Université de Nantes

–

École polytechnique de l'université de Nantes

Projet RIAM ArchiPEG

(convention ANR05RIAM01401)

Lot 4.3 : Manuel d'utilisation du logiciel de conditionnement et de pré-analyse de flux vidéo

O. Brouard, F. Delannay, V. Ricordel et D. Barba

Laboratoire IRCCyN - équipe IVC

juin 2008

Table des matières

Table des figures

Introduction générale

Les travaux présentés dans ce rapport ont été réalisés dans le cadre du projet RIAM ArchiPEG qui relève de la convention ANR05RIAM01401. Ils correspondent à la troisième tâche du sous-projet 4 intitulé : Pré-analyse et conditionnement du flux vidéo en haute définition.

Le dernier standard de codage vidéo développé par le JVT (Joint Video Team) regroupant les experts MPEG et ITU, à savoir MPEG-4 Part 10 (ou encore AVC ou H.624), vise à gagner jusqu'à 50% de la bande passante actuellement utilisée par MPEG-2 pour une qualité visuelle équivalente. On s'accorde donc à décrire ce standard [**??**, **??**] comme le futur de la compression des signaux TV capable de transmettre un programme HD[1] à des débits allant de 6 à 9 Mbits/s. Le schéma du codeur H.264 est présenté en figure 1.

De telles performances ne peuvent être atteintes qu'au prix d'une estimation et d'une compensation de mouvement complexes, afin d'exploiter de façon optimale les redondances spatiales et temporelles présentes au sein des vidéos. Le standard H.264 offre donc une palette large et complexe de possibilités pour l'estimation et la compensation de mouvement, notamment au niveau de :

- la précision des vecteurs déplacement : elle peut aller jusqu'au au quart de pixel pour la luminance et jusqu'au huitième de pixel pour la chrominance ;
- la taille variable des blocs estimés : 7 modes pour la prédiction inter (16×16, 16×8, 8×16, 8×8, 8×4, 4×8, 4×4) et 2 modes pour la prédiction intra (16×16, 4×4) ;
- la sélection des images de référence : le choix de l'image de référence intervient au niveau macrobloc et sous-macrobloc contrairement aux normes précédentes telles que MPEG-2.

Le codeur H.264 réalise, lors de la phase de codage d'une séquence vidéo, une optimisation débit-distorsion pour chaque macrobloc afin d'obtenir le meilleur mode de codage (intra ou inter, taille des sous-partitions de macrobloc). Lors de cette optimisation débit-distorsion, le codeur doit réaliser une estimation de mouvement sur tous les modes inter en testant toutes les images de référence précédemment codées-décodées stockées dans un buffer. Cette phase est donc très coûteuse en temps de calcul, alors qu'elle ne garantie pas la cohérence avec le contenu spatio-temporel de la séquence vidéo.

Cette observation indique qu'une connaissance *a priori* sur le contenu spatio-temporel de la séquence vidéo à coder, permettrait de réduire significativement la charge de calculs du codeur. Il apparaît donc nécessaire de placer, en amont du codeur, un phase de pré-analyse dédiée au mouvement au sein de la vidéo. Il sera possible d'appréhender de façon plus juste le mouvement des objets[2] et leur ancrage temporel. Cette analyse doit pouvoir caractériser le mouvement physique ainsi que la complexité locale de l'image dans le but d'accélérer le codage, en choisissant la meilleure stratégie offerte par le codeur H.264 (i.e. le meilleur jeu de paramètres du codeur). La connaissance approfondie des objets (cycle de vie, suivi spatio-temporel, texture, ...) présents dans une scène permettra notamment de décider, pour chacun d'entre eux, quelles sont les meilleures images de référence pour la prédiction et les modes les mieux adaptés à leur codage.

