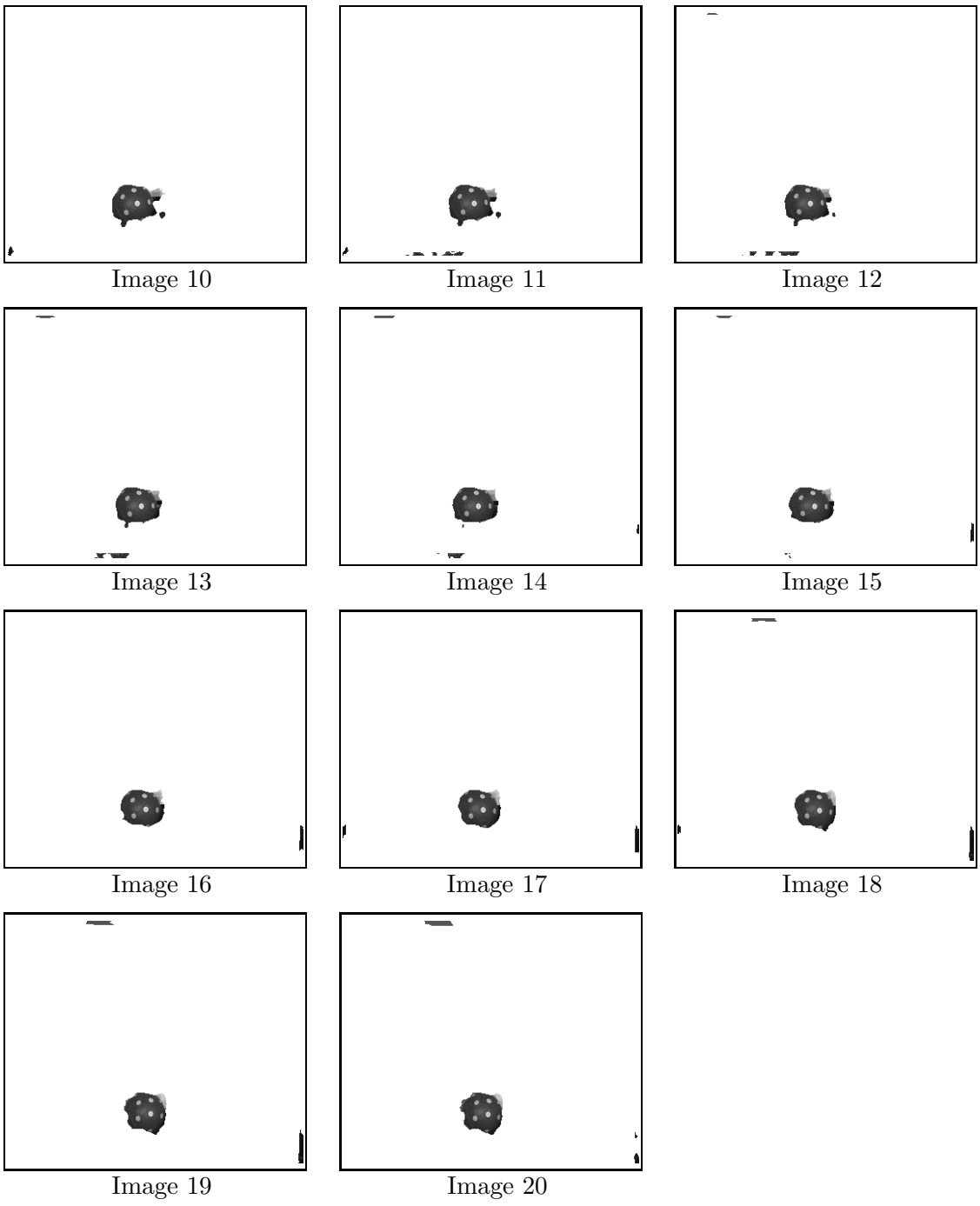


FIG. 3.7 – *Illustration des résultats du clustering 3D sur le ballon*

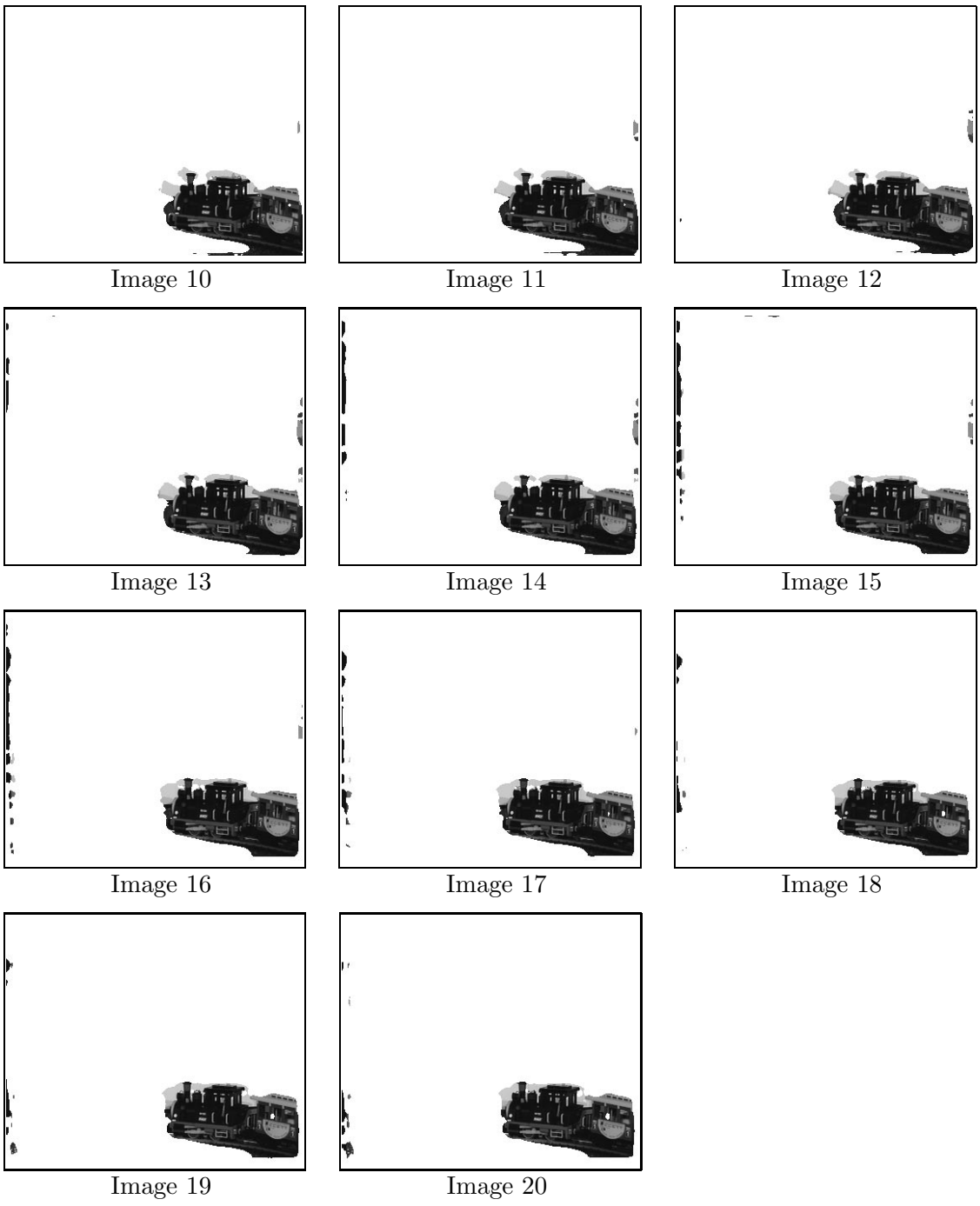


FIG. 3.8 – *Illustration des résultats du clustering 3D sur le train*

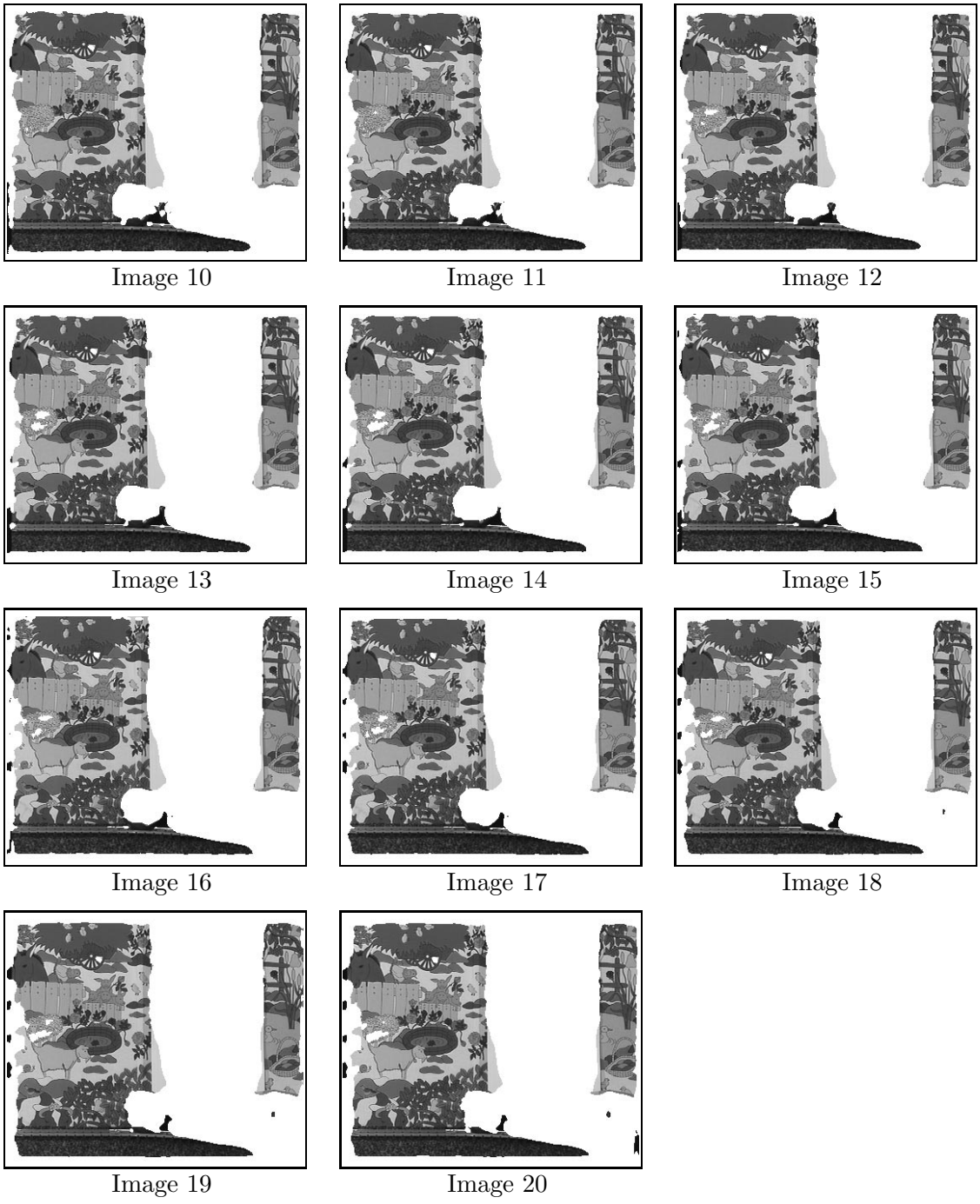


FIG. 3.9 – *Illustration des résultats du clustering 3D sur le fond*

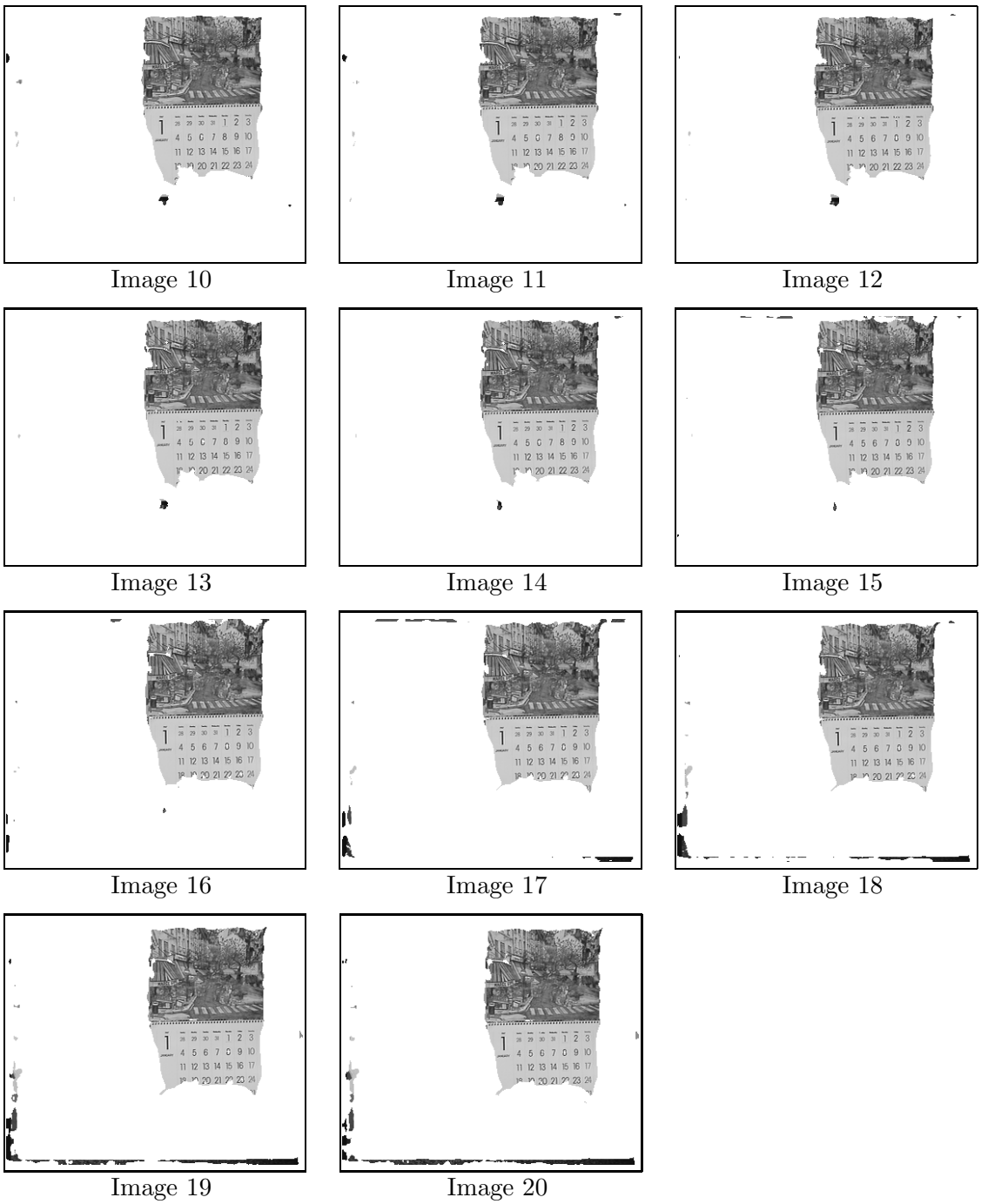


Image 10 Image 11 Image 12

Image 13 Image 14 Image 15

Image 16 Image 17 Image 18

Image 19 Image 20

FIG. 3.10 – *Illustration des résultats du clustering 3D sur le calendrier*

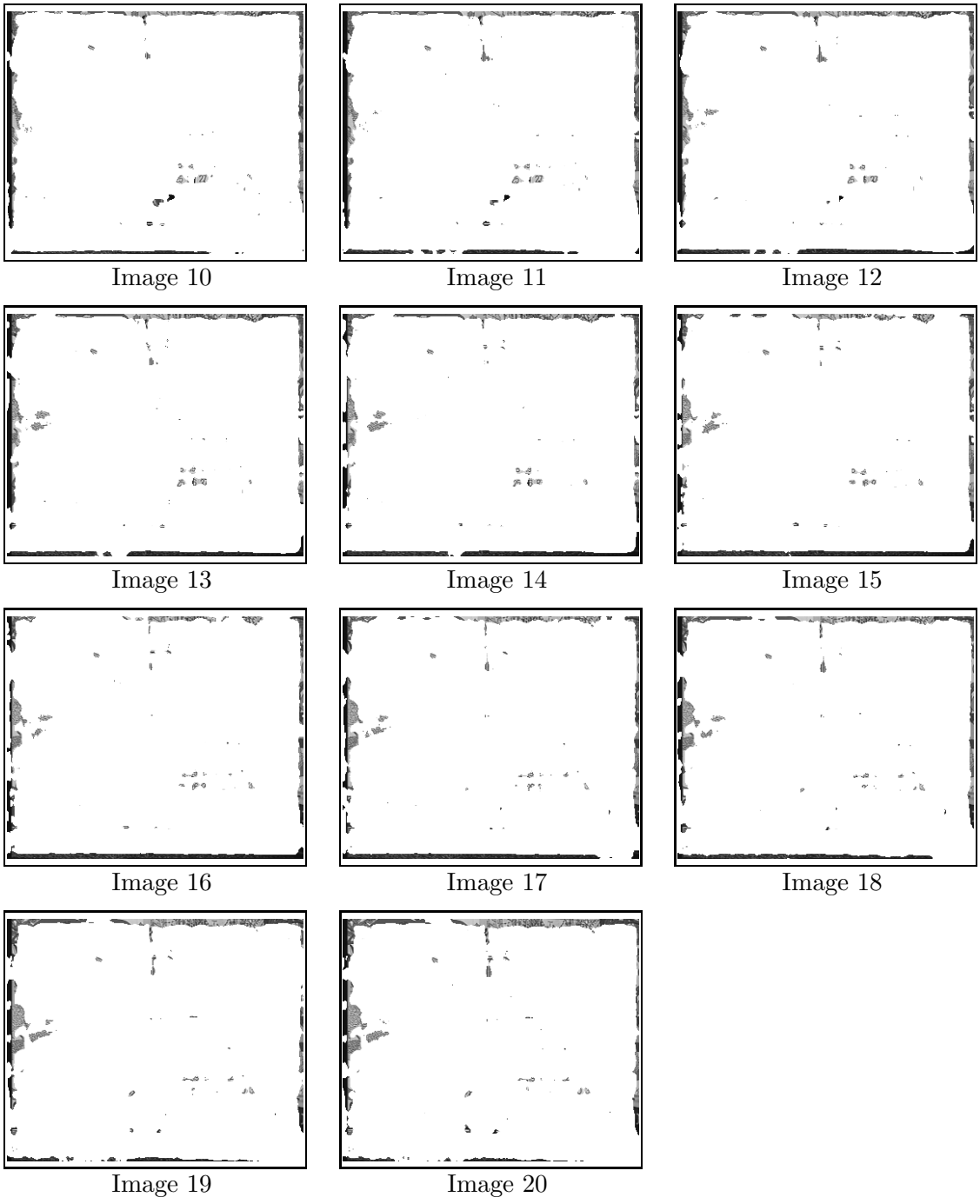
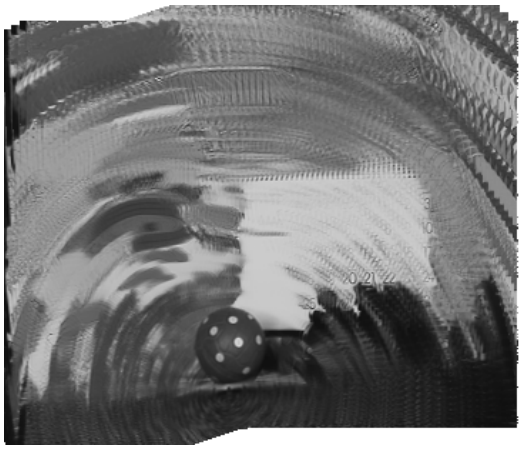


FIG. 3.11 – *Illustration des résultats du clustering 3D sur le cluster rejet*



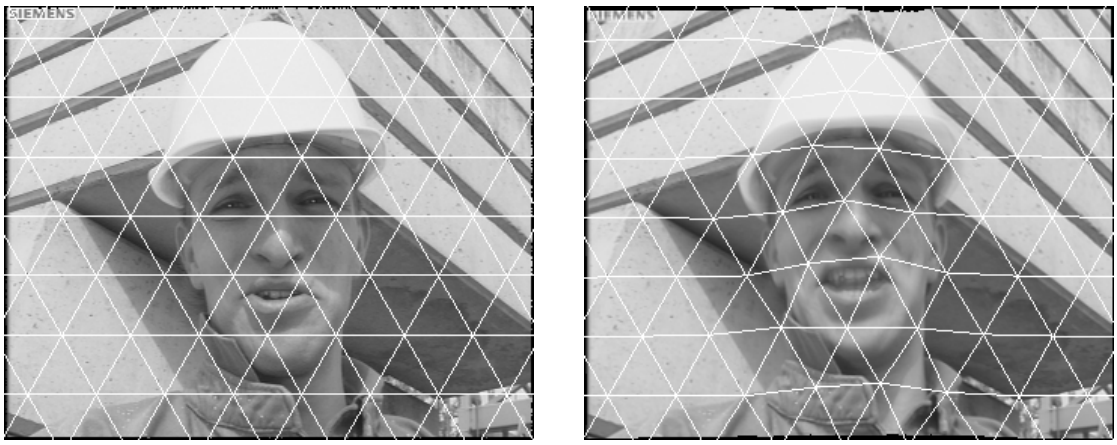
(a) mosaïque du ballon

(b) Mosaïque du train

(c) Mosaïque du fond

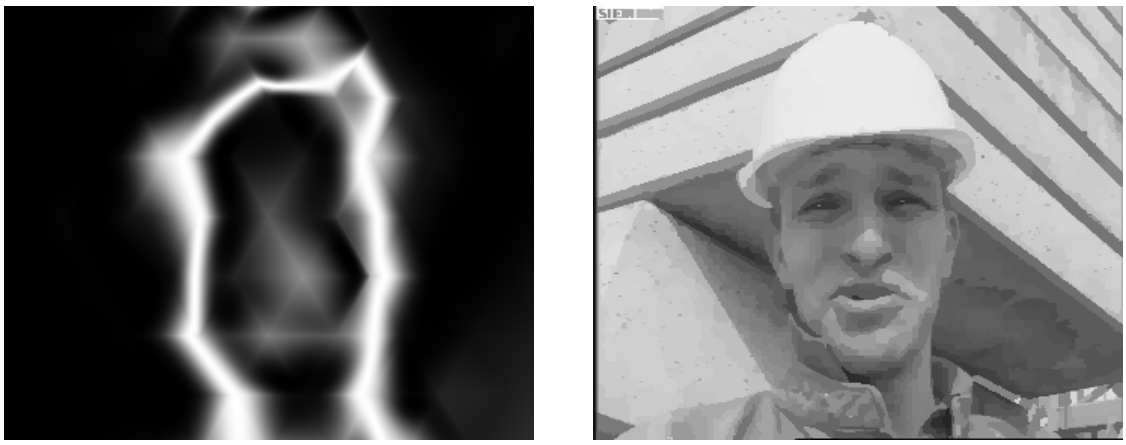
(d) Mosaïque du calendrier

FIG. 3.12 – *Illustration des mosaïques associées à chaque objet*



(a) Maillage sur l'image 50 (b) Maillage sur l'image 55

FIG. 3.13 – *Illustration de l'estimation de mouvement par maillage actif sur la séquence Foreman*



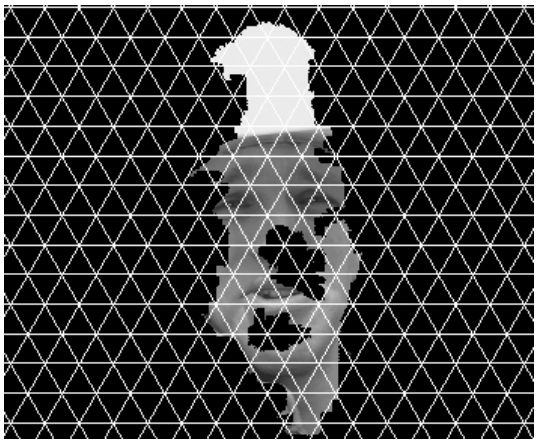
(a) Entropie sur l'image 50 ré-échantillonné entre 0 et 255 (b) Segmentation MDL sur l'image 50

FIG. 3.14 – *Avant l'affectation des régions aux objets*

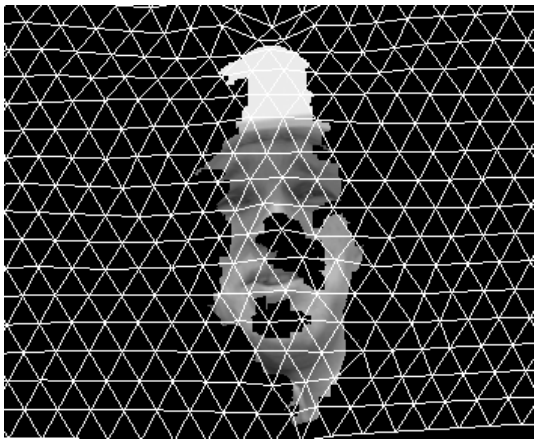
(a) Germe du visage

(b) Germe du fond

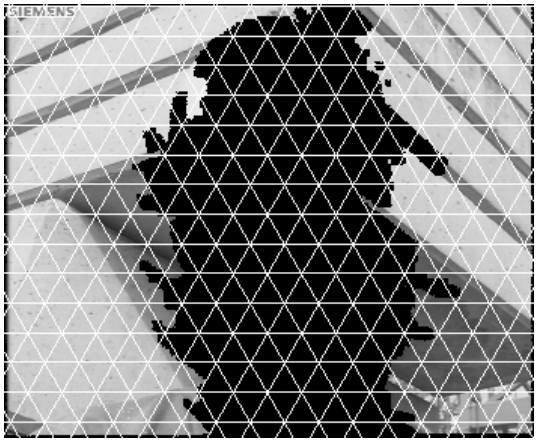
FIG. 3.15 – *Germes issus du clustering affine flou pour la séquence Foreman*



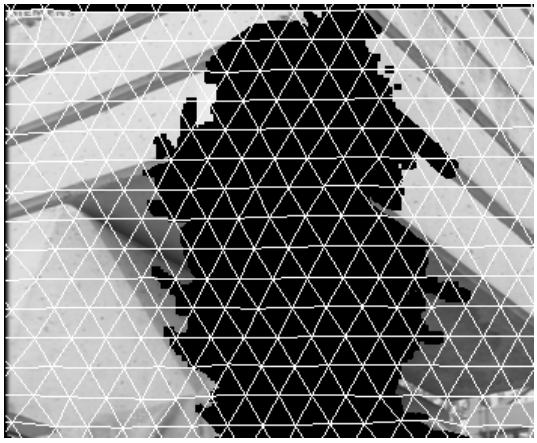
(a) Image 50 : Maillage sur le germe visage



(b) Image 60 : Maillage sur le germe visage



(c) Image 50 : Maillage sur le germe fond



(d) Image 60 : Maillage sur le germe fond

FIG. 3.16 – *Illustration de l'estimation de mouvement sur chaque germe*

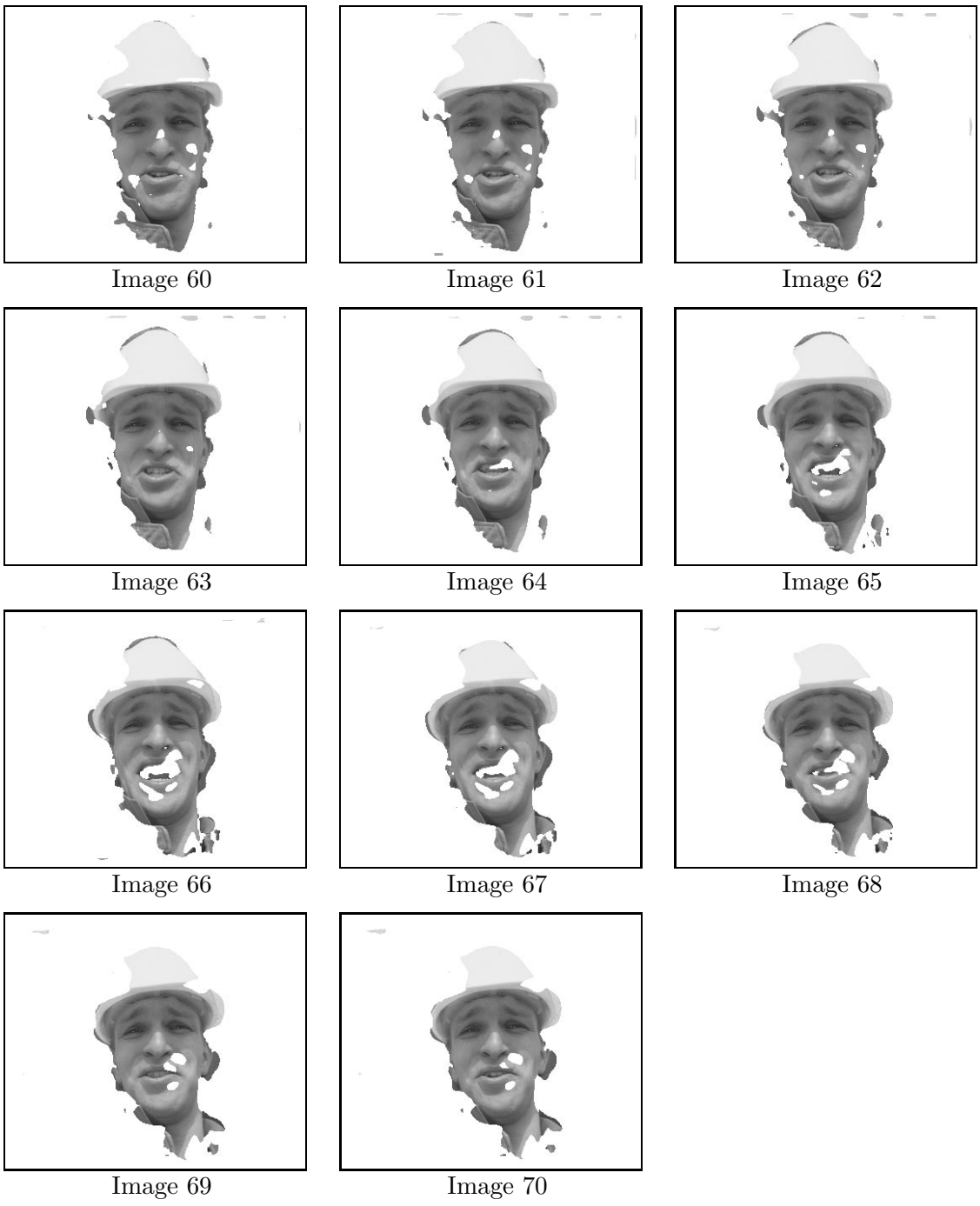


FIG. 3.17 – *Illustration des résultats du clustering 3D sur le visage*

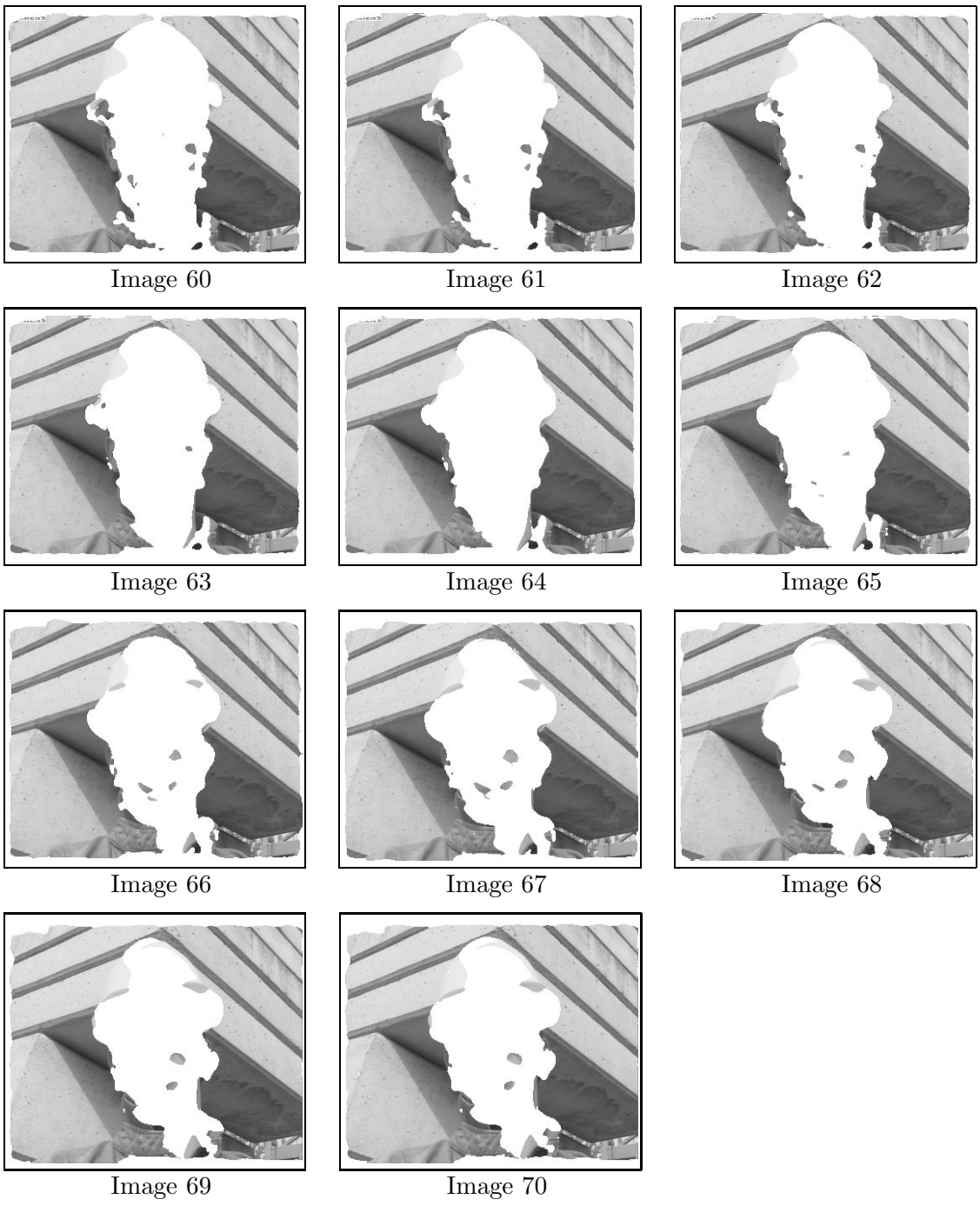


Image 60 Image 61 Image 62

Image 63 Image 64 Image 65

Image 66 Image 67 Image 68

Image 69 Image 70

FIG. 3.18 – *Illustration des résultats du clustering 3D sur le fond*

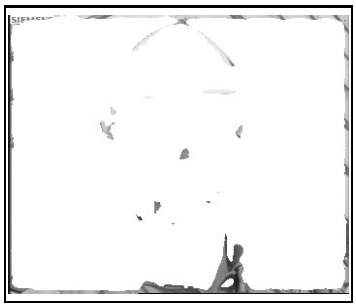


Image 50

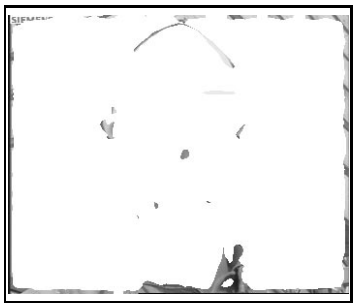


Image 51


Image 52

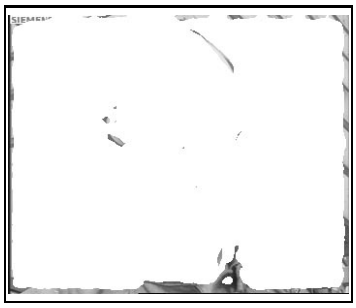


Image 53


Image 54


Image 55

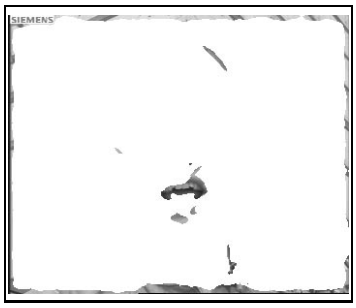


Image 56


Image 57

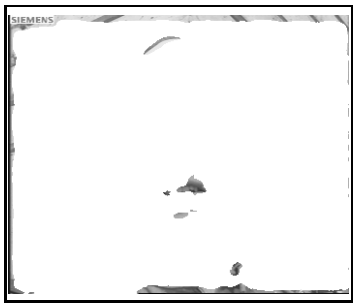


Image 58

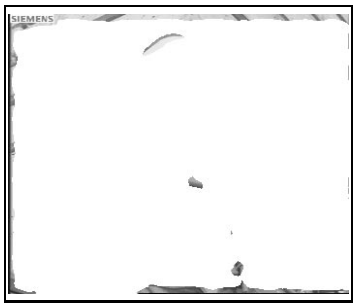


Image 59

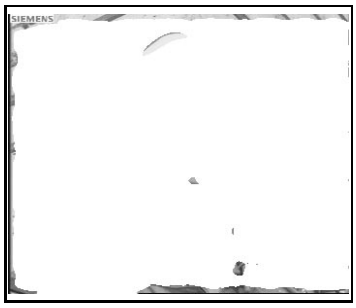


Image 70

FIG. 3.19 – *Illustration des résultats du clustering 3D sur le cluster rejet sur la séquence Foreman*

(a) mosaïque du visage (b) Mosaïque du fond

FIG. 3.20 – *Illustration des mosaïques associées à chaque objet issu du clustering 3D*

texture qui est placée dans la classe rejet. Ce phénomène est correct puisque l'apparition de texture ne fait pas partie du modèle d'objet à texture fixe que nous recherchons. Une deuxième remarque concerne le haut du casque qui est mal affecté. Ce genre de problème est particulièrement difficile à résoudre. En effet, la texture est uniforme et sensiblement la même pour la mosaïque du fond ou celle du visage. Ainsi la mise en concurrence rend le choix d'affectation particulièrement difficile puisque très instable.

3.1.3 La séquence Stefan

Cette sous-section décrit le déroulement de clustering affine puis du clustering 3D sur la séquence Stefan. Cette exemple illustre l'intérêt de la classe rejet. Ici nous réalisons un clustering 3D avec une seule classe de mouvement. Le mouvement est celui de l'objet en arrière plan. Nous utilisons des germes extraits manuellement. Nous voulons montrer que nous pouvons ré-extraire assez bien l'objet fond ainsi que rejeter ce qui n'est pas l'objet fond.

Nous estimons donc le mouvement sachant que l'on dispose de tous les masques. L'estimation est effectuée sur l'intervalle [225 , 250] sur la séquence Stefan 352x240. On dispose donc des germes sur tout le groupe d'images ainsi que du mouvement du fond. Les résultats du Clustering 3D sont donnés sur les figures 3.21, 3.22 et 3.23 après quarante itérations.

La zone en arrière-plan est assez bien retrouvée, par contre la zone avant-plan n'est pas très bien segmentée. Ceci peut s'expliquer par la persistance temporelle que nous avons imposée via la contrainte de régularisation. On obtient alors un effet de « trainée » dans les affectations à l'objet avant plan.

3.1.4 Réflexion sur les résultats du clustering 3D

Le clustering 3D donne des résultats proches du modèle objet que nous recherchons c'est-à-dire un mouvement fin long terme et une texture par objet. Deux points sont particulièrement intéressants : l'utilisation du long terme et l'utilisation d'une classe rejet.

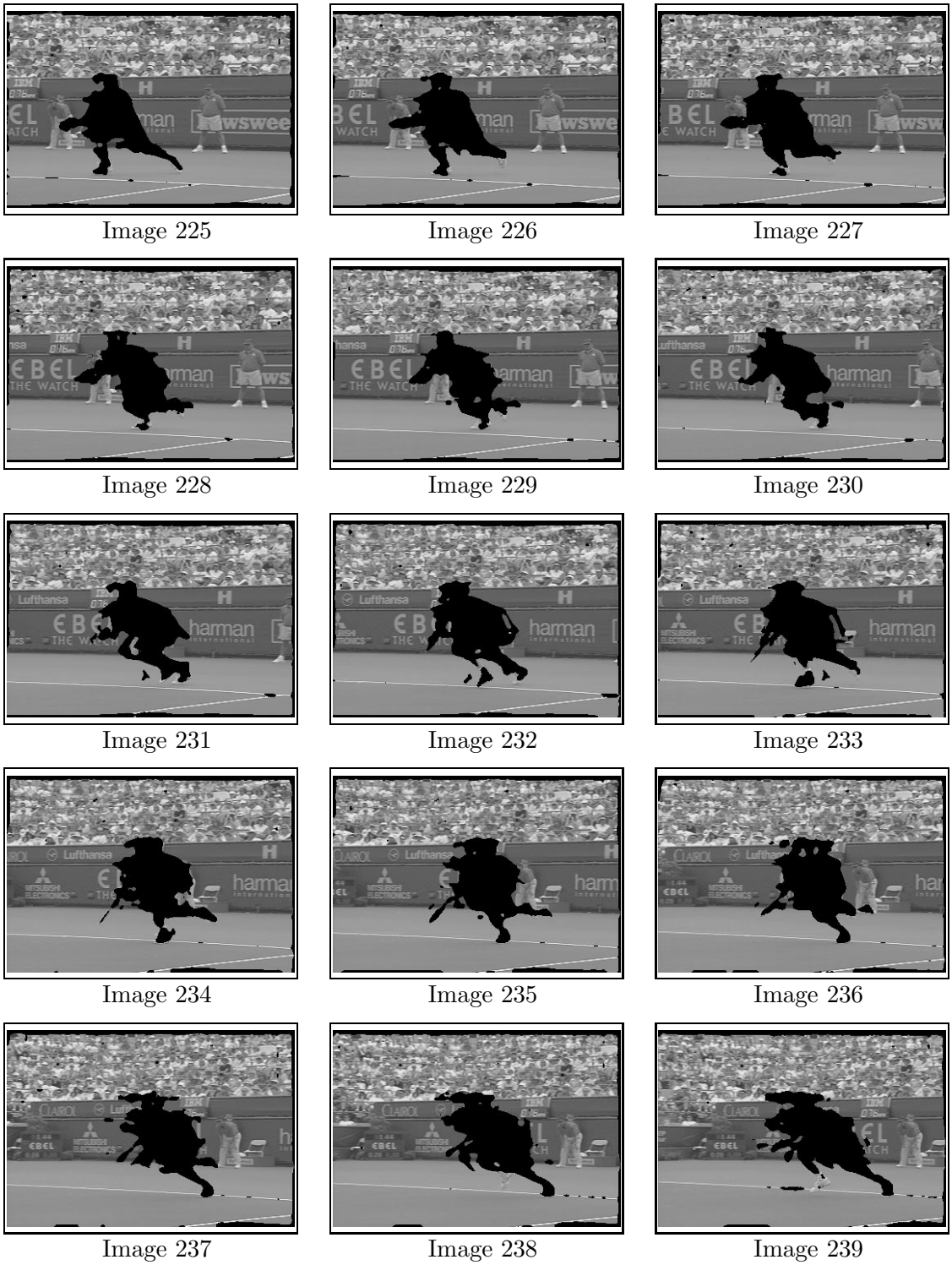


FIG. 3.21 – *Illustration des résultats du clustering 3D sur le fond*

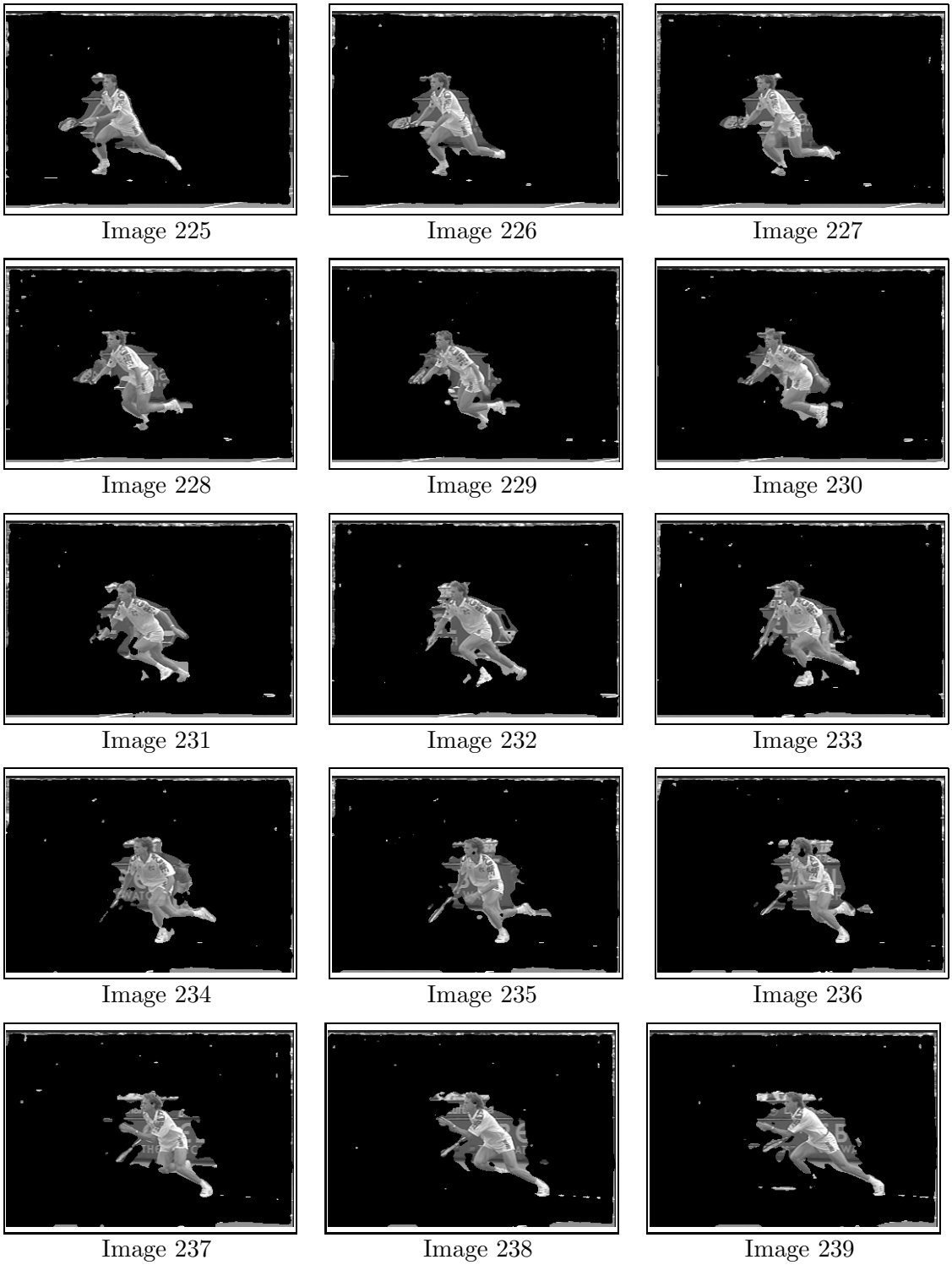


FIG. 3.22 – *Illustration des résultats du clustering 3D sur le cluster rejet*

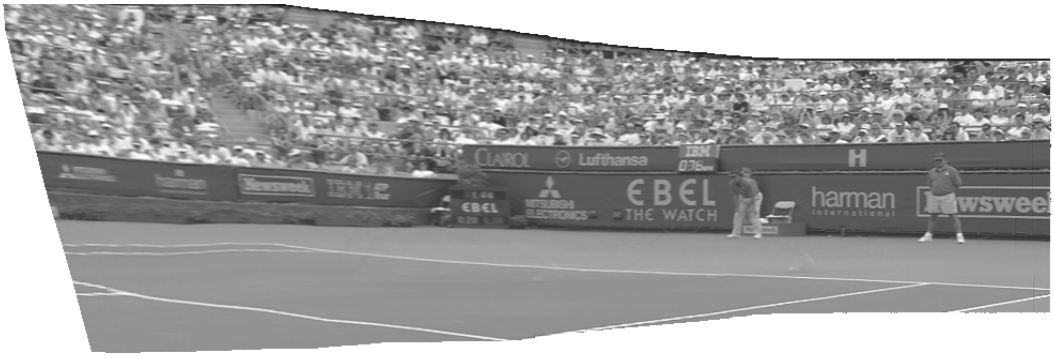


FIG. 3.23 – *Illustration de la mosaïque du fond issus du clustering 3D de la séquence Stefan*

Les résultats du long terme permettent plus facilement d'affecter les zones occultées. C'est le cas par exemple sur la séquence Stefan où l'on obtient la mosaïque du fond. La classe rejet quant à elle permet de rejeter les régions ne correspondant à aucun modèle objet.

Les résultats montrent aussi la difficulté d'initialiser le clustering 3D. La phase d'initialisation nécessite en effet d'obtenir le bon nombre de germes ainsi que des germes pas trop éloignés des frontières d'objets. Comme on a pu le voir sur la séquence Foreman, une initialisation grossière est cependant suffisante pour obtenir une segmentation finale correcte.

Puisque l'approche par clustering 3D requiert une initialisation proche des objets et qu'un certain nombre d'algorithmes résolvent le problème de segmentation en objets vidéo dans des cas particuliers (c'est-à-dire avec a priori), on peut envisager la phase d'initialisation comme une succession d'étapes d'analyse avec a priori. Plus généralement, la section suivante introduit la notion de séquentialité et de découpage pour résoudre le problème de la segmentation automatique en objets vidéo. La séquentialité et le découpage permettent de résoudre le problème par exécution successive de briques algorithmiques ayant chacune un objectif particulier. Intuitivement, l'agencement de ces briques a une complexité croissante. Ainsi, l'étape de clustering 3D n'arrive que tardivement.

3.2 Analyse d'une séquence vidéo : une approche en plusieurs étapes

La segmentation en objets vidéo sans a priori est un problème difficile. Selon la définition donnée dans la section 2.2 un objet vidéo est une zone de mouvement qui se distingue par rapport au reste de l'image. Il existe un certain nombre d'algorithmes qui résolvent le problème de segmentation en objets vidéo dans des cas particuliers. On peut donc envisager la segmentation en objets vidéos sans a priori comme une succession d'étapes d'analyse avec a priori.

Actuellement, il est possible d'obtenir un certain nombre de familles d'objets de manière automatique. Nous retenons donc les familles suivantes :

– l'objet de type **logo** caractérisé par une petite surface et pas de mouvement,

– l'objet de type **zone fixe** caractérisé par une grande surface et pas de mouvement,

– l'objet de type **arrière-plan** ayant un mouvement globale,
– les objets de type **avant plan**.

La catégorie objets en avant plan peut être ensuite redécoupée en objets plus sémantiques comme un visage ou bien un buste.

La segmentation en objets vidéo peut alors être vue comme une succession de briques algorithmiques que nous allons présenter. Ces briques sont les suivantes :

1. détection de la présence de plus d'un objet,
2. extraction des objets logo et zones fixes s'il y en a,
3. extraction du mouvement global s'il y en a,
4. extraction des objets en avant-plan s'il y en a.

Pour illustrer le type de résultats que l'on peut obtenir, nous avons mené des tests sur le premier GOP des séquences : Bus (CIF 15Hz), Foreman (CIF 15Hz), Mobile (CIF 15Hz), Tempête (CIF 15Hz), Lion (CIF), Stefan (352x240), Garden (352x240), Smill (256x256), Rond-point (256x224).

3.2.1 La première brique : détermination du nombre d'objet

La première brique consiste à déterminer s'il existe plus d'un objet. Dans le cadre de notre vision d'un objet vidéo, il faut simplement savoir s'il y a différents mouvements dans la vidéo, c'est-à-dire s'il y a des zones de recouvrement et de découvrement dans la séquence. L'outil de maillage est intéressant pour déterminer ceci puisqu'il est facile de détecter des étirements ou des écrasements de mailles. En prenant des tailles de mailles inférieures à 100 pixels d'aire pour une séquence CIF, on stoppe l'estimation du mouvement lorsque l'on détecte au moins un triangle dont l'aire est de 95% inférieur à sa taille initiale. Ce système permet de détecter les perturbations dans l'estimation du mouvement par maillage dû à la présence de zones de découvrements et de recouvrements. Ces perturbations indiquent la présence de plus d'un objet. Ainsi, si le GOP obtenu est petit (de l'ordre d'une dizaine d'images sur une séquence à 15Hz), on suppose qu'il y a plusieurs objets.

La première brique nous permet d'obtenir l'information suivante :

– un seul objet (et la taille du GOP),
– plus d'un objet (et la taille du GOP).

Nous trouvons pour les séquences des tailles de GOP indiquant qu'il y a probablement plusieurs objets :

séquence	taille du GOP
Bus	[0-2]
Foreman	[0-3]
Mobile	[0-4]
Tempête	[0-9]
Lion	[0-6]
Stefan	[0-2]
Garden	[0-3]
Smill	[0-4]
Rond-point	[0-8]

3.2.2 La deuxième brique : extraction des logos et zones fixes

La deuxième brique consiste à détecter tous les objets sans mouvement, c'est-à-dire les logos ainsi qu'une partie des zones fixes. Le résultat de l'estimation de mouvement de la première brique permet aisément de trouver les zones fixes. En effet, le mouvement dense permet par seuillage d'obtenir les pixels fixes. L'utilisation d'une segmentation spatiale permet alors de conserver les régions fixes. On utilise pour passer de l'affectation pixel à l'affectation région la notion d'entropie par région (équation 2.11). On obtient alors les informations suivantes :

- un objet est classifié comme zone fixe si son aire est supérieur à 50% de l'aire de l'image ou si sa fenêtre englobante est de la même taille que l'image,
- un objet est classifié comme logo si ce n'est pas une zone fixe (l'aire est inférieur à 50% de la taille de l'image et la fenêtre englobante est de taille inférieure à la taille de l'image),
- pas de zone fixe ni de logo.

Remarquons que notre algorithme n'est pas capable de détecter un logo fixe sur un fond fixe ce qui n'est pas très important dans le cadre du codage objet puisque les deux régions ne se distinguent pas en terme de mouvement.

Les figures 3.24 montrent les VOP des logos pour les séquence bus et Foreman. Les figures 3.25 montrent les VOP des zones fixes pour les séquences Smill et Lion. Pour les autres séquences, aucun masque logo ni zone fixe n'est détecté. En effet, aucune région n'a été détectée comme étant fixe.

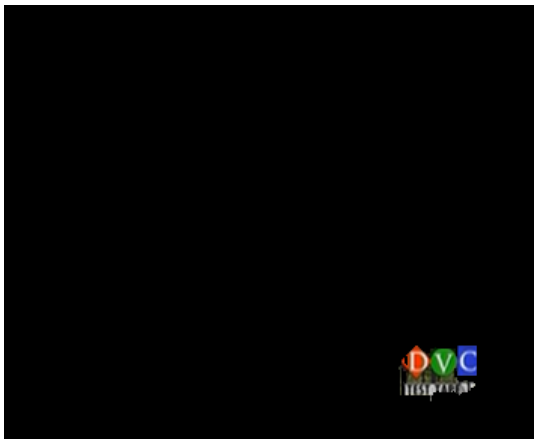
3.2.3 La troisième brique : extraction d'un mouvement global

La troisième brique consiste à extraire un mouvement global affine. Si l'extraction réussit alors on dispose d'un objet qui peut probablement être classé dans la catégorie fond. L'extraction peut échouer et indique que la séquence est plus complexe. L'algorithme que nous avons mis en place prend en compte les objets logos s'il y en a. S'il y a des zones fixes, on ne lance pas l'algorithme. En effet, on a obtenu des zones fixes qui couvrent plus de 50% de l'image. Il est donc fort probable que la zone fixe soit de type fond et qu'il n'y ait pas de mouvement global.

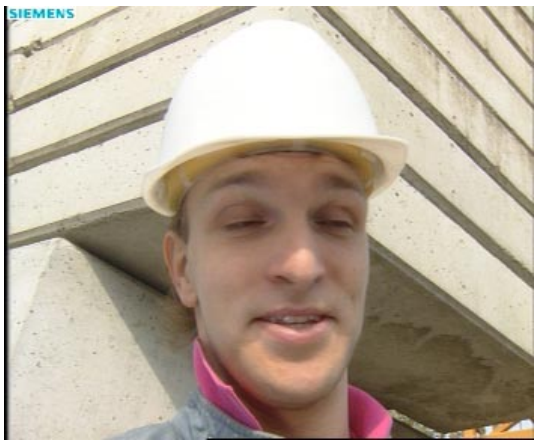
L'algorithme consiste à calculer le mouvement affine robuste sur le GOP en utilisant le clustering affine flou avec rejet présenté dans le chapitre précédent. On lance simplement l'algorithme sur un GOP avec une classe mouvement et une classe rejet. L'algorithme détermine alors les régions fiables selon un mouvement affine. On relance l'estimation du mouvement sur les régions fiables. L'estimation du mouvement est effectuée concurrentiellement sur les régions fiables et non fiables. Ainsi si le mouvement est effectivement le mouvement global d'un objet, on devrait pouvoir l'estimer sur une taille de GOP plus grande que précédemment. On fait alors l'hypothèse qu'il y a un mouvement global si le GOP est plus grand que précédemment. On obtient au final le masque de mouvement global en relançant un clustering affine.

Pour les séquences Foreman, Tempête, Stefan, Garden on obtient alors des masques de mouvement global (figure 3.26). Pour la séquence Bus, aucun mouvement global n'est détecté. Par contre, pour les séquences Mobile et Rond-point l'algorithme donne des résultats erronés. En effet, sur la séquence Mobile une partie de la tapisserie et du ca-

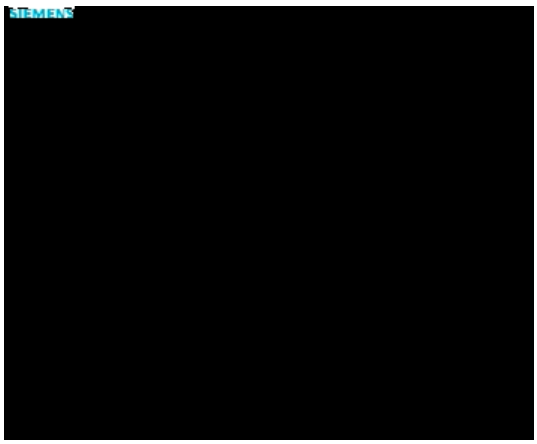
(a) Image 0 séquence bus



(b) vop logo séquence bus

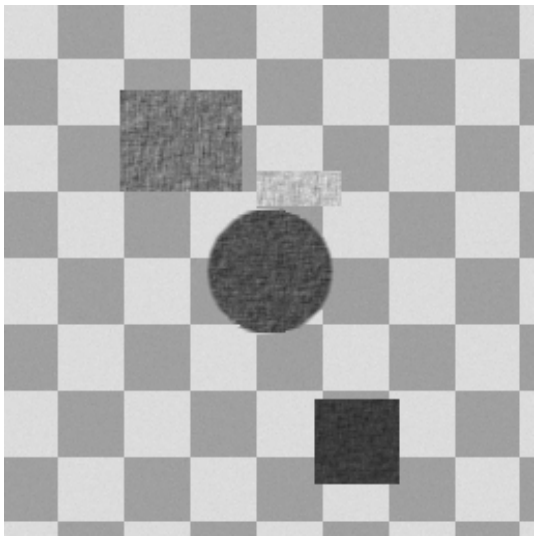


(c) Image 0 séquence Foreman

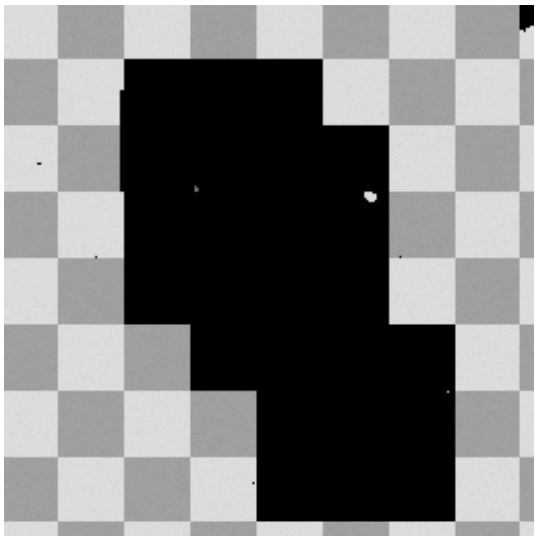


(d) vop logo séquence Foreman

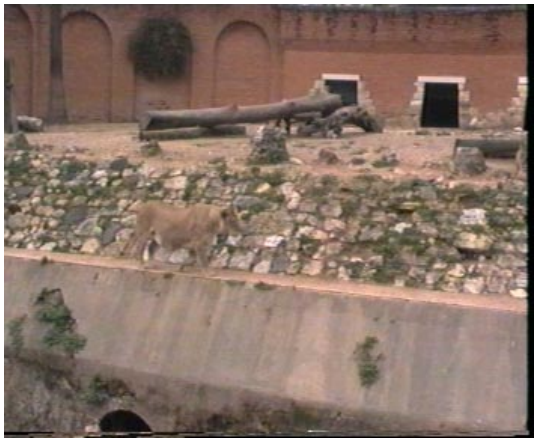
FIG. 3.24 – *Illustration de l'extraction d'objet logo*



(a) Image 0 séquence Smill



(b) vop zone fixe séquence Smill



(c) Image 0 séquence Lion

(d) vop zone fixe séquence Lion

FIG. 3.25 – *Illustration de l'extraction d'objet zone fixe*

lendrier sont considérés comme ayant un mouvement global et sur la séquence rond-point, seul le ciel est considéré comme mouvement global.

3.2.4 La quatrième brique : des segmentations plus complexes

La quatrième brique consiste à extraire des objets qui ne sont ni des logos, ni des zones fixes, ni des mouvements globaux. En fonction des objets précédemment trouvés ou non trouvés, on connaît les zones de textures restant à analyser ainsi que les types d'objets que l'on n'a plus à chercher. Il y a plusieurs cas pouvant ce présenter :

- il n'y a plus d'autres objets : c'est le cas pour la séquence Tempête. Le masque ou les masques couvrent plus de 90% de l'image. On suppose qu'il n'y a qu'un seul objet dans cette séquence.
- il y avait un objet mouvement global : c'est le cas pour les séquences Stefan, Foreman et Garden. Il faut mettre en œuvre des techniques d'extraction d'objets en avant plan sachant le mouvement global. On peut par exemple utiliser le clustering 3D avec une classe rejet.
- il y avait une zone fixe seulement : c'est le cas pour la séquence Lion et Smill. Il faut mettre en œuvre un algorithme de détection des objets avant plan sachant le fond fixe. Encore une fois, on peut utiliser le clustering 3D avec une classe rejet.
- il y avait un logo seulement : c'est le cas pour la séquence bus. C'est le type de séquence le plus dur à traiter. En effet, il n'y a pas de mouvement global mais plutôt un ensemble de mouvements. Ici, on a une séquence dont l'estimation du mouvement est particulièrement perturbée. Dans ce type de séquence, on peut alors mettre en place des approches plus complexes : recherche par segmentation en tube, par clustering 3D etc...

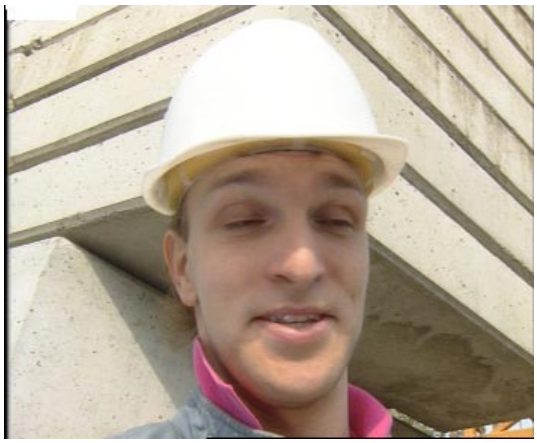
3.2.5 Réflexion sur les résultats de la segmentation en plusieurs étapes

La segmentation toute automatique reste un problème ouvert. On peut cependant obtenir de meilleurs résultats en traitant le problème de manière séquentielle. Ainsi, une suite de briques logicielles peut simplifier l'opération de segmentation. L'idée est donc de décomposer le problème sans a priori en plusieurs petits problèmes avec a priori.

Les résultats que nous proposons illustrent assez bien ce à quoi on peut s'attendre en terme de segmentation. Les objets sont grossièrement localisés sans être exactement sur les frontières de texture. Cependant, on peut remarquer qu'il est possible dans les cas simples d'obtenir une segmentation automatique.

(a) Image 0 séquence Stefan

(b) vop mouvement global



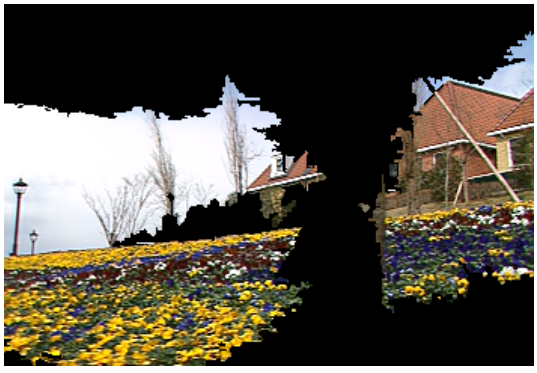
(c) Texture sans logo séquence Foreman

(d) vop mouvement global

(e) Image 0 séquence Tempête

(f) vop mouvement global

(g) Image 0 séquence Garden



(h) vop mouvement global

FIG. 3.26 – *Illustration de l'extraction des régions de mouvement global*

Chapitre 4

Conclusion de la première partie

4.1 Conclusion

Dans cette première partie, nous avons formalisé la notion d'objet vidéo via l'équation 2.2. La définition d'objet vidéo que nous donnons distingue les objets grâce à leur mouvement. Cette définition est intéressante pour le codage vidéo puisque l'on recherche une homogénéité des mouvements, ce qui permet une bonne prédiction des textures, et un nombre limité d'objet cohérents, ce qui réduit le coût de codage global des contours. De plus, nous raisonnons sur des groupes d'images ce qui permet d'avoir une bonne stabilité temporelle et donc de réduire le coût des prédictions lors du codage.

Pour résoudre la formulation énergétique issue de notre modèle d'objet vidéo, nous avons proposé un algorithme de clustering 3D. Ainsi, notre approche prend en compte la notion de mouvement fin long terme. Les problèmes d'occultation sont alors mieux réglés et la stabilité des résultats est augmentée.

Les résultats obtenus sont assez proches de ce que l'on recherche, cependant, la phase d'initialisation reste sensible. On a donc proposé de diviser le problème général de la segmentation sans a priori en sous-problèmes de segmentation avec a priori. Les résultats montrent qu'il est alors possible dans les cas simples d'obtenir des objets de manière quasi-automatique.

4.2 Perspectives

Les résultats obtenus de manière automatique sont un peu trop éloignés des frontières de texture nécessaires pour un codage objet. En effet, à très bas débit, l'effet de débordement de texture d'un objet sur d'autres objets est visible. Ainsi, un certain nombre d'améliorations pourraient être envisagées.

Comme on l'a vu, les zones de textures uniformes sont difficiles à affecter. On peut prendre l'exemple du bas du calendrier dans la zone où le train passe. Certaines parties du bas du calendrier sont mal affectées. Il serait peut être intéressant d'ajouter des contraintes d'homogénéité de textures dans les zones frontières.

Autre chose, l'estimation du mouvement pourrait être améliorée dans les zones d'occultations. En effet, il reste des problèmes trop prononcés d'étirement et d'écrasement dans les zones de découvrement et de recouvrement. Ceci influence de manière trop forte les

zones d'objets limitrophes.

Enfin, une approche multi-résolution et hiérarchique permettrait un raffinement itératif des résultats et permettrait aussi d'accélérer la vitesse de convergence.

deuxième partie

Le codage d'objets vidéos

Chapitre 1

État de l'art : les approches de codage par régions et par objets

Ce chapitre traite du codage par régions qui est apparu au début des années 1980 et du codage par objets qui a suivi une dizaine d'années plus tard. On peut d'ailleurs voir l'approche objets comme une suite logique à l'approche régions dans le sens où les problèmes liés à la représentation par régions sont partiellement résolus : le problème de recouvrement et de découvrement est substitué à un problème de détermination d'ordre de profondeur, l'instabilité temporelle de la carte de segmentation est réduite et la gestion des textures cachées est plus aisée. De plus, on gagne en sémantique grâce à l'utilisation de modèles. L'approche par régions se caractérise par :

- une seule carte de segmentation. Il y a donc peu d'indépendance entre les régions;
- et une texture par région à chaque instant. Il n'y a pas de gestion des parties cachées ;

tandis que l'approche par objets se caractérise par :

- une segmentation par objets ; on ajoute donc l'information d'ordre de profondeur qui permet de recomposer chaque image en plaquant les objets du plus éloigné au plus proche. L'indépendance des objets est totale ;
- et une texture fixe ou faiblement dynamique par objet pour un groupe d'images. Cette texture est donc le résumé des textures découvertes sur un groupe d'images. Il y a une meilleure gestion des parties cachées.

La première section de ce chapitre traite du codage par régions, la deuxième section du codage par objets. La troisième section traite plus particulièrement du codage de forme, spécifique à ces deux formes de codage.

1.1 Les approches de codage par régions

Le codage de deuxième génération est apparu au début des années 1980 [Kunt et al. 85]. L'idée principale est qu'à très faible débit, l'information la plus pertinente pour le système visuel humain (SVH) est représentée par les contours. Les approches par codage de régions ont donc proposé une représentation de l'image en deux composantes :

- les contours définissant les régions,
- et la texture des régions.

Dans cette section sur le codage par régions, nous présenterons le codeur MORPHECO [Salembier et al. 95] car il nous semble décrire d'une assez bonne manière ce qu'est un codeur par régions. D'autres codeurs ont été proposés comme [Yokoyama et al. 95], [Garciagarduño 95] ou plus récemment [Mukherjee et al. 98], [Pateux 98].

1.1.1 Le codeur MORPHECO

La segmentation et l'extraction du mouvement

La segmentation est réalisée grâce à une approche morphologique multi-étapes par ligne de partage des eaux. On procède en 2 phases itérées :

- l'extraction de régions par ligne de partage des eaux qui donne une carte de segmentation avec une valeur de texture moyenne par région,
- et la définition d'une image résidu qui sera utilisée comme image à segmenter à l'étape suivante. Cette image résidu est issue de la différence entre l'image initiale et l'image des valeurs moyennes des régions.

La première image est segmentée sans autre information, les images suivantes utilisent la carte de segmentation précédente comme initialisation. La figure 1.1 illustre le résultat de la segmentation sur la séquence Foreman.

(a) séquence Foreman

(b) carte de segmentation

FIG. 1.1 – *Illustration d'une carte de segmentation obtenue par la technique de [Salembier et al. 95]*

En ce qui concerne le mouvement, on calcule une translation de contour. Les informations de texture ne sont donc pas prises en compte pour le calcul de ce mouvement. En effet, il est nécessaire de prédire au mieux les contours pour rester pertinent avec le SVH. De plus, la carte de segmentation étant coûteuse, il faut privilégier la qualité de prédiction des contours pour bénéficier au mieux de la redondance temporelle lors du codage de la carte de segmentation.

Le codage de la partition

La segmentation permet d'obtenir une carte pour toutes les images. Le codage de la première carte (intra) de segmentation est effectuée par chaîne de Freeman multiple (« Multi-Grid Chain Code » - MMC) [Salembier et al. 96] (cf. paragraphe 1.3.2).

Pour ce qui est des autres cartes, plusieurs étapes sont mises en œuvre pour les coder en utilisant au mieux la redondance temporelle. En effet, l'exploitation de la redondance temporelle permet de réduire le coût de codage des cartes mais est difficile à mettre en

place. Ceci est dû au fait qu'entre deux images, des recouvrements, des découvrements ainsi que des modifications de la forme des régions se produisent. On procède en 4 étapes :

- Première étape : on extrait les **régions en avant-plan**. Pour ceci, on définit l'ordre de profondeur d'une région grâce à l'erreur de prédiction d'une région. Cela permet via un graphe d'adjacence des régions, de trouver les régions qui sont en avant-plan. On utilise la règle suivante : si une région est entourée de régions en avant-plan, alors cette région est en arrière-plan.
- Deuxième étape : on calcule une **carte d'erreurs** de prédiction des régions en avant-plan. Les erreurs de prédiction des régions en avant-plan sont définies par la différence entre la carte de segmentation au temps t et les prédictions, sachant le mouvement, des régions en avant-plan du temps $t-1$.
- Troisième étape : on **filtre** par ouverture morphologique la carte des erreurs de prédiction des régions avant-plan pour supprimer les petites variations de contour qui sont coûteuses en débit.
- Quatrième étape : on code la carte d'erreurs de prédiction des régions en avant-plan par **contour de correction**. Les contours de correction (codés par chaînes de Freeman) représentent les nouveaux contours des régions en avant-plan.

Le codeur code le mouvement de chaque région, l'ordre de profondeur et les contours de correction par codage entropique. Le décodeur met à jour chaque région avant-plan, par ordre de profondeur croissant, grâce au mouvement et au contour de correction de la région. Ensuite, il faut retrouver les régions du fond. On filtre par ouverture la zone fond puis on affecte chaque région dans la zone fond, ceci sachant le mouvement. Une dilatation est ensuite effectuée sur toute les régions fond pour combler toute la zone fond et enfin un filtrage supprime les pixels isolés.

On constate donc que le codage de carte de contour propre au codage de région est assez complexe et que le gros problème vient des découvrements, recouvrements et changements de forme. Il est clair que l'approche objet permet de supprimer le problème de codage de carte. En effet, chaque objet a une forme lui étant propre. De plus cette forme est plus stable temporellement et est donc codée plus efficacement.

Le codage des textures

Le décodeur dispose du mouvement de chaque région ainsi que de la carte de segmentation. Il peut donc prédire la texture de chaque région. Certaines zones sont non prédites, on utilise alors une technique de remplissage de région par dilatation géodésique de texture. Il reste alors à coder l'erreur de prédiction de texture de chaque région. Ceci est effectué par transformée de type cosinus ou polynomial adaptée à la forme. Ensuite une quantification puis un codage entropique sont réalisés. La figure 1.2 illustre le résultat de codage sur la séquence Foreman à 5Hz à 32Kbit/s.

Le codeur MORPHECO a été ensuite enrichi à travers le projet SESAME [Marqués et al. 96] en prenant en compte une répartition du débit sur chacune des régions avec un critère débit-distorsion [Salembier et al. 97]. La figure 1.3 illustre l'amélioration de qualité (33Kb/s à 5Hz, PSNR inférieure à 30 dB), qui cependant reste inférieure à la qualité issue du codage au même débit avec le codeur H.263.

L'approche par région est donc intéressante mais laisse apparaître de fortes pertes en détail de texture et des artefacts de contour lorsqu'une région assez homogène de l'image

Première ligne : séquence originale de Foreman, deuxième ligne : résultat de codage

FIG. 1.2 – *Résultat de codage à 5Hz à 32Kbits/s. Figure extraite de [Torres et al. 96]*

est codée par plusieurs régions. Ces considérations, confirmées par des tests subjectifs [Alpert et al. 97], ont montré la supériorité des approches basées blocs. C'est pour cela que MPEG4 n'a pas retenu l'approche régions mais plutôt l'approche objets et modèles.

1.2 Les approches de codage par objets

Les approches de codage par objets utilisent l'information de forme, de mouvement, de texture et d'ordre de profondeur pour coder chaque objet. Le problème principal provient de la segmentation. Les sections suivantes présentent quelques codeurs incluant cette phase de segmentation. Ces sections donnent une idée des bénéfices que l'on peut attendre d'un codage objet automatique ainsi que les points pouvant être améliorés.

1.2.1 L'analyse-synthèse par modèle : OBASC

Les codeurs basés objet par analyse-synthèse (Object-based analysis-synthesis coder - OBASC) fonctionnent en 2 étapes : tout d'abord une étape de segmentation puis une étape de codage en objets vidéo. Le premier codeur de type OBASC a été introduit en 1989 par [Musmann et al. 89], dans la même période que les codeurs MPEG-1 et H.263 qui codaient par bloc. À la différence des approches par régions, chaque objet est codé indépendamment. Un objet est défini par son modèle de mouvement et est décrit par son mouvement, sa forme et ses paramètres de couleur.

Un OBASC est un codeur basé sur un schéma d'analyse-synthèse :

– dans premier temps le codeur extrait les objets vidéo grâce à une segmentation basée modèle de mouvement. Les zones d'un objet qui ne sont pas prédites par son modèle de mouvement sont appelées régions d'échec du modèle.

À gauche: séquence originale de Foreman, à droite : résultat de codage

FIG. 1.3 – *Résultat de codage par SESAME à 5Hz à 33Kbits/s. Figure extraite de [Torres et al. 96]*

– dans un second temps le codeur code les objets vidéo. Le codeur code le mouvement de l'objet puis la forme par prédiction via la compensation de mouvement. Enfin, la texture et la forme des régions d'échec du modèle sont codées.

L'efficacité du codage par OBASC dépend principalement du choix des modèles et de la possibilité d'extraire automatiquement ceux-ci lors de l'analyse de la séquence vidéo. Différents modèles ont été testés, comme des objets 2D rigides avec un mouvement 2D, des objets 2D rigides avec un mouvement 3D [Hötter 90], des objets 3D rigides [Galpin 02], et non rigides avec un mouvement 3D [Martinez 99].

Les approches de type OBASC montrent des performances très intéressantes, particulièrement en visiophonie lorsque l'on utilise des modèles 3D de corps humain. Ainsi, pour un codage à très faible débit, [Eisert et al. 99] obtiennent un gain en débit de 35% par rapport à H.263 TMN-10 à PSNR égal. La figure 1.4 montre la supériorité de l'approche MAC (Model-Aided Coder) de [Eisert et al. 99] par rapport à H263.

(a) codage H263 TMN-10 (b) codage par MAC

FIG. 1.4 – *Résultat de codage avec modèle de visage à 10Hz à 7Kbits/s. PSNR(H263) = 31,08 dB ; PSNR(MAC) = 33,19 dB. Figure extraite de [Eisert et al. 99]*

Cependant, le grand problème reste l'analyse de la séquence :

– d'une part les modèles sont souvent complexes et donc bien plus coûteux en temps

de calcul qu'une simple approche basée bloc, sans segmentation,

– d'autre part, les modèles ne sont pas génériques et donc l'approche échoue si les hypothèses du modèle ne sont pas suffisament remplies. On peut remarquer que le codage dynamique (cf. III) est une solution pour avoir plus de généricité : si l'on dispose d'une batterie de modèles, on retient celui qui fonctionne le mieux.

Ces deux problèmes ont fait que MPEG 4 s'est limité à définir un codeur/décodeur d'objets et à laisser les problèmes de segmentation de côté. En effet, la norme porte seulement sur le codage et le décodage et rien n'est fourni pour ce qui est de l'extraction d'objets vidéo. Dans les deux sous-sections suivantes, nous donnons deux exemples de codeurs OBASC génériques qui utilisent des modèles de mouvement 2D non rigides.

1.2.2 L'analyse-synthèse par modèle de mosaïque

Une mosaïque, appelée aussi « sprite », est une image qui est composée de l'ensemble des textures d'un objet découvertes sur plusieurs images. Par exemple, dans la séquence Stefan, il y a un plan montrant la montée au filet d'un joueur de tennis. On définit en général deux objets : le joueur de tennis en avant-plan et le reste de l'image en arrière-plan. La mosaïque résumant la texture de l'objet arrière-plan est alors la somme des textures découvertes lors du plan. La figure 1.5 montre le résultat du calcul de la mosaïque.

FIG. 1.5 – *Mosaïque de l'arrière-plan de la séquence Stefan*

Le codage par mosaïque consiste donc à obtenir une mosaïque pour un objet. Ainsi, cet objet est composé d'une seule image ainsi que des paramètres de mouvement permettant de reconstituer chaque moment de la séquence. Les autres objets sont codés différemment puisqu'ils n'entrent pas dans la catégorie mosaïque. Ainsi, lors du décodage, il est nécessaire de recomposer la séquence sachant la mosaïque et les autres objets.

Les auteurs de [Okada et al. 01] proposent une approche de codage mosaïque qui fonctionne sur un mode analyse-synthèse (cf. 1.2.1) et où la segmentation est effectuée de sorte que l'on dispose d'un objet en arrière-plan et d'objets en avant-plan. De plus, l'approche est générique. En effet, si certains critères ne sont pas réunis, le codage par mosaïque est abandonné au profit d'un codage par blocs sans notion d'objet. L'approche est donc très proche des notions de codage dynamique traitées dans la troisième partie du manuscrit.

Les critères de sélection du mode de codage par mosaïque sont :

– un minimum de 30 images pour générer la mosaïque,

– un mouvement de caméra de type zoom, rotation, inclinaison, ou panoramique, suffisamment prononcé,

– un avant-plan de taille inférieure à 30% de l'image.

Le codage de l'objet mosaïque est effectué par un codeur MPEG4 qui prend en compte ce type d'objet. Le codage consiste à coder par blocs DCT l'image mosaïque, et à coder les mouvements homographiques. Les résultats obtenus montrent pour la même distorsion à faible débit, un débit jusqu'à 5 fois inférieur pour le codage par mosaïque par rapport à un codage sans objet par MPEG4 Version 1.

1.2.3 L'analyse-synthèse par modèle de mouvement affine

Quelques codeurs d'objets par segmentation spatio-temporelle basée mouvement affine ont récemment été mis en œuvre. Ce type de codeurs fonctionne sur un mode analyse-synthèse de type OBASC (cf. 1.2.1). La partie codage exploite la dimension temporelle des segmentations par l'utilisation d'ondelettes 3D.

Dans [Han et al. 98], la segmentation est effectuée conjointement avec l'estimation du mouvement. Pour ceci, on maximise une probabilité a posteriori composée d'un terme d'attache aux données et de deux termes markoviens contraignant la régularité spatiale et temporelle de la carte de segmentation. La partie codage consiste à appliquer une ondelette temporelle sachant le mouvement affine de l'objet, puis une ondelette spatiale. La répartition des bits est effectuée via une optimisation débit-distorsion, le codage de forme est réalisé par codage de chaîne puis par prédiction. La figure 1.6 illustre le schéma de codage.

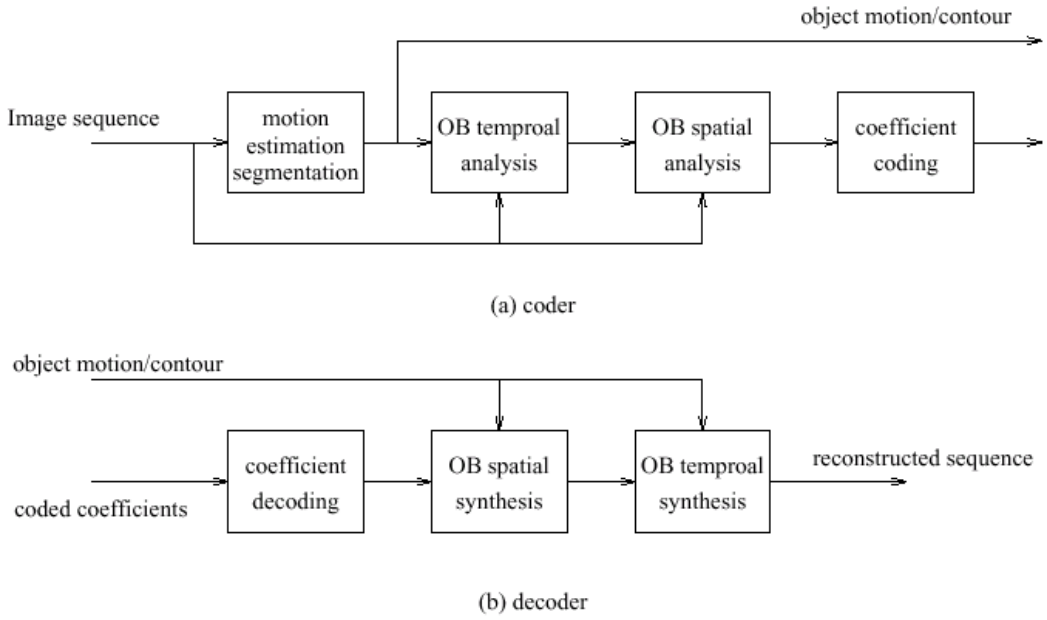
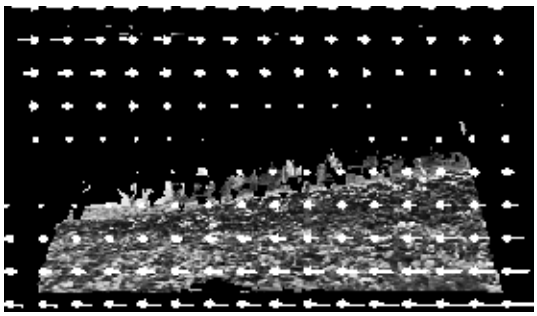


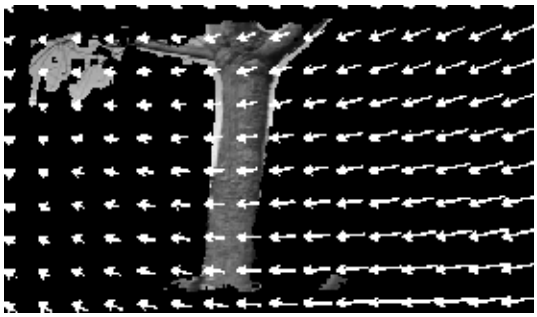
FIG. 1.6 – *Schéma de codage de [Han et al. 97]*

Dans [Schwarz et al. 00], on segmente de sorte d'obtenir la représentation en couches introduite par [Wang et al. 94] (voir figure 1.7). Ainsi, on dispose d'un certain nombre d'objets avec ordre de profondeur. Cette segmentation est réalisée sur un groupe d'images.



couche correspondant au parterre de fleur

couche correspondant aux maisons



couche correspondant a l'arbre

image reconstruite à partir des couches

FIG. 1.7 – *Résultat d'une segmentation en couche par l'algorithme de [Wang et al. 94]. Figure extraite de [Wang et al. 94]*

Ensuite, le mouvement de type affine et la forme d'un objet pour le groupe d'images est codé et décodé. Puis, toutes les images de texture de l'objet sont compensées en mouvement vers une image référence, pour aligner les textures sur un axe temporel. On peut alors effectuer la transformation ondelette 3D sur le groupe des textures de l'objet. Le codage des coefficients ondelette est alors réalisé grâce à une version étendue de l'algorithme de SPIHT [Said et al. 96]. La figure 1.8 illustre le schéma de codage. Les résultats présentent une supériorité à très faible débit de 2 à 3 dB par rapport à MPEG2 TM5.

1.2.4 Le codage par MPEG4

À la différence d'un OBASC, MPEG4 n'est pas un codeur qui prend en entrée une séquence brute. En effet, il faut alimenter le codeur d'une segmentation en objets. On fait donc bien la distinction entre la partie segmentation et la partie codage. MPEG4 propose ainsi plusieurs type de codage :

- **un codage objet** pour lequel chacun des objets est défini par un mouvement, une forme, une texture et un ordre de profondeur.
- **un codage de mosaïque** pour lequel l'objet mosaïque est défini par l'image mosaïque et les paramètres de mouvement homographique.
- **un codage de visage et de corps par modèle 3D** (Syntactic Natural Hybrid Coding - SNHC) pour lequel l'objet visage ou corps est défini par un maillage 3D, les déformations de ce maillage et une texture.

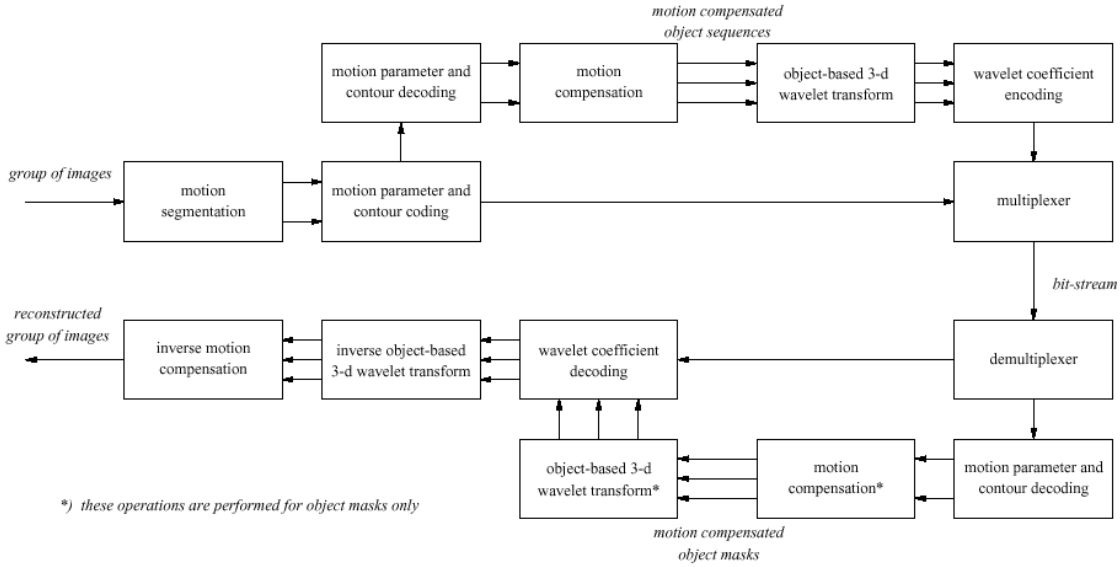


FIG. 1.8 – *Schéma de codage de [Schwarz et al. 00]*

La figure 1.9 illustre le schéma de codage utilisé dans MPEG4. On peut constater que trois informations sont à coder : le mouvement, la forme et la texture. Par contre, il faut bien noter que ces trois informations sont codées de manière dépendantes. Il y a trois étapes :

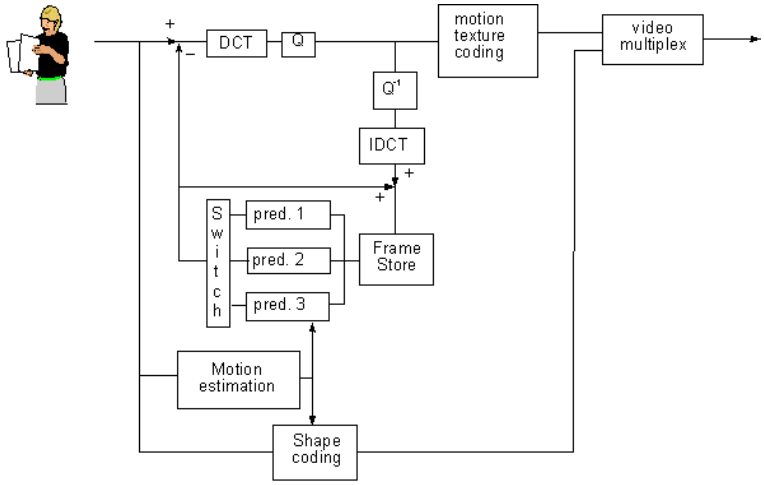


FIG. 1.9 – *Schéma de codage du codeur vidéo MPEG4. Figure extraite de [ISO/IEC 02]*

– **première étape : le codage du mouvement** (s'il y en a) qui peut être codé par :
  – un **mouvement global** de l'objet (Global Motion Compensation - GMC). Le nombre de paramètres du mouvement est ainsi très faible ;
  – les déformations d'un **maillage 2D**,
  – des **paramètres d'animation** de visage (« Face Animation Parameters ») ou des paramètres d'animation de corps (« Body Animation Parameters ») lors de

l'utilisation de SNHC ;

- **deuxième étape : le codage de forme** par MPEG CAE (cf. 1.3.1). On utilise le mouvement pour améliorer le codage de la forme. Avec SNHC, il n'y a pas de phase de codage de forme, mais plutôt une étape de pré-encodage spécifiant les caractéristiques du visage (« Facial Definition Parameters ») ou du corps (« Body Definition Parameter ») ;
- **troisième étape : le codage des textures** ou des erreurs de textures par DCT ou ondelette adapté à la forme (Shape adapted DCT- SA-DCT ou Shape adapted Wavelet - SA-WLT). Dans le cas du codage par mosaïque, la texture peut être fixe ou dynamique. Dans le cas du codage SNHC la texture est optionnelle et codée une seule fois lors de la définition des caractéristiques du visage (« Facial Definition Parameters ») ou du corps (« Body Definition Parameter »).