Ce document présente le manuel d'utilisation du logiciel de filtrage et pré-analyse du flux vidéo. Le deuxième chapitre est consacré à la description du logiciel *gui4x264*. Ce logiciel est un moyen simple et interactif de créer des jeux de paramètres interprétables par un encodeur AVC. Le dernier chapitre de ce rapport présente les modifications apportées au codeur *x264*, afin de prendre en compte les directives

[1] TVHD : télévision haute-définition.

[2] Un objet désigne un ensemble de macroblocs dont le mouvement, la couleur et la texture sont homogènes.

de codage générées par notre logiciel de filtrage et de pré-analyse de flux vidéo.

Les dernières étapes de l'outil de pré-analyse de flux vidéo haute définition en vue d'un encodage en temps réel sous le standard H.264 sont présentées en annexes (les premières étapes ayant été présentées dans les rapports précédents). L'objectif est donc de fournir au codeur H.264 un jeu de paramètres adapté au codage d'une séquence vidéo et présentant une cohérence spatio-temporelle fonction des objets présents dans la scène. Le première partie décrit la segmentation spatio-temporelle par une approche markovienne combinant des informations de mouvement, de couleur, de texture et de connexité spatiale et temporelle. Ensuite, une description d'une méthode de modélisation spatio-temporelle de l'attention visuelle pré-attentive est réalisée. La dernière partie présente une application de compression sélective directe utilisant les informations issues de notre modèle d'attention visuelle pré-attentive.

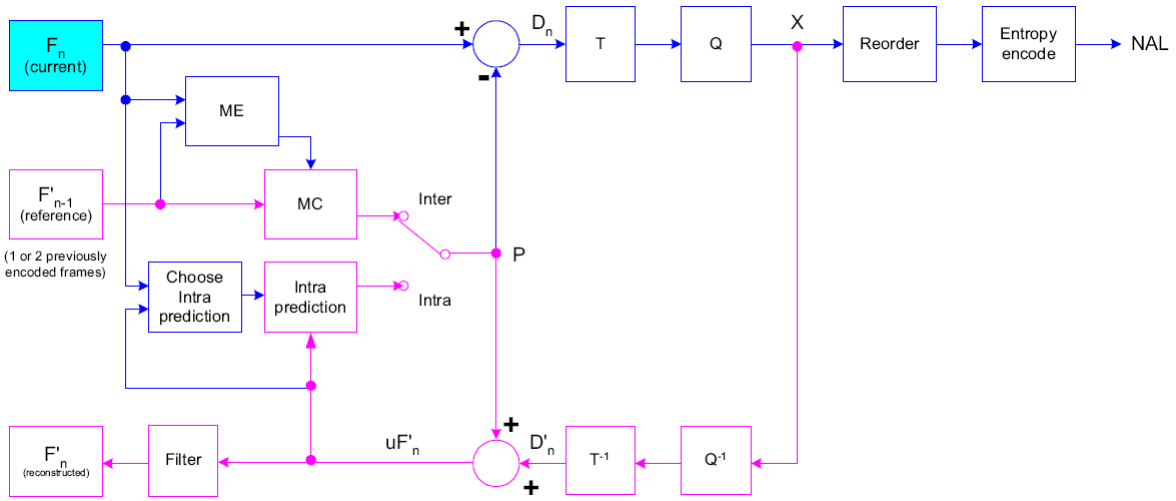

FIG. 1 – Schéma du codeur H.264/AVC [**??**].

Chapitre 1

Le logiciel de filtrage et de pré-analyse de flux vidéo

1.1 Introduction

L'outil de conditionnement et de pré-analyse de flux vidéo doit permettre de guider les choix d'un encodeur H.264 AVC pour le codage d'objets spatio-temporels contenus dans une séquence vidéo. Notre outil doit donc générer des informations interprétables par un encodeur AVC. Les paramètres à piloter dans le codeur ont été fixés dans le cadre du projet, il s'agit:

- du mode de codage (Intra, Inter P, Inter B);
- de la taille de partition des macroblocs;
- du paramètre de quantification (QP);
- de l'ordre de codage des images.