On peut donc voir MPEG4 comme une boîte à outils logicielle où se rajoute les nouvelles techniques de codage. En effet, MPEG4 est une norme ouverte aux extensions et se veut la référence en codage vidéo. Une attention particulière peut être portée sur l'unité « Joint Video Team (JVT) » qui réunit le monde ISO et le monde ITU. Cette unité travaille sur le codeur-décodeur **MPEG4 partie 10, Advanced Video Coding (MPEG4 AVC)**, appelé aussi **H264/AVC**, qui devrait être achevé fin 2003. H264/AVC repose sur le travail qu'a réalisé l'ITU sur le codeur-décodeur H.26L. Ce codeur est actuellement le plus efficace dans la famille des codeurs par blocs.

L'approche MPEG4 est très intéressante dans son aspect de normalisation du codage. De plus, elle confronte les deux tendances actuelles du codage qui sont :

- augmenter le nombre de modèles. On dispose d'une gamme qui va du mouvement dense par bloc jusqu'aux modèles 3D en passant par des mouvement affines ou bien des maillages 2D,
- augmenter la complexité de chaque modèle. Le nombre de paramètres des modèles augmente (exemple de H264/AVC qui pousse très loin l'approche par bloc en proposant des tailles de blocs 4x4 jusqu'à 16x16 et des optimisations débit-distorsion).

La suite du manuscrit présente le codage de forme qui est l'information spécifique au codage par région ou par objet.

1.3 Le codage de forme

Dans les codeurs basés régions tels que MORPHECO [Salembier et al. 95] ou basés objets tels que MPEG4 [ISO/IEC 98], les informations issues de la segmentation doivent être codées. Généralement, on classe les techniques de codage de formes en deux catégories : le codage basé image et le codage basé contours.

1.3.1 Le codage de forme basé image

Le codage par Modified-Modified Read (MMR)

La méthode de codage par Modified-Modified Read est basée sur le parcours ligne à ligne d'une image binaire [ITUT 98]. On code la longueur de chaque segment noir et chaque segment blanc. Pour améliorer l'efficacité de la méthode, on utilise l'information

de la ligne précédente qui fait office de prédiction. Cet algorithme a été mis en place dans les systèmes de télécopie par la recommandation ITU-T T.6 [ITUT 98].

[Yamaguchi et al. 97] ont proposé une extension appelée Modified MMR qui exploite une compensation en mouvement par bloc et propose également un codage avec pertes par l'intermédiaire d'une conversion d'échelle sur les blocs de pixels traités.

Le codage par Context-Based Arithmetic Encoding (CAE)

L'approche par codage arithmétique basé contexte [Brady et al. 97] est une technique de codage très efficace. Elle consiste à coder l'appartenance d'un pixel à la forme sachant la connaissance de ses voisins. Le voisinage définit un contexte qui est utilisé pour accéder à une table contenant une distribution de probabilités. L'appartenance du pixel à la forme et sa distribution de probabilité sont alors utilisées par le codeur arithmétique pour coder le pixel.

Le codage de l'image est alors réalisé en parcourant celle-ci et en apprenant les distributions de probabilité durant le parcours dans le cas de CAE adaptatif. JBIG [ITUT 93] met en œuvre cette approche.

Le codage MPEG4 CAE

MPEG 4 a retenu l'approche par CAE pour le codage de carte de segmentation en ajoutant l'utilisation de macro-blocs (BABs - Binary Alpha Blocks) ainsi que la prédiction temporelle. Il y a donc deux étapes dans le codage de forme MPEG CAE.

- La première étape fonctionne au niveau macro-bloc (16x16). Le codeur dispose de 5 modes de codage :
  - le mode transparent,
  - le mode opaque,
  - le mode codage intra,
  - le mode codage inter,
  - le mode codage par compensation de mouvement ;
- La deuxième étape consiste à coder les pixels des macro-blocs inter et intra. Le contexte utilisé pour le codage arithmétique de chaque pixel dépend du type de mode : inter ou intra. Dans le cas inter, le contexte englobe les points provenant de la carte de segmentation précédente. La figure 1.10 illustre les pixels contexte.

C'est actuellement la solution référence pour le codage de forme dans le domaine de la vidéo. Cependant, à très bas débit, la perte d'information est telle que l'effet de bloc est visible. En effet, à très faible débit, on ne code plus que l'information bloc opaque ou bloc transparent.

Le codage par décomposition en squelette

La décomposition en squelette, aussi appelée par transformation par axe médian, ou transformation par axe symétrique, est issue de la morphologie mathématique [Ballard et al. 82]. L'approche consiste à appliquer à l'image un élément structurant : disque, carré, croix ... On extrait ce que l'on appelle un squelette.

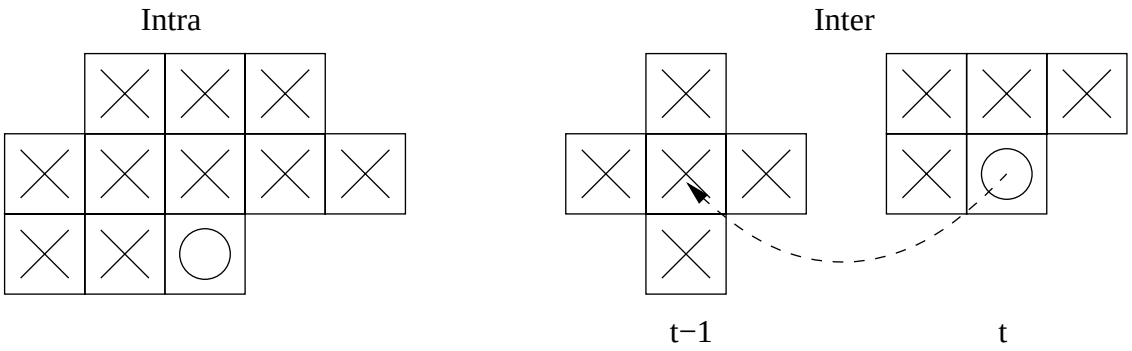


FIG. 1.10 – *Contexte intra et inter utilisés pour le codage par MPEG CAE. Le rond indique le point à coder, les croix le contexte de codage. Dans le cas d'un codage inter, le contexte pris dans l'image précédente est composé du point mis en correspondance par compensation de mouvement et de ses quatre voisins.*

Le squelette est alors codé par exemple par plage (« run-length ») puis par codage arithmétique adaptatif [Brigger 95]. Le coût de codage est assez élevé mais on peut le réduire en cherchant un ensemble de squelettes déconnectés ou bien en minimisant le nombre de points squelette reconstruisant l'aire maximum de la forme. La figure 1.11 illustre la notion de squelette déconnecté (squelette géodésique). L'extraction du squelette est souvent réalisée grâce à l'algorithme de Rosenfeld-Pfaltz [Rosenfeld et al. 82].

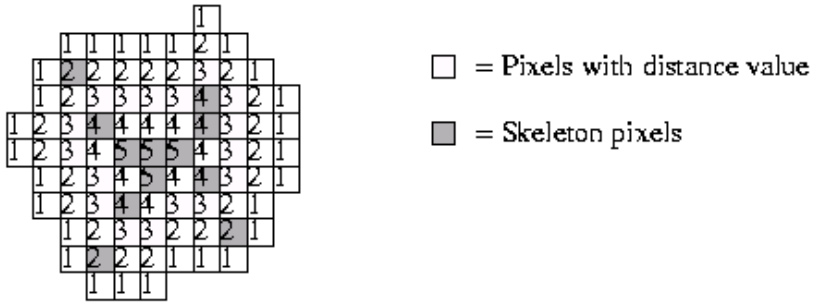


FIG. 1.11 – *Illustration du squelette d'une forme quelconque. Figure extraite de [Herrmann et al. 97]. Chaque carré représente un pixel. Les valeurs présentes dans chaque carré représentent la distance au contour extérieur (distance de Chamfer). Les carrés grisés représentent le squelette de la forme.*

Une particularité intéressante du codage par squelette est de permettre la hiérarchisation progressive du flux. En effet, un squelette est composé de points dont les grandes valeurs sont plus représentatives que les petites : elles permettent de reconstruire une aire plus grande. Ainsi, un ordonnancement décroissant des valeurs nœud permet de définir une hiérarchisation du flux vidéo. Le flux progressif obtenu est robuste aux erreurs de transmission. En effet, la perte d'un nœud du squelette n'a qu'une influence très locale sur la forme reconstruite.

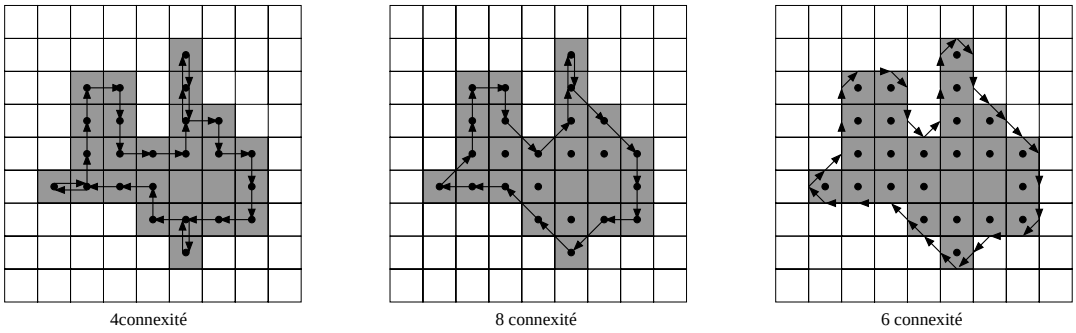
1.3.2 Le codage de forme basé contours

Le contour par chaîne de Freeman

Le codage de contour par chaîne de Freeman [Freeman 61], aussi appelé codage par chaînage, consiste à décrire un contour par une succession de déplacements. On définit un point de départ sur le contour et ensuite on donne une suite de directions permettant le parcours du contour. On distingue alors plusieurs choix possibles pour définir les directions de déplacement.

- Une discrétisation 4-connexe : on utilise alors 4 directions de déplacements : Nord, Ouest, Sud, Est. On se déplace du centre du pixel au centre du pixel suivant.
- Une discrétisation 8-connexe : on utilise alors 8 directions de déplacements : Nord, Nord-Ouest, Ouest, Sud-Ouest, Sud, Sud-Est, Est, Nord-Est. On se déplace du centre du pixel au centre du pixel suivant.
- Une discrétisation 6-connexe : on parcourt les arêtes des pixels. Le déplacement est donc différent dans le cas où l'on est sur une arête horizontale ou bien verticale.

La figure 1.12 illustre les différentes représentations d'un même contour par utilisation de différentes connexités.



(a) Parcours 4-connexe (b) Parcours 8-connexe (c) Parcours 6-connexe

FIG. 1.12 – *Illustration de représentations 4-connexe, 8-connexe, 6-connexe*

La chaîne de Freeman est alors codée de manière différentielle, c'est-à-dire que l'on utilise un schéma prédictif pour coder l'erreur de prédiction d'une direction. Ensuite, on utilise en général un codeur arithmétique. Ce schéma de codage est l'un des plus efficaces parmi les codage sans perte. On peut aussi faire du codage avec perte en lissant le contour par filtrage morphologique, en simplifiant le parcours en augmentant les zones de direction constante, ou bien en sous échantillonnant.

La description 4-connexe définit une chaîne qui est en moyenne 33% plus longue que la description 8-connexe [Rosenfeld et al. 82]. De plus, la représentation en 4-connexe n'est pas efficace pour représenter les diagonales. C'est donc couramment la 8-connexité qui est utilisée pour coder un contour par chaînage. La 6-connexité et la 8-connexité donnent des coût de codages similaires. En général le codage sans perte par chaînage donne un débit de l'ordre de 1.2 bits par pixels contour [Eden et al. 85]. Bien entendu, pour atteindre ce débit, il faut coder de manière efficace en utilisant par exemple des chaînes de Markov [Pateux et al. 98].

Le codage par ligne de base

Le codage par ligne de base [Lee et al. 97] consiste à représenter le contour par une liste de distances entre le contour et un des deux axes de l'image. Dans un premier temps, on place le contour dans un système de coordonnées 2-D et l'on choisit l'axe ayant la projection du contour la plus longue. Cet axe est appelé la ligne de base et sert à mesurer la distance au contour. Le parcours du contour permet d'extraire de manière régulière les distances entre le contour et la ligne de base. Les points où se produit un changement de sens de parcours vis-à-vis de la ligne de base sont appelés « turning point ». Le contour est subdivisé en segments de 16 pixels de long et la reconstruction du contour est effectuée par interpolation linéaire entre deux points connus du contour. Il est possible, de sur-échantillonner ou bien sous échantillonner tant que l'on reste sous un seuil de distorsion fixé.

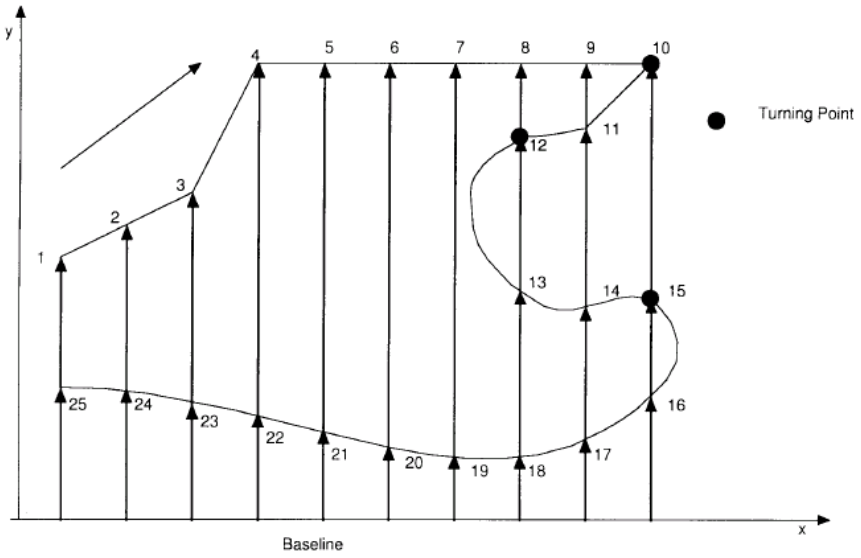


FIG. 1.13 – *Illustration de la représentation d'un contour par ligne de base. Figure extraite de [Katsaggelos et al. 98]*

La figure 1.13 illustre la représentation par ligne de base. On peut constater que cette forme de représentation n'est pas adaptée pour représenter des contours chahutés. De plus elle est beaucoup moins flexible que les approches par contour polygonal ou bien par B-Spline.

Le codage par contour polygonal

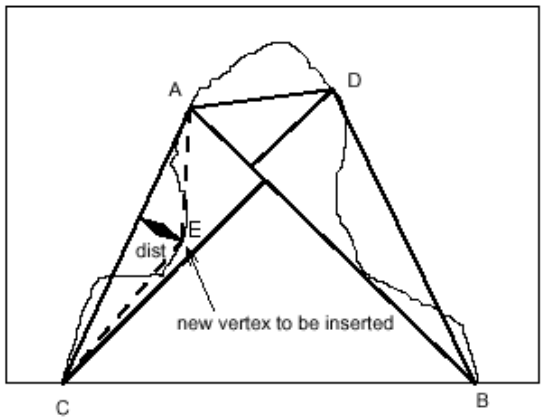
Le codage par contour polygonal consiste à représenter le contour par quelques sommets. Pour reconstruire le contour, on relit les sommets par des segments de droite. Bien entendu, plus on réduit le nombre de sommets, plus la forme reconstruite est anguleuse.

L'algorithme d'extraction des sommets consiste le plus souvent à minimiser la distorsion entre le contour original et le contour reconstruit. La mesure de distorsion peut par exemple être la distance Euclidienne maximum entre les deux contours.

Une solution non optimale a été initialement proposée par [Ramer 72]. La recherche est effectuée de manière itérative :

- dans un premier temps, on extrait les deux points les plus éloignés du contour original. Les deux sommets extraits définissent la première approximation du contour (sur la figure 1.14 on donne ACBDA comme exemple d'approximation du contour) ;
- dans un deuxième temps, on extrait le point le plus éloigné entre le contour original et l'approximation courante. Ce point définit un nouveau sommet qui permet de raffiner l'approximation (sur la figure 1.14, le sommet ajouté est le sommet E et la nouvelle approximation est alors AECBDA). On itère alors l'extraction de sommets jusqu'à ce que la distance entre l'approximation et le contour original soit inférieure à un seuil fixé.

La figure 1.14 illustre l'extraction itérative.



AB : axe principal - ABCD : 4 sommets initiaux - ACBDA : approximation polygonal courante

FIG. 1.14 – *Sélection itérative des sommets. Figure extraite de [Jordan et al. 98]*

Des algorithmes plus sophistiqués ont été développés pour sélectionner de manière optimale les sommets. Une revue des approximations polygonales est donnée dans [Dunham 86].

Le codage par contour B-Spline

Le codage par contour de type spline cubique consiste à représenter le contour par un ensemble de points de contrôle. Ces points de contrôle permettent une reconstruction basée sur une interpolation polynomiale entre les points de contrôle. Les contours reconstruits présentent des aspects lisses qui sont agréables visuellement. Par contre, cette approche n'est pas bien adaptée pour représenter des angles aigus.

L'algorithme consiste à extraire les points de contrôle par la minimisation d'une distance entre le contour original et la B-Spline.

Une B-Spline est définie sur une abscisse curviligne s, appartenant à l'intervalle $[1, S]$, vers un point de $\mathbb{R}^2$ par :

$$B : [1, S] \rightarrow \mathbb{R}^2, s \rightarrow B(s).$$

La B-Spline est une fonction paramétrique dépendant de K points de contrôle c_k et de K fonctions noyaux ϕ_k telle que :

$$B(s) = \sum_{k=1}^{k=K} c_k \phi_k(s). \tag{1.1}$$

Les fonctions noyaux ϕ_k sont définies via la fonction de forme ϕ de sorte que l'on ait :

$$\phi_k(s) = \phi(s - k).$$

La figure 1.15 illustre la fonction de forme ϕ pour une B-Spline cubique. On a ainsi la

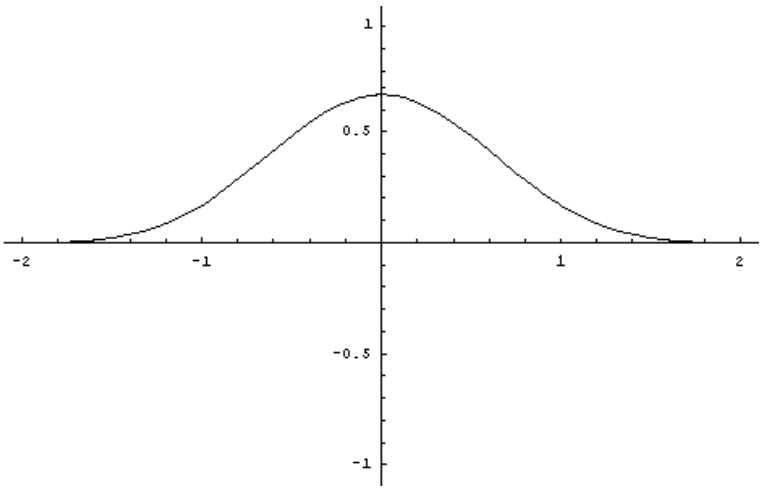


FIG. 1.15 – *Fonction de forme ϕ pour une B-Spline bicubique*

définition générale d'une B-Spline fermée :

$$B(s) = \sum_{k=1}^{k=K} c_k \phi((s - k) mod(K)).$$

On peut remarquer que seulement quelques points de contrôle interviennent sur la définition d'un point de la B-Spline. Dans le cas de la B-Spline cubique, il n'y a que 4 points de contrôle qui influencent la position d'un point. En effet, les autres points de contrôle ont une valeur de fonction de forme nulle.

Pour trouver la B-Spline B qui représente au mieux un contour C, on peut choisir de minimiser :

$$\min_{c_k} \sum_{i=1}^{N} (B(s) - C(i))^2,$$

avec i les indices de parcours du contour C. Deux problèmes se posent alors : le choix du nombre de points de contrôles et leur répartition. Une solution sous-optimale couramment mise en œuvre consiste à répartir de manière uniforme les points de contrôle puis à perturber chaque point de contrôle dans les huit directions possibles et conserver les positions d'erreur minimale. L'algorithme est itératif et s'achève quand l'erreur passe sous un seuil.

D'autres solutions pour résoudre l'approximation par B-Spline on été proposées. Par exemple, [Lu et al. 94] optimisent les positions des points de contrôle. Dans un premier temps, le nombre de points de contrôle est choisi en fonction des zones de fortes courbures. Puis, l'optimisation des points de contrôle est réalisée. Des approches débit-distorsion peuvent aussi être mise en œuvre [Katsaggelos et al. 98]. Une dernière solution consiste à calculer une spline lissée ("smoothing spline") [Precioso et al. 03]. Les conditions d'interpolation ne sont plus totalement satisfaites mais en contrepartie la spline est plus lisse.

Le codage par transformée

Le codage par transformée regroupe toutes les techniques basées sur la transformation d'un signal. Cela consiste à changer la représentation constituée de la suite des points du contour. Ce changement de représentation est utilisé pour concentrer les informations dans un petit nombre de coefficients. Ainsi, la représentation par B-Spline peut être vue comme une transformée.

Dans [Otterloo 91], la transformée de Fourrier est utilisée pour coder un contour. La liste des points du contour (x_i, y_i) est transformée en une liste ordonnée de couples $(i, (y_{i+1} - y_i)/(x_{i+1} - x_i))$, avec i l'indice de parcours du contour et $(y_{i+1} - y_i)/(x_{i+1} - x_i)$ le changement de direction du contour. Grâce à la périodicité d'échantillonnage des points du contour, on peut alors réaliser la transformation de Fourrier sur la liste de couples. Pour préserver les caractéristiques du contour, seuls les forts coefficients de Fourier sont conservés. Dans [Spaan et al. 97], on utilise une transformée par DCT avec au préalable une représentation en coordonnées polaires des points du contour. D'autres transformées peuvent être utilisées comme la transformée en ondelette [Yoshida et al. 98].

1.4 Résumé du chapitre

Ce chapitre présente l'approche de codage par région ainsi que celle par objet. L'approche par objet permet d'avoir une plus grande indépendance entre objets et permet de gérer plus facilement les cas de recouvrement et de découvrement de texture. En effet, chaque objet possède une texture propre qui peut être occultée à certains moments de la séquence. Bien entendu, il est nécessaire de connaître l'ordre de profondeur de chaque objet, ce qui n'est pas aisé.

L'approche objet permet aussi d'avoir une plus grande indépendance sur le codage puisque l'on peut proposer différents modèles d'objet. Les codeurs de type OBASC avec une phase d'analyse suivie d'une phase de synthèse illustrent le gain que l'on peut obtenir par rapport au codage non objet. Les quelques exemple d'OBASC donnés ici laissent à penser que le codage d'objet vidéo est possible de manière automatique dans les cas ou l'on est capable de trouver un modèle. Cela peut être un modèle 3D de scène rigide, un modèle 3D de visage, un modèle 3D de corps humain, des modèles de mouvement affine (segmentation en couche), un mouvement global (mosaïque) etc. De plus, les quelques résultats présentés montrent des performances intéressantes.

Enfin, nous avons vu les différentes techniques de codage de contour qui, bien qu'efficaces, utilisent peu la notion de stabilité temporelle long terme du contour. En effet, la

plupart des approches ne prennent pas en compte la redondance temporelle long terme. De plus, les diverses représentations ne proposent pas ou peu la hiérarchisation.

Le chapitre suivant s'appuie sur le fait que le codage d'objets vidéo est prometteur et donc propose d'améliorer la technique de codage d'objets vidéo grâce à une meilleure répartition de l'information de mouvement, de texture et de forme. Cette répartition passe par une décorellation des trois informations : mouvement, texture et forme ainsi que par une représentation long terme de ces trois informations. Un codage de contour est proposé de sorte que la redondance temporelle soit mieux prise en compte. Ainsi, la représentation du contour est aisément hiérarchisable. Nous proposons alors un schéma hiérarchique de codage d'objet vidéo.

Chapitre 2

Vers une hiérarchisation totale d'un flux vidéo

2.1 Une décomposition plus hiérarchique

Comme on l'a vu dans le chapitre précédent, le codage objet présente de nombreux avantages par rapport au codage par régions. Vis-à-vis du codage par blocs, le codage objet propose de nouvelles fonctionnalités telles que la manipulation des objets ou la hiérarchisation du flux par objets.

Pour aller plus loin dans la hiérarchisation du flux vidéo, il est alors possible de séparer les informations de mouvement, de texture et de forme. Chacune de ces trois informations peut alors être codée indépendamment. Des distorsions sont alors possibles sur les paramètres de mouvement et de forme ce qui peut affecter la reconstruction de l'objet.

Bien que ces distorsions soient présentes, elles sont peu visibles par le système visuel humain lorsqu'elles sont faibles. Cette hypothèse sur le système de vision humain est largement utilisée pour réduire le coût de codage du mouvement et de la forme. Cela permet ainsi de donner plus de débit à la texture et de permettre une hiérarchisation complète du flux d'un objet en trois informations indépendantes : mouvement, texture et forme.

Cette technique de décomposition en trois caractéristiques (mouvement, forme, texture) nous permet de définir un codeur d'objet vidéo complètement hiérarchique. Ainsi, par rapport aux schémas blocs de type H264/AVC [Wiegand et al. 03] ou par rapport aux techniques de codage d'objets par ondelettes [Han et al. 97] [Schwarz et al. 00], présentés dans la section 1.2.3, nous proposons une notion de hiérarchisation bien plus puissante.

Le schéma 2.1 illustre le fonctionnement de notre codeur objet. Une estimation du mouvement est effectuée, ce qui permet de décorréler le mouvement, la texture et la forme. La partie codage de texture et de mouvement est succinctement décrite dans la section suivante (voir le codeur ondelette 3D [Cammas et al. 03b] pour plus de détails). Ensuite, nous traitons le codage de forme de manière indépendante par rapport au mouvement et à la texture.

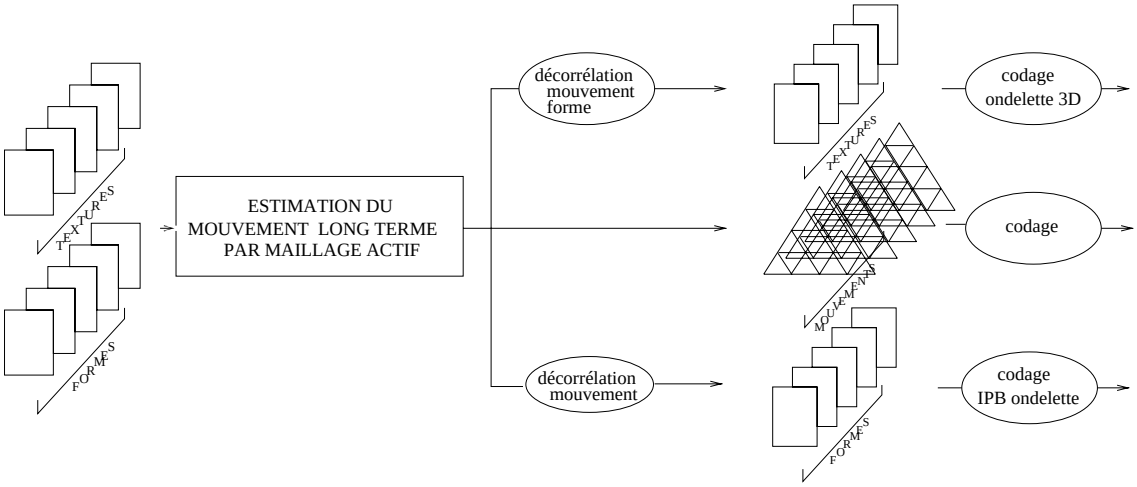


FIG. 2.1 – *Illustration du schéma de codage objet avec décorrélation des information mouvement, texture, forme*

2.2 Le codeur ondelette 3D

Le codeur par ondelette 3D non objet est celui présenté dans [Cammas et al. 03a] [Cammas et al. 03b]. L'approche consiste à effectuer tout d'abord une analyse long terme du mouvement grâce à une estimation par maillage dynamique [Pateux et al. 01] [Marquant 00]. Une fois que l'estimation est effectuée, on décorrèle l'information de texture en projetant toutes les textures dans un ou plusieurs temps référence. On dispose alors de deux informations complètement décorrélées que l'on peut donc coder indépendamment.

Le codage de la texture consiste à faire une décomposition en ondelettes temporelles puis spatiales. Un schéma de lifting avec redressement des textures est mis en œuvre lors de l'application de la transformée ondelette temporelle. Ensuite, un codage hiérarchique des sous-bandes est effectué par EBCOT [Taubman 00]. Le codage de mouvement est effectué en utilisant les propriétés hiérarchiques du maillage ainsi que par l'utilisation d'un codeur arithmétique en plan de bits [Marquant et al. 00].

Par rapport aux autres techniques ondelette 3D [Taubman et al. 94], [Ohm 94], [Choi et al. 99], [Secker et al. 01], [Luo et al. 01], le maillage, le lifting ainsi que l'indépendance entre le codage de texture et le codage de mouvement sont des éléments qui permettent au codeur ondelette 3D non objet d'égaler les performance de H26Lv8 sur certaines séquences. De plus, le codeur ondelette 3D non objet a la propriété de permettre la hiérarchisation grâce à l'utilisation des ondelettes.

Une remarque importante est que le système visuel humain est peu sensible aux faibles distorsions géométriques introduites par le codage avec perte du mouvement. Ainsi, il est possible de gagner du débit pour la texture en en prenant sur le mouvement. Le gain obtenu par cette indépendance ainsi que par l'utilisation d'un maillage n'est pas négligeable puisqu'à faible débit (environ 100Kb/s sur CIF Foreman à 15Hz), les schémas ondelette 3D par bloc, utilisent 45% du débit pour le mouvement, alors que le codeur ondelette 3D non objet n'utilise que 13% du débit [Cammas et al. 03a] [Vieron et al. 02].

Le codeur a aussi la possibilité de passer en mode objet. En effet, il est possible d'utiliser

(a) VOP du visage (b) texture prolongée

FIG. 2.2 – *Illustration du résultat de prolongement de texture*

l'estimation par maillage sur une zone définie par un masque. De plus, la transformée ondelette 3D est réalisée avec auparavant l'utilisation d'un prolongement de texture pour remplir les zones non définies (padding). La figure 2.2 illustre le prolongement de texture sur l'image 0 de la séquence CIF Foreman. Ainsi, la transformée ondelette 3D est appliquée sur des images rectangulaires. Cette technique permet de rendre le codage de texture complètement indépendant du codage de forme.

De plus, la technique de prolongement de texture utilisée introduit peu de haute fréquence et est spécialement conçue pour ne pas introduire de fort coût de codage. Ainsi, le codeur ondelette 3D objet permet de définir trois composantes : la texture, le mouvement, et la forme, qui peuvent être codées de manière indépendante.

Une remarque est que les régions de découvrement et de recouvrement posent problèmes lors de la décomposition ondelettes 3D non objet (voir chapitre résultats section 3.3). Le fait de proposer un codage objet permet de supprimer ces zones à problème et donc potentiellement d'améliorer les performances du codage ondelette 3D. Ainsi, avec l'hypothèse supplémentaire que le système de vision humain est moins sensible à la perte sur la forme, on peut coder certaines partie de l'image (objets) avec des GOP de tailles différentes et des répartitions de débit différentes et dépasser les performances du codage ondelette 3D non objet tout en conservant la propriété de hiérarchisation du flux vidéo.

Pour passer en mode objet le codeur ondelette 3D, il nous faut disposer d'un codeur de forme proposant la hiérarchisation. Dans la section suivante nous proposons un codeur de contour qui possède des propriétés long terme et qui propose une structure aisément hiérarchisable.

2.3 Le codage spatio-temporel long terme de contour

Le codage de contour que nous proposons ici a un double objectif. Le premier est de prendre en compte de manière plus importante que les approches existantes la stabilité temporelle d'un contour. Cette approche long terme permet d'être plus efficace mais

aussi permet d'avoir une représentation très facilement hiérarchisable. Le second objectif consiste à renforcer la stabilité temporelle dans les zones d'occultation. Ainsi, on souhaite s'approcher des contours réels d'un objet et non des contours dûs aux occultations.

Dans les sous-sections suivantes nous allons expliquer la représentation que nous avons choisie. Puis nous donnerons la technique de codage de cette représentation. Le schéma 2.3 résume l'ensemble des étapes permettant le codage spatio-temporel long terme de contour.

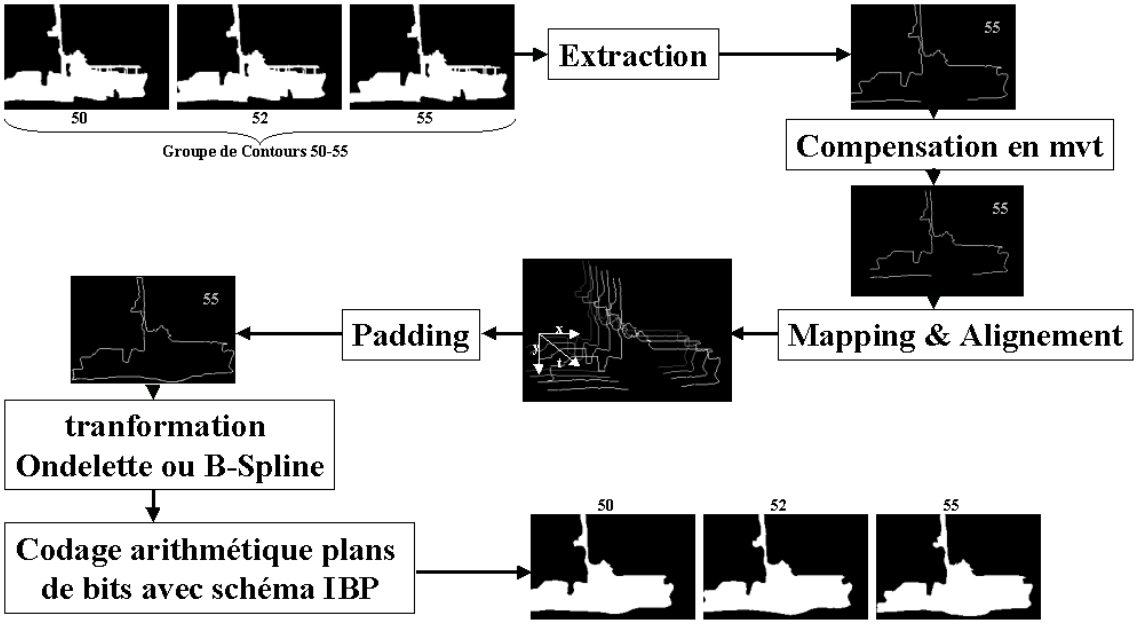


FIG. 2.3 – *Schéma de codage de contour*

2.3.1 Extraction, alignement et prolongement des contours

Notre objectif est d'obtenir une mise en correspondance des points du contour au cours du temps. Cette mise en correspondance permet alors de représenter l'évolution du contour par deux plans spatio-temporels (comme dans [Yoshida et al. 98]) donnant la position (x, y) d'un point du contour sachant l'indice s sur ce contour et le numéro de l'image t (voir figure 2.14).

Les deux plans spatio-temporels ont la propriété d'être lisses et stables temporellement et spatialement (grâce à la Compensation en mouvement, au Mapping, à l'Alignement et au Padding. cf. schéma 2.3) et ceci sur une grande fenêtre temporelle. Cette propriété permet un codage efficace par ondelette et donc une hiérarchisation aisée du flux. De plus, la décorrélation des contours et du mouvement (grâce à la compensation en mouvement. cf. schéma 2.3) nous permet d'avoir un codage de contour indépendant du codage de mouvement. Cela permet d'optimiser indépendamment le codage de contour et le codage de mouvement.

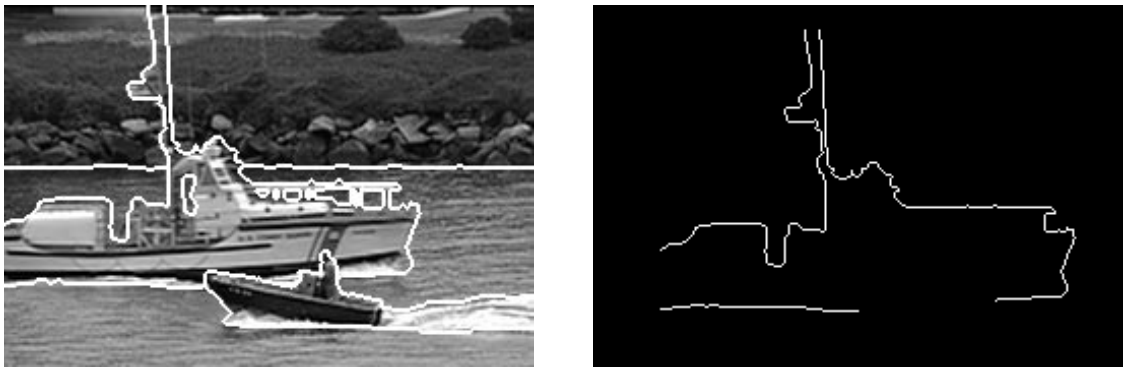
Les différentes étapes pour obtenir les deux plans spatio-temporels sont :

1. extraction des contours et compensation en mouvement,
2. mise en correspondance des contours consécutifs (mapping),

3. alignement du groupe de contours et sur-échantillonnage,
4. prolongement spatio-temporel dans le cas de contours ouverts.

Extraction des contours et compensation en mouvement

Une forme est composée de contours internes (les « trous » à l'intérieur de l'objet) et externes (la forme globale). Ce que nous appelons contour externe correspond à l'enveloppe d'un objet vidéo sans les parties dûes aux occultations. La figure 2.4 montre le contour externe du grand bateau de l'image 50 de la séquence Coastguard. Sans perte de généralité, nous nous restreignons au contour externe. Les contours internes peuvent être codés de la même façon ou bien par un autre codeur.



(a) Carte de segmentation initiale (b) Contour partiel extérieur

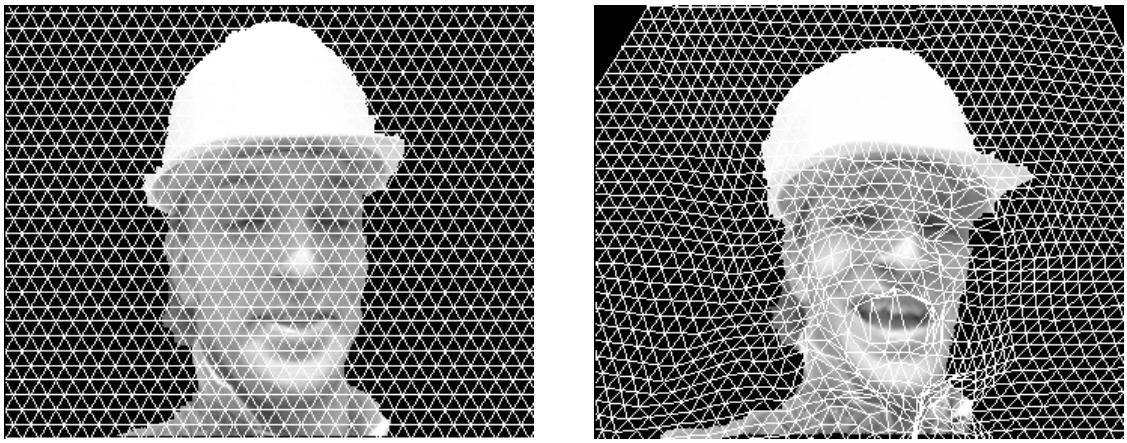
FIG. 2.4 – *Extraction du contour apparent de l'objet vidéo grand bateau de la séquence Coastguard*

La connaissance des parties valides (segments qui ne sont pas dûs à une occultation) de l'enveloppe externe d'un objet est déduite grâce à l'ordre de profondeur associé à chaque segment composant l'enveloppe. On suppose donc que cette information de « z-order » local est connue et qu'elle peut être obtenue comme dans [Bonnaud et al. 97]. Cette information est réutilisée après décodage lors de la composition des différents objets composants la scène.

Ainsi, nous décrivons une forme par une liste chaînée de positions extraites à partir du contour externe. Cette représentation possède éventuellement des « ruptures » puisque le contour peut être partiellement occulté. On peut remarquer que le bord de l'image est vu comme une occultation.

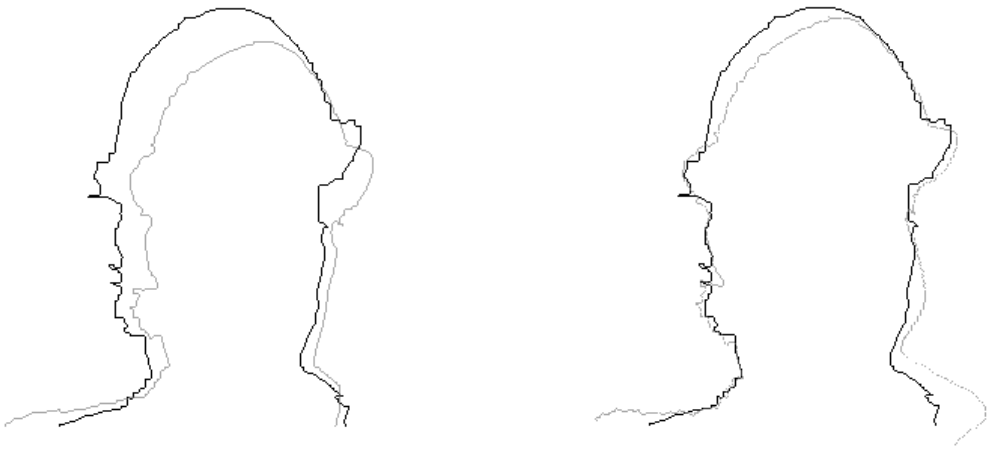
Une fois que les listes de positions sont obtenues (une par image), les listes sont projetées vers un temps de référence en utilisant le mouvement de la texture. Cette projection facilite le processus de mise en correspondance des contours et renforce la stabilité temporelle de notre représentation. Le fait de considérer le mouvement de texture et non celui de contour permet de décorréler mouvement et forme; cela rend possible un codage pleinement progressif pour chacune de ces deux informations [Chaumont et al. 03a]. L'utilisation du mouvement de la texture permet aussi de ne pas avoir à coder l'information de mouvement des contours.

Les figures 2.5(a) et 2.5(b) illustrent l'estimation de mouvement texture obtenu par maillage actif [Marquant et al. 00]. Le maillage est initialisé sur l'image 0 (figure 2.5(a)) de la séquence Foreman et est suivi jusqu'à l'image 8 (figure 2.5(b)). La figure 2.6(a) montre le contour de l'image 0 et celui de l'image 8. La figure 2.6(b) montre le contour de l'image 8 déplacée au temps de référence 0, grâce au mouvement texture (celui estimé par le maillage) et le contour de l'image 0. On peut constater que les contours sont plus stables temporellement si tous les contours sont projetés dans le même référentiel temporel.



(a) Maillage sur l'image 0 (b) Maillage sur l'image 8

FIG. 2.5 – *Estimation du mouvement texture par maillage entre l'image 0 et l'image 8 de la séquence Foreman*



(a) C_8 (gris) et C_0 (noir) (b) $C_8^{déplacé}$ (gris) et C_0 (noir)

FIG. 2.6 – *Contour de l'image 0 et de l'image 8, pour le visage de la séquence Foreman, avec ou sans compensation de mouvement texture*

Mise en correspondance de deux contours consécutifs

L'étape de mise en correspondance de deux contours consiste à trouver une relation de correspondance entre des points d'un contour C_t et des points d'un contour C_{t+1}. Cette relation de correspondance peut être obtenue par une technique de programmation dynamique ou bien par la technique que nous présentons ici.

On définit tout d'abord la relation de correspondance unidirectionnelle par une fonction injective $Map_{t\to t+1}$ qui associe à tous les points du contour t un point du contour $t+1$. Cette fonction est obtenue en parcourant simultanément les deux contours C_t et C_{t+1} et en déterminant, pour chaque point $P1$ ($P1 \in C_t$), le point P_2 ($P2 \in C_{t+1}$) qui est à une distance Euclidienne minimale c'est-à-dire tel que :

$$P_2 = arg \min_{\{P \in C_{t+1}\}} dist(P1, P).$$

On définit alors la correspondance bidirectionnelle Map qui est déduite des deux fonctions de correspondance unidirectionnelle ($Map_{t\to t+1}$ et $Map_{t+1\to t}$) de sorte que l'on ait en correspondance tous les points $(P1, P2)$ tels que :

$$P1 = Map_{t+1\to t}(P2) \quad et \quad P2 = Map_{t\to t+1}(P1).$$

Comme on peut le constater, la relation de correspondance bidirectionnelle Map ne lie pas tous les points des contours C_t et C_{t+1}. On va donc ajouter quelques « liens » c'est-à-dire quelques couples de points à la relation de correspondance Map (voir figure 2.7). Les liens sont ajoutés entre les points les plus proches des deux contours à condition qu'il n'y ait pas de zones de rupture. On construit donc une nouvelle relation de correspondance $MapA$, de sorte qu'entre deux « liens » il reste au plus un seul contour ayant des points non liés. La relation $MapA$ est la relation de correspondance finale.

Sur la figure 2.7, les contours C_t et C_{t+1} sont représentés par deux graphes orientés. Les indices i sont les points du contour C_t et les indices j sont les points du contour C_{t+1}. La figure représente deux cas pouvant survenir lorsque l'on veut enrichir la relation de correspondance bidirectionnelle Map. Ainsi, l'ajout de « liens » à la relation de correspondance bidirectionnelle Map est réalisé lorsque cela est possible. On peut remarquer que le cas des contours ouverts (lorsqu'il y a une « rupture ») n'autorise pas l'ajout de liens.

Principe de l'Alignement du groupe de contour et du sur-échantillonnage

Après mise en correspondance deux à deux des contours, il est nécessaire de sur-échantillonner l'ensemble des contours pour pouvoir avoir une correspondance bijective entre tous les points de tous les contours. On veut donc avoir un alignement global du groupe de contours. Pour cela, des points « virtuels » vont être ajoutés sur chaque contour.

Pour résoudre ce problème d'insertion de points « virtuels », on introduit la notion d'abscisse universelle (l'algorithme d'obtention de l'abscisse universelle est expliqué dans la sous-section suivante). C'est un nombre entier donné à l'ensemble des points et qui permet d'identifier les trajectoires (points alignés temporellement). En effet, l'abscisse universelle est identique pour tous les points qui appartiennent à la même trajectoire.

Cette abscisse universelle permet, une fois calculée, d'aligner le groupe de contour c'est-à-dire d'insérer des points virtuels pour compléter les trajectoires. La figure 2.8 illustre

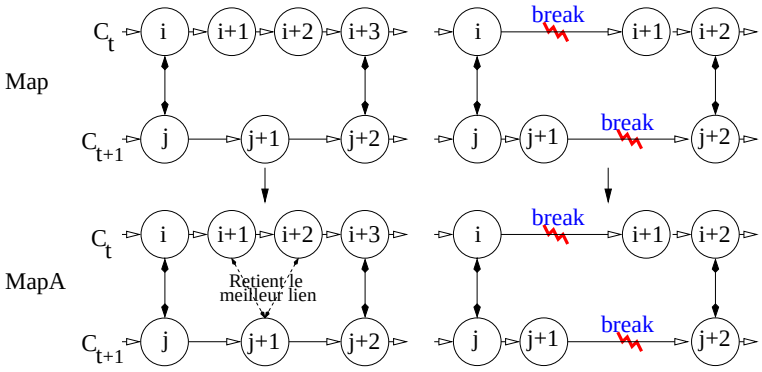


FIG. 2.7 – *Relation de correspondance MapA entre deux contours consécutifs (avant et après l'ajout de «liens» la relation Map)*

le résultat du calcul de l'abscisse universelle. Chaque point possède une valeur d'abscisse universelle. On peut constater sur le schéma que la trajectoire d'abscisse universelle 223 est incomplète. En effet, les contours 0, 1 et 2 n'ont pas de points pour cette abscisse universelle. Il y aura donc à ajouter des points virtuels pour compléter la trajectoire d'abcisse universelle 223.

Ainsi, une fois que chaque contour possède une correspondance vers l'abscisse universelle, les points « virtuels » sont ajoutés partout où la valeur de l'abscisse universelle manque (voir figure 2.9).

Dans le cas où l'on n'est pas dans une zone de « rupture », on définit tout de suite les positions de ces points « virtuels ». En effet, il suffit tout simplement de positionner les points virtuels à la même position que le point de gauche ou de droite (le point de gauche ou de droite étant un voisin spatiale non « virtuel »). On effectue ce choix pour ne pas ajouter des positions inexistantes.

S'il n'y a aucune zones de « rupture », on dispose alors de toutes les positions pour tous les contours. On obtient donc immédiatement les deux plans spatio-temporels de la figure 2.14. Ces deux plans spatio-temporels donnent la position (x, y) d'un point du contour de l'image t sachant l'abscisse universel s sur ce contour.

Dans le cas où l'on est dans une zone de « rupture », c'est le prolongement spatio-temporel (sous-section « Prolongement spatio-temporel des contours ouverts ») qui est utilisé pour positionner les points « virtuels ».

Principe d'obtention de l'abscisse universelle

Pour obtenir l'abscisse universelle associée à chaque point du groupe de contours, on représente chaque contour par un graphe valué (voir figure 2.10). Un nœud correspond à un point du contour, et un arc représente le passage d'un point au point suivant sur le contour. La valeur portée par un arc reliant deux nœuds correspond à l'incrément sur l'abscisse universelle. Ainsi, initialement, tous les arcs de tous les contours sont valués à 1.

De plus, on ajoute des arcs non valués entre les graphes de contours consécutifs tem-

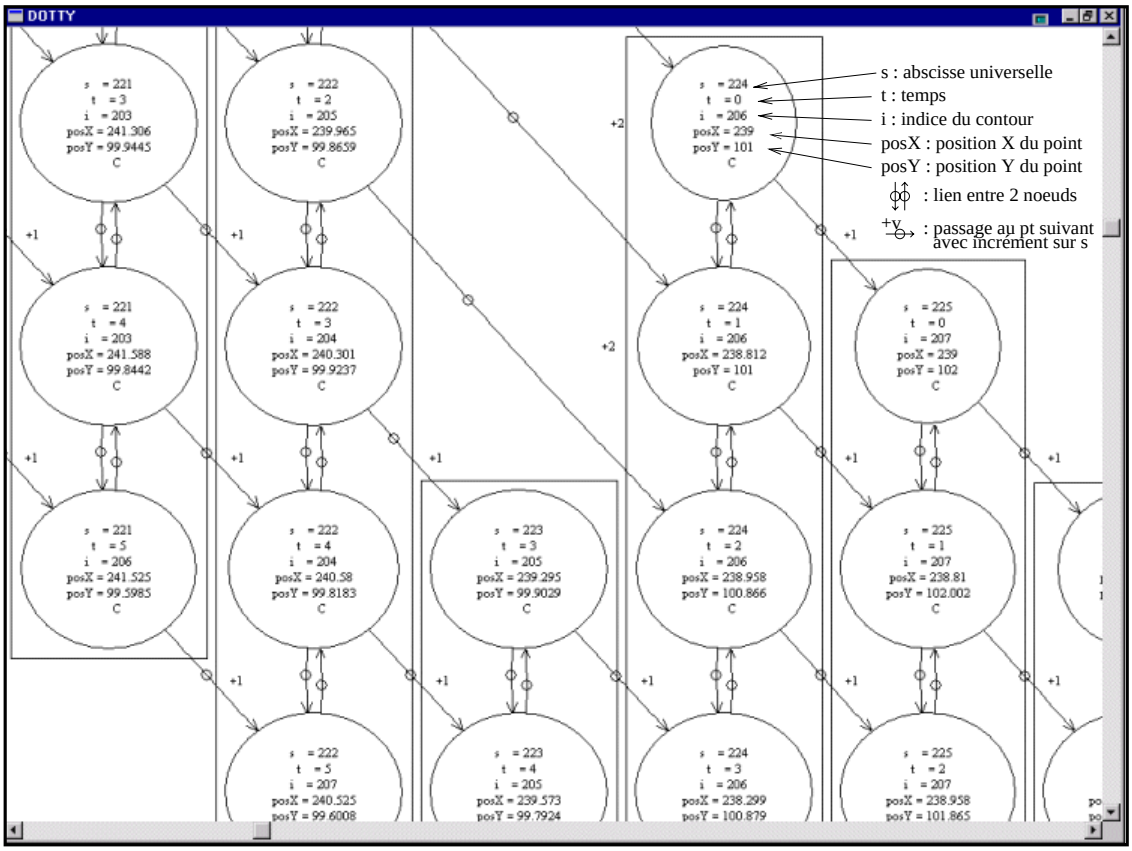


FIG. 2.8 – *Visualisation sous dotty d'une partie du graphe résultant de l'alignement du groupe de contour. Chaque point de chaque contour possède une abscisse universelle*

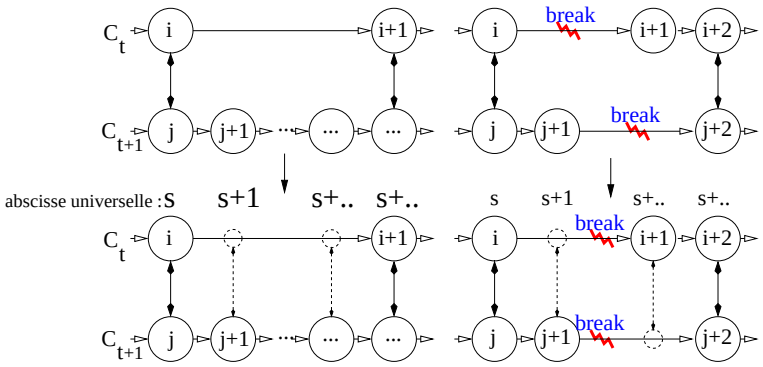


FIG. 2.9 – *Alignement du groupe de contours par le calcul d'une abscisse universelle. La notion d'abscisse universelle permet d'ajouter des points «virtuels» (sur-échantillonage). Les points «virtuels» sont représentés par les cercles en pointillés*

porellement. Ces arcs non valués correspondent aux points mis en correspondance entre contours consécutifs (relation $MapA$ de la section 2.3.1). On appelle « lien » les arcs entre les graphes consécutifs temporellement.

On dispose maintenant d'une représentation du groupe de contours sur laquelle on va modifier les valeurs des arcs valués. Ainsi, plutôt que de raisonner sur une abscisse universelle on va plutôt raisonner sur l'incrément sur l'abscisse universelle.

On part alors du constat suivant : entre deux « liens » consécutifs, la somme des incréments d'abscisse universelle **doit être** la même sur les deux contours mis en jeu.

On raisonne alors en deux passes. La première passe consiste à mettre à jour les arcs dans le cas où il y a au plus une rupture entre deux liens. La deuxième passe consiste à mettre à jour les arcs où il y a deux ruptures entre deux liens (une rupture sur chaque contour).

Cas où il y a au plus une rupture entre deux liens

La figure 2.10 représentent deux contours (deux graphes valués) qui sont « liés ». La somme des incréments sur l'abscisse universelle sur le contour C_t est de $+dI$ et sur le contour C_{t+1} de $+dJ$. L'algorithme doit donc faire en sorte qu'après mise à jour $dI = dJ = max$, avec $max = max(dI, dJ)$.

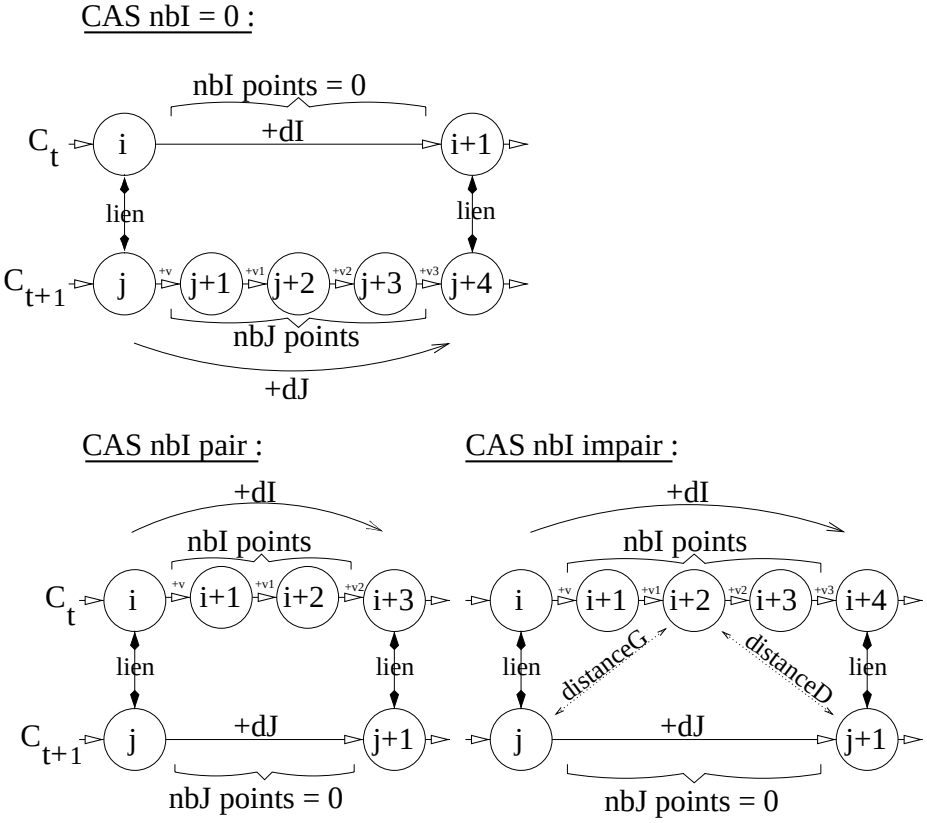


FIG. 2.10 – *Deux contours «liés» avec différents cas de figure pour le nombre de points nbI*

L'algorithme consiste donc à mettre à jour les valeurs des arcs en traitant à chaque

SI $(dI < dJ)$
  SI $(nbI == 0)$
    la valeur de l'arc $(i, i+1)$ est mise à la valeur dJ
  SINON //c'est nbJ qui vaut 0
    SI (il y a une rupture sur C_t)
      on ajoute à la valeur de l'arc rupture de C_t la valeur $dJ - dI$
    SINON
      SI (nbI est pair)
        on ajoute à la valeur de l'arc milieu de C_t la valeur $dJ - dI$
      SINON
        SI $(distanceG <= distanceD)$
          on ajoute à l'arc milieu droit de C_t la valeur $dJ - dI$
        SINON $//distanceG > distanceD$
          on ajoute à l'arc milieu gauche de C_t la valeur $dJ - dI$

ALG 2.1: Règle de mise à jour des arcs des graphes lorsqu'il y a au plus un seul contour en «rupture» entre deux «liens». Voir la figure 2.10 pour comprendre les notations employées

fois seulement deux contours consécutifs temporellement. Le schéma traite ainsi chaque groupe de deux contours et va du premier contour jusqu'au dernier puis du dernier jusqu'au premier. La règle de mise à jour pour un groupe de deux contours, entre deux « liens » et lorsqu'il y a au plus une rupture sur C_t ou C_{t+1} est donnée par l'algorithme 2.1. Il est facile de déduire la deuxième règle (cas où l'on a : $dI > dJ$).

Cas où il y a deux ruptures entre deux liens (une rupture sur chaque contour)

Le cas où on a une rupture entre deux liens à la fois sur C_t et C_{t+1} est traité dans une deuxième passe. On raisonne ici sur tous les contours en même temps et ceci entre la trajectoire à gauche et la trajectoire à droite de la rupture. On cherche alors le coût pour chaque contour à gauche et à droite de la rupture. Nous conservons le coût maximum à gauche et à droite. La somme de ces deux coûts plus 1 donne l'incrément de passage que chaque contour doit posséder. Il est alors assez simple de mettre à jour les valeurs des arcs de rupture de sorte que l'on respecte la somme représentant l'incrément de passage. On peut remarquer que s'il existe un des contours qui n'a pas de rupture, c'est celui-ci qui donne la valeur d'incrément de passage.

Prolongement spatio-temporel des contours ouverts

Chaque point du groupe de contours est indicé par l'abscisse universelle. Or il est possible que certains contours ne soient pas fermés. Ainsi, dans les zones de « rupture », il y a un saut d'abscisse universelle. On va donc prolonger spatio-temporellement les contours dans les zones de « rupture » de sorte que l'on puisse fermer les contours.

Dans un premier temps, nous ajoutons des points « virtuels » pour fermer les contours (le nombre de points ajoutés est fonction de la distance entre les points extrémités de la « rupture »). La figure 2.11 montre l'ajout de points « virtuels » quand il y a une « rupture » de contour. Les points « virtuels » seront représentés par l'ensemble Ω_{Out} tandis que les

points originaux seront représentés par l'ensemble Ω_{In}.

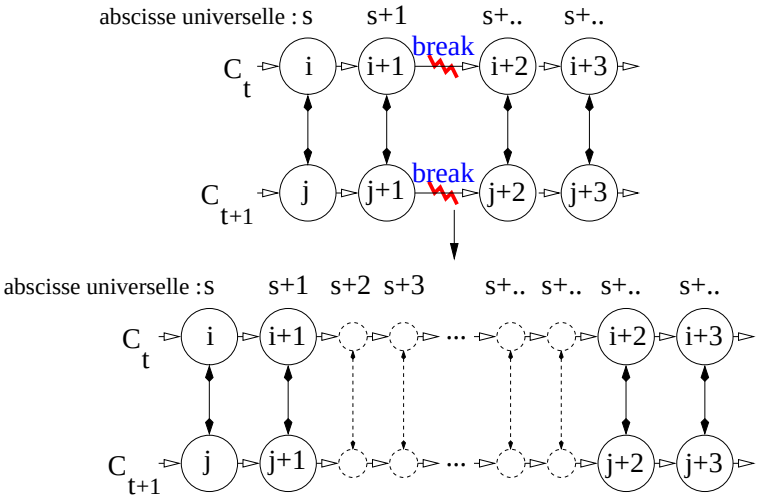


FIG. 2.11 – *Ajout de point «virtuels» pour fermer les contours ouverts*

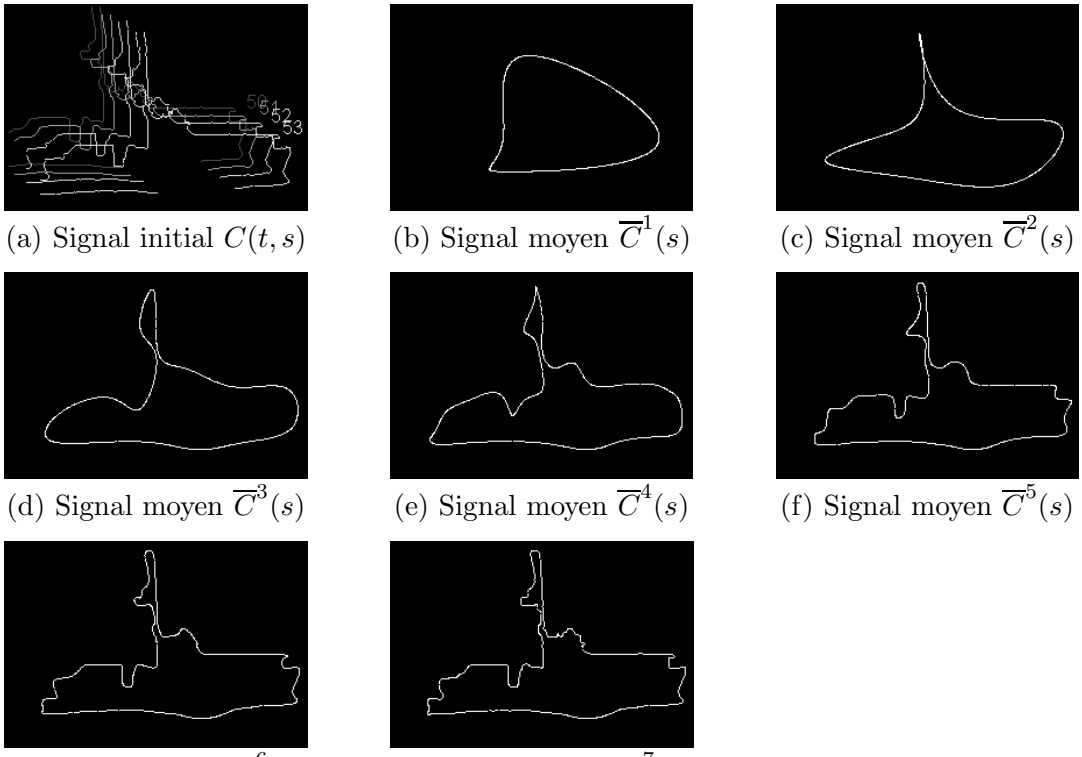
Dans un second temps, nous définissons les positions des points « virtuels » via le calcul du prolongement de contour. Pour prolonger un signal (pour nous, un groupe de contours consécutifs appartenant au même objet), l'idée est de trouver un signal paramétrique $\overline{C}^L(s)$ qui est identique là où le signal est défini et qui est une extension lisse ailleurs. Un contour peut ainsi être modélisé paramétriquement par une combinaison linéaire de fonctions finies $C(s) = \sum_{k=1}^{k=K} c_k \phi_k(s)$ (e.g. B-splines), s étant l'abscisse universelle. Cette représentation peut être généralisée pour représenter l'évolution d'un contour avec $C(t,s) = \sum_{k=1}^{k=K} c_k \phi_k(t,s)$, t étant le temps. Cependant, le signal $\overline{C}^L(s)$ que nous recherchons n'a pas besoin d'être trop dépendant du temps, puisque les contours sont rendus suffisamment stables temporellement par la compensation en mouvement des contours. De plus, afin de proposer une extension lisse, nous nous intéressons plus particulièrement à une représentation hiérarchique de $\overline{C}^L(s)$:

$$\begin{aligned} \overline{C}^L(s) &= \sum_{l=1}^{L} \Delta C^l(s), \\ \Delta C^l(s) &= \sum_{k=1}^{k=2^{l+1}} \delta c_k^l \phi_k^l(s), \end{aligned}$$

avec ϕ_k^l les fonctions multi-échelles, et δc_k^l les coefficients à estimer.

On exploite la représentation hiérarchique de $\overline{C}^L(s)$ en proposant un calcul itératif consistant à raffiner le signal par une succession de phases d'analyse et de synthèse aux différentes résolutions. Ce choix permet d'avoir une extension des contours qui soit peu coûteuse à coder. La figure 2.12 montre l'évolution du signal moyen $\overline{C}^L(s)$ lors des itérations succesives.

La phase de synthèse permet de mettre à jour le signal moyen $\overline{C}^l(s)$ pour un niveau l grâce au signal de raffinement $\Delta C^l(s)$. Le signal de raffinement $\Delta C^l(s)$ est trouvé durant la phase d'analyse par calcul des coefficients incrémentaux δc_k^l. La mise à jour du signal



(a) Signal initial $C(t,s)$ (b) Signal moyen $\overline{C}^1(s)$ (c) Signal moyen $\overline{C}^2(s)$

(d) Signal moyen $\overline{C}^3(s)$ (e) Signal moyen $\overline{C}^4(s)$ (f) Signal moyen $\overline{C}^5(s)$

(g) Signal moyen $\overline{C}^6(s)$ (h) Signal moyen $\overline{C}^7(s)$

FIG. 2.12 – *Signal moyen $\overline{C}^L(s)$ à différents niveaux L*

moyen est effectuée sur les deux ensembles Ω_{In} et Ω_{Out} :

$$\begin{aligned} \overline{C}^0(s) &= 0, \\ \overline{C}^l(s) &= \overline{C}^{l-1}(s) + \Delta C^l(s), \end{aligned}$$

et on définit un résidu sur Ω_{In} :

$$\begin{aligned} Res^0(t,s) &= C(t,s), \\ Res^l(t,s) &= C(t,s) - \overline{C}^l(s). \end{aligned}$$

La phase d'analyse a pour objectif de trouver le signal de raffinement $\Delta C^l(s)$ qui représente le mieux le résidu Res^{l-1} du niveau $l-1$.

Sur l'ensemble Ω_{In}, on minimise la différence entre le signal de raffinement $\Delta C^l(s)$ et le résidu $Res^{l-1}(t,s)$ du niveau $l-1$ (terme $\epsilon(t,s)$), ce qui est équivalent à minimiser la différence entre le signal original $C(t,s)$ et le signal moyen $\overline{C}^l(s)$ du niveau l.

Sur l'ensemble Ω_{Out}, on cherche une extension lisse du signal de raffinement $\Delta C^l(s)$ en introduisant un terme de pénalisation sur la valeur du résidu Res^{l-1}, pondéré par λ. Ce terme est aussi utile pour éviter les phénomènes d'oscillations aux bords de Ω_{In}. Les coefficients incrémentaux δc_k^l sont obtenus par la minimisation sur le groupe de T images de :

$$E = \sum_{t=1}^{t=T} \left[\sum_{s \in \Omega_{In}(t)} \varepsilon(t,s)^2 + \sum_{s \in \Omega_{Out}(t)} \lambda \| Res^{l-1}(t,s) \|^2 \right]$$
$$avec : \quad \varepsilon(t,s) = \| \Delta C^l(s) - Res^{l-1}(t,s) \|.$$

Une fois que le signal $\overline{C}^L$ est trouvé, le groupe de contours est facilement prolongé (voir figure 2.13).

(a) Avant prolongement (b) Après prolongement

FIG. 2.13 – *Illustration du prolongement spatio-temporel sur le contour bateau de l'image 50 de coastguard*

2.3.2 Codage spatio-temporel du contour

Le schéma IPB

Une fois que les contours ont été alignés et prolongés, nous possédons une surface d'évolution du contour (les deux plans spatio-temporels) paramétrée par s (l'abscisse universelle) et t (le temps) (voir figure 2.14). Cette surface représente un groupe de contours consécutifs fermés.

Deux méthodes de paramétrisation sont comparées pour encoder la surface spatio-temporelle du contour. La première est basée sur les B-splines et la seconde sur une décomposition en ondelettes. De plus, les contours sont codés en utilisant un schéma de type IPB. Le premier contour sera codé en intra (I) et les autres seront codés en utilisant une simple prédiction (P) ou une prédiction bidirectionnelle (B). Une seule image B est insérée entre deux images I ou P. Dans le cas des ondelettes, le schéma doit être effectué en boucle ouverte si l'on désire obtenir une hiérarchisation totale du flux.

Les coefficients sont codés avec un codeur arithmétique en plans de bits. Pour le codage par ondelettes, la quantification avec perte est obtenue en choisissant le nombre de plans de bits encodés. Pour le codage par B-spline, la quantification est effectuée sur les coefficients, puis ceux-ci sont représentés en différentiels (DPCM/MICD) et enfin codés via le codeur arithmétique en plans de bits (ici, tous les plans de bits sont codés).

Le codage en plan de bits consiste simplement à coder plage par plage (plan par plan) en allant des bits plus significatif jusqu'aux bits les moins significatifs. Le schéma 2.15 illustre une décomposition ondelette en sous-bandes et les coefficients sont représentés de

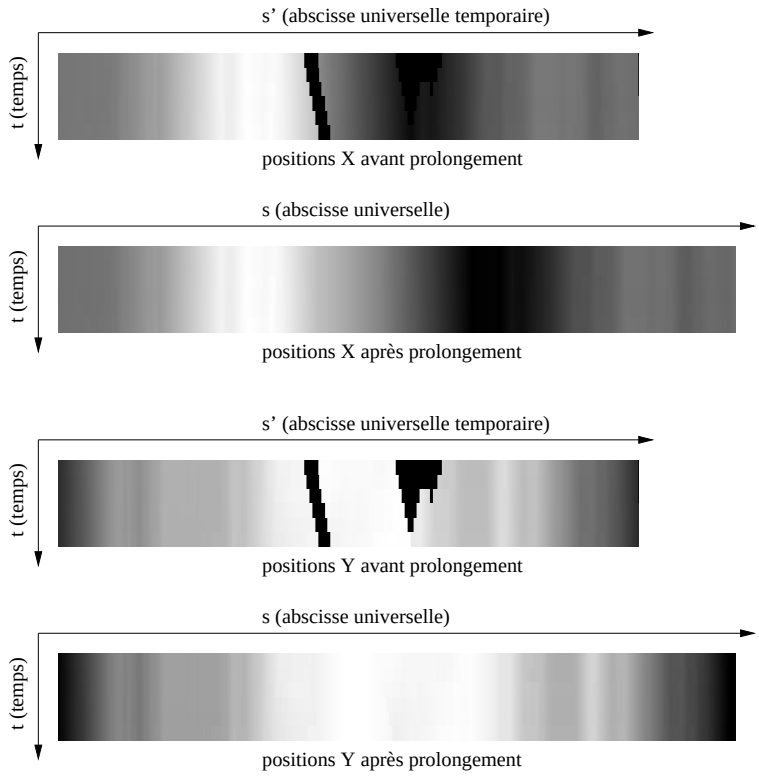


FIG. 2.14 – *Plans spatio-temporels pour les positions X et Y. Les zones noires représentent les «ruptures» de contour présentes avant le prolongement de contour*

manière à illustrer la notion de plage de codage ou plans de bits, et de bits significatifs et non significatifs. Le codage en plan de bits consiste donc à remettre en forme le message à coder de sorte qu'il soit bien comprimé. De plus, le flux généré a des propriétés de progressivité puisque la troncature du train de bits fait perdre uniquement les bits les moins significatifs.

Le codage arithmétique en plans de bits des sous-bandes issu d'une décomposition ondelette est un codage par plan de bits avec l'utilisation d'un codeur arithmétique. On insère à l'intérieur du flux des informations indiquant si pour un plan donnée, la sous-bande traitée vient d'apparaître ou n'est jamais apparue dans les plans précédent. Ainsi, cette information est équivalente à un routage ou un parcours d'arbre indiquant si une sous-bande est ou non active. On ajoute aussi l'information de fin de plan indiquant qu'il n'est pas nécessaire de coder les sous-bandes suivantes du plan puisqu'elle ne sont pas significatives. On peut remarquer que ce principe de codage est celui utilisé dans SPIHT [Said et al. 96].

Nous pouvons remarquer que nous n'avons pas retenu un codage arithmétique en plan de bits utilisant un critère debit-distorsion. En effet, on aurait pu coder par paquets les coefficients choisis par optimisation (chaque paquet contient l'information sur l'état d'avancement dans les plans de bits pour chaque sous-bande), comme cela est fait dans

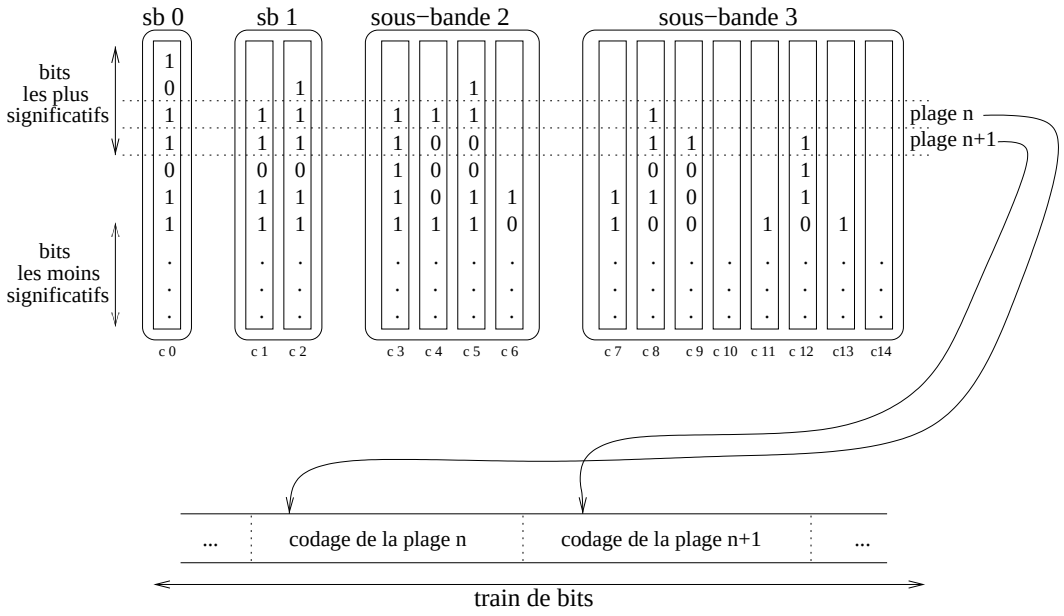


FIG. 2.15 – *Illustration du codage en plan de bits*

EBCOT [Taubman 00]. Cependant, ce choix implique un coût de description des paquets (état d'avancement dans les plans de bits) qui est fort comparé au coût des données. Ceci est lié au fait que nous manipulons un petit nombre de données. La technique de quantification uniforme (par codage arithmétique en plans de bits des sous-bandes) s'avère donc ici plus performante en terme d'optimisation débit-distorsion.

La représentation en B-splines

Un contour peut être exprimé sous forme d'une B-spline par :

$$B(u) = \sum_{k=1}^{k=K} \phi(u - u_k).P_k,$$

où les P_k sont les points de contrôle, les ϕ sont les noyaux d'une B-spline, u est l'abscisse curviligne et u_k est l'abscisse curviligne correspondant au P_k.

Les points de contrôle P_k sont calculés de sorte que la B-spline représente au mieux le contour original C. On les obtient en minimisant l'expression E :

$$\begin{aligned} E &= \sum_{s=1}^{s=S} \|B(u_s) - C(s)\|^2 \\ &= \sum_{s=1}^{s=S} \| \sum_{k=1}^{k=K} (\phi(u_s - u_k).P_k) - C(s)\|^2, \end{aligned}$$

avec s l'abscisse universelle.

Cette minimisation, via annulation des dérivées, mène au système linéaire creux (A.X=B) qui suit, et peut être résolue de manière efficace par un outil classique d'algèbre linéaire (par exemple par gradient conjugué). On obtient alors les points de contrôle Intra :

$$\begin{aligned}
\frac{\partial \left(\sum_{s=1}^{s=S}\sum_{l=1}^{l=K}(\phi(u_s-u_l).P_l - C(s))^2\right)}{\partial P_k} &= \sum_{s=1}^{s=S} 2\phi(u_s-u_k)\left(\sum_{l=1}^{l=K}(\phi(u_s-u_l).P_l) - C(s)\right) \\
\Rightarrow \quad \sum_{s=1}^{s=S}\sum_{l=1}^{l=K}\phi(u_s-u_k)\phi(u_s-u_l).P_l &= \sum_{s=1}^{s=S}\phi(u_s-u_k).C(s) \\
\Rightarrow \quad \sum_{l=1}^{l=K}\underbrace{\left(\sum_{s=1}^{s=S}\phi(u_s-u_k)\phi(u_s-u_l)\right)}_{A_{k,l}}.P_l &= \underbrace{\sum_{s=1}^{s=S}\phi(u_s-u_k).C(s)}_{B_l}.
\end{aligned}$$

On a retenu un schéma de codage de type IPB. Pour bénéficier des propriétés d'alignement temporel des deux plans spatio-temporels, nous allons faire en sorte de calculer par minimisation les déplacements ΔP_k des points de contrôle P_k entre deux contour. Ainsi, lorsque l'on dispose des points de contrôle P_k d'un premier contour, nous calculons les déplacements ΔP_k permettant de définir les points de contrôle d'un second contour. Ce sont alors les valeurs de déplacements ΔP_k qui seront codées pour ce second contour.

Ainsi, on calcule le déplacement des points de contrôle ΔP_k d'un contour (contour Prédit), sachant que l'on connaît les points de contrôle P_k d'un premier contour (contour Intra), par la minimisation de l'expression suivante :

$$E = \sum_{s=1}^{s=S} \| \sum_{k=1}^{k=K}(\phi(u_s-u_k).(P_k+\Delta P_k)) - C(s)\|^2.$$

Ce sont les valeurs de déplacement ΔP_k qui seront codées.

La représentation en ondelettes

Afin d'avoir une représentation hiérarchique d'un groupe de contours et de fournir une scalabilité, on propose une décomposition en ondelettes dyadiques. Avant de réaliser la transformation, nous ré-échantillonnons le groupe de contours (ré-échantillonnage des deux plans spatio-temporels) pour avoir une longueur égale à un multiple d'une puissance de 2. Ce ré-échantillonnage permet d'effectuer une suite de décompositions 1D circulaires jusqu'à obtenir un seul coefficient basse fréquence. On peut facilement effectuer cette décomposition puisque le signal est circulaire (les contours sont fermés) et puisque le signal possède une longueur en puissance de 2. Pour cette décomposition, on utilise une décomposition avec les filtres 9/7 de Daubechies [Antonini et al. 92].

Le schéma 2.16 illustre la forme obtenue lors de la décomposition ondelettes en sous-bandes pour une même ligne sur les deux plans spatio-temporel avec différents nombre de plans de bits. On décode une composante X ou Y d'un contour en utilisant un nombre donné de plan de bits (les images ont été réduite en résolution spatial en fonction du nombre de plan de bits pour illustrer la hiérarchie des informations).

FIG. 2.16 – *Illustration du codage en plan de bits avec différents nombre de plans de bits*

2.4 Résumé du chapitre

Ce chapitre traite la notion de hiérarchisation totale. Deux notions importantes sont introduites : la décorrélation de la texture, du mouvement et de la forme puis la hiérarchisation.

La décorrélation de la texture, du mouvement et de la forme est rendue possible grâce à l'utilisation de l'outil de maillage ainsi qu'à l'utilisation de prolongement de texture. Cette décorrélation permet de coder séparément chacune de ces informations permettant ainsi de répartir plus aisément les débits, c'est-à-dire de donner proportionnellement plus de débit à l'information de texture. L'hypothèse sous-jacente est que le système visuel humain est moins sensible aux distorsions sur le mouvement et la forme que sur la texture.

La hiérarchisation des trois informations mouvement, texture et forme est obtenue grâce à l'indépendance des trois informations mais aussi parce que nous utilisons des représentations mouvement, texture et forme qui sont long terme. Le fait que nous ayons des représentation long terme permet d'exploiter la redondance temporelle et ainsi rend efficace l'utilisation de transformées ondelettes qui permettent d'obtenir un flux hiérarchique.

Nous avons abordé dans ce chapitre plus spécifiquement le codage de contours s'inscrivant dans cette recherche de décorrélation et de hiérarchisation du flux vidéo objet. Le codage de contour que nous avons proposé repose sur la construction d'une représentation long terme d'un groupe de contours (plans spatio-temporels). Cette représentation permet alors aisément de coder un groupe de contours de manière hiérarchique grâce à l'utilisation de transformées ondelettes.

Le chapitre suivant illustre et commente les résultats de codage de contour et de codage objets hiérarchique.

Chapitre 3

Présentation des résultats : codage objet et codage de contour

Ce chapitre donne des résultats sur le codage de contour avec perte que nous proposons. Les résultats permettent d'évaluer la performance de notre approche. Ensuite nous abordons le codage objet par l'utilisation du codeur ondelette 3D objet.

3.1 Résultats du codage de contour

Cette section compare trois méthodes de codage de forme avec perte: l'approche MPEG4 CAE, notre schéma IPB avec une représentation B-spline et celui avec une transformée ondelette. Le débit correspond au débit moyen par élément de contour. La distorsion choisie une des mesures proposée par MPEG4 : $d_n = \frac{\text{nombre de pixels mal affectés}}{\text{nombre de pixels total du masque original}}$, et l'on observe sa moyenne sur un groupe d'images.

La figure 3.1 montre la distorsion en fonction du débit par élément de contour pour la séquence Foreman. Remarquerons tout d'abord que les résultats, pour un codage avec perte, ne présentent un intérêt que pour des débits inférieurs à 1,2 bits/élément de contour. En effet, un codage de Freeman avec codeur arithmétique permet d'atteindre ce débit sans perte.

En utilisant l'approche IPB B-spline ou ondelette, il est possible de descendre à des débits très bas (0,4 bits/élément de contour) alors que MPEG4 CAE est borné environ à 0,8 bits/élément de contour. De plus, comme illustré sur la figure 3.2, à très faible débit (autour de 0,7 bits/élément de contour), la qualité est toujours acceptable, ce qui n'est pas le cas pour MPEG4 CAE. Il faut noter ici que MPEG4 CAE est utilisé dans ses limites qu'il n'est pas vraiment prévu pour être utilisé dans de telles conditions.

Enfin, l'approche ondelette donne de meilleurs résultats que l'approche B-spline. Ceci peut être expliqué par le fait que la décorrelation spatiale est plus efficace avec les ondelettes. De plus, la quantification des coefficients ondelettes sélectionne automatiquement les coefficients les plus représentatifs tandis que pour l'approche B-spline, il est plus difficile de choisir la meilleure quantification avec le bon nombre de points de contrôle.

Les figures 3.3, 3.4, 3.5 ainsi que le tableau 3.1 illustrent le gain obtenu sur MPEG4, à faible débit, en utilisant notre schéma. À un niveau de distorsion acceptable, 30% à 60% de bits sont gagnés par rapport à MPEG4 CAE et ceci tout en offrant une représentation

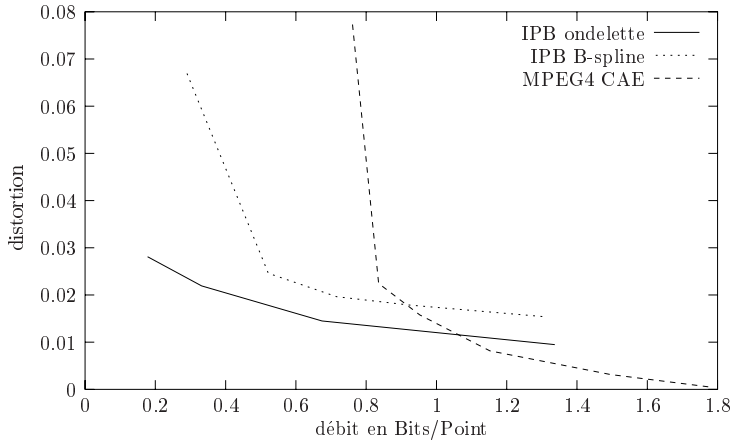


FIG. 3.1 – *Distorsion en fonction du débit pour la séquence Foreman*

progressive.

Séquence	MPEG4 CAE	IPB Ondelette	Gain (bits)
Children	D=0,058 412 bits/fr	D=0,054 318 bits/fr	30%
Coastguard (grand bateau)	D=0,066 302 bits/fr	D=0,064 153 bits/fr	49%
Foreman	D=0,022 673 bits/fr	D=0,022 267 bits/fr	60%

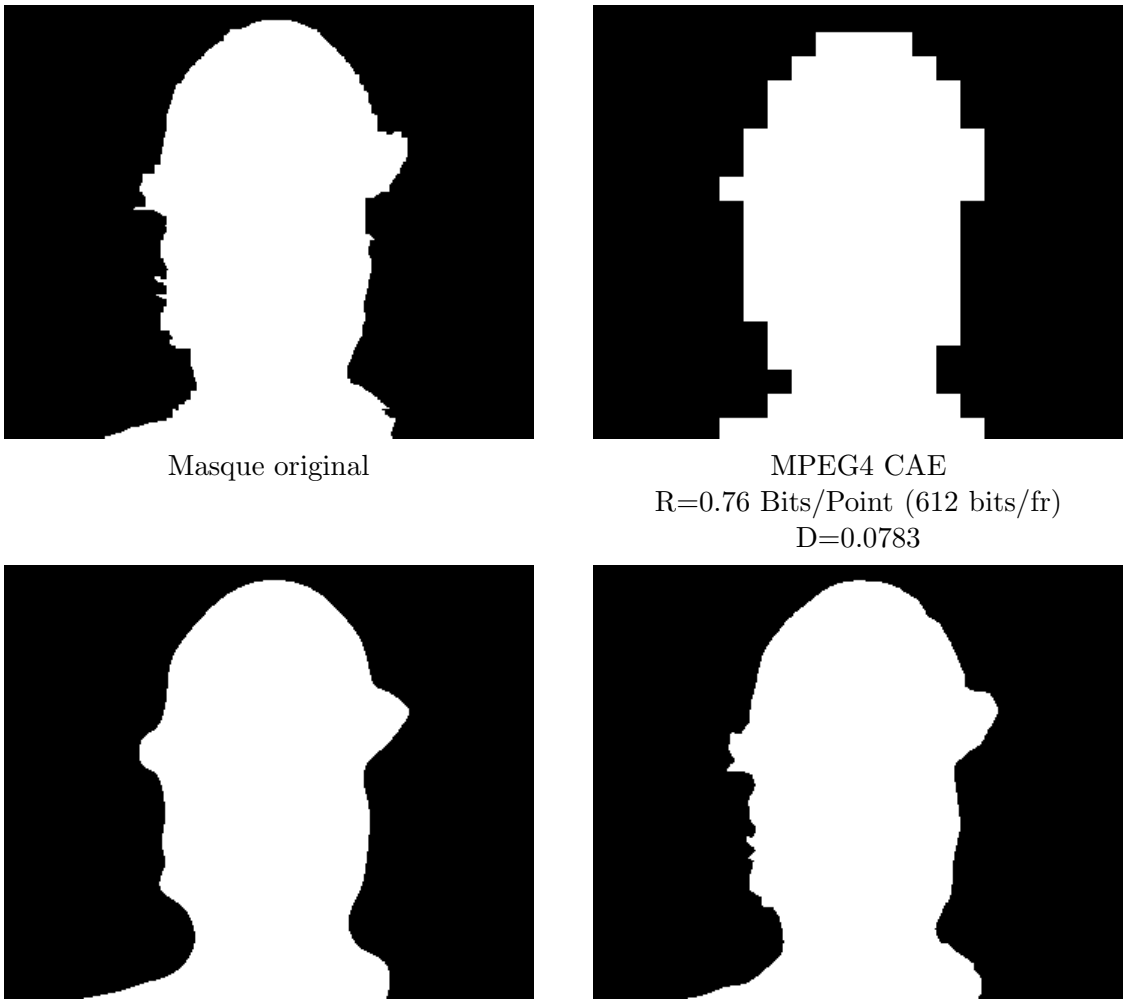
TAB. 3.1 – *Gain de l'approche IPB ondelette sur MPEG4*

La figure 3.6 illustre l'aspect hiérarchique de la représentation ainsi que l'effet visuel du codage de contour avec perte. Le VOP de la figure 3.6(c) est obtenu en utilisant le masque de la figure 3.2, codé-décodé en IPB Ondelette (Q=8). Le VOP de la figure 3.6(d) quant à lui utilise le masque de la figure 3.5 codé-décodé en IPB Ondelette (Q=16). De manière générale, le codage de masque avec une quantification de Q=8 donne des résultats supérieurs à MPEG4. De plus, avec une quantification de Q=8, le léger lissage obtenu semble rendre le contour plus agréable que l'original qui issu d'une segmentation manuelle par étiquetage de régions spatiales.

3.2 Réflexion sur le codage de contour

L'alignement d'un groupe de contours ainsi que la fermeture de contour par prolongement de contour permettent d'obtenir une structure en plans spatio-temporels qui présente de bonnes propriétées pour le codage. En effet, cette structure bénéficie d'une bonne stabilité temporelle. Les résultats de codage confirment que cette représentation permet d'obtenir de bons résultats de codage à faible débit.

La structure en plans spatio-temporels permet aussi de proposer aisément une hiérarchisation du flux. De plus, le codage proposé en IPB est assez robuste aux pertes d'image



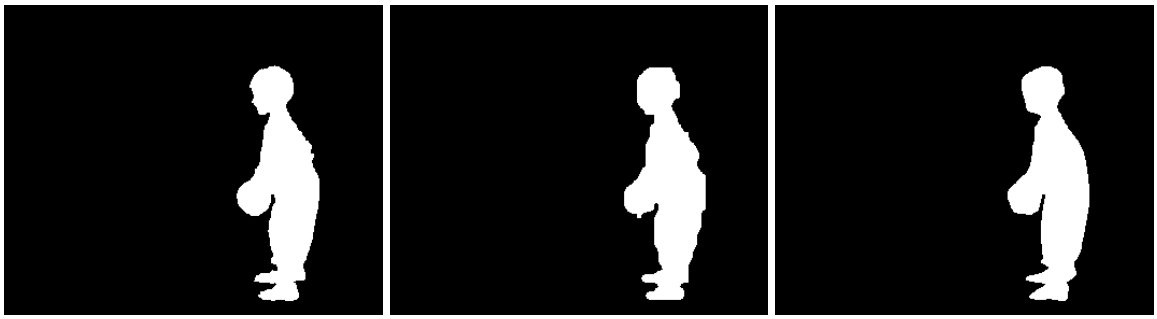
Masque original

MPEG4 CAE
R=0.76 Bits/Point (612 bits/fr)
D=0.0783

IPB B-spline
R=0.72 Bits/Point (576 bits/fr)
D=0.0196

IPB Ondelette
R=0.67 Bits/Point (543 bits/fr)
D=0.0145

FIG. 3.2 – *Comparaison des techniques de codage pour un débit d'environ 0.7 bits par élément de contour (environs 600 bits par images à 15Hz)*



Masque image 0 children | MPEG4 CAE R=412 bits/im D=0.058 | IPB Ondelette R=318 bits/im D=0.054

FIG. 3.3 – *Comparaison MPEG4 CAE et IPB Ondelette à distorsion pratiquement égale pour la séquence Children sur le GOP [5-10]*

Masque 50 coastguard | Masque extérieur

Masque prolongé (La distorsion est calculée par rapport à ce masque) | MPEG4 CAE R=302 bits/im D=0.066 | IPB Ondelette R=153 bits/im D=0.066

FIG. 3.4 – *Comparaison MPEG4 CAE et IPB Ondelette à distorsion égale sur le GOP [50-55]*

B puisque les prédictions bidirectionnelles d'images sont très proches des images B.