Á partir de la décomposition d'une séquence en objets spatio-temporels et des cartes de saillance, notre outil développé doit permettre de choisir ces paramètres pour chaque objet. Le jeu de paramètres sera alors formaté et stocké dans un fichier texte. Ce fichier sera donné en entrée d'un encodeur H.264 modifié et permettra de guider les choix de codage.

Nous avons développé notre logiciel de conditionnement et de pré-analyse de flux vidéo à l'aide du logiciel *Microsoft Visual Studio 2005* (MVS 2005). La figure 1.1 présente la vue du logiciel de pré-analyse de flux vidéo obtenue avec MVS 2005.

1.2 Configuration du logiciel de pré-analyse

L'exécution du logiciel de conditionnement et de pré-analyse de flux vidéo se réalise via une application en lignes de commande (*CLI : Command Line Interface*). Plusieurs options doivent être précisées concernant la vidéo à traiter et les paramètres de pré-analyse. Ces possibilités sont données lors d'un appel « ArchiPEG2.0.exe –h » dans l'invité de commandes :

```
1 E:\ArchiPEG2.0.exe -h
2     Syntax: ArchiPEG2.0.exe -i input_file -o output_directory -f
          [int] -width [int] -height  [int] -frames [int] [options]
3         -i input_file
4         -o output_directory
```

```
5         -f [int] (video format (444;422;420))
6         -width [int] (frame width)
7         -height [int] (frame height)
8         -interlaced [bool] (default false)
9         -frames [int] (number of frames)
10        -GOPsize [int] (default 9)
11        -bWidth [int] (block size, default 16)
12        -bHeight [int] (block size, default 16)
13        -wSize [int] (size of the search window, default 20)
14  Appuyez sur une touche pour continuer...
```

Les différents paramètres à préciser lors de l'exécution du logiciel de pré-analyse de flux vidéo sont détaillés ci-dessous ainsi que les options :

- **-i vidéo_en_entrée** : vidéo en entrée de l'outil de conditionnement et de pré-analyse de flux vidéo;

```
1  -i "E:\\input\\mobcal.yuv"
```

- **-o répertoire_de_sortie** : répertoire où sont sauvegardés les fichiers créés en sortie du logiciel de conditionnement et de pré-analyse de flux vidéo;

```
1  -o "E:\\output\\mobcal"
```

- **-f [entier]** : format de la vidéo à traiter (444 ; 422 ; 420);

```
1  -f 420
```

- **-width [entier]** : largeur en pixels de la vidéo à traiter;

```
1  -width 1280
```

- **-height [entier]** : hauteur en pixels de la vidéo à traiter;

```
1  -height 720
```

- **-interlaced [booléen]** : type de la vidéo (entrelacée (*true*) ou progressive (*false*)). Par défaut l'option **-interlaced** à pour valeur *false*;

```
1  -interlaced false
```

- **-frames [entier]** : nombre d'images à traiter de la vidéo en entrée;

```
1  -frames 500
```

- **-GOPsize [entier]** : taille en nombre d'images des segments temporels. Par défaut l'option **-GOPsize** à pour valeur 9, ce qui correspond à $180ms$ dans le cas d'une vidéo progressive avec un taux d'affichage de 50 images par seconde;

```
-GOPsize 9
```

- **-bWidth [entier]** : largeur en pixels des blocs pour la pré-analyse du flux vidéo. Par défaut, la largeur des blocs est de 16 pixels;

```
-bWidth 16
```

- **-bHeight [entier]** : hauteur en pixels des blocs pour la pré-analyse du flux vidéo. Par défaut, la hauteur des blocs est de 16 pixels;

```
-bHeight 16
```

- **-wSize [entier]** : taille en pixels de la fenêtre de recherche pour l'estimation de mouvement réalisée au début de la pré-analyse de flux vidéo. Par défaut, la taille de la fenêtre de recherche est 20 pixels;

```
-wSize 20
```

1.3 Le mode DEBUG

Il s'agit ici de permettre à l'utilisateur de ne plus utiliser le logiciel de pré-analyse comme une "boite noire". Pour cela, il doit pouvoir parcourir le code en mode DEBUG, ce qui n'est pas possible lorsque des entrées consoles sont nécessaires.