La section suivante présente les résultats du codage objet hiérarchique par ondelettes 3D objet incluant notre schéma de codage de contour.

3.3 Résultat de codage objet hiérarchique

L'objectif de cette section est d'illustrer l'intérêt de la décorrélation des trois informations de texture, de mouvement et de forme ainsi que du codage ondelette 3D objet. L'objectif est aussi de signifier qu'il est possible d'obtenir un codage totalement hiérarchique

La première comparaison que nous pouvons effectuer est celle du codage ondelette 3D non objet et du codage ondelette 3D objet. La figure 3.7 montre le résultat du codage de la séquence Foreman avec et sans objet. Le PSNR que nous mesurons est effectué dans le domaine texture c'est-à-dire que la mesure est faite non pas entre la séquence reconstruite et la séquence originale mais entre la texture codée et la texture originale. En effet, on suppose qu'une faible distorsion sur le mouvement et la forme a peu d'impact sur le SVH. Ainsi, on ne conserve que la mesure de distorsion de texture.

Que cela soit pour le codage ondelette 3D objet ou non objet, les résultats sont sensiblement équivalents pour un débit de 160Kb/s avec un léger avantage (en PSNR texture) pour le codage par ondelette 3D non objet. Cependant, cette comparaison ne prend pas en compte la distorsion géométrique sur le mouvement (seul la distorsion texture est mesurée). Ainsi, si l'on regarde le résultat de la séquence reconstruite avec l'approche non objet, les artefacts dûs aux étirements de mailles dans les zones de recouvrement et de découvrement sont particulièrement visibles. Ces artefacts ne sont pas présents dans l'approche objet. De plus, visuellement, le visage est de meilleur qualité dans l'approche objet (particulièrement sur la bouche du personnage).

L'avantage de l'approche objet est de permettre de régler les problèmes d'étirement de mailles dûs aux occultations entre objets. Elle permet aussi de choisir une taille de GOP variable par objets ce qui revient à s'adapter à l'activité de chaque objet. Cette possibilité est intéressante car plus le GOP est grand et plus l'efficacité du codage ondelette est grande. Pour l'exemple donné, le codage du fond est effectué sur des GOP de 30 images alors que le codage de l'avant-plan est effectué sur des GOP de 8 images.

Par ailleurs, le codage objet offre la possibilité de répartir les débits entre objets. Dans l'exemple du codage de la séquence Foreman, 83% du débit va au codage de la texture et du mouvement de l'objet avant-plan et seulement 13% du débit va au codage de la texture et du mouvement de l'objet arrière plan pour la même distorsion. Quant à la forme de l'objet en avant-plan, elle ne représente que 4% du débit total. Il est évident que la souplesse du codage objet (répartition des débits, taille du GOP par objet) permet de mieux adapter le codage au signal que l'on traite et donc d'être plus efficace.

Le schéma de codage objet hiérarchique que nous avons présenté permet d'obtenir des résultats à très faible débit (sous les 100 Kbit/s pour du CIF à 15Hz) autorisant la comparaison avec H26Lvm8. La séquence test que nous présentons ici est la séquence CIF Erik à 15 Hz.

La figure 3.8 montre qu'à faible débit, la qualité subjective du codage ondelette 3D avec deux objets est équivalente. Les artefacts sont différents : d'un coté H26Lvm8 produit des effets de blocs et un flou sur les textures et de l'autre l'ondelette 3D produit des rebonds.

Par contre, on bénéficie de fonctionnalités nouvelles :

- la hierarchisation objet,
- la hierarchisation du mouvement, de la forme et de la texture.

Le tableau 3.2 illustre les comparaisons de débit ainsi que la répartition du débit pour la forme, le mouvement et la texture entre H26L VM8.4 et notre approche hiérarchique. Pour le codage de mouvement nous avons fixé une distorsion maximum de 1/2 pixels. Pour le codage de contour, nous avons retenu une quantification de 8. On a alors pour le codage de la séquence Erik à 55 Kb/s une répartition du débit sur l'objet avant-plan de 66% pour la texture, 27% pour le mouvement et 7% pour la forme. On peut remarquer que bien que le coût de codage de la texture soit important, il n'en reste pas moins qu'à faible débit, le mouvement et la forme ont une proportion non négligeable. Cette remarque milite en faveur du codage avec perte sur le mouvement et sur la forme.

Comme on peut le constater, il manque une répartition des débits entre chacune des trois informations : mouvement, texture, forme. On voit bien qu'il faudrait un modèle psychovisuel humain pour définir quelle est l'influence de chacune des distorsions mouvement, texture et forme.

	Approche ondelette 3D objet			H26L	
Mouvement (kbs)	11.9	12.23	12.25		
Texture (kbs)	28.6	37.8	47.4		
Forme (kbs)	3	3	3		
Arrière-plan (kbs)	11.5	11.5	11.5		
Débit total (kbs)	55	64	74	52	70
PSNR(dB)	27.9	28.4	29	26.9	28.2

TAB. 3.2 – *Comparaison entre notre schéma hiérarchique et H26L VM8.4. La répartition des débits est donnée pour la séquence Erik entre l'avant-plan (mouvement, texture et forme) et l'arrière-plan. Le PSNR est calculé uniquement sur l'avant-plan rogné de quelques pixels*

Masque image 0 Foreman | MPEG4 CAE R=673 bits/im D=0.022 | IPB Ondelette R=267 bits/im D=0.022

FIG. 3.5 – *Comparaison MPEG4 CAE et IPB Ondelette à distorsion égale pour la séquence Foreman sur le GOP [0-5]*

(a) VOP avec masque original
(coût MPEG4 CAE sans perte = 1616 bits/fr)

(b) VOP avec masque à Q=4, R=1077bits/fr

(c) VOP avec masque à Q=8, R=543bits/fr

(d) VOP masque à Q=16, R=267bits/fr

(e) VOP avec masque à Q=32, R=143bits/fr

FIG. 3.6 – *VOP de l'image 0 de la séquence Foreman pour différents niveaux de perte sur le contour en utilisant le codage IPB Ondelette. Ces figures illustrent l'effet visuel dû au codage de forme avec perte. La quantification correspond à un nombre de plans de bits supprimés. Ainsi, une quantification de 4 correspond à la suppression des 2 derniers plans de bits, une quantification de 8 correspond à la suppression des 3 derniers plans de bits ...*

(b) Codage ondelette 3D non objet
R=160 Kb/s
$PSNRtext$=30,4

(c) Codage ondelette 3D objet
R_{fond}=21 Kb/s
R_{visage}=133 Kb/s
R_{forme}=6 Kb/s
R=160 Kb/s
$PSNRtext$=30,1

FIG. 3.7 – *Comparaison entre ondelette 3D objet et non objet à un débit de 160Kb/s sur la séquence Foreman CIF 15Hz*

(a) Image 20 Erik

(b) Codage ondelette 3D
R=55 Kb/s
$PSNR_{avant-plan}$=27,9

(c) Codage H26L VM 8
R=52 Kb/s
$PSNR_{avant-plan}$=26,9

FIG. 3.8 – *Comparaison entre H26L VM8 (2 images B, CABAC, 5 frames de référence, optimisation RD) et l'ondelette 3D objet (deux objets : avant-plan et fond) à un débit très faible. Image 20 de la séquence Erik CIF à 15Hz*

Chapitre 4

Conclusion de la deuxième partie

4.1 Conclusion

Dans cette partie, nous avons présenté la notion de codeur d'objet vidéo avec flux pleinement progressif (forme, mouvement, texture). L'indépendance de chacun des flux permet d'optimiser indépendamment chacune des 3 informations. Ainsi, en faisant l'hypothèse que le système psychovisuel humain est peu sensible aux faibles distorsions sur le mouvement et sur la forme, la décomposition en 3 flux hiérarchiques indépendants permet de gagner du débit sur l'information de mouvement et de forme au profit de l'information de texture.

Dans cette partie, nous avons aussi proposé une technique efficace de codage de forme s'ajoutant au codeur ondelette 3D. Celle-ci tire bénéfice de l'alignement de contours ainsi que du prolongement spatio-temporel. En utilisant un codage ondelette et un schéma IPB nous obtenons de meilleurs résultats que MPEG4 CAE à faible débit et grâce aux propriétés spatio-temporelles intrinsèques, l'aspect visuel est acceptable. De plus, les 2 plans spatio-temporels autorisent une hiérarchisation du flux codé et permettent d'avoir une certaine robustesse aux pertes de données sur un réseau.

4.2 Perspectives

Comme on l'a vu pour le codage hiérarchique, chacune des informations : forme, texture, mouvement peut être dégradée. Actuellement, lorsque nous codons à faible débit, le mouvement et la forme sont dégradés de manière fixe (la distorsion sur le mouvement est inférieure à 0.5 pixels et la quantification sur le contour est de Q=8). Il manque un modèle de vision humaine pour savoir s'il faut dégrader conjointement ces trois informations et si oui, comment les dégrader sachant une qualité ou bien un débit objectif.

Dans le même ordre d'idée, la métrique de distorsion pour les contours bien qu'étant couramment utilisée est peu représentative d'un niveau de distorsion visuelle. Il serait probablement plus intéressant de se fier à une mesure de type distance maximum entre le contour initial et le contour codé-décodé. Dans [Katsaggelos et al. 98] les évaluations subjectives menée sur des séquences CIF ont montré qu'une distance Euclidienne maximum de 1,4 permet de représenter de manière propre les objets dans les applications à faible débit. Pour l'instant, nous avons remarqué qu'une quantification de Q=8 rendait des

résultats satisfaisants pour les très faibles débits (Les figures 3.2, 3.3 et 3.4 sont obtenues pour le codage IPB Ondelette avec une quantification de 8).

La technique de codage de contour IPB ondelette peut être rendue plus performante en approfondissant la technique de prolongement de contour constant dans le temps par un prolongement variant dans le temps. La technique de mise en correspondance de deux contours qui est expliquée dans la section 2.3.1 peut mener à une répartition peu régulière et peu homogène des « liens ». Une approche par programmation dynamique pourrait être envisagée. La hiérarchisation du flux pourrait être approfondie en envisagent une hiérarchisation en résolution. Enfin, la généralisation du schéma de codage de contour aux contours internes permettrait une totale intégration dans un codeur objet.

troisième partie

Le codage dynamique

Chapitre 1

État de l'art : le codage dynamique

1.1 Présentation générale

Le codage dynamique a été introduit en 1995 [Ebrahimi et al. 95] durant la période de normalisation de MPEG4. L'EPFL[1] a donc proposé un codec vidéo. Le codec était adapté à la compression à faible débit, entre 10 kbit/s et 112 kbit/s, et fournissait une fonctionnalité objet à 48kbit/s pour des séquences avec peu de mouvement et peu de détails (exemple : séquence QCIF Akiyo à 5 Hz).

Le codage dynamique repose sur le constat suivant : chaque codeur est plus ou moins performant en fonction du signal qu'il traite. L'exemple donné dans [Reusens et al. 97] pour le codage d'image illustre bien ce constat (cf. figure 1.1). Ils comparent trois types de codage :

- codage basé DCT (JPEG),
- codage basé fractal,
- codage basé dessin (bi-niveau).

et l'on remarque que (cf. figure 1.1) :

- le codage basé dessin est le plus adapté pour l'image avec du texte,
- le codage basé fractal est le plus adapté pour le dessin,
- le codage basé DCT est plus adapté pour l'image médicale d'un cœur.

Ainsi, en fonction du signal, certains codages sont plus adaptés que d'autres. L'idée de base du codage dynamique est alors de partitionner l'image en régions et de coder chacune d'elles en utilisant une des techniques de codage dont on dispose. Cette approche permet de sélectionner de manière adaptative la technique de codage en fonction des caractéristiques de la région.

De manière générale, le codage dynamique nécessite dans un premier temps une spécification du profil de codec que l'on désire. Cette spécification tient compte de plusieurs paramètres :

- le type de donnée (image, vidéo, images stéréo, ...),
- le débit (faible-débit, moyen-débit, haut-débit)
- le type d'application (temps réel ou non),
- le type de communication (point à point, point à multi-points, multi-points à multipoints),

1. EPFL : École Polytechnique Fédérale de Lausane

(a) une image avec du texte (b) un dessin (c) une image médical

(a) JPEG 0,5 bit/pixel (b) JPEG 0,25 bit/pixel (c) JPEG 0,4 bit/pixel

(a) Fractal 0,5 bit/pixel (b) Fractal 0,25 bit/pixel (c) Fractal 0,4 bit/pixel

(a) Graphique 0,5 bit/pixel (b) Graphique 0,25 bit/pixel (c) Graphique 0,4 bit/pixel

FIG. 1.1 – *Illustration des performances de différents codeurs en fonction du signal (figure extraite de [Reusens et al. 97])*

– les fonctionnalités (hiérarchisation, objet, progressivité, édition, ...).

Cette spécification permet donc de choisir parmi :

– plusieurs techniques de segmentation,

– et plusieurs techniques de codage.

On limite ainsi notre codec à un ensemble de solutions admissibles. Dans un second temps, il faut définir un critère d'évaluation permettant de choisir selon ce critère la partition et le codage le plus performant.

Le codec doit donc réaliser une segmentation ainsi qu'un codage. Il est alors possible de procéder de deux façons différentes : soit segmenter puis coder, soit segmenter et coder de manière conjointe.

On peut remarquer que la notion de codage dynamique peut se retrouver à une échelle plus petite, dans la sélection de mode de codage intra, inter etc de MPEG2 et MPEG4. La segmentation étant ici réduite à une découpe en blocs. On retrouve aussi cette notion de codage dynamique dans H264/AVC lors de la découpe de macroblocs par optimisation débit-distorsion.

1.2 Le codage dynamique appliqué au codage d'objet vidéo

Dans [Reusens et al. 97], le codeur dynamique d'objets vidéo est réalisé pour une application de type vidéo-conférence. Les contraintes de cette application sont des délais de codage et de décodage faibles et une compression à très faible débit. Ainsi, on se restreint à une partition en arbre quaternaire (quad-arbre). Cinq modèles de codage sont alors retenus :

– un codage DCT de texture,

– un codage DCT de l'erreur de compensation de mouvement,

– une compensation de mouvement sans codage de l'erreur,

– un codage de texte et de graphique (codage binaire de l'image),

– un codage par fractal.

On décrit d'abord la séquence en objets avant-plan et objet arrière-plan grâce à des régions d'intérêt. Ensuite, la segmentation en quad-arbre de chaque objet est réalisée conjointement avec le codage, grâce à une optimisation débit-distorsion. La figure 1.2 illustre le partitionnement avant-plan et arrière-plan ainsi que le résultat du codage avant-plan et arrière-plan de l'image 0 de la séquence QCIF Akiyo.

On peut constater que la segmentation en objets vidéo de [Reusens et al. 97] est fournie par un opérateur humain. Plus généralement, dans le cadre du codage par analyse-synthèse, la segmentation en objets vidéo est faite de manière disjointe par rapport au codage. En effet, on cherche des objets qui ont un sens ou bien qui respectent un modèle. Ainsi, on ne cherche pas nécessairement un optimum en terme de débit-distorsion lors de la segmentation. La segmentation et le codage sont donc réalisés consécutivement.

La segmentation doit donc pouvoir donner des familles d'objets qui respectent des modèles (la première partie de ce manuscrit aborde le problème d'extraction d'objets vidéo). Cette disjonction entre la segmentation et le codage permet d'utiliser des modèles d'objets convenants parfaitement à la définition d'une région de l'image. On gagne alors en richesse de représentation par rapport une approche conjointe de la segmentation et du codage. De plus le codage bénéficie de modèles qui sont adaptés et peu coûteux.

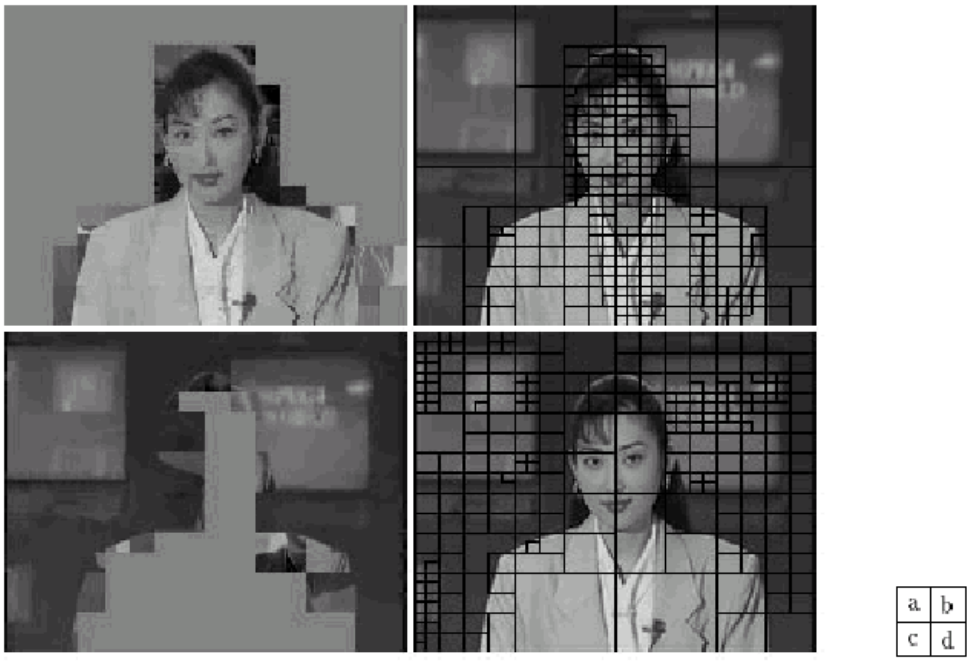


FIG. 1.2 – *Codage dynamique d'objet de [Reusens et al. 97]. (a) image avant-plan intra décodée (8Kbit), (b) quad-arbre de l'image avant-plan, (c) image arrière-plan intra décodée (7Kbit), (d) quad-arbre de l'image arrière-plan. Figure extraite de [Reusens et al. 97])*

En conclusion, le codage dynamique par objets nécessite dans un premier temps une segmentation en objets. Ensuite, on peut procéder au codage par mise en concurrence de différentes techniques de codage.

1.3 Les mesures de distorsion

Il existe de nombreuses mesures de distorsion permettant d'évaluer la qualité d'une image décodée. On utilise en général la mesure de PSNR (Peak Signal to Noise Ratio) qui exprime l'amplitude de la détérioration de l'image par rapport à l'image originale, c'est-à-dire l'amplitude du bruit par rapport au signal. Pour une image en niveaux de gris codée sur 8 bits, on a :

$$PSNR = -10\ log\left(\frac{EQM}{255^2}\right). \tag{1.1}$$

L'EQM représente l'Erreur Quadratique Moyenne entre l'image originale et l'image dégradée. Plus l'image a une faible valeur de PSNR plus l'image est détériorée.

Cette mesure n'est pourtant pas bien adaptée. En effet, une transformation légère de type rotation, translation zoom, affine, etc donne souvent une mesure de qualité faible alors que l'humain ne va pas noter de dégradation. De la même façon, un éclaircissement ou un assombrissement de l'image originale mène à un PSNR très faible alors que l'image est pour l'humain de très bonne qualité. Cependant, bien qu'elle ne corresponde pas très bien au système de vision humaine, c'est cette mesure qui a été retenue pour l'évaluation des

algorithmes soumis aux normes comme celle de MPEG4. Il faut noter que la détermination de mesures d'évaluation de la qualité d'une vidéo représente une activité de recherche à part entière. Par exemple, le groupe VQEG[2] (Video Quality Experts Group) propose de nouvelles méthodes de mesure de la qualité d'une vidéo en émulant le Système de Vision Humain.

Pour effectuer une mise en concurrence de plusieurs codeurs, on doit utiliser une mesure commune d'évaluation de la qualité des résultats de codage. La définition d'une telle mesure est problématique et reste l'objet de recherche. Par exemple, le PSNR ne mesure pas bien la distorsion, et il semble vain de vouloir comparer chaque résultat de codage avec cette mesure. De plus, chaque codage introduit des types de distorsions différents. La figure 1.3 illustre les artefacts associés aux techniques de codage par DCT, par ondelettes et par régions :

- le codage par régions a tendance à faire perdre les textures et à introduire de faux contours,
- le codage par DCT laisse apparaître des effets de bloc,
- le codage par ondelettes introduit des rebonds et un flou.

(a) codage région (b) codage basé DCT (c) codage basé ondelette

FIG. 1.3 – *Illustration des artefacts selon le codage*

On peut citer [Fleury 99] qui a repris une des métriques de [vandenBrandenLambrecht 96] pour construire une métrique basé région (RBQM - Region Based Quality Metric) proche du Système de Vision Humain. Il a alors défini un algorithme de prédiction du meilleur choix de codage par régions sachant cette métrique. Ainsi, sachant une contrainte de débit et la connaissance de quelques caractéristiques d'une région (taille, moyenne, variance, catégorie de la région (homogène, texturée, frontière), débit, fraction du rectangle englobant), son algorithme prédit le codage (basé DCT, basé ondelette, basé région, codage mouvement uniquement) donnant la meilleure qualité RBQM. Une bonne revue des mesures de qualité proches du système humain est présentée dans [Nadenau et al. 00].

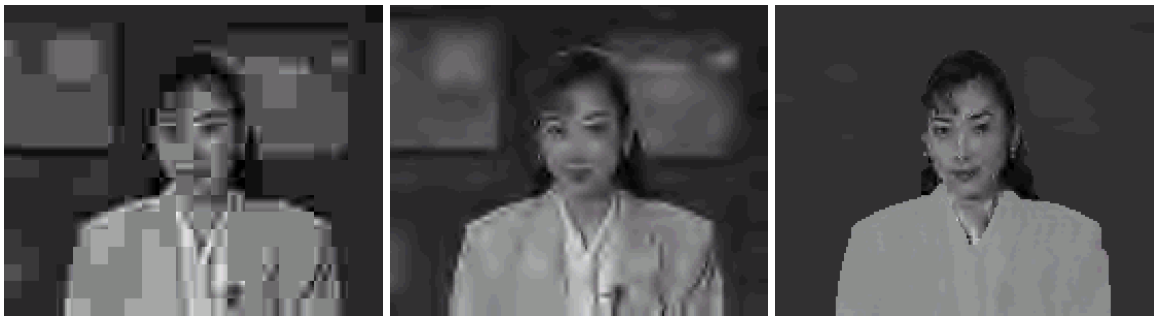
En conclusion, si l'on veut mettre en concurrence différentes techniques de codage, il faut se donner une mesure de qualité qui permette de comparer équitablement les différents résultats de codage.

2. http://www.vqeg.org/

1.4 La répartition des débits

La répartition des débits entre chaque région ou chaque objet est classiquement modélisée par une approche débit-distorsion. Dans la section 1.1.2, nous avons présenté l'approche de segmentation en quad-arbre de [Pardàs et al. 96], formulée par un critère débit-distorsion, qui permet de calculer la partition et le codage le plus adapté sachant un débit donné.

La figure 1.4(c) illustre le résultat du codage dynamique sur l'image 0 de la séquence Akiyo. L'image a été partitionnée en trois régions : le fond, la tête et le buste. La répartition des débits entre ces trois zones est alors obtenue de sorte que la qualité RBQM (cf. section 1.3) soit la même sur chaque zone. Cette répartition ainsi que le choix du codage est effectué via une approche débit-distorsion. La figure 1.4(a) (resp. b) illustre le résultat du codage de l'image par Jpeg (resp. par ondelette). Dans le cas du codage dynamique (image 1.4(c)), la plupart du débit est attribué au visage. En effet, l'activité sur le visage du personnage est forte, ainsi les distorsions ont tendance à être visibles. Pour avoir une qualité RBQM égale sur les trois objets il faut donc donner plus de débit au visage.



(a) codage Jpeg (b) codage basé ondelette (c) codage dynamique

FIG. 1.4 – *Comparaison entre un codage dynamique en objets vidéo et deux méthodes de codage non objet. Figure extraite de [Fleury 99]*

D'autre propositions de répartition de débit entre objets ont été proposées. Elle mettent en avant la notion de région d'intérêt ou bien de région de focalisation visuelle. Les régions d'intérêt peuvent être spécifiées par un opérateur ou bien par des critères qui sont pour l'instant empiriques. On notera que le mouvement, l'ordre de profondeur et la taille sont couramment utilisés. Par exemple, dans [Chai et al. 00], on favorise l'avant-plan par rapport à l'arrière-plan en quantifiant beaucoup moins celui-ci. Le partage des débits est nuancé par le ratio de surface et celui du mouvement. La figure 1.5 illustre la comparaison d'un codage H261 avec répartition (visage - reste de l'image) ou sans répartition de débit. On peut remarquer qu'à débit identique le fait de donner moins de débit sur l'arrière-plan mène à une qualité subjective bien meilleure. Notons encore une fois qu'il manque un modèle de vision humaine permettant de justifier ces choix.

En conclusion, le problème de répartition des débits nécessiterait une connaissance plus approfondie du système de vision humain (SVH). On pressent tout de même que les critères de mouvement, d'ordre de profondeur et de taille sont pertinents pour définir le SVH.

(a) H261 R=39Kb/f
Q_{fond}=25 - $PSNR_{fond}$=29,45
Q_{visage}=25 - $PSNR_{visage}$=30,91

(b) H261 R=39Kb/f
Q_{fond}=31 - $PSNR_{fond}$=28,45
Q_{visage}=11 - $PSNR_{visage}$=34,87

FIG. 1.5 – *Illustration de la répartition de débit par région d'intérêt. Figure extraite de [Chai et al. 00]*

1.5 Résumé du chapitre

Ce chapitre introduit la notion de codage dynamique qui consiste à segmenter puis à coder de manière optimale le résultat de la segmentation. Dans le cadre du codage objet, cela consiste à mettre en concurrence plusieurs codeurs par objet et à choisir le codeur le plus performant pour chaque objet.

Il se pose alors le problème de la métrique d'évaluation des résultats de codage. En effet, chaque codage génère des artefacts différents qu'il est difficile de comparer tant qu'on ne disposera pas d'un modèle du système de vision humaine. Se pose aussi le problème du choix de codage par objet ainsi que de la répartition des débits.

Le chapitre suivant propose un schéma de codage dynamique avec la justification des mesures de distorsion. Une approche d'optimisation débit distorsion est aussi donnée pour choisir le codeur et le débit pour chaque objet. Enfin, les problèmes de composition sont abordés.

Chapitre 2

Le schéma de codage dynamique par objets

Ce chapitre donne les solutions retenues pour la mise en place de notre codeur dynamique. Il a été défini en fonction des contraintes suivantes : codage à faible débit, communication point à point, fonctionnement non temps réel et fonctionnalité objet.

Pour ce faire, nous proposons un codage mettant en concurrence le m3dcoder [Galpin et al. 01] [Balter et al. 03b], un codeur H264/AVC [Schäfer et al. 03] modifié, le codeur ondelette 3D [Cammas et al. 03b] et un codage par image mosaïque (« sprite ») avec un mouvement affine.

Le schéma 2.1 illustre le fonctionnement général du codage dynamique sur deux objets. Chaque codeur code chaque objet. Les courbes débit-distorsion obtenues permettent de choisir le codeur et la répartition des débits pour chaque objet. Pour ce codage dynamique, des solutions sont proposées pour la métrique de qualité, pour la répartition des débits ainsi que pour la composition.

Les sections suivantes présentent les différents codeurs et la métrique de distorsion retenue pour chacun d'eux, la technique de répartition des débits, et les techniques de compositions pour la synthèse de la vidéo décodée.

2.1 Descriptions des codeurs utilisés

Les sous sections suivantes présentent respectivement : le codeur m3dcoder, le codeur H264/AVC adapté, le codeur ondelette 3D et le codeur par mosaïque et mouvement affine.

2.1.1 Le codeur m3dcoder

Le codeur m3dcoder [Galpin et al. 01] [Balter et al. 03b] est un OBASC basé sur la construction d'un modèle 3D d'une scène rigide à partir d'une séquence vidéo. L'analyse est effectuée sur un GOP (groupe d'images) dont on extrait :

- un modèle 3D de l'objet rigide,
- la position de la caméra au cours du temps,
- la première et la dernière image du GOP.

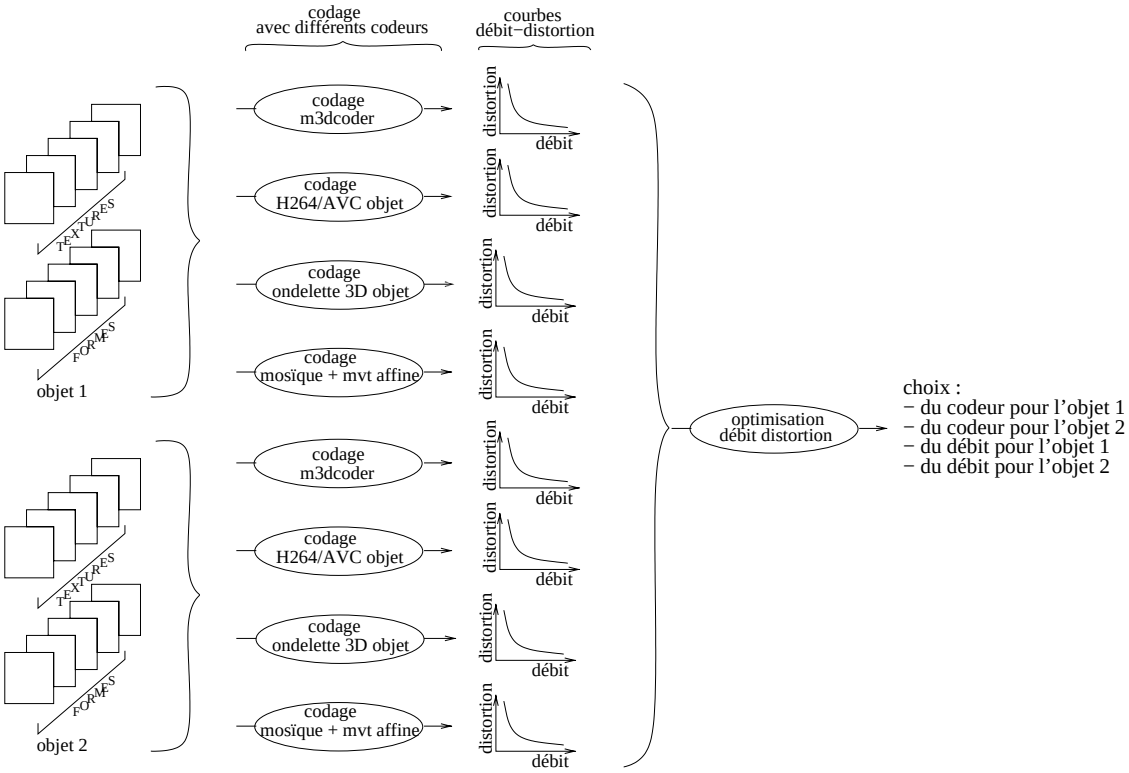
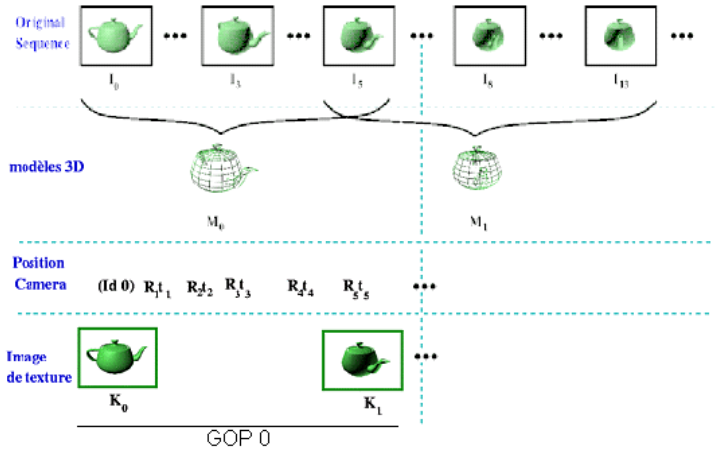


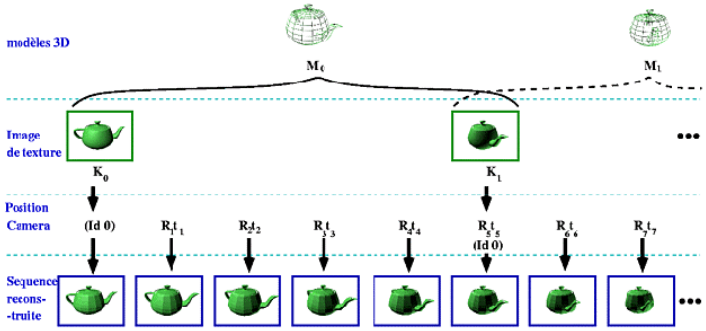
FIG. 2.1 – *Schéma du codage dynamique que nous proposons. L'illustration est donné pour le codage dynamique de deux objets*

Le codage est alors réalisé en 3 étapes. Tout d'abord le modèle 3D est représenté par un maillage uniforme dont les positions des noeuds sont quantifiés et codé par EBCOT (jpeg2000). Ensuite, les positions de la caméra sur le GOP sont codées en différentiel. Enfin, la dernière image est codée en Inter via EBCOT (jpeg2000). La première image est codée uniquement pour le premier GOP de la séquence. On la code alors en Intra via EBCOT (jpeg2000).

La compression est réalisée avec une contrainte de débit. On code dans un premier temps le modèle 3D du GOP et les positions de la caméra sur le GOP puis le débit restant est utilisé pour coder la texture de l'image clef du GOP en prenant en compte l'amortissement du coût de codage de l'image Intra du premier GOP. Le schéma 2.2 illustre le fonctionnement de la partie d'analyse puis de synthèse du codeur.



(a) Partie analyse du m3dcoder. Extraction à partir d'un GOP : du modèle 3D, des positions caméra et de 2 images de texture



(b) partie synthèse du m3dcoder. Reconstruction de la vidéo à partir : du modèle 3D, des positions caméra et de 2 images de texture

FIG. 2.2 – *Analyse-synthèse pour le codeur m3dcoder. Figure extraite de [Balter et al. 03b]*

L'analyse et la synthèse

La séquence doit respecter les trois hypothèses suivantes pour que la modélisation soit possible :

- la scène (objet) doit être statique,
- la scène doit contenir peu d'objets spéculaires,
- le mouvement de la caméra ne doit pas être dégénéré.

L'analyse de la séquence consiste alors à trouver la taille du GOP c'est-à-dire trouver une image clef. Pour ceci, trois hypothèses doivent être respectées :

- le mouvement moyen des points suivis entre la première et la dernière image doit être supérieur à 10 pixels,
- le pourcentage de points communs entre la première et la dernière image doit être supérieur à 30%,
- l'erreur de mise en correspondance des points du modèle avec les points suivis doit être inférieure à 1/2 pixels.

Si l'une des hypothèses n'est pas valide, l'analyseur échoue. Les étapes nécessaires pour le choix de l'image clef sont donc :

- une estimation du mouvement via un maillage actif,
- le choix de points d'intérêt pour le calcul du modèle,
- l'estimation de la matrice fondamentale (F) avec les paramètres intrinsèques de la caméra fixés. L'estimation de F est réalisée de manière robuste par minimisation de la distance symétrique entre les droites épipolaires et les points mis en correspondance (approche RANSAC - Random Sample Consensus) ;
- obtention du modèle 3D grâce à la matrice fondamentale,
- calcul des paramètres extrinsèques : obtention de la matrice essentielle et calcul des matrices de rotation et translation permettant de passer de la première caméra à la dernière caméra (méthode de Luong-Faugeras),
- calcul de pose : calcul de toutes les positions caméra par minimisation entre les points 3D projetés et leur correspondant 2D (méthode de Dementhon).

De plus, le calcul d'un modèle 3D sur un GOP est contraint par le modèle 3D du GOP précédent ainsi que le modèle 3D du GOP suivant, pour avoir une cohérence entre chaque modèle 3D. Cette cohérence est de type contrainte d'échelle et contrainte de reprojection 2D. Cette approche est appelée : ajustement glissant (« bundle adjustement »).

Le codage de chaque GOP met en jeu :

- le modèle 3D,
- la position des caméras,
- la première et de la dernière image du GOP.

Le schéma de codage retenu est un codage IP. La quantité d'information par GOP est très faible. En effet, à distorsion égale et à faible débit, pour un GOP défini par m3dcoder, H263+ va consacrer 2 à 3 Kb/image pour le champs de mouvement alors que m3dcoder va seulement utiliser environ 200 bits/images pour le modèle 3D et les positions caméra ([Galpin 02] p.166). Pour ce qui est de la texture, H263+ va coder toutes les images avec un schéma IPB alors que m3dcoder va coder une image prédite pour le GOP avec prise en compte de l'amortissement du coût de codage de la première image de la séquence.

La mesure de qualité

La figure 2.3 illustre le problème d'utilisation du PSNR lorsque la séquence codée-décodée a subi des transformations géométriques. En effet, pour un codage à 82 Kb/s, le PSNR de la séquence codée par m3dcoder est de l'ordre de 26 dB alors que chacune des images clef est codée avec un PSNR d'environ 34 dB. Ainsi, bien que subjectivement, la qualité de reconstruction de m3dcoder est nettement supérieure à H26L, la mesure de PSNR n'indique qu'une faible supériorité de 1dB (H26L dégrade la vidéo à environs 25 dB). Comme nous l'avons vu précédemment (section 1.3), le PSNR n'est pas adapté pour évaluer la qualité dans le cas de distorsions géométriques comme celles introduites par le codeur m3dcoder.

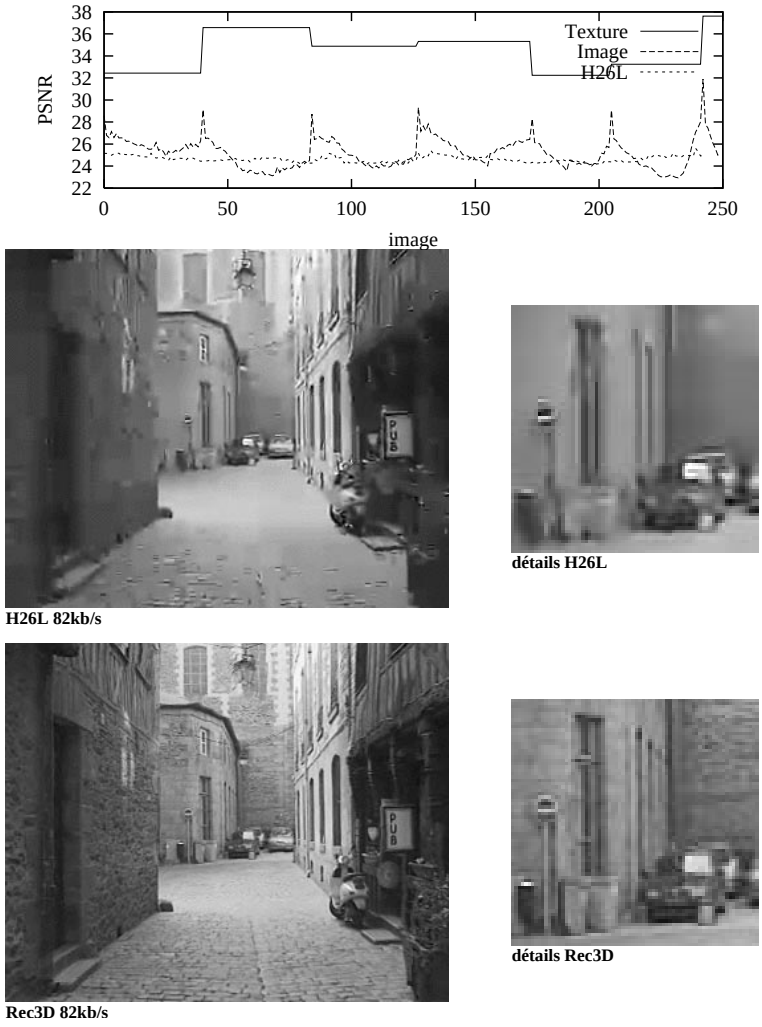


FIG. 2.3 – *Comparaison H26L et m3dcoder à 82Kb/s sur la séquence Rue CIF à 25 Hz. Les qualités PNSR sont d'environ 26dB pour m3dcoder et 25 dB pour H26L. Par contre la qualité subjective est très largement supérieure pour le m3dcoder*

Des mesures de qualité ont été proposées pour modéliser de façon plus réaliste le

Système de Vision Humain (section 1.3). Cependant ces techniques sont bien souvent adaptées aux distorsions associées au type de codage. De ce fait, nous conservons toutefois comme mesure de qualité le PSNR qui reste une mesure intéressante tant qu'il n'y a pas de transformation géométrique ou de variation de luminosité. Nous allons ainsi utiliser le PSNR dans un espace où la distorsion géométrique ne sera plus présente lors du calcul. On décide alors de mesurer deux distorsions :

- d'une part les erreurs de distorsion sur la géométrie,
- d'autre part les erreurs de distorsion sur la texture.

Pour mesurer les erreurs de distorsion texture avec le PSNR sans être perturbé par les erreurs géométrique, on peut alors procéder de deux façons:

- Soit **mesurer le PSNR entre l'image clef et l'image clef codée-décodée** et attribuer cette mesure à tout le GOP. C'est la mesure actuellement utilisée dans [Galpin et al. 01] et [Balter et al. 03b]. On parle de mesure dans le domaine texture. Cette approche est indépendante des distorsions géométriques mais elle pose deux problèmes :
  - elle ne prend pas en compte le système de reconstruction des images du GOP. En effet, deux textures participent à la reconstruction : la texture de la première image et la texture de la dernière image (ce problème est cependant partiellement résolu par la technique de morphing [Balter et al. 03a]) ;
  - elle ne propose qu'une seule valeur de qualité valable pour tout le GOP.
- Soit **mesurer le PSNR en construisant une nouvelle séquence qui servira de référence** et qui prend en compte les distorsions géométriques. On parle de mesure dans le domaine image. Cette approche permet d'utiliser beaucoup plus de textures et ainsi d'avoir une mesure de PSNR pour chaque image du GOP. La séquence référence peut être obtenue de deux façons :
  - en construisant une séquence ou le modèle 3D et les positions caméras sont codées et décodées avec les textures de début et de fin de GOP. Les variations de luminosité ou les changements/apparitions de textures entre les deux images clef de la séquence originale ne seront pas présentes dans la séquence de référence. Cette technique est cependant un peu artificielle et a tendance a sur-évaluer la qualité ;
  - en construisant une séquence ou le modèle 3D et les positions caméras sont codées et décodées avec toutes les textures. Cette technique est cependant dépendante des variations de résolutions locales et il n'est pas évident qu'elle soit plus représentative que les autres.

Pour être cohérents avec les autres mesures de qualité que nous avons prises pour le codeur Ondelette 3D et le codeur par mosaïque, nous avons adopté la mesure effectuée dans le domaine texture. Cette mesure donne une unique valeur PSNR pour tout un GOP.

2.1.2 Le codeur H264/AVC adapté

Le codeur H264/AVC

L'unité « Joint Video Team (JVT) » qui réunit le monde ISO et le monde ITU ont finalisé la conception du codeur décodeur **H264/AVC** aussi appelé **MPEG4 partie 10**

ou **MPEG4 Advanced Video Coding (MPEG4 AVC)** fin 2002. La figure 2.4 illustre la chronologie des travaux sur H26L et MPEG4 qui ont mené au nouveau codec H264/AVC.

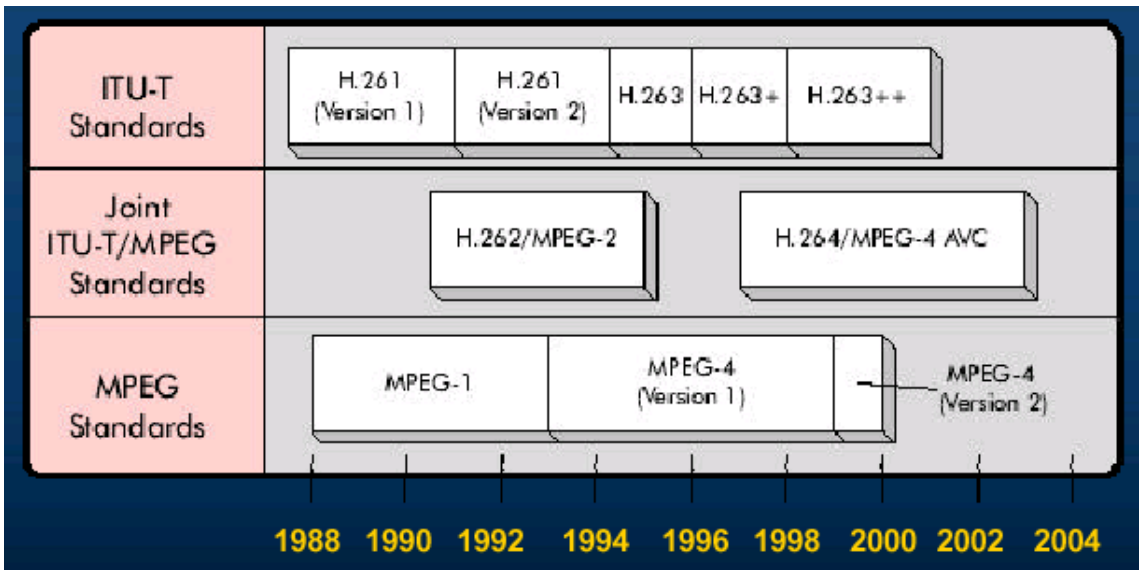


FIG. 2.4 – *Chronologie des standards menant à H264/AVC*

Le schéma de codage du codeur décodeur H264/AVC est très similaire au schéma de codage MPEG2. La figure 2.5 illustre le codage pour un macro-bloc.

La première image de la séquence ainsi que les images utilisées pour un accès aléatoire sont codées en INTRA, c'est-à-dire sans utiliser d'autres informations que celles présentes dans l'image elle-même. Chaque bloc est prédit en utilisant le voisinage spatial des blocs déjà codés.

Pour toutes les autres images de la séquence, on utilise le codage INTER. Le codage INTER utilise une prédiction par compensation de mouvement à partir des images précédemment décodées. Le processus de codage consiste à choisir les données de mouvement, c'est-à-dire l'image de référence et le mouvement appliqué aux blocs.

Le résidu de prédiction (INTRA ou INTER) - qui est la différence entre le bloc original et le bloc prédit - est transformé par une transformation entière. Les coefficients sont alors quantifiés puis codés de manière entropique.

Lorsque le décodeur reçoit l'erreur de prédiction (INTRA ou INTER) codée d'un bloc ou d'une image, il effectue alors le processus inverse : décodage entropique, transformation entière inverse, ajout de l'image de prédiction et enfin post-filtrage pour supprimer les effets de bloc (deblocking filter).

La performance de H264/AVC

Chaque étape du codage a été optimisée et a vu le nombre de paramètres considérablement augmenter. Voici quelques caractéristiques qui font la supériorité de H264/AVC sur les autres codeurs basé blocs :

– découpe des macroblocs telle que l'on dispose de tailles variables : 16x16, 8x16, 16x8, 8x8, 4x8, 8x4, 4x4,

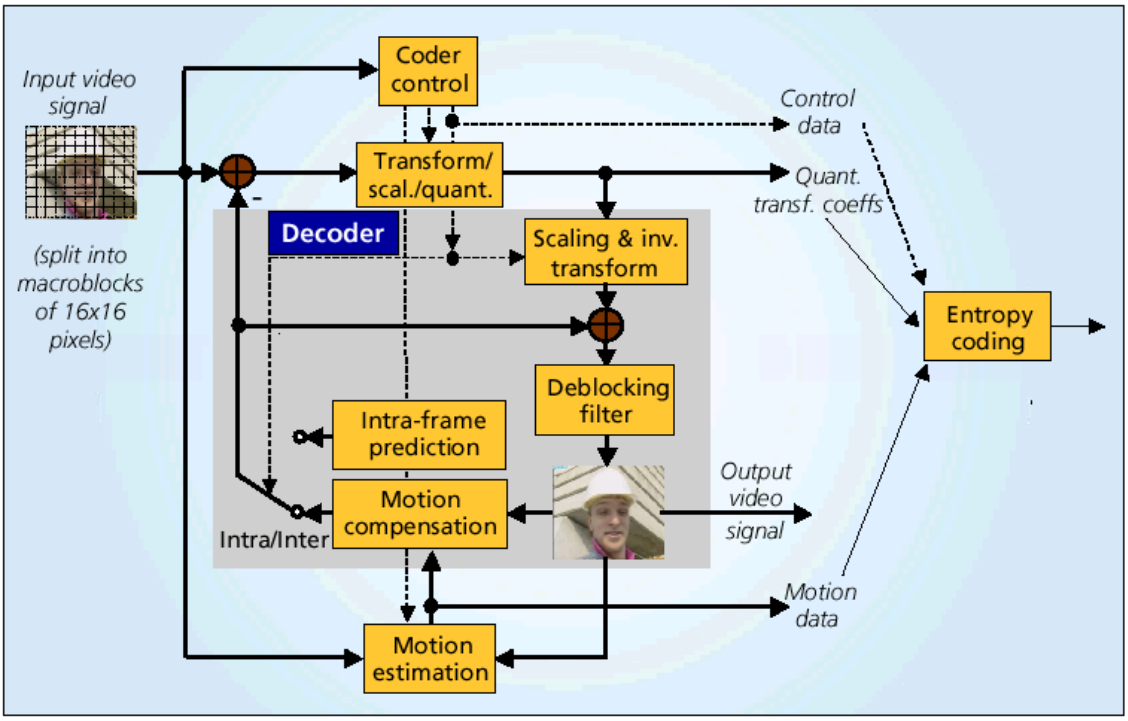


FIG. 2.5 – *Structure basique pour un codeur H264/AVC pour un macro-bloc. Figure extraite de [Schäfer et al. 03]*

- choix du codage du macrobloc INTER, INTRA, SKIP par un critère débit distorsion,
- calcul du mouvement par un critère débit-distorsion avec boucle sur l'estimation du mouvement pour supprimer les artefacts et une précision allant jusqu'au quart de pixel.
- prédiction des INTER grâce à plusieurs images références.
- transformation entière,
- plus grand contexte pour le codeur arithmétique CABAC,
- deblocking filter dans la boucle de codage.

Des évaluations ont été menées récemment [Schwarz et al. 02] sur des séquences QCIF (10Hz et 15Hz) et CIF(10Hz et 15Hz) et donnent les performances telles qu'en moyenne H264/AVC (version JVT/H.26L JM-2) gagne à distorsion équivalente :

- 39% de débit sur MPEG4 (Advanced Simple Profile),
- 49% de débit sur H.263 (High Latency Profile),
- 64% de débit sur MPEG2.

La figure 2.6 montre les courbes de distorsion en fonction du débit pour différents codeurs sur la séquence CIF Tempete à 15Hz. H264/AVC a plusieurs décibels d'avance sur MPEG4 pour le même débit.

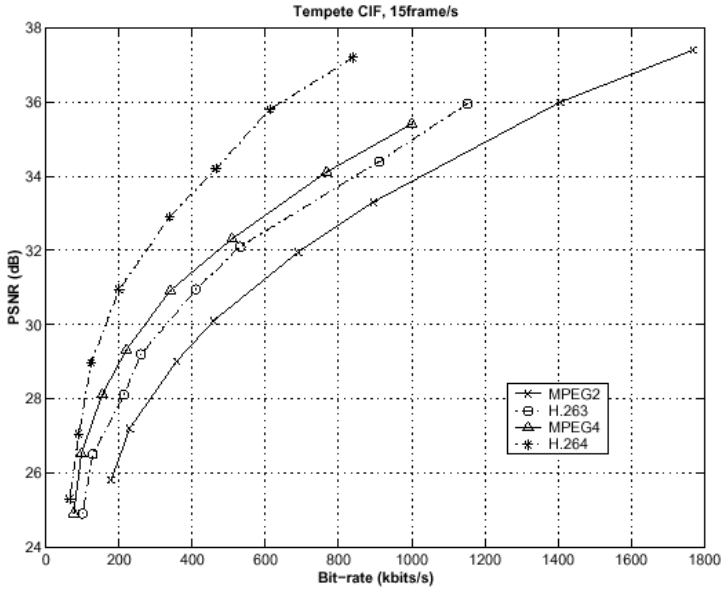


FIG. 2.6 – *Comparaison débit-distorsion avec différents codeurs sur la séquence CIF Tempete à 15Hz. Figure extraite de [Schwarz et al. 02]*

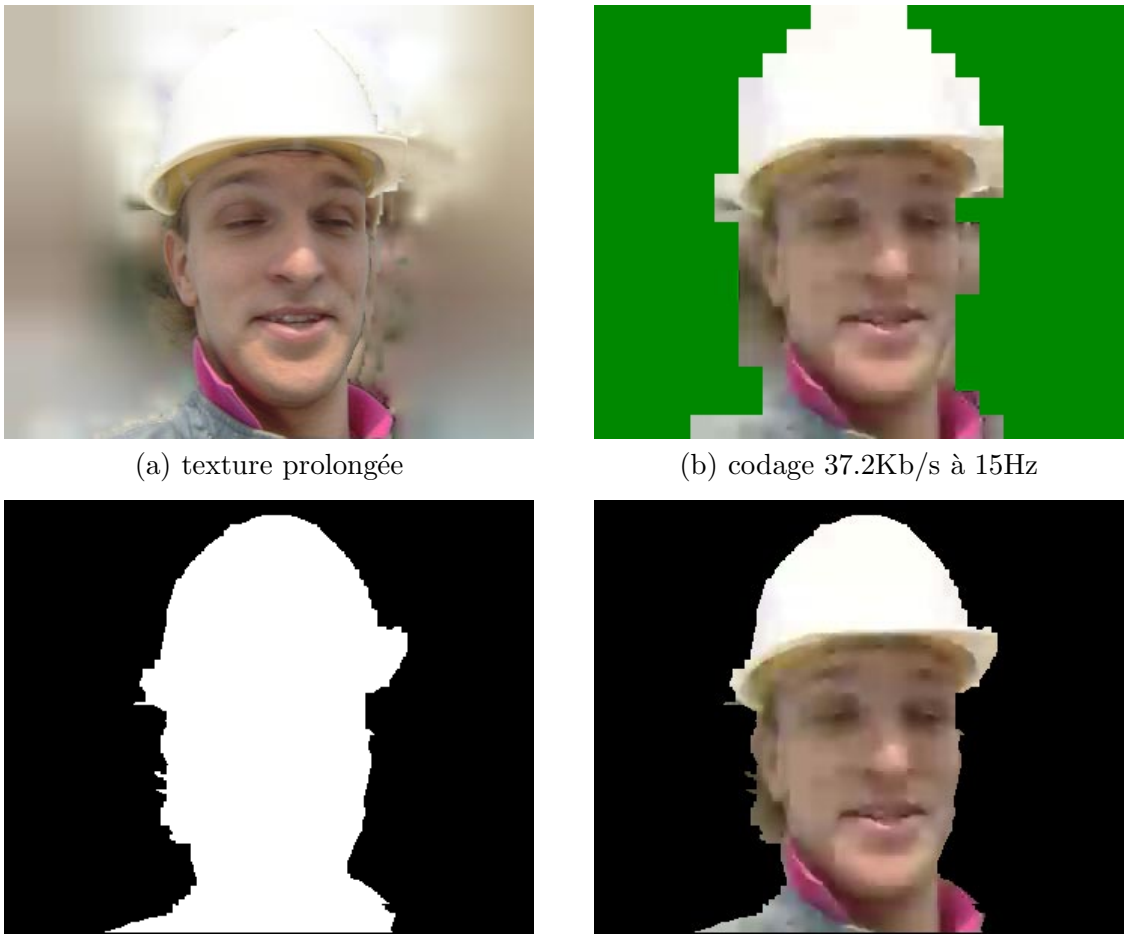
Modification de H264/AVC pour coder des objets vidéo

Nous avons modifié le codeur H264/AVC pour pouvoir l'adapter au codage de forme arbitraire. Pour ceci, nous avons fait le choix suivant : un macrobloc n'est pas codé s'il n'appartient pas à la forme. Ainsi, les macroblocs interceptant la forme sont codés. Lorsque le macrobloc n'est pas totalement inclus dans la forme, une solution est de faire du codage adapté à la forme (shape adapted). Or, nous voulons un codage qui soit indépendant de la notion de forme, c'est-à-dire que le décodage ne nécessite pas la connaissance de la forme.

Pour ne plus être dépendant de la forme lors du codage, nous décidons de coder par bloc. Cependant, pour les blocs à la frontière de l'objet, on effectue un prolongement (padding) de texture. La technique de prolongement permet d'introduire peu de hautes fréquences et ainsi, de ne pas être trop coûteuse lors du codage du macrobloc. De plus, les macroblocs décodés et utilisés comme prédiction par H264/AVC ont la propriété, grâce au padding, d'être prolongés en texture. Les erreurs de prédictions sont ainsi peu coûteuses.

La figure 2.7(a) illustre la technique de padding sur l'image 0 de Foreman. La figure 2.7(b) illustre le résultat du codage de la texture et du mouvement à 37.2Kb/s avec H264/AVC adapté aux objets. La figure 2.7(c) illustre la forme utilisée pour le choix du codage des macroblocs. La figure 2.7(d) montre l'objet vidéo plan (Video Object Plan - VOP) obtenu en utilisant la forme initiale et le résultat de codage de la texture.

La propagation (padding) de texture dans les zones non définies peut être vue comme la définition d'un signal étendu par rapport à la texture originale qui est le signal initial. La technique est similaire à celle utilisé pour le prolongement de contours (voir partie 2 section 2.3.1). L'approche est itérative et basée sur un calcul multi-résolution basé quad-arbre avec raffinement successif du signal étendu. Il y a trois étapes itérées pour calculer le signal étendu c'est-à-dire pour calculer la texture à utiliser dans les zones non définies :



(a) texture prolongée (b) codage 37.2Kb/s à 15Hz

(c) masque utilisé (d) VOP résultat

FIG. 2.7 – *Illustration du résultat de codage H264/AVC JM5 par objets sur l'objet visage de la séquence CIF Foreman à 15 Hz sur les 90 premières images*

la définition d'un résidu, l'approximation de ce résidu, le raffinement du signal étendu.

À la première itération, le résidu Res^0 est égal à la texture de l'objet. L'approximation de ce résidu nous donne un quad-arbre ΔI^l possédant un seul nœud et dont la valeur est la valeur moyenne du résidu Res^0. Le raffinement du signal étendu nous donne la première approximation du signal étendu de sorte que $\overline{I}^0 = \Delta I^0$. À chaque itération les feuilles du quad-arbre sont divisées en quatre. De manière générale, les étapes de l'itération sont :

- **la définition du signal résidu** Res^l. L'image Res^l est le résultat de la différence entre l'image des résidus de l'itération précédente Res^{l-1} et l'image avec les textures étendues $\overline{I}^{l-1}$. On a $Res^l = Res^{l-1} - \overline{I}^{l-1}$.
- **Le calcul d'un incrément de raffinement** ΔI^l. Le quad-arbre ΔI^l pour un niveau l a comme valeur de feuille la valeur moyenne de l'image des résidus Res^l.
- **Le calcul du signal étendu** $\overline{I}^l$. L'image $\overline{I}^l$ au niveau l est égale à $\overline{I}^l = \sum_{lev=0}^{lev=l} \Delta I^{lev}$

Les itérations cessent lorsque les noeuds du quad-arbre ΔI^l représentent un pixel. Une fois l'image $\overline{I}^l$ calculée, on utilise les valeurs obtenues pour remplir les zones non définies de l'image originale. La figure 2.7(a) illustre le résultat de remplissage de texture en utilisant un volume d'images pour calculer un signal étendu moyen.

Notons que la mesure de qualité retenue pour le codeur H264/AVC est le PSNR. En effet, puisqu'il n'y a pas de distortion sur le mouvement estimé et que le codage de chaque texture est effectué dans le référentiel de chaque image de la séquence, la mesure de PSNR est alors comparable à la mesure $PSNR_{texture}$ utilisée pour les trois autres codeurs : m3dcoder, ondelette 3D objet, et codage par mosaïque.

2.1.3 Le codeur ondelette 3D objet

Le codeur par ondelette 3D est celui présenté dans [Cammas et al. 03a] [Cammas et al. 03b] (voir partie 2 section 2.2) utilisé en mode objet. L'approche consiste dans un premier temps à estimer le mouvement sur les zones définies par les masques objets et ceci grâce à un maillage dynamique [Marquant 00]. Dans un second temps, les textures et les formes sont redressées dans un ou plusieurs temps de références. Un prolongement de texture (padding) (cf. 2.1.2) est alors effectué pour décorréler les formes des textures. Dans un troisième temps le codage de chacune des informations mouvement, texture, et forme est effectué.

Nous avons proposé dans [Chaumont et al. 03a] un codeur objet basé ondelette 3D avec utilisation du codage de forme présenté dans [Chaumont et al. 03b]. On a constaté (voir section 2.2) que sur la séquence Erik, à faible débit, la qualité PSNR résultant du codage par codeur ondelette 3D objet était supérieure à celle résultant du codeur H26L VM8. Ainsi, le codeur ondelette 3D est potentiellement un bon outil pour le codage dynamique.

La mesure de qualité que nous retenons est le PSNR dans le domaine texture c'est-à-dire que la mesure est faite non pas entre la séquence reconstruite et la séquence originale mais entre la texture codée et la texture originale. En effet, on suppose qu'une faible distortion sur le mouvement et la forme a peu d'impact sur le SVH. Ainsi, on ne conserve que la mesure de distortion de texture.

2.1.4 Le codage par mosaïque

La construction de mosaïque est basée sur l'estimateur de mouvement basé maillage [Marquant 00] et la construction est identique à [Pateux et al. 01]. On fixe un seuil d'arrêt de construction lorsque les mailles deviennent trop petites par rapport à leur taille initiale lors de l'estimation de mouvement. Ainsi, on définit la taille du GOP sur laquelle la mosaïque est construite. La figure 2.8 illustre la mosaïque du fond de la séquence Foreman sur 64 images.

FIG. 2.8 – *Illustration de la création d'une mosaïque grâce à l'estimateur de mouvement par maillage. Arrière plan de la séquence Foreman CIF avec 64 images*

Le codage de cette image est alors réalisée via jpeg2000. Le mouvement est simplifié en un mouvement affine à six paramètres par un calcul robuste (on utilise l'algorithme de clustering avec rejet présenté dans la partie 1 du manuscrit). Lorsque l'on génère la séquence décodée, on utilise donc un mouvement différent du mouvement original. Cependant, pour des mosaïques de type fond ayant un mouvement global simple, la distorsion sur le mouvement est imperceptible ce qui confirme que l'utilisation d'un mouvement affine est suffisante.

La distorsion de l'image mosaïque par rapport à la séquence originale est donc calculée uniquement en terme de $PSNR_{texture}$. Pour effectuer cette mesure, on projette toutes les images initiales ainsi que leur masque dans le référentiel de la mosaïque. On mesure alors le $PSNR_{texture}$, entre la texture projetée et la mosaïque codée-décodée, sachant le masque

projeté.

La technique de codage par mosaïque présente cependant l'inconvénient d'avoir une texture complètement statique. Ainsi, sur l'exemple de la mosaïque de l'arrière-plan de Foreman (figure 2.8), la mosaïque a un $PSNR_{texture}$ de 35.6. On ne peut donc jamais dépasser cette qualité. Le codage par mosaïque statique est donc plutôt intéressant dans la gamme de codage à très bas débit.

2.2 La répartition des débits entre objets

Cette section traite de la répartition des débits entre objets ainsi que du choix du codeur pour chaque objet. Cette répartition permet le codage dynamique et est efficace si l'on utilise une optimisation débit-distorsion.

Ainsi, la répartition des débits est faite par optimisation débit-distorsion de sorte que l'on ait une contrainte de débit global et que l'on choisisse pour chaque objet le meilleur des codeurs ainsi que le meilleur débit pour cet objet.

Il est donc nécessaire comme nous l'avons expliqué dans le premier chapitre d'utiliser une mesure de distorsion qui soit équitable. Pour cela, nous prenons :

- pour le codeur H.264 objet, la mesure d'erreur quadratique ;
- pour les codeurs : ondelette 3D, m3dcoder et codage par mosaïque, la mesure d'erreur quadratique dans le domaine texture. Si l'erreur sur le mouvement ou le modèle 3D est suffisamment faible, alors on néglige cette erreur, sinon la technique de codage est rejetée. Bien entendu tout système de pondération peut être ajouté pour privilégier un codeur, ou un objet.

2.2.1 Répartition par contrainte de débit

Après codage et décodage des séquences, nous disposons de courbes débit-distorsion $\mathcal{C}$ pour chaque codeur *cod* et chaque objet *obj*. Nous souhaitons alors obtenir la distorsion minimale sachant un débit contraint. On souhaite donc minimiser la distorsion D, résultant de la somme des distorsions sur chaque objet *obj*, sachant que le débit total R doit être inférieur au débit contrainte R_{cont} :

$$\min_{\{p_{obj,cod,i}\}\in\mathcal{C}} D = \sum_{obj} D_{obj}(p_{obj,cod,i}) : R = \sum_{obj} R_{obj}(p_{obj,cod,i}) < R_{cont} \tag{2.1}$$

Ainsi pour chaque objet *obj*, on a deux inconnues : le codeur *cod* et le point débit-distorsion $p_{obj,cod,i}$ de ce codeur. Remarquons qu'ici la distorsion D_{obj} utilisée est l'erreur quadratique. En effet, on veut minimiser l'erreur totale.

La formulation Lagrangienne de ce problème consiste à transformer ce problème contraint en un problème non contraint via l'introduction d'un multiplicateur de Lagrange λ. On définit alors le coût Lagrangien J:

$$J(\{p_{obj,cod,i}\},\lambda) = \sum_{obj} J_{obj}(p_{obj,cod,i},\lambda) \tag{2.2}$$

avec :

$$J_{obj}(p_{obj,cod,i},\lambda) = D_{obj}(p_{obj,cod,i}) + \lambda R_{obj}(p_{obj,cod,i}) \tag{2.3}$$

Minimiser J revient alors à minimiser chaque J_{obj} sachant la valeur de λ optimale.

On résoud le problème par une méthode de programmation convexe [Shoham et al. 89] qui suppose la monotonie des courbes débit-distorsion. La résolution est itérative et consiste :

- à trouver chaque J_{obj} minimal sachant la valeur de λ fixée. Pour ce faire, on calcule pour chaque objet la valeur de J_{obj} en parcourant tous les points $p_{obj,cod,i} \in \mathcal{C}$ de l'objet obj. On obtient donc pour chaque objet la valeur J_{obj} minimale et le point débit-distorsion $p_{obj,cod,i}$ associé. On déduit alors le débit total $R = \sum_{obj} R_{obj}(p_{obj,cod,i})$;
- à modifier la valeur de λ en fonction du résultat de débit R calculé précédemment pour approcher la contrainte de débit R_{cont}. Si la valeur de R est sous le seuil R_{cont}, on augmente la valeur du λ sinon, on diminue la valeur du λ.

2.2.2 Répartition par contrainte de débit et qualité uniforme

L'approche donnée ici, permet d'avoir la solution optimale. Par contre, il est possible que la répartition des qualités entre chaque objet ne soit pas très équitable. Pour qu'il y ait uniformité des qualités entre chaque objet, on formule le problème par l'utilisation d'un Lagrangien par objet. Ainsi, la valeur de J_{obj} dépend de λ_{obj} :

$$J_{obj}(p_{obj,cod,i}, \lambda_{obj}) = D_{obj}(p_{obj,cod,i}) + \lambda_{obj} R_{obj}(p_{obj,cod,i}) \tag{2.4}$$

Remarquons qu'ici, la distorsion D_{obj} est nécessairement l'erreur quadratique moyenne et non l'erreur quadratique. En effet, on veut que chaque objet ait le même PSNR et donc on doit utiliser l'EQM.

La résolution itère les deux étapes suivantes :

- dans une première étape on respecte une contrainte de qualité D_{cont} identique pour chaque objet. Il y a donc un certain nombre d'itérations sur chaque objet consistant :
  - à trouver le J_{obj} minimal sachant la valeur de λ_{obj} fixé. Pour ce faire, on calcule la valeur de J_{obj} en parcourant tout les points $p_{obj,cod,i} \in \mathcal{C}$ de l'objet obj. On obtient donc la valeur J_{obj} minimale et le point débit-distorsion $p_{obj,cod,i}$ associé. On a ainsi la qualité de l'objet D_{obj} ;
  - à modifier la valeur de λ_{obj} en fonction de la qualité D_{obj} obtenue précédemment pour approcher la contrainte de distorsion D_{cont}. Si la valeur de D_{obj} est sous le seuil de D_{cont}, on diminue la valeur du λ_{obj} sinon, on augmente la valeur du λ_{obj}.
- dans une seconde étape à modifier la contrainte D_{cont} pour respecter la contrainte R_{cont}. En effet, dans cette étape, nous disposons d'un ensemble de points $p_{obj,cod,i}$ respectant la contrainte D_{cont}. On peut calculer le débit résultant $R = \sum_{obj} R_{obj}(p_{obj,cod,i})$ et ainsi augmenter ou bien diminuer la valeur de D_{cont}.

2.3 La composition

Comme on l'a vu dans la deuxième partie du manuscrit, un objet est beaucoup plus qu'un ensemble de régions. Un objet c'est :

- une forme qui évolue dans le temps. Cette forme est l'enveloppe externe de l'objet. Ainsi, comme on l'a vu dans le chapitre sur le codage de formes (voir partie 2

section 2.3), les contours dûs à une occultation ne font pas partie de la forme. À la différence du codage de région, les problèmes de zones de texture non définies lors de la composition sont beaucoup moins présents du fait qu'une partie des textures cachées sont connues de l'objet. La figure 2.9 montre le phénomène de « trou » qui ne devrait pas être présent si l'objet fond était défini comme nous l'entendons c'est-à-dire sans contours. Cependant, le phénomène de « trou » reste possible ;

– une texture qui est fixe ou qui évolue et qui résume les textures apparues sur un certain nombre d'images ;

– un mouvement qui peut aussi générer des « trous » si on le code avec perte.

FIG. 2.9 – *Illustration du problème de composition d'objet ayant des fréquences d'échantillonnage temporelles différentes. La figure de gauche illustre un codage-décodage de l'avant-plan et de l'arrière-plan à 30Hz. La figure de droite illustre le codage de l'arrière-plan avec un sous échantillonnage à 15Hz et l'avant-plan codé-décodé à 30Hz. Les figures sont extraites de [Lee et al. 03].*

Les objets peuvent :

– être codés avec des fréquences temporelles différentes,

– subir des distorsions sur la forme et le mouvement,

– être codés avec des techniques de codage différentes, dans le cadre du codage dynamique.

Lorsque l'on recompose la scène plusieurs problèmes apparaissent :

– il est possible que certaines zones de l'image n'ont pas de texture. On voit alors apparaître des « trous » dans l'image reconstruite. Pour pallier ce problème, on utilise la technique de padding présentée dans la section 2.3.1 ;

– l'effet de placage des objets est particulièrement visible. Ceci est dû à l'aliasing qui est produit de part la discrétisation des contours d'objets. Pour pallier ce problème, on met en œuvre une technique d'antialiasing.

2.3.1 Le padding

Le padding ou prolongement de texture est utile lorsqu'une partie de la texture n'est pas définie (même une accumulation temporelle de texture par calcul d'une mosaïque ne permet pas de définir toutes les zones). Par exemple dans le cas du codage de forme avec perte, la forme décodée est différente de la forme initiale. Ainsi, certaines zones sont nouvelles en terme de textures.

L'exemple 2.10 montre qu'une distorsion sur la forme n'est pas visible tant que la texture utilisée pour remplir les zones non définies est un bon prolongement des textures existantes. Ainsi, le padding multirésolution autorise le codage de forme avec perte et permet de résoudre le problème délicat de remplissage de texture non définie.

De plus le codage avec perte (partie 2 du manuscrit) lisse les contours qui peuvent être bruités si la segmentation est automatique ou semi-automatique. Ainsi, dans l'exemple 2.10 on constate ce résultat étonnant : la composition est meilleure visuellement lorsque la forme est codée avec perte que lorsque l'on utilise la forme initiale. Cela s'explique par le fait que la segmentation originale est issue d'une segmentation par régions. Ainsi, les frontières n'étaient pas régulières et pas toujours bien positionnées. Le codage de contour avec perte rend plus régulier le contour codé-décodé.

2.3.2 L'antialiasing

L'antialiasing est une approche de post-traitement qui a été introduite pour réduire l'effet d'escalier produit lorsque l'on plaque une forme ayant un contour qui a été discrétisé.

L'approche consiste à calculer pour chaque pixel le pourcentage d'appartenance du pixel à la forme. Les pixels frontière ont alors un pourcentage d'appartenance proportionnel à l'aire couverte par le contour réel. L'antialiasing permet donc de calculer un coefficient de « mélange » par pixel [Porter et al. 84]. À la place d'un masque définissant la forme d'un objet, on dispose alors d'une carte de coefficients (alpha-map).

Le problème est alors de calculer le contour réel sachant que l'on dispose du contour discret. Les auteurs de [Braquelaire et al. 97] proposent de calculer les points de passage d'un contour réel. Ainsi, chaque pixel se voit attribuer un point de passage du contour réel. On appelle alors chemin Euclidien l'ensemble des points de passage du contour réel. Les points de passage permettent de déterminer le point d'entrée et le point de sortie dans chaque pixel. On déduit alors pour chaque pixel le pourcentage de couverture ("blending coefficient").

D'une manière similaire à [Braquelaire et al. 97], nous allons rechercher un contour réel à partir de masque discret. Pour cela, nous calculons une B-Spline ayant une distance au plus de 0.6 pixels par rapport au contour discret. On fait varier le nombre de points de contrôle pour obtenir la B-Spline possédant le moins de points de contrôle et respectant la contrainte de distance. Une fois la B-Spline obtenue, on extrait les points entrants et les points sortants de chaque pixel intercepté par la B-Spline. On obtient alors la surface de couverture pour chaque pixel grâce à la délimitation obtenue par la ligne joignant le point entrant et le point sortant.

La figure 2.11 illustre l'effet de l'antialiasing. La séquence est Foreman CIF 15Hz. L'avant-plan est codé à 79Kb/s avec la technique par ondelettes 3D, l'arrière-plan est codé par mosaïque à 15Kb/s et le masque est codé à 6Kb/s avec la technique IPB-Wavelet. La colonne de gauche montre le résultat de composition sans antialiasing et la colonne de

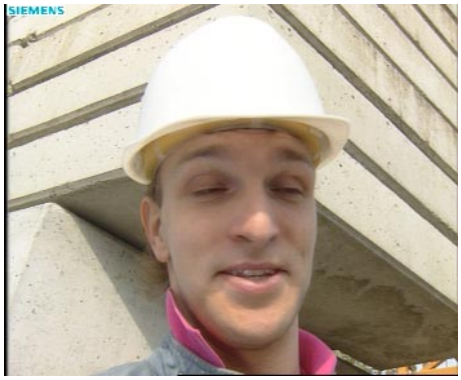
droite le résultat avec antialiasing. L'amélioration dûe à l'antialiasing peut paraître peu significative sur une seule image, mais elle apporte une amélioration visuelle significative en séquence. En effet, la composition sans antialiasing, en plus de provoquer des effets de marche d'escalier, donne l'impression que les objets sont superposés ce qui est particulièrement désagréable. De plus le codage-décodage des objets renforce l'aspect « saut » de luminance entre objets. On a encore plus l'impression d'avoir une image composée de « patchs ». Il est donc nécessaire de fondre les objets entre eux lors de la composition par une technique d'antialiasing.

2.4 Résumé de chapitre

Dans ce chapitre, nous avons présenté les différents codeurs que nous allons utiliser pour le codage dynamique d'objet vidéo. Ces codeurs font actuellement partis des plus performants. Nous avons aussi donné la mesure de distorsion que nous utiliserons pour comparer les résultats de chacun des codeurs. Enfin, nous avons présenté la technique d'optimisation débit-distorsion qui va nous permettre de choisir le codeur et le débit les plus adaptés pour chaque objet.

Nous avons aussi évoqué les problèmes de composition bien spécifiques au codage d'objets et qui sont assez rarement traités. Or, il semble que si l'on veuille se comparer à des techniques non objet, certains artefacts doivent être supprimés. Le padding de texture permet ainsi de faire du codage de forme avec perte sans que les textures manquantes ne posent problème. L'antialiasing permet d'éviter l'effet « patch » qui apparaît lors de la composition des objets.

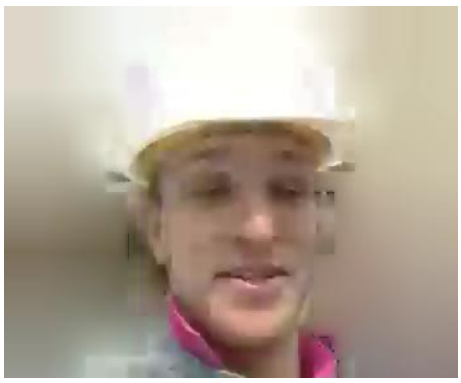
Le chapitre suivant présente les résultats du codage dynamique et des techniques de composition.



(a) séquence CIF Foreman 15 Hz

(b) padding de la mosaïque fond

(c) codage H264/AVC objet 37,2Kb/s

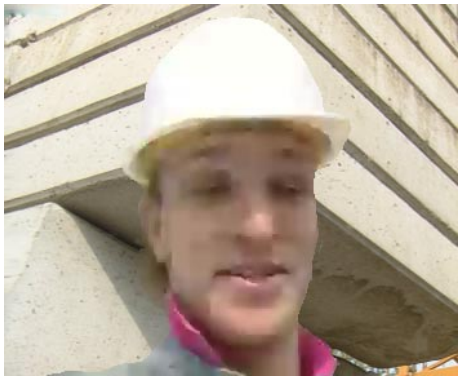


(d) padding sur les blocs non codés

(e) masque initial

(f) masque codé par IPB wavelet 6Kb/s

(g) composition visage à 37,2Kb/s et fond avec masque original



(h) composition visage à 37,2Kb/s et fond avec contour codé à 6Kb/s

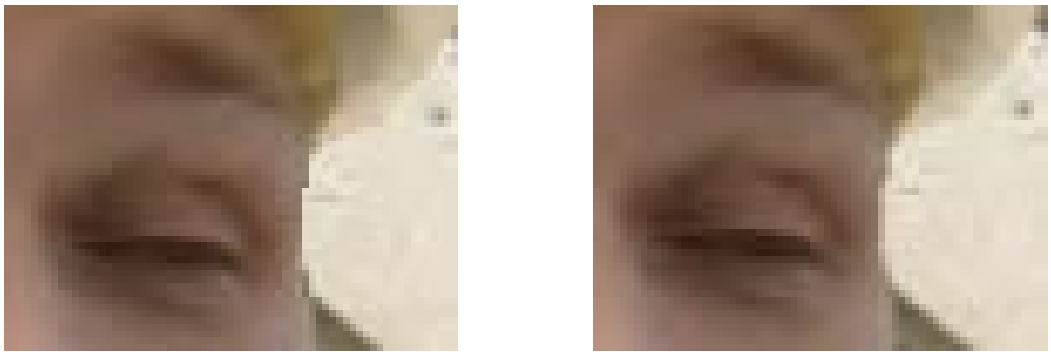
FIG. 2.10 – *Illustration de la composition d'objet. La technique de padding multirésolution permet de régler les problèmes de zones non définies lors du décodage. De plus, le padding permet de détériorer la forme sans que cela soit visible au décodage.*

(a) masque IPBWavelet 6Kb/s

(b) masque avec antialiasing

(c) composition sans antialiasing

(d) composition avec antialiasing



(e) agrandissement sans antialiasing

(f) agrandissement avec antialiasing

FIG. 2.11 – *Illustration de la composition d'objets avec ou sans antialiasing. L'antialiasing recrée l'effet naturel de discrétisation d'un contour lors de l'acquisition d'une image par un capteur*

Chapitre 3

Présentation des résultats : intérêt du codage objet et du codage dynamique

Ce chapitre illustre les résultats du codage dynamique sur trois séquences : Foreman, Stefan, Escalier. Les courbes de débit-distortion ainsi que la répartition des débits entre chaque objet sont données. Plusieurs figures permettent une évaluation visuelle.

3.1 La séquence Foreman et Stephan

Pour la séquence Foreman CIF à 15Hz et la séquence Stefan 352 × 240 à 15Hz, l'arrière-plan est codé avec trois codeurs : H264/AVC objet, ondelette 3D et mosaïque (le codeur m3dcoder échoue par manque d'information 3D). L'avant-plan est codé par H264/AVC objet et par ondelette 3D pour la séquence Foreman et seulement avec H264/AVC objet pour la séquence Stefan. En effet, le maillage suit difficilement le mouvement avant-plan et le codage par ondelette 3D devient bien moins performant par rapport à H264/AVC. Les courbes débit-distortion fig.3.1, fig.3.2 et fig.3.4, fig.3.5 illustrent les résultats de codage pour l'avant-plan et l'arrière plan.

On met en place le codage dynamique sachant les courbes débit-distortion. Le tableau 3.1 illustre la répartition des débits ainsi que le choix du codeur par objet pour la séquence Foreman. On constate qu'il peut y avoir des écarts allant jusqu'à 2 décibels entre les deux objets pour une contrainte de débit donnée. On répartit donc les débits tout en imposant une répartition des qualités de manière équitable. Les tableaux 3.2 et 3.4 donnent les résultats obtenus avec cette approche. Les qualités sont alors mieux réparties et pour un débit donné, on observe une faible réduction de qualité globale pour un équilibrage des qualités.

Les résultats des deux tableaux 3.2 et 3.4 montrent qu'en fonction du débit imposé, le codage de l'arrière-plan est effectué par mosaïque (très faible débit), par ondelette 3D (faible débit) ou bien par H264/AVC objet (moyen débit). Le codage dynamique change de codeur en fonction du débit objectif. Ainsi, il n'y a pas un codeur surpassant tous les autres. Pour une application désirant une propriété de hiérarchisation, l'approche dynamique devient alors problématique. En effet, il faut d'abord que tous les codeurs proposent

la hiérarchisation et ensuite, on ne peut générer un flux hiérarchisé que sur le petit intervalle de débit ou chaque codeur est le plus performant. Ainsi, en imaginant dans notre cas que les codeurs aient la propriété de hiérarchisation désirée, on aurait pour la séquence Foreman un flux hiérarchique sur l'intervalle]...,172], un flux hiérarchique sur l'intervalle [233,265], et un flux hiérarchique sur l'intervalle [300,...[et pour la séquence Stefan les intervalles]...,114], [193,268] et [312,...[.

Pour un codage dynamique à 100Kb/s, pour la séquence Foreman, on obtient une qualité légèrement supérieure à H264/AVC non objet (33,4 dB contre 32,9 dB), tout en proposant une hiérarchisation objet (voir tableau 3.3). Pour la séquence Stefan, à 100Kb/s on obtient aussi une qualité supérieure avec le codage dynamique (27.2 dB contre 26,7 dB à 105 Kb/s pour H264/AVC non objet) (voir tableau 3.5).

Les figures 3.3 et 3.6 montrent quelques images des séquences codées à 100 Kbits/s par codage dynamique objet et par H264/AVC non objet. Il est intéressant de remarquer que pour le codage dynamique sur la séquence Foreman (respectivement sur la séquence Stefan) 81% (resp. 70%) du débit total est consacré au codage du mouvement et de la texture du visage (resp. du joueur de tennis) et seulement 13% (resp. 25%) au codage de la texture et du mouvement du fond et 6% (resp. 5%) au codage du contour. Cela s'explique par le fait que le débit est donné aux objets étant peu stables temporellement c'est à dire ayant des variations d'illumination, des mouvements brusques, des mouvement non rigide, et des auto-occultations.

Remarquons que le codage dynamique à 100Kb/s de la séquence Stefan donne un arrière-plan dont les lignes du terrain de tennis ne sont pas droites. Cet artefact est dû à une erreur d'estimation dans le mouvement. D'autres expérimentations ont montré qu'il est possible d'obtenir une meilleure estimation du mouvement et de construire une mosaïque de meilleure qualité. Ainsi, les résultats du codage dynamique seraient plus agréables visuellement.

Sur la séquence Stefan, autour de 30 dB, c'est le codage H264/AVC non objet qui est le plus performant (voir tableau 3.6 et figure 3.7). On peut remarquer qu'en règle général, le codage dynamique d'objet vidéo est plus performant qu'H264/AVC non objet pour les faibles débits. Par contre, pour les débits plus élevés, en général, chaque objet est codé en H264/AVC ce qui est moins performant que de coder la séquence entière par H264/AVC non objet. En effet, le codage objet nécessite un coût supplémentaire pour la carte de segmentation.

Pour ce qui est de la comparaison entre H264/AVC et le codeur ondelette 3D, on peut remarquer que le codeur ondelette 3D ne peut être supérieur au codeur H264/AVC que lorsque l'objet possède un mouvement peu complexe et est estimé sur un GOP de grande taille. En effet, sur des objets ayant un mouvement comme celui du visage de la séquence Foreman, le codeur H264/AVC est largement plus performant. À 100Kbits/s il y a 8 dB de différence pour le codage du visage avec 34,6 dB pour H264/AVC objet et 26,6 dB pour ondelette 3D objet.

On constate visuellement sur la figure 3.3, pour le codage à 100 Kb/s, que la texture du visage est légèrement meilleure pour le codage dynamique. Par contre il est difficile de comparer la texture de l'arrière-plan : les artefacts ne sont pas du même type. Pour le codage dynamique, les artefacts sont les rebonds dûs au codage ondelette. Pour le codage par H264/AVC, les artefacts sont l'effet de bloc et le lissage entraînant la perte de texture. Ce qui nous fait dire encore une fois qu'il manque d'une mesure d'évaluation ou

de comparaison de qualité.

On peut aussi remarquer que l'on a omis le logo dans notre séquence reconstruite. Cependant, le coût de codage d'une zone aussi petite pour toute la séquence (150 images) est faible puisqu'il est inférieur à 1 Kbits/s. Enfin, le codage avec perte sur la forme n'a pas d'effet visible puisque lors de la reconstruction, les techniques de padding de texture ainsi que d'antialiasing permettent de faire disparaître les artefacts de composition.

Pour la séquence Stefan, on peut remarquer que les masques utilisés pour l'avant-plan (joueur de tennis) sont légèrement trop grands. Ainsi, lors de la composition, l'effet de « patch » est visible. De plus, du débit est perdu lors du codage de l'avant-plan puisque l'on a à coder des textures, situées aux zones fontière et ayant une forte activité spatial, qui ne devraient pas être présentes. L'influence de la segmentation est donc particulièrement importante.

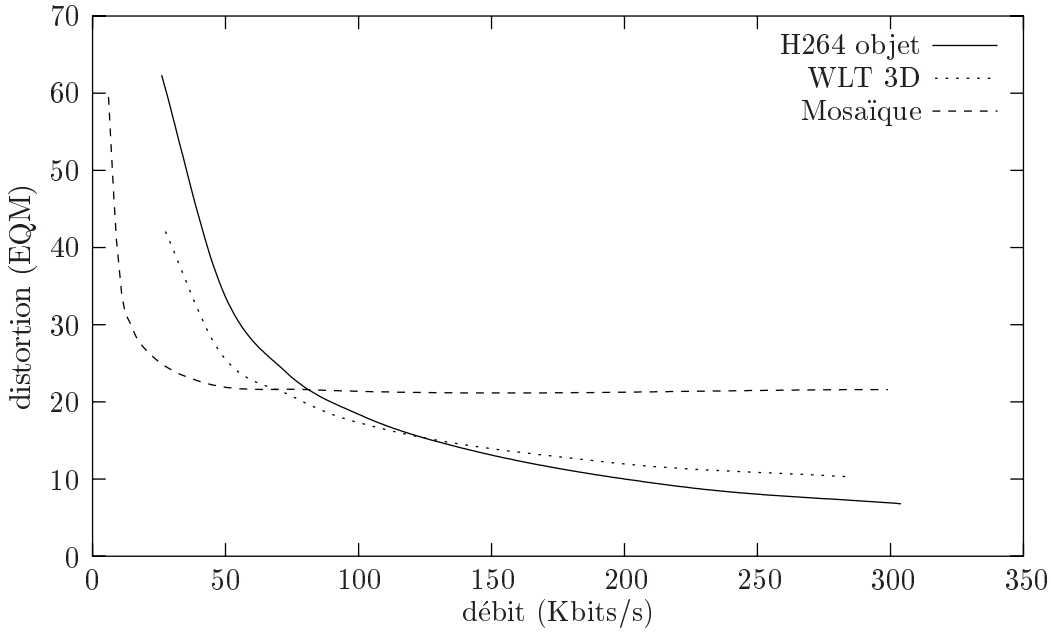


FIG. 3.1 – *Courbe de la distortion (moyenne sur 60 images de l'erreur quadratique moyenne) en fonction du débit. Le codage porte sur l'arrière-plan de la séquence CIF Foreman à 15Hz. Les codeurs utilisés sont : H264/AVC adapté objet, ondelette 3D (WLT 3D) et mosaïque avec un mouvement affine*

3.2 La séquence Escalier

La séquence Escalier 360 × 288 à 25 Hz est présentée pour illustrer le codage dynamique non objet puisque l'on a retenu un seul objet dans cette séquence : la scène tout entière. Trois codeurs sont mis en concurence : H264/AVC, ondelette 3D et m3dcoder. Les résultats (voir figure 3.8) montrent que le codeur m3dcoder permet d'obtenir une bonne qualité de texture à faible débit (pour un débit de 100Kb/s le PSNR texture de m3dCoder est de 31.3 (voir figure 3.9)). Bien entendu, la mesure de PSNR texture pour le codeur m3dcoder ne prend pas en compte les variations de texture à l'intérieur d'un GOP. Ainsi, cette mesure

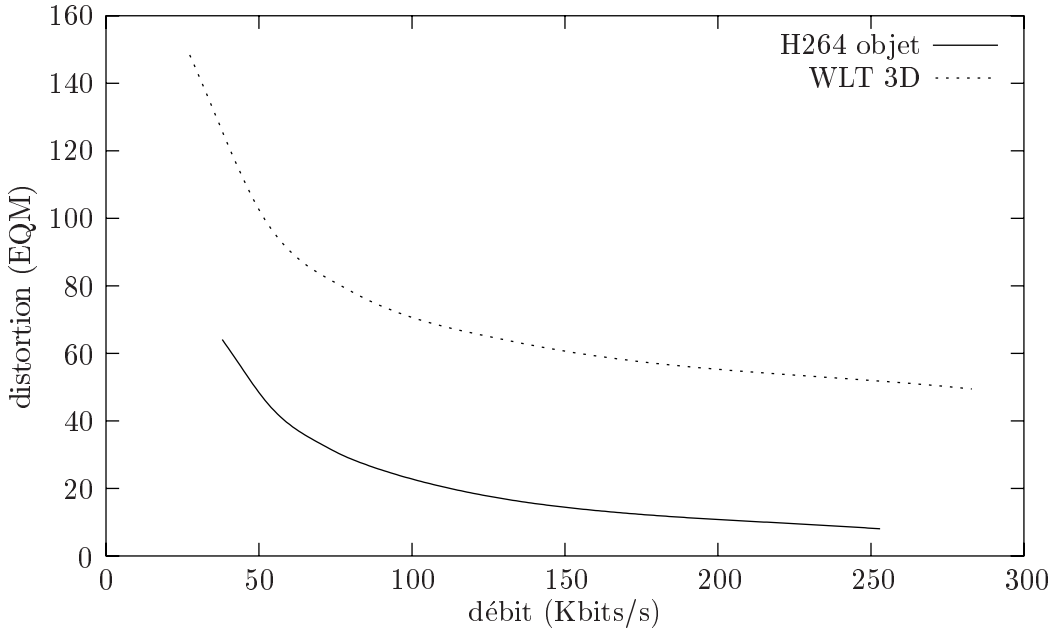


FIG. 3.2 – *Courbe de la distortion (moyenne sur 60 images de l'erreur quadratique moyenne) en fonction du débit. Le codage porte sur l'avant-plan de la séquence CIF Foreman à 15Hz. Les codeurs utilisés sont : H264/AVC adapté objet, et ondelette 3D (WLT 3D)*

ne reflète pas toutes les distortions. En particulier, la zone de l'image où l'eau ruisselle est statique après décodage, ce qui n'est pas réaliste.

Ainsi, le codage dynamique permet cette souplesse de ne proposer non pas un seul mais plusieurs types de codage. Cette souplesse est bien entendu gagnée au prix d'une grande complexité calculatoire.

débit	arrière-plan			avant-plan			PSNR global
	débit	EQM	PSNR	débit	EQM	PSNR	$-10log(\frac{\sum_{obj}\frac{EQM_{obj}}{nbObj}}{255^2})$
75 Kb/s	SPRITE			H264 objet			32.8
	16 Kb/s	28.6735	33.56	59 Kb/s	39.6573	32.14	
100 Kb/s	SPRITE			H264 objet			33.7
	20 Kb/s	26.7716	33.85	80 Kb/s	28.7663	33.54	
150 Kb/s	SPRITE			H264 objet			34.8
	30 Kb/s	24.0745	34.32	120 Kb/s	18.5618	35.44	
175 Kb/s	SPRITE			H264 objet			35.2
	36 Kb/s	23.1680	34.48	139 Kb/s	15.6802	36.17	
200 Kb/s	SPRITE			H264 objet			35.6
	43 Kb/s	22.3775	34.63	157 Kb/s	13.7494	36.75	
263 Kb/s	WLT 3D			H264 objet			36.3
	104 Kb/s	16.9436	35.84	159 Kb/s	13.5678	36.08	
272 Kb/s	WLT 3D			H264 objet			36.4
	109 Kb/s	16.5115	35.95	163 Kb/s	13.2214	36.92	
305 Kb/s	H264 objet			H264 objet			36.8
	142 Kb/s	13.7552	36.76	163 Kb/s	13.2214	36.92	

TAB. 3.1 – *Répartition de débit entre l'avant-plan et l'arrière-plan et choix du codeur pour la séquence Foreman CIF 15Hz. La répartition est obtenue par une optimisation débit distortion avec contrainte sur le débit (la distortion utilisée est l'erreur quadratique moyenne)*

débit	arrière-plan			avant-plan			PSNR global
	débit	EQM	PSNR	débit	EQM	PSNR	$-10log(\frac{\sum_{obj}\frac{EQM_{obj}}{nbObj}}{255^2})$
74 Kb/s	SPRITE			H264 objet			32.6
	11 Kb/s	33.7435	32.85	63 Kb/s	37.0029	32.45	
93 Kb/s	SPRITE			H264 objet			33.4
	13 Kb/s	31.0084	33.22	80 Kb/s	33.5420	33.54	
100 Kb/s	SPRITE			H264 objet			33.7
	17 Kb/s	28.0313	33.65	83 Kb/s	27.7055	33.71	
149 Kb/s	SPRITE			H264 objet			34.7
	43 Kb/s	22.3775	34.63	106 Kb/s	21.3857	34.83	
172 Kb/s	SPRITE			H264 objet			35.2
	43 Kb/s	22.3775	34.63	129 Kb/s	17.0702	35.81	
233 Kb/s	WLT 3D			H264 objet			35.8
	104 Kb/s	16.9437	35.8	129 Kb/s	17.0702	35.81	
250 Kb/s	WLT 3D			H264 objet			36.1
	109 Kb/s	16.5115	35.95	141 Kb/s	15.4328	36.25	
265 Kb/s	WLT 3D			H264 objet			36.3
	109 Kb/s	16.5115	35.95	156 Kb/s	13.8422	36.72	
300 Kb/s	H264 objet			H264 objet			36.8
	143 Kb/s	13.6717	36.77	157 Kb/s	13.7494	36.75	

TAB. 3.2 – *Répartition de débit entre l'avant-plan et l'arrière-plan et choix du codeur pour la séquence Foreman CIF 15Hz. La répartition est obtenue par une optimisation débit distortion avec contrainte sur le débit et à qualité égale entre objets (la distortion utilisée est l'erreur quadratique moyenne)*

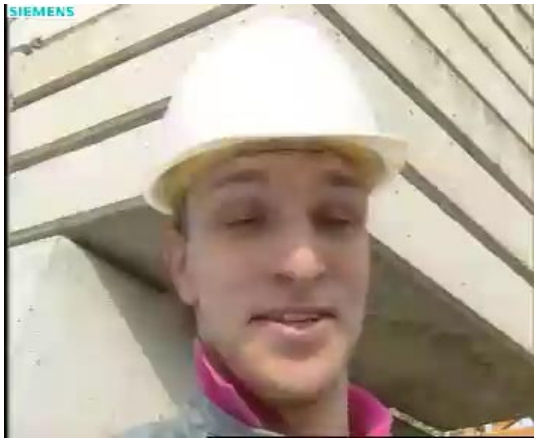
	Codage Dynamique	H264 non objet
objet avant-plan :	**H264 objet**	
débit	80 Kb/s	
PSNR	33.54	
codage de forme :	**IPB Wavelet**	
débit	6 Kb/s	
objet arrière-plan :	**SPRITE**	
débit	13 Kb/s	
PSNR	33.22	
séquence reconstruite :		
débit :	99 Kb/s	100 Kb/s
PSNR :	33.4	32.9

TAB. 3.3 – *Comparaison entre un codage dynamique objet et un codage H264/AVC JM5 pour la séquence Foreman CIF 15Hz. Le codage dynamique obtient une meilleure qualité PSNR*

débit	arrière-plan			avant-plan			PSNR global
	débit	EQM	PSNR	débit	EQM	PSNR	$-10log(\frac{\sum_{obj}\frac{EQM_{obj}}{nbObj}}{255^2})$
95 Kb/s	SPRITE			H264 objet			27.17
	25 Kb/s	128.7934	27.03	70 Kb/s	120.7614	27.31	
97 Kb/s	SPRITE			H264 objet			27.23
	25 Kb/s	128.7934	27.03	72 Kb/s	117.0708	27.45	
114 Kb/s	SPRITE			H264 objet			27.51
	41 Kb/s	110.4191	27.7	73 Kb/s	115.3583	27.51	
193 Kb/s	WLT3D			H264 objet			28.38
	120 Kb/s	73.3487	29.48	73 Kb/s	115.3584	27.51	
250 Kb/s	WLT3D			H264 objet			30.02
	142 Kb/s	64.5742	30.03	108 Kb/s	64.7878	30.02	
268 Kb/s	WLT3D			H264 objet			30.3
	156 Kb/s	60.42924	30.32	112 Kb/s	60.8965	30.28	
312 Kb/s	H264 objet			H264 objet			30.7
	200 Kb/s	49.69316	31.16	112 Kb/s	60.8965	3.28	
350 Kb/s	H264 objet			H264 objet			31.53
	219 Kb/s	45.50757	31.55	131 Kb/s	46.0389	31.5	
400 Kb/s	H264 objet			H264 objet			32.22
	256 Kb/s	39.12503	32.21	144 Kb/s	38.7956	32.2	

TAB. 3.4 – *Répartition de débit entre l'avant-plan et l'arrière-plan et choix du codeur pour la séquence Stefan 240 × 352 à 15Hz. La répartition est obtenue par une optimisation débit distortion avec contrainte sur le débit et à qualité égale entre objets (la distortion utilisée est l'erreur quadratique moyenne)*

(a) image 0 - Codage Dynamique objet



(b) image 0 - Codage H264 non objet

(c) image 10 - Codage Dynamique objet

(d) image 10 - Codage H264 non objet

(e) image 20 - Codage Dynamique objet

(f) image 20 - Codage H264 non objet

FIG. 3.3 – *Images de la séquence Foreman codé décodé par codage dynamique objet à 99Kb/s et par codage H264/AVC non objet à 100Kb/s. Le tableau 3.3 donne la répartition des débits*

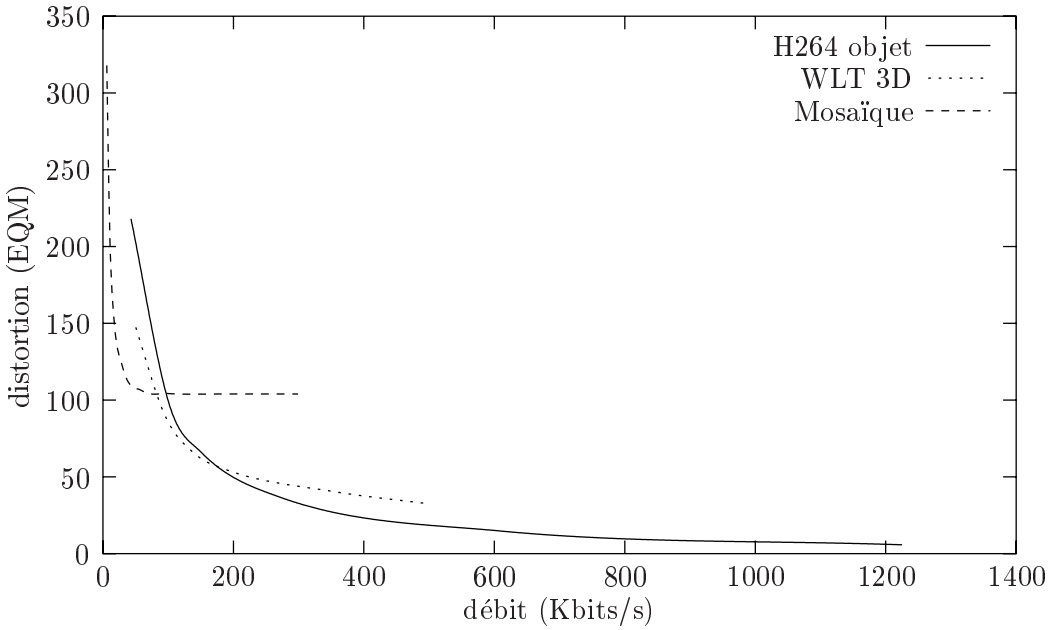


FIG. 3.4 – *Courbe de la distortion (moyenne sur 90 images de l'erreur quadratique moyenne) en fonction du débit. Le codage porte sur l'arrière-plan de la séquence Stefan 240 × 352 à 15Hz. Les codeurs utilisés sont : H264/AVC adapté objet, ondelette 3D (WLT 3D) et mosaïque avec un mouvement affine*

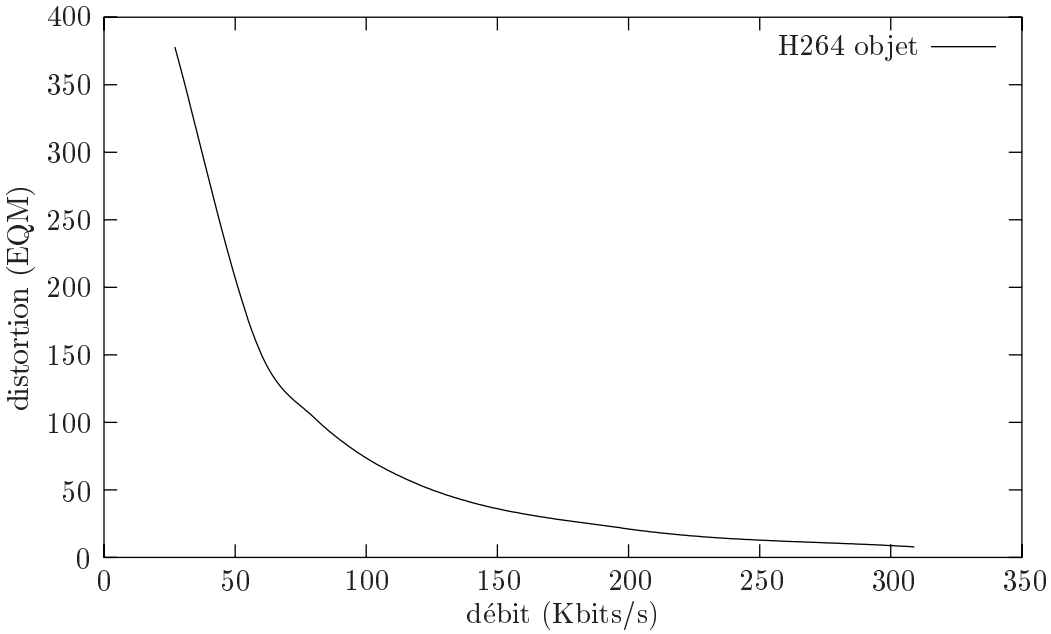


FIG. 3.5 – *Courbe de la distortion (moyenne sur 90 images de l'erreur quadratique moyenne) en fonction du débit. Le codage porte sur l'avant-plan de la séquence Stefan 240 × 352 à 15Hz. Le codeur utilisé est : H264/AVC adapté objet*

	Codage Dynamique	H264 non objet
objet avant-plan :	**H264 objet**	
débit	70 Kb/s	
PSNR	27.31	
codage de forme :	**IPB Wavelet**	
débit	5 Kb/s	
objet arrière-plan :	**SPRITE**	
débit	25 Kb/s	
PSNR	27.03	
séquence reconstruite :		
débit :	100 Kb/s	105 Kb/s
PSNR :	27.2	26.7

TAB. 3.5 – *Comparaison entre un codage dynamique objet et un codage H264/AVC JM5 pour la séquence Stefan 240 × 352 à 15Hz. Le codage dynamique obtient une meilleure qualité PSNR bien que de manière subjective (voir figure 3.6), le codage H264/AVC non objet semble meilleur. Ceci vient du fait que la mosaïque est mal construite (erreur sur l'estimation du mouvement)*

	Codage Dynamique	H264 non objet
objet avant-plan :	**H264 objet**	
débit	112 Kb/s	
PSNR	30.28	
codage de forme :	**IPB Wavelet**	
débit	5 Kb/s	
objet arrière-plan :	**WLT 3D**	
débit	155 Kb/s	
PSNR	30.18	
séquence reconstruite :		
débit :	272 Kb/s	256 Kb/s
PSNR :	30.3	30.9

TAB. 3.6 – *Comparaison entre un codage dynamique objet et un codage H264/AVC JM5 pour la séquence Stefan 240 × 352 à 15Hz. Ici, le codage H264/AVC non objet est plus performant*

(a) image 0 - Codage Dynamique objet

(b) image 0 - Codage H264/AVC non objet

(c) image 10 - Codage Dynamique objet

(d) image 10 - Codage H264/AVC non objet

(e) image 20 - Codage Dynamique objet

(f) image 20 - Codage H264/AVC non objet

FIG. 3.6 – *Images de la séquence Stefan codée décodée par codage dynamique objet à 100Kb/s (PSNR = 27,2) et par codage H264/AVC non objet à 105Kb/s (PSNR = 26,7)*

(a) image 0 - Codage Dynamique objet

(b) image 0 - Codage H264/AVC non objet

(c) image 10 - Codage Dynamique objet

(d) image 10 - Codage H264/AVC non objet

(e) image 20 - Codage Dynamique objet

(f) image 20 - Codage H264/AVC non objet

FIG. 3.7 – *Images de la séquence Stefan codée décodée par codage dynamique objet à 272Kb/s (PSNR = 30.3) et par codage H264/AVC non objet à 256Kb/s (PSNR = 30.9)*

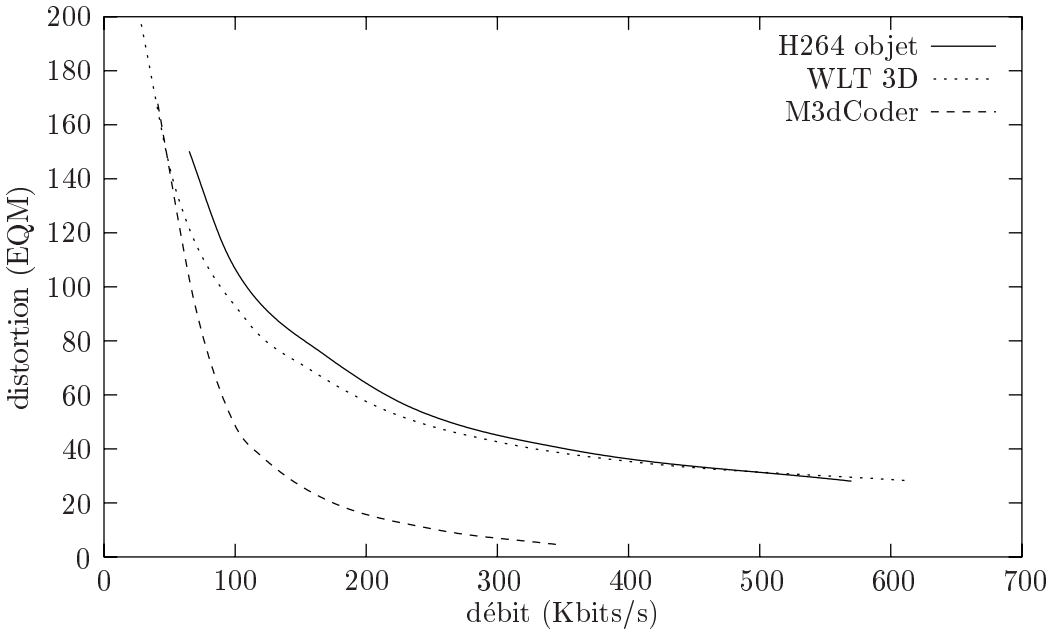
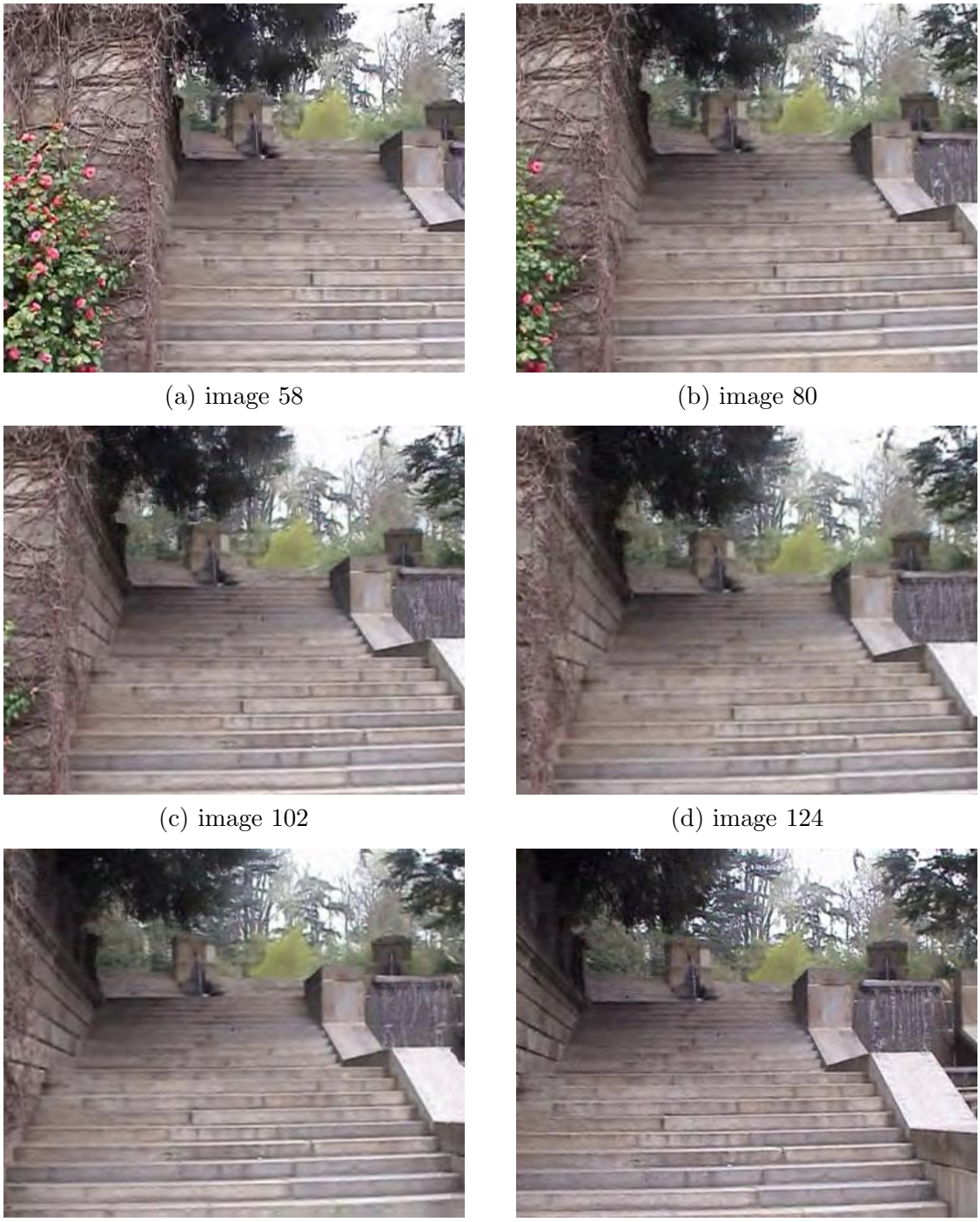


FIG. 3.8 – *Courbe de la distortion (moyenne sur 110 images de l'erreur quadratique moyenne) en fonction du débit. Le codage porte sur la séquence Escalier à 25Hz. En entrée de m3dCodeur, la séquence est de taille 360 × 288. En entrée de H264/AVC et ondelette 3D (WLT 3D), la séquence est rognée à une taille de 352 × 288*



(a) image 58 (b) image 80

(c) image 102 (d) image 124

(e) image 146 (f) image 168

FIG. 3.9 – *Images de la séquence Escalier codée décodée par m3dCoder à 100Kb/s (PSNR = 31.3)*

Chapitre 4

Conclusion de la troisième partie

4.1 Conclusion

Dans cette partie, nous avons présenté le codage dynamique et proposé un schéma de codage dynamique d'objet vidéo. Ce schéma de codage passe par la définition de métrique de distortion et par la mise en concurrence de codeurs via une optimisation débit-distortion. Les codeurs retenus sont le codeur m3dcoder, le codeur H264/AVC objet, le codeur ondelette 3D et le codage par mosaïque et mouvement affine.

Les résultats obtenus montrent que le codage dynamique permet de valoriser l'approche objet car elle permet d'obtenir de très bonnes performances. En particulier, sur les exemples proposés, à faible débit, le codage dynamique surpasse en PSNR, H264/AVC non objet. De plus l'utilisation du padding et de l'antialliasing lors de la composition de la séquence après du décodage permet d'avoir des résultats de comparaisons visuels satisfaisants.

4.2 Perspective

Le codage dynamique est intéressant car il permet d'utiliser le meilleur des codages par type de signal c'est-à-dire par type d'objet. Il permet de même de valoriser l'approche objet. Par contre, la complexité opératoire est particulièrement importante. De plus, comme on l'a vu, la propriété de hiérarchisation est difficile à mettre en œuvre.

Pour avoir une hiérarchisation il peut être préférable d'utiliser un seul codeur hiérarchique pour toutes les gammes de débit, et ceci même si l'on code de manière moins performante. Cependant, autant le codeur ondelette 3D se comporte assez bien pour les faibles débits et sur des signaux ayant un mouvement facilement estimable par maillage, autant il est vraiment peu performant lorsqu'il y a de forts mouvements ou des mouvements complexes (voir les développements effectués dans la thèse de N. Cammas). Le codeur H264/AVC quant à lui ne propose pas encore une totale hiérarchisation du flux. Il y a donc des efforts à faire sur ces deux codeurs.

Conclusion générale

Le codage par objets est-il intéressant ?

Dans cette thèse nous avons abordé le schéma de codage automatique d'objet vidéo. Les objectifs étaient : premièrement de valider l'hypothèse selon laquelle le codage par objets vidéo permet d'obtenir des gains significatifs en utilisant le codage dynamique, et deuxièmement de savoir si la segmentation automatique était réaliste.

Les résultats des trois parties permettent de conclure que le codage par objets vidéo permet d'obtenir des résultats significatifs par rapport au codage standard non objet. Ainsi, pour le codage ondelette 3D, à même débit et même distorsion, on obtient une meilleure qualité visuelle dans les zones d'occultations pour le codage objet. L'approche de codage par objet permet de plus de proposer une hiérarchisation objet.

En mettant en place un codage dynamique, c'est-à-dire en mettant en concurrence plusieurs codeurs par objets, on obtient des résultats dépassant H264/AVC à très faible débits, et équivalent H264/AVC à faible débit. Pour exemple, sur la séquence Foreman CIF 15Hz, on obtient à 100Kb/s un gain de 0,5 dB par rapport à H264/AVC JM5. Ainsi, le codage d'objet vidéo permet d'obtenir des gains significatifs par rapport aux approches non objet tout en proposant la hiérarchisation en objet.

Pour ce qui est de la segmentation automatique ou semi-automatique, les résultats de l'approche séquentielle pour obtenir des objets spécifiques suivie éventuellement d'une approche par clustering 3D montrent qu'il est possible d'obtenir des objets de manière automatique dans des cas simples. Par contre, dès que l'on a des séquences ayant beaucoup de mouvements, il est difficile de trouver précisément les objets.

De plus, nous avons vu que la hiérarchisation du flux d'un objet est une propriété qu'il est possible d'obtenir. Cette propriété permet de répartir indépendamment le débit entre la forme, le mouvement, et la texture. Ainsi, le codage par objet présente de nombreux intérêts, le seul réel problème reste son automatisation par segmentation automatique. Les 2 points essentiels à retenir en ce qui concerne la segmentation automatique sont que d'une part il est possible d'obtenir des segmentations peu éloignées des objets mais que d'autre part les segmentations pour le codage objet ont besoin d'être sur les frontières de texture. En effet, dans le cas contraire l'effet de « patch » est visible lors de la composition.

Contributions

Dans la première partie, nous avons étudié le domaine de la segmentation automatique et proposé une modélisation d'un objet vidéo. Notre modèle fait intervenir la notion de

suivi long terme via la représentation d'un objet sous la forme mouvement/texture. Nous utilisons un maillage actif pour représenter le mouvement. Un algorithme de clustering 3D a été développé basé sur ce modèle. Les atouts de cette approche sont l'utilisation du long terme dans la segmentation (ce qui permet de disposer d'une grande quantité d'information et ainsi de gérer plus facilement les problèmes d'occultations) et l'utilisation d'un modèle de mouvement fin. De plus, l'introduction de la séquentialité dans la segmentation automatique permet d'obtenir des objets pour les séquences simples.

Dans la seconde partie du manuscrit, nous avons exploré la hiérarchisation du flux objet. Pour cela, nous utilisons un schéma de codage ondelette 3D et nous introduisons notamment un codage de contours avec perte. La décorrélation des trois informations texture, mouvement et forme permet de mieux répartir les débits. Cela permet aussi de hiérarchiser aisément le flux sur chacune de ces informations. En ce qui concerne le codage de contour, nous proposons d'introduire une plus grande stabilité temporelle des contours. L'utilisation du padding pour fermer les contours renforce cette stabilité. Le codage issu de cette représentation permet d'obtenir pour le codage de contour jusqu'à 60% de gain par rapport à MPEG4 CAE à très faible débit. De plus, cette représentation est très facilement hiérarchisable.

Dans la dernière partie du manuscrit, nous avons étudié le codage dynamique d'objets vidéo (mise en concurrence de plusieurs codeurs pour chaque objet vidéo). Les codeurs utilisés sont le codeur H264/AVC, un codeur ondelette 3D, un codeur 3D et un codeur par mosaïque. Nous proposons une répartition automatique des débits ce qui permet d'obtenir des résultats supérieur à chaque codeur pris séparément, tout en offrant le découpage du flux en objets vidéo. Nous proposons de gérer les zones de textures non définies par un padding multirésolution. Nous proposons aussi de recomposer la séquence en supprimant l'effet d'aliasing. Enfin de nombreux résultats permettent d'évaluer l'approche par codage dynamique.

Perspectives

Le travail présenté dans cette thèse peut être étendu que ce soit dans le domaine de la segmentation ou dans celui du codage. Nous présentons ici une liste non exhaustive de perspectives.

En ce qui concerne la segmentation il peut être intéressant de trouver une technique de détermination du nombre d'objets présents dans une séquence. L'approche séquentielle de la segmentation permet de trouver quelques objets (on part de modèles simples vers des modèles compliqués, on ajoute des a priori). Le clustering affine permet de trouver un nombre de germes déterminés. Les approches par tubes permettent de donner un nombre limité de régions ayant des propriétés d'homogénéité spatio-temporelle. L'ensemble de ces techniques pourraient être utilisées pour résoudre ce problème.

En ce qui concerne le codage hiérarchique ondelette 3D objet, la technique objet ou bien non objet représente une très bonne approche du codage pour obtenir les fonctionnalités de hiérarchisation. Ainsi, l'amélioration du schéma de codage semble particulièrement intéressante. Par exemple, la construction de mosaïques, par projection des textures via un maillage, posent des problèmes d'échantillonnage. On pourrai envisager un échantillonage variable ainsi qu'une représentation hiérarchique de cette échantillonage.

Plus généralement, un positionnement par rapport aux autres techniques hiérarchique pourrait être envisagé ainsi qu'une évaluation de l'impact de la répartition des débit entre le mouvement la texture et la forme. Pour le codage de contour, il faudrait pouvoir généraliser l'approche à n'importe quel type de forme. Les notions de taille de GOP, de z-order local, de mesure de distorsion serait à approfondir.

Pour le codage dynamique, il serait intéressant de comparer l'impact psychovisuel entre codage objet, non objet, dynamique et non dynamique. Dans le même ordre d'idées, il serait intéressant de comparer en gain et en qualité psychovisuelle le codage d'objets issus d'une segmentation automatique et le codage non objet H264/AVC. La scalabilité du codage dynamique pourrait aussi être envisagé via l'utilisation de codeurs hiérarchiques et en faisant évoluer les codeurs suivant le débit. On peut très bien envisager comme couche basse un flux codé par un premier codeur et comme couche haute un flux codé par un deuxième codeur et utilisant le flux décodé de la couche basse.

Annexe A

Détail des calculs du clustering affine

A.1 Détail des calculs des paramètres affines $A_{k,t}$ et $T_{k,t}$

La minimisation de la fonctionnelle E (équation 2.7) permet d'obtenir les paramètres du mouvement affine (2.8) : ax, ay, $a00$, $a01$, $a10$ et $a11$. On effectue l'annulation de la dérivée partielle de E pour un temps t et un germe k, en fonction de chaque paramètre mvt (équation A.1). L'annulation de chaque dérivée donne ax et ay (équation A.2 et A.3) et un système linéaire (équations en annexe A.4, A.5 A.6, A.7 de l'index) à résoudre pour obtenir $a00$, $a01$, $a10$ et $a11$:

$$\frac{\partial E}{\partial mvt} = \frac{\partial}{\partial mvt}\left(\sum_{i=1}^{i=N} P_i^m \left\| \begin{pmatrix} X_{obs}(i) \\ Y_{obs}(i) \end{pmatrix} - \begin{pmatrix} ax + a00(X_{ref}(i) - \overline{X_{ref}}) + a01(Y_{ref}(i) - \overline{Y_{ref}}) \\ ay + a10(X_{ref}(i) - \overline{X_{ref}}) + a11(Y_{ref}(i) - \overline{Y_{ref}}) \end{pmatrix} \right\|^2\right), \tag{A.1}$$

$$\frac{\partial E}{\partial ax} = 0 \quad \Rightarrow \quad ax = \frac{\sum_{i=1}^{i=N} P_i^m X_{obs}(i)}{\sum_{i=1}^{i=N} P_i^m} = \overline{X_{obs}}, \tag{A.2}$$

$$\frac{\partial E}{\partial ay} = 0 \quad \Rightarrow \quad ay = \frac{\sum_{i=1}^{i=N} P_i^m Y_{obs}(i)}{\sum_{i=1}^{i=N} P_i^m} = \overline{Y_{obs}}, \tag{A.3}$$

$$\begin{aligned}
\frac{\partial E}{\partial a00} = 0 \quad \Rightarrow \quad & a00 \times \underbrace{\sum_{i=1}^{i=N} P_i^m (X_{ref}(i) - \overline{X_{ref}})(X_{ref}(i) - \overline{X_{ref}})}_{A00} \\
+ \quad & a01 \times \underbrace{\sum_{i=1}^{i=N} P_i^m (X_{ref}(i) - \overline{X_{ref}})(Y_{ref}(i) - \overline{Y_{ref}})}_{A01} \\
= \quad & \underbrace{\sum_{i=1}^{i=N} P_i^m (X_{ref}(i) - \overline{X_{ref}})(X_{obs}(i) - \overline{X_{obs}}),}_{B0}
\end{aligned} \tag{A.4}$$

$$\begin{aligned}
\frac{\partial E}{\partial a01} = 0 \quad \Rightarrow \quad & a00 \times \underbrace{\sum_{i=1}^{i=N} P_i^m (Y_{ref}(i) - \overline{Y_{ref}})(X_{ref}(i) - \overline{X_{ref}})}_{A10} \\
+ \quad & a01 \times \underbrace{\sum_{i=1}^{i=N} P_i^m (Y_{ref}(i) - \overline{Y_{ref}})(Y_{ref}(i) - \overline{Y_{ref}})}_{A11} \\
= \quad & \underbrace{\sum_{i=1}^{i=N} P_i^m (Y_{ref}(i) - \overline{Y_{ref}}))(X_{obs}(i) - \overline{X_{obs}}),}_{B1}
\end{aligned} \tag{A.5}$$

$$\begin{aligned}
\frac{\partial E}{\partial a10} = 0 \quad \Rightarrow \quad & a10 \times \underbrace{\sum_{i=1}^{i=N} P_i^m (X_{ref}(i) - \overline{X_{ref}})(X_{ref}(i) - \overline{X_{ref}})}_{A00} \\
+ \quad & a11 \times \underbrace{\sum_{i=1}^{i=N} P_i^m (X_{ref}(i) - \overline{X_{ref}})(Y_{ref}(i) - \overline{Y_{ref}})}_{A01} \\
= \quad & \underbrace{\sum_{i=1}^{i=N} P_i^m (X_{ref}(i) - \overline{X_{ref}})(Y_{obs}(i) - \overline{Y_{obs}}),}_{B2}
\end{aligned} \tag{A.6}$$

$$\begin{aligned}\frac{\partial E}{\partial a11} = 0 \quad \Rightarrow \qquad & a10 \times \underbrace{\sum_{i=1}^{i=N} P_i^m (Y_{ref}(i) - \overline{Y_{ref}})(X_{ref}(i) - \overline{X_{ref}})}_{A10} \\ + \quad & a11 \times \underbrace{\sum_{i=1}^{i=N} P_i^m (Y_{ref}(i) - \overline{Y_{ref}})(Y_{ref}(i) - \overline{Y_{ref}})}_{A11} \\ = \qquad & \underbrace{\sum_{i=1}^{i=N} P_i^m (Y_{ref}(i) - \overline{Y_{ref}})(Y_{obs}(i) - \overline{Y_{obs}})}_{B3},\end{aligned} \tag{A.7}$$

Les équations A.4, A.5, A.6, A.7, mènent au système linéaire suivant qui, après résolution, permet d'obtenir les paramètres $a00$, $a01$, $a10$ et $a11$:

$$\begin{pmatrix} A00 & A01 & 0 & 0 \\ A10 & A11 & 0 & 0 \\ 0 & 0 & A00 & A01 \\ 0 & 0 & A11 & A11 \end{pmatrix} \times \begin{pmatrix} a00 \\ a01 \\ a10 \\ a11 \end{pmatrix} = \begin{pmatrix} B0 \\ B1 \\ B2 \\ B3 \end{pmatrix}. \tag{A.8}$$

A.2 Détail du calcul des probabilités $P_{i,k}$

Pour calculer les probabilités $P_{i,k}$ il faut ajouter la contrainte suivante : $\forall i, \sum_{l=1}^{l=K} P_{i,l} = 1$. Ainsi, l'équation énergétique E (équation 2.7) se réécrit avec intégration de cette contrainte via l'utilisation des multiplicateurs de Lagrange :

$$E = \sum_{t=1}^{T-\Delta t} \sum_{k=1}^{K} \sum_{i=1}^{N} (P_{i,k}^m \times d_{i,k,t}^2) - \sum_{t=1}^{T-\Delta t} \sum_{i=1}^{N} \lambda_{i,t} \sum_{k=1}^{K} (P_{i,k} - 1).$$

La dérivée partielle de E en fonction de $P_{i,k}$ vaut alors :

$$\frac{\partial E}{\partial P_{i,k}} = m P_{i,k}^{m-1} \sum_{t=1}^{T-\Delta t} (d_{i,k,t}^2) - \sum_{t=1}^{T-\Delta t} (\lambda_{i,t}).$$

En annulant la dérivée partielle on obtient ainsi $P_{i,k}$:

$$\frac{\partial E}{\partial P_{i,k}} = 0 \quad \Rightarrow \quad P_{i,k} = \left(\frac{\sum_{t=1}^{T-\Delta t} \lambda_{t,i}}{m \sum_{t=1}^{T-\Delta t} d_{i,k,t}^2} \right)^{\frac{1}{m-1}}. \tag{A.9}$$

Il ne reste plus qu'à calculer la valeur de $\lambda_{t,i}$, en observant :

$$\forall i, \sum_{l=1}^{l=K} P_{i,l} = 1,$$

d'où :

$$\begin{aligned}
\sum_{l=1}^{l=K} \left(\frac{\sum_{t=1}^{T-\Delta t} \lambda_{t,i}}{m \sum_{t=1}^{T-\Delta t} d_{i,l,t}^2} \right)^{\frac{1}{m-1}} &= 1 \\
\left(\frac{\sum_{t=1}^{T-\Delta t} \lambda_{t,i}}{m} \right)^{\frac{1}{m-1}} &= \frac{1}{\sum_{l=1}^{l=K} \left(\frac{1}{\sum_{t=1}^{T-\Delta t} d_{i,l,t}^2} \right)^{\frac{1}{m-1}}}. \qquad \text{(A.10)}
\end{aligned}$$

On utilise alors l'équation A.10 pour substituer le Lagrangien dans l'équation A.9 et l'on obtient l'expression de la probabilité $P_{i,k}$:

$$\begin{aligned}
P_{i,k} &= \frac{1}{\sum_{l=1}^{l=K} \left(\frac{1}{\sum_{t=1}^{T-\Delta t} d_{i,l,t}^2} \right)^{\frac{1}{m-1}}} \times \left(\frac{1}{\sum_{t=1}^{T-\Delta t} d_{i,k,t}^2} \right)^{\frac{1}{m-1}} \\
P_{i,k} &= \frac{1}{\sum_{l=1}^{K} \left(\frac{\sum_{t=1}^{T-\Delta t} d_{i,k,t}^2}{\sum_{t=1}^{T-\Delta t} d_{i,l,t}^2} \right)^{\frac{1}{m-1}}}. \qquad \text{(A.11)}
\end{aligned}$$

Annexe B

Détail des calculs du clustering 3D

B.1 Détail des calculs

La résolution de l'équation E (équation 2.2) est effectuée par clustering 3D. Nous prenons des distances Euclidiennes et nous modifions l'équation énergétique E pour pouvoir conserver une équation du second degré :

$$E_{mod} = \sum_{t=1}^{T} \sum_{k=1}^{K} \sum_{i=1}^{N} \left\{ E^{d}_{i,k,t} + E^{rs}_{i,k,t} + {E^{rt}_{i,k,t}}' \right\}, \tag{B.1}$$

où :

$${E^{rt}_{i,k,t}}' = \beta Q^{2}_{i,k,t} \times dP^{2}_{i,k,t} + \gamma [P_{i,k,t} - Q_{i,k,t}]^{2}.$$

Les probabilités $Q_{i,k,t}$ sont introduites pour conserver une équation du second degré et représentent les probabilités qui imposent une continuité temporelle le long des trajectoires des objets. Le terme $\gamma[P_{i,k,t} - Q_{i,k,t}]^2$ permet alors d'assurer aux probabilités $Q_{i,k,t}$ d'être en accord avec les probabilités $P_{i,k,t}$.

Les trois sections suivantes détaillent le calcul de $M_k(i)$, $P_{i,k,t}$ et $Q_{i,k,t}$ en partant de l'équation B.1. Le principe est d'annuler les dérivées partielles. Pour $P_{i,k,t}$ et $Q_{i,k,t}$, on impose en plus les contraintes suivantes $\forall i, \sum_{l=1}^{l=K} P_{i,l} = 1$ et $\forall i, \sum_{l=1}^{l=K} Q_{i,l} = 1$.

Pour rappel, nous donnons l'équation détaillée de E_{mod} (équation B.1) dont nous devons calculer les dérivées partielles :

$$\begin{aligned}
E_{mod} &= \sum_{i,k,t} P^{2}_{i,k,t} \times [I_t(i) - M_k(\Theta_k^{t \to t_{ref}}(i))]^2 \\
&+ \sum_{i,k,t} \alpha \sum_{j \in \mathcal{V}(i)} [P_{i,k,t} - P_{j,k,t}]^2 \\
&+ \sum_{i,k,t} \beta Q^{2}_{i,k,t} \sum_{l=1}^{K} \left[[P_{i,l,t} - P_{\Theta_k^{t \to t-1}(i),l,t-1}]^2 + [P_{i,l,t} - P_{\Theta_k^{t \to t+1}(i),l,t+1}]^2 \right] \\
&+ \sum_{i,k,t} \gamma [P_{i,k,t} - Q_{i,k,t}]^2.
\end{aligned} \tag{B.2}$$

E_{mod} peut se réécrire en notation simplifiée par :

$$\begin{aligned} E_{mod} &= \sum_{i,k,t} P_{i,k,t}^2 \times dI_{i,k,t}^2 \\ &+ \sum_{i,k,t} \alpha \sum_{j \in \mathcal{V}(i)} [P_{i,k,t} - P_{j,k,t}]^2 \\ &+ \sum_{i,k,t} \beta Q_{i,k,t}^2 dP_{i,k,t}^2 \\ &+ \sum_{i,k,t} \gamma [P_{i,k,t} - Q_{i,k,t}]^2. \end{aligned}$$

B.2 Détail du calcul de l'équation des mosaïques $M_k(j)$

Le mouvement que nous utilisons est un mouvement réversible. Ainsi, nous avons pour n'importe quelle position i d'une image au temps t :

$$i = \Theta_k^{t_{ref} \to t}(\Theta_k^{t \to t_{ref}}(i)).$$

Le terme d'attache aux données mettant en jeu la distance entre une mosaïque M_k projetée et une image I_t est :

$$\sum_{i,k,t} P_{i,k,t}^2 \times [I_t(i) - M_k(\Theta_k^{t \to t_{ref}}(i))]^2, \tag{B.3}$$

Il faut noter qu'en soit, la minimisation de la fonctionelle B.3 est un problème difficile de par les problèmes d'échantillonages. C'est un problème de résolution inverse que l'on peut par exemple observer en super-résolution [Dekeyser 01]. On considère donc que la fonctionnelle B.3 peut se réécrire en la fonctionnelle B.4 pondérée par le jacobien de changement de coordonnées $J(i \to j)$ (voir [Marquant 00] p 277).

$$\sum_{j,k,t} P_{j,k,t}^2 \times [I_t(\Theta_k^{t_{ref} \to t}(j)) - M_k(j)]^2 \times J(i \to j). \tag{B.4}$$

L'optimisation de la fonction E_{mod} (B.2) est alors effectuée en substituant la somme B.4 à la somme B.3. Nous pouvons alors aisément calculer la dérivée partielle de E_{mod} en fonction de $M_k(j)$ et l'annuler. Nous retrouvons ainsi l'équation 2.12 de mise à jour des mosaïque de chaque objet. Il faut remarquer que la minimisation de l'équation E_{mod} dans un domaine continu mènerait à cette solution mais le passage dans le domaine discret est

effectué au prix de quelques approximations.

$$\begin{aligned}
\frac{\partial E_{mod}}{\partial M_k(j)} &= \sum_{t=1}^{t=T} -2P^2_{\Theta_k^{t_{ref}\to t}(j),k,t}[I_t(\Theta_k^{t_{ref}\to t}(j)) - M_k(j)] \\
&= 0 \\
\sum_{t=1}^{t=T} P^2_{\Theta_k^{t_{ref}\to t}(j),k,t} M_k(j) &= \sum_{t=1}^{t=T} P^2_{\Theta_k^{t_{ref}\to t}(j),k,t} I_t(\Theta_k^{t_{ref}\to t}(j)) \\
M_k(j) &= \frac{\sum_{t=1}^{t=T} P^2_{\Theta_k^{t_{ref}\to t}(j),k,t} I_t(\Theta_k^{t_{ref}\to t}(j)}{\sum_{t=1}^{t=T} P^2_{\Theta_k^{t_{ref}\to t}(j),k,t}}.
\end{aligned}$$

B.3 Détail du calcul de l'équation des probabilités $P_{i,k,t}$

On désire annuler la dérivée partielle de E_{mod} (équation B.2) sachant la contrainte suivante : $\forall i, \forall t, \sum_{k=1}^{k=K} P_{i,k,t} = 1$. L'équation E_{mod} avec introduction du Lagrangien se ré écrit en :

$$E_{modL} = E_{mod} + \lambda \sum_{i,k,t} (P_{i,k,t} - 1). \tag{B.5}$$

En effectuant le changement de variable $l \to k$ et $k \to l$ sur le terme participant à la régularisation spatiale dans l'équation E_{mod} :

$$\begin{aligned}
&\textstyle\sum_{i,k,t} \beta Q^2_{i,k,t} \sum_{l=1}^{K} \left[[P_{i,l,t} - P_{\Theta_k^{t\to t-1}(i),l,t-1}]^2 + [P_{i,l,t} - P_{\Theta_k^{t\to t+1}(i),l,t+1}]^2 \right] \\
&= \textstyle\sum_{i,l,t} \beta Q^2_{i,l,t} \sum_{k=1}^{K} \left[[P_{i,k,t} - P_{\Theta_l^{t\to t-1}(i),k,t-1}]^2 + [P_{i,k,t} - P_{\Theta_l^{t\to t+1}(i),k,t+1}]^2 \right],
\end{aligned}$$

on peut aisément calculer la dérivée partielle de E_{modL} en fonction de $P_{i,k,t}$:

$$\begin{aligned}
\frac{\partial E_{modL}}{\partial P_{i,k,t}} &= 2P_{i,k,t} dI^2_{i,k,t} \\
&+ 2\alpha \sum_{i,k,t} \sum_{j \in \mathcal{V}(i)} [P_{i,k,t} - P_{j,k,t}] \\
&+ 2\beta \sum_{l=1}^{K} Q^2_{i,l,t} \left[[P_{i,k,t} - P_{\Theta_l^{t\to t-1}(i),k,t-1}] + [P_{i,k,t} - P_{\Theta_l^{t\to t+1}(i),k,t+1}] \right] \\
&+ 2\gamma [P_{i,k,t} - Q_{i,k,t}]^2 \\
&- \lambda.
\end{aligned} \tag{B.6}$$

En annulant la dérivée, on obtient alors :

$$
\begin{aligned}
\frac{\partial E_{modL}}{\partial P_{i,k,t}} &= 0 \\
P_{i,k,t} &= \frac{\overbrace{[(\alpha \sum_{j \in \mathcal{V}(i)} P_{j,k,t}) + (\beta \sum_{l=1}^{K} Q_{i,l,t}^2 (P_{\Theta_l^{t \to t-1}(i),k,t-1} + P_{\Theta_l^{t \to t+1}(i),k,t+1})) + (\gamma Q_{i,k,t})]}^{\alpha' \widehat{P}_{i,k,t}} + \lambda/2}{\underbrace{[(\alpha \sum_{j \in \mathcal{V}(i)} 1) + \gamma + (2\beta \sum_{l=1}^{K} Q_{i,l,t}^2)]}_{\alpha'} + dI_{i,k,t}^2} \\
P_{i,k,t} &= \frac{\alpha' \widehat{P}_{i,k,t} + \lambda/2}{\alpha' + dI_{i,k,t}^2}. \qquad \text{(B.7)}
\end{aligned}
$$

Il ne reste plus qu'à calculer la valeur de λ. On va faire en sorte que λ soit exprimée par une expression ne faisant pas intervenir de différence. On s'assure ainsi que lorsque l'on effectue le calcul numérique de $P_{i,k,t}$ la valeur est obligatoirement positive. Le calcul de la valeur de λ est alors le suivant :

$$
\begin{aligned}
\forall i, \forall t, \sum_{k=1}^{k=K} P_{i,k,t} &= 1, \\
\forall i, \forall t, \sum_{k=1}^{k=K} \widehat{P}_{i,k,t} &= \frac{1}{(\alpha \sum_{j \in \mathcal{V}(i)} 1) + \gamma + (2\beta \sum_{l=1}^{K} Q_{i,l,t}^2)} \times \\
&\quad [\sum_{k=1}^{k=K} (\alpha \sum_{j \in \mathcal{V}(i)} P_{j,k,t}) \\
&\quad + \sum_{k=1}^{k=K} (\beta \sum_{l=1}^{K} Q_{i,l,t}^2 (P_{\Theta_l^{t \to t-1}(i),k,t-1} + P_{\Theta_l^{t \to t+1}(i),k,t+1})) \\
&\quad + \sum_{k=1}^{k=K} (\gamma Q_{i,k,t})], \\
&= 1
\end{aligned}
$$

d'où :

$$\begin{aligned}
\sum_{k=1}^{k=K} P_{i,k,t} &= \sum_{k=1}^{k=K} \widehat{P}_{i,k,t} \\
\sum_{k=1}^{k=K} \frac{\alpha' \widehat{P}_{i,k,t} + \lambda/2}{\alpha' + dI^2_{i,k,t}} &= \sum_{k=1}^{k=K} \widehat{P}_{i,k,t} \\
\frac{\lambda}{2} &= \frac{\sum_{k=1}^{K} \frac{\widehat{P}_{i,k,t} \times dI^2_{i,k,t}}{\alpha' + dI^2_{i,k,t}}}{\sum_{k=1}^{K} \frac{1}{\alpha' + dI^2_{i,k,t}}}.
\end{aligned}$$

On peut alors remplacer la valeur de λ dans l'équation B.7 et on retrouve alors l'équation 2.13 :

$$\begin{aligned}
P_{i,k,t} &= \frac{\alpha' \widehat{P}_{i,k,t} + \frac{\sum_{l=1}^{K} \frac{\widehat{P}_{i,l,t} \times dI^2_{i,l,t}}{\alpha' + dI^2_{i,l,t}}}{\sum_{l=1}^{K} \frac{1}{\alpha' + dI^2_{i,l,t}}}}{\alpha' + dI^2_{i,k,t}} \\
P_{i,k,t} &= \frac{\frac{\sum_{l=1}^{K} \frac{\alpha' \widehat{P}_{i,k,t} + \widehat{P}_{i,l,t} \times dI^2_{i,l,t}}{\alpha' + dI^2_{i,l,t}}}{\sum_{l=1}^{K} \frac{1}{\alpha' + dI^2_{i,l,t}}}}{\alpha' + dI^2_{i,k,t}} \\
P_{i,k,t} &= \frac{\sum_{l=1}^{K} \frac{\alpha' \widehat{P}_{i,k,t} + \widehat{P}_{i,l,t} \times dI^2_{i,l,t}}{\alpha' + dI^2_{i,l,t}}}{\sum_{l=1}^{K} \frac{\alpha' + dI^2_{i,k,t}}{\alpha' + dI^2_{i,l,t}}}.
\end{aligned}$$

B.4 Détail du calcul de l'équation des probabilités $Q_{i,k,t}$

On désire annuler la dérivée partielle de E_{mod} (équation B.2) sachant la contrainte suivante : $\forall i, \forall t, \sum_{k=1}^{k=K} P_{i,k,t} = 1$. L'équation E_{mod} avec introduction du Lagrangien se réécrit :

$$E_{modL} = E_{mod} + \lambda \sum_{i,k,t} (P_{i,k,t} - 1).$$

On peut aisément calculer la dérivée partielle de E_{modL} en fonction de $Q_{i,k,t}$:

$$\begin{aligned}
\frac{\partial E_{modL}}{\partial Q_{i,k,t}} &= 2\beta Q_{i,k,t} \overbrace{\sum_{l=1}^{K} \left[[P_{i,l,t} - P_{\Theta_k^{t \to t-1}(i),l,t-1}]^2 + [P_{i,l,t} - P_{\Theta_k^{t \to t+1}(i),l,t+1}]^2 \right]}^{dP^2_{i,k,t}} \\
&+ 2\gamma [Q_{i,k,t} - P_{i,k,t}] \\
&- \lambda.
\end{aligned} \tag{B.8}$$

En annulant la dérivée, on obtient alors :

$$\begin{aligned} \frac{\partial E_{modL}}{\partial Q_{i,k,t}} &= 0 \\ Q_{i,k,t} &= \frac{\gamma P_{i,k,t} + \lambda/2}{\gamma + \beta d P_{i,k,t}^2}. \end{aligned} \tag{B.9}$$

Il ne reste plus qu'à calculer la valeur de λ. On va faire en sorte que λ soit exprimée par une expression ne faisant pas intervenir de différence. On s'assure ainsi que, lorsque l'on effectue le calcul numérique de $Q_{i,k,t}$, la valeur en soit obligatoirement positive. Le calcul de la valeur de λ est alors le suivant :

$$\begin{aligned} \forall i, \forall t, \sum_{k=1}^{k=K} Q_{i,k,t} &= 1, \\ \forall i, \forall t, \sum_{k=1}^{k=K} P_{i,k,t} &= 1, \end{aligned}$$

d'où :

$$\begin{aligned} \sum_{k=1}^{k=K} Q_{i,k,t} &= \sum_{k=1}^{k=K} P_{i,k,t} \\ \sum_{k=1}^{k=K} \frac{\gamma P_{i,k,t} + \lambda/2}{\gamma + \beta d P_{i,k,t}^2} &= \sum_{k=1}^{k=K} P_{i,k,t} \\ \frac{\lambda}{2} &= \frac{\sum_{k=1}^{K} \frac{P_{i,k,t} \times \beta d P_{i,k,t}^2}{\gamma + \beta d P_{i,k,t}^2}}{\sum_{k=1}^{K} \frac{1}{\gamma + \beta d P_{i,k,t}^2}}. \end{aligned}$$

On peut alors remplacer la valeur de λ dans l'équation B.9 et on retrouve alors l'équation 2.15 :

$$\begin{aligned} Q_{i,k,t} &= \frac{\gamma P_{i,k,t} + \frac{\sum_{l=1}^{K} \frac{P_{i,l,t} \times \beta d P_{i,l,t}^2}{\gamma + \beta d P_{i,l,t}^2}}{\sum_{l=1}^{K} \frac{1}{\gamma + \beta d P_{i,l,t}^2}}}{\gamma + \beta d P_{i,k,t}^2} \\ Q_{i,k,t} &= \frac{\frac{\sum_{l=1}^{K} \frac{\gamma P_{i,k,t} + P_{i,l,t} \times \beta d P_{i,l,t}^2}{\gamma + \beta d P_{i,l,t}^2}}{\sum_{l=1}^{K} \frac{1}{\gamma + \beta d P_{i,l,t}^2}}}{\gamma + \beta d P_{i,k,t}^2} \\ Q_{i,k,t} &= \frac{\sum_{l=1}^{K} \frac{\gamma P_{i,k,t} + \beta d P_{i,l,t}^2 P_{i,l,t}}{\gamma + \beta d P_{i,l,t}^2}}{\sum_{l=1}^{K} \frac{\gamma + \beta d P_{i,k,t}^2}{\gamma + \beta d P_{i,l,t}^2}}. \end{aligned}$$

Bibliographie

[Akaike 74] Akaike (H.). – A new look at statistical model identification. – *IEEE Trans. on automatic control*, vol. 19, pp. 716–723, Décembre 1974.

[Alatan et al. 98] Alatan (A.), Onural (L.), Wollborn (M.), Mech (R.), Tuncel (E.) et Sikora (T.). – Image sequence analysis for emerging interactive multimedia services - the european COST 211 framework. – *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, pp. 802–813, Novembre 1998.

[Alpert et al. 97] Alpert (T.), Baroncini (V.), Choi (D.), Contin (L.), Koenen (R.) et Peterson (F.P.). – Subjective evaluation of MPEG-4 codec proposals: Methodological approach and test procedures. – *Signal Process.*, vol. 9, pp. 303–325, Mai 1997.

[Antonini et al. 92] Antonini (M.), Barlaud (M.), Mathieu (P.) et Daubechies (I.). – Image coding using the wavelet transform. – *IEEE Trans. Image Processing*, vol. 1, pp. 205–220, Février 1992.

[Aubert et al. 99] Aubert (G.) et Blanc-Feraud (L.). – Some remarks on the equivalence between 2D and 3D classical snakes and geodesic active contours. *International Journal of Computer Vision*, 34:19–28, Septembre 1999.

[Azencott 87] Azencott (R.). – Markov fields and image analysis. – *6eme Congrès de Rec. des Formes et Intelligence Artificielle*, pp. 1183–1191, Novembre 1987.

[Ball et al. 66] Ball (G. H.) et Hall (D. J.). – ISODATA, a novel method of data analysis and pattern classification. – *In Proceedings of the International Communication Conference*, Juin 1966.

[Ball et al. 67] Ball (G. H.) et Hall (D. J.). – A clustering technique for summarizing multivariate data. *Behav. Sci.*, 12:153–155, Mars 1967.

[Ballard et al. 82] Ballard (D.) et Brown (C.). – *Computer Vision*, chap. 8. – Prentice-Hall, 1982.

[Balter et al. 03a] Balter (R.) et Morin (L.). – Morphing 3D bidimensionnel entièrement automatique. – *Journées Francophones des Jeunes Chercheurs en Vision par Ordinateur, ORASIS'2003*, Mai 2003.

[Balter et al. 03b] Balter (R.), Morin (L.) et Galpin (F.). – Very low bitrate compression of video sequence for virtual navigation. – *Picture Coding Symposium, PCS'2003*, pp. 305–308, Avril 2003.

[Benois et al. 92] Benois (J.) et Barba (D.). – Image segmentation by region-contour cooperation for image coding. – *International Conference on Pattern Recognition, ICPR'1992*, vol. C, pp. 331–334, Août 1992.

[BenoisPineau et al. 02] Benois-Pineau (J.) et Nicolas (H.). – A new method for region-based depth ordering in a video sequence: application to frame interpolation. *Journal of Visual Communications and Image Représentation*, 13(3):363–385, Septembre 2002.

[Besag 86] Besag (J.). – On the statistical analysis of dirty pictures (with discussion). *Journal of the Royal Statistical Society Series B*, 48:259–302, 1986.

[Bezdek 81] Bezdek (J.C.). – *Pattern Recognition with Fuzzy Objective Function Algorithm*. – Plenum Press, 1981.

[Bonnaud 98] Bonnaud (L.). – *Schémas de suivi d'objets vidéo dans une séquence animée : application à l'interpolation d'images intermédiaires*. – Thèse de Doctorat, Universitété de Rennes 1, IRISA, France, Octobre 1998.

[Bonnaud et al. 97] Bonnaud (L.) et Labit (C.). – Multiple occluding objects tracking using a non-redundant boundary-based representation for image sequence interpolation after decoding. – *International Conference on Image Processing, ICIP'1997*, vol. 2, pp. 426–429, Santa Barbara, CA, USA, Octobre 1997.

[Bouthemy et al. 87] Bouthemy (P.) et Rivero (J. Santillana). – A hierarchical likelihood approach for region segmentation according to motion-based criteria. – *1st Internationale Conference on Computer Vision*, pp. 463–467, Juin 1987.

[Brady et al. 97] Brady (N.), Bossen (F.) et Murphy (N.). – Context-based arithmetic encoding of 2D shape sequence. – *International Conference on Image Processing, ICIP'1997*, vol. 1, pp. 29–32, Santa Clara, California, USA, Octobre 1997.

[Braquelaire et al. 97] Braquelaire (J-P.) et Viallard (A.). – A new antialiasing approach for image compositing. *Visual Computer*, 13 issue 5(5):218–227, 1997.

[Brigger 95] Brigger (P.). – *Morphological shape representation using the skeleton decomposition: application to image coding*. – PhD. Thesis, EPFL, Lausane, Switzerland, 1995.

[Cammas et al. 03a] Cammas (N.) et Pateux (S.). – Codage vidéo scalable par maillage et ondelettes 3D. – *COmpression et REprésentation des Signaux Audiovisuels, CORESA'2003*, pp. 151–154, Janvier 2003.

[Cammas et al. 03b] Cammas (N.) et Pateux (S.). – Fine grain scalable video coding using 3D wavelets and active meshes. – *IS&T/SPIE's 15th Electronic Imaging Science and Technology - SPIE'2003*, vol. 5022, Janvier 2003. – France Telecom and IRISA.

[Canny 83] Canny (J.F.). – *Finding edges and lines in images*. – Rapport de Recherche n° 720, MIT, 1983.

[Caplier et al. 01] Caplier (A.), Bonnaud (L.) et Chassery (J-M.). – Robust fast extraction of video objects combining frame differences ans adaptive reference image. – *International Conference on Image Processing, ICIP'2001*, Octobre 2001.

[Caselles et al. 95] Caselles (V.), Kimmel (R.) et Sapiro (G.). – Geodesic active contours. – *In IEEE International conference on Computer Vision*, pp. 694–699, Boston, USA, Juin 1995.

[Caselles et al. 97] Caselles (V.), Kimmel (R.) et Sapiro (G.). – Geodesic active contours. *International Journal of Computer Vision*, 22:61–79, 1997.

[Castagno 98] Castagno (R.). – *Video segmentation based on multiple features for interactive and automatic multimedia applications*. – Lausane, PhD. Thesis, Ecoles polytechnique fédérale de Lausane, 1998.

[Castagno et al. 98] Castagno (R.) et Sodomaco (A.). – Estimation of image feature reliability for an interactive video segmentation scheme. – *International Conference on Image Processing, ICIP'1998*, vol. 1, pp. 938–942, Chicago, USA, Octobre 1998.

[Chai et al. 00] Chai (D.), Ngan (K. N.) et Bouzerdoum (A.). – Foreground/background bit allocation for region-of-interest coding. – *IEEE International Conference on Image Processing, ICIP'2000*, vol. 2, pp. 923–926, Septembre 2000.

[Chaumont et al. 02] Chaumont (M.), Pateux (S.) et Nicolas (H.). – Segmentation of non-rigid video objects using long term temporal consistency. – *International Conference on Image Processing, ICIP'2002*, Septembre 2002.

[Chaumont et al. 03a] Chaumont (M.), Cammas (N.) et Pateux (S.). – Fully scalable object based video coder based on analysis-synthesis scheme. – *International Conference on Image Processing, ICIP'2003*, Septembre 2003.

[Chaumont et al. 03b] Chaumont (M.), Pateux (S.) et Nicolas (H.). – Efficient lossy contour coding using spatio-temporal consistency. – *Picture Coding Symposium, PCS'2003*, pp. 289–294, Avril 2003.

[Choi et al. 99] Choi (S-J.) et Woods (J.W.). – Motion-compensated 3-D subband coding of video. – *IEEE Transactions on Image Processing*, vol. 8, pp. 155–167, Février 1999.

[Chou et al. 90] Chou (P.B.) et Brown (C.M.). – The theory and practice of bayesian image modeling. *International Journal of Computer Vision*, 4:185–210, 1990.

[Cocquerez et al. 95] Cocquerez (J.P.) et Philipp (S.). – *Analyse d'images : filtrage et segmentation.* – Masson, Collection enseignement de la physique, 1995.

[Déforges et al. 99] Déforges (O.) et Ronsin (J.). – Nonuniform sub-sampling using squares elements : a fast still image coding at low bit rate. – *Picture Coding Symposium, PCS'1999*, Portland, USA, Avril 1999.

[Dekeyser 01] Dekeyser (F.). – *Restauration de séquences d'images par approches spatio-temporelles : filtrage et super-résolution par le mouvement.* – Thèse de Doctorat, Université de Rennes 1, Septembre 2001.

[DeMenthon 02] DeMenthon (D.). – Spatio-temporal segmentation of video by hierarchical mean shift analysis. – *Statistical Methods in Video Processing Workshop*, Juin 2002.

[Deriche 87] Deriche (R.). – Using canny's criteria to derive an optimal edge detector recursively implanted. *International Journal of Computer Vision*, 1(2):167–187, 1987.

[Diehl 91] Diehl (N.). – Signal processing: Image communication. – *Object-oriented motion estimation an segmentation in image sequences*, vol. 3, pp. 23–56, 1991.

[Dunham 86] Dunham (J.). – Object-oriented analysis-synthesis coding of moving images. – *IEEE. trans. PAMI*, vol. 8, pp. 67–75, Janvier 1986.

[Dunn 74] Dunn (J. C.). – A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. *Journal of Cybernetics*, 3:32–57, 1974.

[Ebrahimi et al. 95] Ebrahimi (T.) et al. – *Dynamic Coding of Visual Information.* – Technical report n° ISO/IEC JTC1/SC29/WG11/M0320, ISO/IEC, Octobre 1995.

[Eden et al. 85] Eden (M.) et Kocher (M.). – On the performance of a contour coding algorithm in the context of image coding. part 1: Contour segment coding. – *Signal Processing*, vol. 8, pp. 381–386, Juillet 1985.

[Eisert et al. 99] Eisert (P.), Wiegand (T.) et Girod (B.). – Rate-distortion-efficient video coding compression using a 3-D head model. – *International Conference on Image Processsing, ICIP'1999*, vol. 4, pp. 217–221, Octobre 1999.

[Fablet et al. 99] Fablet (R.), Bouthemy (P.) et Gelgon (M.). – Etiquettage statistique d'un graphe de régions pour la détection d'objets mobiles dans des séquences d'images couleur. – *17eme Colloque Gretsi sur le traitement du signal et les images*, vol. 2, pp. 499–502, Vannes, France, Septembre 1999.

[Fleury 99] Fleury (P.). – *Dynamic Scheme Selection in Image Coding.* – PhD. Thesis, École Polytechnique Fédérale de Lausane, Juillet 1999.

[Foret et al. 02] Foret (G.), Bertolino (P.) et Cibaud (D.). – Partition projection in videos by global and local block-matching. – *In IEEE International Conference on Image Processing, ICIP'2002*, Rochester, USA, Septembre 2002.

[Freeman 61] Freeman (H.). – On the encoding of arbitrary geometric configurations. – *Transaction on Electronic Computers*, vol. 10, pp. 260–268, Juin 1961.

[Galpin 02] Galpin (F.). – *Représentation* 3D *de séquence vidéo.* – Rennes 1, Thèse de Doctorat, Institut de Formation Supérieur en Informatique et Communication (IFSIC), Université de Rennes 1, Janvier 2002.

[Galpin et al. 01] Galpin (F.) et Morin (L.). – Computed 3D models for very low bitrate video coding. – *Proceedings of the IEEE conference on Visual Communications and Image Processing, VCIP'2001*, vol. 4310, Janvier 2001.

[Garciagarduño 95] Garcia-garduño (V.). – *Une approche de compression orientée-objets par suivi de segmentation basée mouvement pour le codage de séquences d'images numériques.* – Rennes 1, Thèse de Doctorat, UFR Structure et propriétés de la matière, Université de Rennes 1, Mai 1995.

[Geman et al. 84] Geman (S.) et Geman (D.). – Stochastic relaxation, gibbs distributions and the bayesian restoration of images. – *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 6, pp. 721–741, Novembre 1984.

[Greenspan et al. 02] Greenspan (H.), Goldberger (J.) et Meyer (A.). – A probabilistic framework for spatio-temporal video representation and indexing. – *European Conference on Computer Vision*, vol. 4, pp. 461–75, Springer-Velag, Berlin, Germany, Mai 2002.

[Han et al. 97] Han (S.C.) et Woods (J.). – Spatiotemporal subband/wavelet coding of video with object-based motion information. – *International Conference on Image Processing, ICIP'1997*, vol. 2, Washington, DC, USA, Octobre 1997.

[Han et al. 98] Han (S.C.) et Woods (J.). – Adaptive coding of moving objects for very low bit rates. *IEEE Journal on Selected Areas in Communications, issue on very low bit-rate video coding*, 16(1), Janvier 1998.

[Herrmann et al. 97] Herrmann (S.) et Mooshofer (H.). – *Shape Analysis by Geodesic Skeletons.* – Information, Technical University of Munich, Inst. F. Integr. Circuits, COST211ter Simulation Group, Novembre 1997.

[Hötter 90] Hötter (M.). – Object-oriented analysis coding based on moving two-dimensional objects. – *Signal Processing: Image Communication*, vol. 2, pp. 409–428, Avril 1990.

[ISO/IEC 93] ISO/IEC (JTC1/SC29/WG11 Coding Of Moving Pictures & Audio). – *Information technology – Coding of moving pictures and associated audio for digital*

storage media at up to about 1,5 Mbit/s. – standard, International Organisation For Standardisation ISO/IEC, 1993.

[ISO/IEC 98] ISO/IEC (JTC1/SC29/WG11 Coding Of Moving Pictures & Audio). – *Generic coding of audio-visual objects: Part II, Visual.* – Final draft, International Organisation For Standardisation ISO/IEC, Novembre 1998.

[ISO/IEC 02] ISO/IEC (JTC1/SC29/WG11 Coding Of Moving Pictures & Audio). – MPEG4 *Overwiew - (V.21 Jeju Version).* – Final draft, International Organisation For Standardisation ISO/IEC, Mars 2002.

[ITUT 93] ITU-T. – *Coded representation of picture and audio information - progressive bi-level image compression.* – Recommendation T.82, ITU-T, Mars 1993.

[ITUT 98] ITU-T (Group 4 Facsimile Apparatus). – *Facsimile Coding Schemes and Coding Control Functions for Group 4 Facsimile Apparatus.* – Recommendation T.6, ITU-T, Novembre 1998.

[JehanBesson et al. 01] Jehan-Besson (S.), Barlaud (M.) et Aubert (G.). – Region-based active contours for video object segmentation with camera compensation. – *International Conference on Image Processing, ICIP'2001*, Thessaloniki, Greece, Octobre 2001.

[JehanBesson et al. 03] Jehan-Besson (S.) et Barlaud (M.). – Dreams: Deformable regions driven by an eulerian accurate minimization method for image and video segmentation. *International Journal of Computer Vision, IJCV'2003*, 1(53):45–70, 2003.

[Jordan et al. 98] Jordan (C. Le Buhan), Bhattacharjee (S.), Bossen (F.), Jordan (F.) et Ebrahimi (T.). – Shape representation and coding of visual objects in multimedia applications – an overview. – *Ann. Telecommun*, 1998.

[Kass et al. 88] Kass (M.), Witkin (A.) et Terzopoulos (D.). – Snakes: Active contour models. *Inernational Journal of Computer Vision*, 1:321–332, 1988.

[Katsaggelos et al. 98] Katsaggelos (A.K.), Kondi (L.P.), Meier (F.W.), Ostermann (J.) et Schuster (G.M.). – MPEG-4 and rate-distortion-based shape-coding techniques. – *IEEE Proc., special issue on Multimedia Signal Processing*, vol. 86, pp. 1126–1154, Juin 1998.

[Kichenassamy et al. 95] Kichenassamy (S.), Kumar (A.), Olver (P.), Tannenbaum (A.) et Yezzi (A.). – Gradient flows and geometric active contour models. – *In IEEE International Conference on Computer Vision*, pp. 810–815, Boston, USA, 1995.

[Kunt et al. 85] Kunt (M.), Ikonomopoulos (A.) et Kocher (M.). – Second-generation image-coding techniques. – *Proceedings of the IEEE*, vol. 73, pp. 549–574, Avril 1985.

[Leclerc 89] Leclerc (Y. G.). – Constructing simple stable descriptions for image partitioning. *International Journal of Computer Vision*, 3:73–102, 1989.

[Lee et al. 97] Lee (S.), Cho (D.), Son (S.), Jang (E.) et Shin (J.). – Binary shape coding using 1-D distance values from baseline. – *International Coneference on Image Processing, ICIP'1997*, vol. 1, pp. 508–511, Santa Clara, California, USA, Octobre 1997.

[Lee et al. 03] Lee (J.-W.) et Ho (Y.-S.). – Object composition for MPEG-4 video coding. – *Picture Coding Symposium, PCS'2003*, pp. 189–192, Avril 2003.

[Lorette 99] Lorette (A.). – *Analyse de texture par méthodes markoviennes et par morphologie mathématique : application à l'analyse des zones urbaines sur des images satellitales.* – Thèse de Doctorat, INRIA sophia-antipolis, Septembre 1999.

[Lu et al. 94] Lu (F.) et Milios (E. E.). – Optimal spline fitting to planar shape. – *Signal Processing*, vol. 37, pp. 129–140, 1994.

[Luo et al. 01] Luo (L.), Li (J.), Li (S.), Zhuang (Z.) et Zhang (Y-Q.). – Motion-compensated lifting wavelet and its application in video coding. – *IEEE International Conference on Multimedia and Expo*, Août 2001.

[Mahalanobis 30] Mahalanobis (P.C.). – On tests and measures of groups divergence I. *Journal of the Asiatic Society of Benagal*, 26:541, 1930.

[Mansouri et al. 00] Mansouri (A.-R.), Olivier (A.) et Konrad (J.). – Topology-independent region tracking with level sets. – *in Proc. IEEE International Conference on Image Processing, ICIP'2000*, vol. 3, pp. 66–69, Septembre 2000.

[Marquant 00] Marquant (G.). – *Représentation par maillage déformable pour la manipulation et la communication d'objets vidéo.* – Thèse de Doctorat, Université de Rennes 1, IRISA, France, Décembre 2000.

[Marquant et al. 00] Marquant (G.), Pateux (S.) et Labit (C.). – Mesh and "crack lines": Application to object-based motion estimation and higher scalability. – *International Conference on Image Processing, ICIP'2000*, vol. 2, pp. 554–557, Vancouver, BC, Canada, Septembre 2000.

[Marqués et al. 96] Marqués (F.), Salembier (P.), Pardàs (M.), Morros (R.), Corset (I.), Jeannin (S.), Marcotegui (B.) et Meyer (F.). – A segmentation-based coding system allowing manipulation of objects (SESAME). – *In IEEE International Conference on Image Processing, ICIP'1996*, vol. 3, pp. 679–682, Lausanen, Switzerland, Septembre 1996. – Invited Paper.

[Marqués et al. 98] Marqués (F.) et Llach (J.). – Tracking of generic objects for video object generation. – *In IEEE International Conference on Image Processing, ICIP'1998*, Chicago, USA, Octobre 1998.

[Martinez 99] Martinez (G.). – Analysis-synthesis coding based on the model of articulated three-dimensional objects. – *Picture Coding Symposium, PCS'1999*, pp. 213–216, Portland, Oregon, USA, Avril 1999.

[Mazière et al. 00] Mazière (M.) et Chassaing (F.). – Segmentation and tracking of video objects: Suited to content-based video indexing, interarctive television and production systems. – *International Conference on Image Processing, ICIP'2000*, Vancouver, Canada, Septembre 2000.

[McKenna et al. 98] McKenna (S.), Raja (Y.) et Gong (S.). – Object tracking using adaptive colour mixture models. – *In Proc. Asian Conference on Computer Vision, ACCV'1998*, vol. 1, pp. 615–622, Hong Kong, Janvier 1998.

[Megret et al. 02] Megret (R.) et DeMenthon (D.). – *A Survey of Spatio-Temporal Grouping Techniques.* – CS-TR-4403 n° CS-TR-4403, LAMP, University of Maryland, USA, Language And Media Processing (LAMP), University of Maryland, Octobre 2002. Submitted to Computer Vision and Image Understanding (CVIU), an archival journal published by Academic Press.

[Meyer et al. 90] Meyer (F.) et Beucher (S.). – Morphological segmentation. *Journal of visual communication image representation*, 1(1):21–46, Septembre 1990.

[Moulet et al. 88] Moulet (D.) et Barba (D.). – Temporal following of spatial segmentation in image sequences. – *Proceding European Signal Processing Conference*, pp. 39–42, Grenoble, France, Septembre 1988.

[Mukherjee et al. 98] Mukherjee (D.), Deng (Y.) et Mitra (S.). – A region-based video coder using edge flow segmentation and hierarchical affine region matching. – *Proceedings of SPIE, Visual Communications and Image Processing*, vol. 3309, pp. 338–49, San Jose, California, Janvier 1998.

[Musmann et al. 89] Musmann (H.G.), Hötter (M.) et Ostermann (J.). – Object-oriented analysis-synthesis coding of moving images. – *Signal Processing: Image Communication*, vol. 1, pp. 117–138, Octobre 1989.

[Nadenau et al. 00] Nadenau (M. J.), Winkler (S.), Alleysson (D.) et Kunt (M.). – Human vision models for perceptually optimized image processing - a review. – Septembre 2000. Final Draft, Submitted to Proceedings of the IEEE.

[Ndili et al. 01] Ndili (U.), Nowak (R. D.) et Figueiredo (M. A. T.). – Coding theoretic approach to image segmentation. – *International Conference On Image Processing, ICIP'2001*, Octobre 2001.

[Nguyen et al. 93] Nguyen (H. H.) et Cohen (P.). – Gibbs random fields, fuzzy clustering, and the unsupervised segmentation of textured images. – *CVGIP*, vol. 55, pp. 1–19, Janvier 1993.

[Odobez 94] Odobez (J.M.). – *Estimation, détection et segmentation du mouvement: une approche robuste et markovienne.* – Thèse de Doctorat, Université de rennes 1, Décembre 1994.

[Ohm 94] Ohm (J.R). – Three-dimensional subband coding with motion compensation. – *IEEE Transactions on Image Processing*, vol. 3, pp. 559–571, Septembre 1994.

[Okada et al. 01] Okada (S.), Jinzenji (K.) et Kobayashi (N.). – An approach to MPEG-4 multi mode coding using sprite coding. – *International Picture Coding Symposium, PCS'2001*, pp. 421–424, Avril 2001.

[Osher et al. 88] Osher (S.) et Sethian (J.). – Fronts propagating with curvature-dependent speed : algorithms based on hamilton-jacobi formulation. *Journal of Computational Physics*, 79:12–49, 1988.

[Otterloo 91] Otterloo (P.J. Van). – *A Contour-Oriented Approach for shape Analysis.* – Prentice Hall International (UK) Ltd. Hertfordshire, 368p., 1991.

[Paragios 00] Paragios (N. K.). – *Geodesic Active Regions and Level Set Methods : Contribution and Applications in Artificial Vision.* – PhD. Thesis, INRIA, Sophia Antipolis, France, Janvier 2000.

[Paragios et al. 98] Paragios (N.) et Deriche (R.). – *Geodesic Active Regions for Texture Segmentation.* – Technical report 3440, INRIA, France, Juin 1998.

[Pardàs et al. 96] Pardàs (M.), Salembier (P.), marqués (F.) et Morros (R.). – Partition tree for a segmentation-based video coding system. – *International Conference on Acoustics Speech and Signal Processing, ICASSP'1996*, pp. 1982–1985, Atlanta, USA, Mai 1996.

[Parker et al. 01] Parker (B.) et Magarey (J.). – Three dimensional video segmentation using a variational method. – *International Conference on Image Processing, ICIP'2001*, Thessaloniki, Greece, Octobre 2001.

[Pateux 98] Pateux (S.). – *Segmentation spatiotemporelle et codage orienté-régions de séquences vidéo basés sur le formalisme* MDL. – Thèse de Doctorat, université de Rennes 1, Septembre 1998.

[Pateux 00] Pateux (S.). – Spatial segmentation of color images according to the MDL formalism. – *International conference on Color in Graphics and Image Processing, CGIP'2000*, number 37, pp. 89–93, Octobre 2000.

[Pateux et al. 98] Pateux (S.) et Labit (C.). – Calcul de carte de segmentation pour le codage vidéo à très bas-débit. – *Actes des journées CORESA'98, Lannion*, pp. 125–131, juin 1998.

[Pateux et al. 01] Pateux (S.), Marquant (G.) et Chavira-Martinez (D.). – Object mosaicking via meshes and crack-lines technique. application to low bit-rate video coding. – *Picture Coding Symposium, PCS'2001*, Seoul, Korea, Avril 2001.

[Perez et al. 99] Perez (D.G.), Sun (M.-T.) et Gu (C.). – Semantic video object extraction based on backward tracking of multivalued watershed. – *International Conference on Image Processing, ICIP'1999*, vol. 2, pp. 145–149, Octobre 1999.

[Porikli et al. 01] Porikli (F. M.) et Wang (Y.). – An unsupervised multi-resolution object extraction algorithm using video-cube. – *International Conference on Image Processing, ICIP'2001*, Thessaloniki, Greece, Octobre 2001.

[Porter et al. 84] Porter (T.) et Duff (T.). – Compositing digital images. – *SIGGRAPH'84 Conference Proceedings, Association for Computing Machinery*, vol. 18, pp. 253–259, Juillet 1984.

[Precioso et al. 03] Precioso (F.), Barlaud (M.), Blu (T.) et Unser (M.). – Smoothing b-spline active contour for fast and robust image and video segmentation. – *International Conference on Image Processing, ICIP'2003*, pp. 137–140, Septembre 2003.

[Prewitt 70] Prewitt (J.). – Object enhancement and extraction. – *Picture Processing and Psychopictorics, Academic Press, B Linkin and A. Rosenfeld editors*, pp. 74–149, 1970.

[Ramer 72] Ramer (U.). – An iterative procedure for polygonal approximation of plane curves. – *Computer Graphics and Image Processing*, vol. 1, pp. 244–256, Novembre 1972.

[Reusens et al. 97] Reusens (E.), Ebrahimi (T.), Buhan (C. Le), Castagno (R.), Vaerman (V.), Piron (L.), de Solà Fàbregas (C.), Bhattacharjee (S.), Bossen (F.) et Kunt (M.). – Dynamic approach to visual data compression. – *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, pp. 197–211, Février 1997.

[Rissanen 78] Rissanen (J.). – Modeling by shortest data description. – *Automatica*, pp. 465–471, 1978.

[Roberts 65] Roberts (L. G.). – Machine perception of three dimensional solids. *Optical and Electro-optical Information Processing, MIT Press*, pp. 159–197, 1965.

[Rosenfeld et al. 82] Rosenfeld (A.) et Kak (A.C.). – *Digital Picture Processing.* – Academic Press, New York, USA, 2nd édition, vol. 2, 1982.

[Said et al. 96] Said (A.) et Pearlman (W. A.). – A new fast and efficient image codec based on set partitioning in hierarchical trees. – *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, pp. 243–250, Juin 1996.

[Salembier et al. 95] Salembier (P.), Torres (L.), Meyer (F.), et Gu (C.). – Region-based video coding using mathematical morphology. – *Proceedings of IEEE, Special Issue on Digital TV*, vol. 83, pp. 843–857, Septembre 1995. – Invited paper.

[Salembier et al. 96] Salembier (P.), Marqués (F.) et Gasull (A). – *Video Coding: The second generation approach*, chap. Coding of Partition Sequences, pp. 125–169. – L. Torres and M. Kunt, Kluwer Academic Publishers, 1996.

[Salembier et al. 97] Salembier (P.), Marqués (F.), Pardàs (M.), Morros (R.), Corset (I.), Jeannin (S.), Bouchard (L.), Meyer (F.) et Marcotegui (B.). – Segmentation-based video coding system allowing the manipulation of objects. – *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, pp. 60–73, Février 1997.

[Schäfer et al. 03] Schäfer (R.), Wiegand (T.) et Schwarz (H.). – The emerging H.264/AVC standard. – *EBU Technical Review - European Broadcasting Union - Union Européenne de Radio-Télévision*, Janvier 2003. – No. 293.

[Schroeter 96] Schroeter (P.). – *Unsupervised Two-Dimensional and Three-Dimensional Image Segmentation.* – PhD. Thesis, Swiss Federal Institute of Technology, Lausane Switzerland, 1996.

[Schwarz et al. 00] Schwarz (H.) et Müller (E.). – Object-based 3-D wavelet coding using layered object representation. – *IEEE International Conference on Image Processing, ICIP'2000*, vol. 1150, Vancouver, Canada, Septembre 2000.

[Schwarz et al. 02] Schwarz (H.) et Wiegand (T.). – The emerging JVT/H.26L video coding standard. – *International Broadcasting Convention, IBC'2002*, Amsterdam, NL, Septembre 2002. – invited paper.

[Secker et al. 01] Secker (A.) et Taubman (D.). – Motion-compensated highly scalable video compression using an adaptive 3D wavelet transform based on lifting. – *IEEE International Conference on Image Processing, ICIP'2001*, pp. 1029–1032, Octobre 2001.

[Shapiro 93] Shapiro (J. M.). – Embedded image coding using zerotrees of wavelet coefficients. – *IEEE Transactions On Signal Processing*, vol. 41, pp. 3445–3462, Décembre 1993.

[Shi et al. 98] Shi (J.) et Malik (J.). – Motion segmentation and tracking using normalized cuts. – *in IEEE International Conference on Computer Vision*, pp. 1151–1160, Bombay, India, Janvier 1998.

[Shoham et al. 89] Shoham (Y.) et Gersho (A.). – Efficient bit allocation for an arbitrary set of quantizers. – *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 36, pp. 1445–1453, Septembre 1989.

[Sobel 70] Sobel (I. E.). – *Camera Models and Machine Perception.* – PhD. Thesis, Electrical Engineering Department, Stanford University, 1970.

[Spaan et al. 97] Spaan (F.), Lagendijk (R.L.) et Biemond (J.). – Shape coding using polar coordinates and the discrete cosine transform. – *International Conference on Image Processing, ICIP'1997*, vol. 1, pp. 516–518, Santa Barbara, USA, Octobre 1997.

[Taubman 00] Taubman (D.). – High performance scalable image compression with EBCOT. – *IEEE Transactions On Image Processing*, vol. 9, pp. 1158–1170, Juillet 2000.

[Taubman et al. 94] Taubman (D.) et Zakhor (A.). – Multirate 3-D subband coding of video. – *IEEE Transactions on Image Processing*, vol. 3, pp. 572–588, Septembre 1994.

[Terzopoulos et al. 87] Terzopoulos (D.), Platt (J.), Barr (A.) et Fleischer (K.). – Elastically deformable model. *Computer Graphics*, 21(4):205–214, Juillet 1987.

[Torres et al. 96] Torres (L.), Kunt (M.) et Pereira (F.). – Second generation video coding schemes and their role in MPEG-4. – *European Conference on Multimedia Applica-*

tions, Services and Techniques, pp. 799–824, Louvain-la-Neuve, Belgium, Mai 1996. – Invited paper.

[vandenBrandenLambrecht 96] van den Branden Lambrecht (C. J.). – *Perceptual Models and Architectures for Video Coding Applications*. – PhD. Thesis, École Polytechnique Fédérale de Lausane, 1996.

[Vieron et al. 02] Vieron (J.), Guillemot (C.) et Pateux (S.). – Motion compensated 2D+t wavelet analysis for low rate FGS video compression. – *International Thyrrhenian workshop on digital communications 2002*, Capri, Italy, Septembre 2002. – Invited paper.

[Wang et al. 94] Wang (J. Y. A.) et Adelson (E. H.). – Representing moving images with layers. – *IEEE Transactions on Image Processing*, vol. 3, pp. 625–638, Septembre 1994.

[Wareham et al. 96] Wareham (P. C.) et Blostein (S. D.). – Region-oriented video coding using the MDL principle and quad-tree optimization. – *International Conference on Image Processing, ICIP'1996*, vol. 1, Lausanne, Switzerland, Septembre 1996.

[Wiegand et al. 03] Wiegand (T.) et Sullivan (G.). – *Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264—ISO/IEC 14496-10 AVC)*. – Draft ISO/IEC 14496-10:2002(E) - draft ITU-T rec. H.264(2002 E), Pattaya, Thailand, ISO/IEC MPEG and ITU-T VCEG, Mars 2003.

[Yamaguchi et al. 97] Yamaguchi (N.), Ida (T.) et Waranabe (T.). – A binary shape coding method using modified MMR. – *International Conference on Image Processing, ICIP'1997*, vol. 1, pp. 504–507, Santa Barbara, California, USA, Octobre 1997.

[Yokoyama et al. 95] Yokoyama (Y.), Miyamoto (Y.) et Ohta (M.). – Very low bit rate video coding using arbitrarily shaped region-based motion compensation. – *IEEE Transactions on circuits and Systems for Video Technology*, vol. 5, pp. 500–507, Décembre 1995.

[Yoshida et al. 98] Yoshida (T.), Asami (T.) et Sakai (Y.). – Compression of moving contours on spatio-temporal 2-D plane. – *International Conference on Image Processing, ICIP'1998*, vol. 1, pp. 276–280, Chicago, USA, Octobre 1998.

[Zhu et al. 96] Zhu (S.C.) et Yuille (A.). – Region competition: Unifying snakes, region growing, and bayes/MDL for multiband image segmentation. – *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, pp. 884–900, Septembre 1996.

Publications

Congrès Internationaux :

1. M. Chaumont, N. Cammas et S.Pateux. – Fully scalable object based video coder based on analysis-synthesis scheme. – *International Conference on Image Processing, ICIP'2003*, Septembre 2003.
2. M. Chaumont, S. Pateux et H. Nicolas. – Efficient lossy contour coding using spatio-temporal consistency. – *Picture Coding Symposium, PCS'2003*, pp. 289–294, Avril 2003.
3. M. Chaumont, S. Pateux et H. Nicolas. – Segmentation of non-rigid video objects using long term temporal consistency. – *International Conference on Image Processing, ICIP'2002*, Septembre 2002.

Congrès Nationaux :

1. M. Chaumont, S. Pateux et H. Nicolas. – Codage efficace de contour avec perte utilisant la consistance spatio-temporelle. – *Colloque GRETSI sur le traitement du signal et des images, GRETSI'2003*, Septembre 2003.

Video object representation for a progressive and competing coding of pictures sequences.

Abstract

The goal of this thesis is to valid the hypothesis such that the video object coding allows significant improvement thanks to the used of dynamic coding. The dynamic coding is the setting on competition of a set of coders on each video object and the retaining of the best coder for each object.

To reach that goal, different points are studied. The first one concerns the study of automatic video object segmentation. We are proposing an object model with the notion of long term tracking. This model represents an object thanks to its motion (by the used of an active mesh) and its texture (by the used of a mosaic). A 3D clustering algorithm is developed based on that model.

In a second part, we are improving object coding techniques via the scalability on the video bitstream. To that purpose, we are using a wavelet 3D coding scheme and we are notably introducing a lossy contour coding.

Finally the last point studied concerns the dynamic coding. Coders used are H264/AVC, a 3D wavelet coder, a 3D coder and a mosaic coder. The automatic rate distribution allows to obtained better results than those given by each coder took separately, in all offering a cut-out of the bitstream in video objects.

Key words

Video object segmentation, long term segmentation, video object model, energetic function, clustering, active mesh, mosaic, video coding, object video coding, decorrelation motion texture form, scalability, spatio-temporal wavelet, 3D wavelet, contour coding, form coding, padding, dynamic coding, competing coding, rate-distortion optimization, rate distribution, antialiasing.

Représentation en objets vidéo pour un codage progressif et concurrentiel des séquences d'images.

Résumé

L'objectif de cette thèse est de valider l'hypothèse selon laquelle le codage par objets vidéo peut permettre d'obtenir des gains significatifs en utilisant le codage dynamique (mise en concurrence de plusieurs codeurs pour chaque objet vidéo).

Afin de répondre à cet objectif, différents points ont été étudiés. Le premier point concerne l'étude de la segmentation en objet vidéo de manière automatique. Nous avons proposé un modèle d'objet faisant intervenir la notion de suivi long terme via la représentation d'un objet sous la forme mouvement/texture (avec l'utilisation d'un maillage actif pour représenter le mouvement). Un algorithme de clustering 3D a été développé basé sur ce modèle.

Dans un deuxième temps, nous nous sommes attaché à l'amélioration des techniques de codage objet via la hiérarchisation (« scalabilité ») du flux vidéo. Pour cela, nous utilisons un schéma de codage ondelette 3D et nous introduisons notamment un codage de contours avec perte.

Enfin le dernier point étudié concerne le codage dynamique d'objets vidéo (mise en concurrence de plusieurs codeurs pour chaque objet vidéo). Les codeurs utilisés sont : le codeur H264/AVC, un codeur ondelette 3D, un codeur 3D et un codeur par mosaïque. La répartition automatique des débits permet d'obtenir des résultats dépassant ceux produits par chaque codeur pris séparément, tout en offrant le découpage du flux en objets vidéo.

Mots clés

Segmentation en objets vidéo, segmentation long terme, modèle d'objet vidéo, fonctionelle d'énergie, clustering, maillage actif, mosaïque, codage vidéo, codage d'objet vidéo, décorrélation mouvement texture forme, hiérarchisation : scalabilté, ondelettes spatio-temporelles, ondelette 3D, codage de contour, codage de forme, prolongement : padding, codage dynamique, codage concurrentiel, optimisation débit-distorsion, répartition des débits, antialiasing.