Dans la fonction ***main()*** du fichier archipeg.cpp, nous ajoutons donc une variable DEBUG_MODE qui indique si le code sera parcouru en mode DEBUG :

```
int DEBUG_MODE = 1; //logiciel de pré-analyse en mode DEBUG
```

Dans ce cas, les entrées de la CLI ne seront plus disponibles et il faut les passer directement dans la fonction ***main()*** :

```
int main(int argc, char *argv[]) {

   ...

   if ( DEBUG_MODE == 1 )
   {
       // Vidéo en entrée:
       userParameters.inputVideoFileName = new char[500];
       snprintf(userParameters.inputVideoFileName,500,"%s","E:\\
          input\\mobcal.yuv");
       // Répertoire de sortie:
       userParameters.outputDir = new char[500];
       snprintf(userParameters.outputDir,500,"%s","E:\\output\\
          mobcal");

       // Format de la vidéo en entrée (444 ; 422 ; 420):
       userParameters.format = 420;
       // Largeur des images (en pixels):
       userParameters.width  = 1280;
```

```
18          // Hauteur des images (en pixels):
19          userParameters.height = 720;
20
21          // Vidéo entrelacée ou progressive:
22          userParameters.isInterlaced = false;
23          // Taille de la vidéo (nombre d'images):
24          userParameters.nbFrames = 500;
25          // Taille d'un segment temporel:
26          userParameters.GOPsize = 9;
27          // Largeur des blocs (en pixels):
28          userParameters.bWidth  = 16;
29          // Hauteur des blocs (en pixels):
30          userParameters.bHeight = 16;
31          // Taille de la fenêtre de recherche (en pixels):
32          userParameters.wSize = 20;
33
34          //options permettant de générer les fichiers de sortie
35          userParameters.generate_MV  = true;    // vecteurs de
               mouvement des tubes
36          userParameters.generate_RAW = false;   //file.raw: raw
               data for x-and-y-coordinates of tubes
37          userParameters.generate_TXT = true;    //file.txt with
               information computed about tubes
38          userParameters.generate_SEGMENTATION_RESULT = true; //file
                with only-{a1,...,a4} compensated mvs for a full GOP
39      }
40
41      ...
42
43  }
```

1.4 Fichiers générés en sortie

Notre outil de conditionnement et de pré-analyse de flux vidéo génère en sortie un fichier de directives par segment temporel. Celui-ci contient les informations de codage qui sont alors formatées et écrites dans le fichier texte correspondant au segment temporel traité sous la forme:

GOF = 0;
ordre = 8,7,6,5,4,3,2,1,0 ;
frame = 0 ; mbx = 0 ; mby = 0 ; mode = 2 ; QP = 2 ; partition = 0 ;
...
frame = 4 ; mbx = 13 ; mby = 8 ; mode = 1 ; QP = 1 ; partition = 0 ; ref = -1;
...

Le fichier formaté contient donc le numéro du segment courant (ici $GOF = 0$), l'ordre de codage des images et les informations de codage pour chaque macrobloc:

- le numéro d'image à laquelle appartient le macrobloc;
- la position (mbx,mby) du macrobloc dans cette image;
- le mode de codage (voir l'annexe);
- l'indice de quantification en fonction de la saillance du bloc;
- la taille des partitions testées (voir l'annexe);

- le numéro de l'image référence utilisée, c'est-à-dire l'indice de l'image utilisée dans le buffer des images de référence.

Le fichier texte ainsi généré peut alors être fourni en entrée d'un codeur modifié. Le codeur utilisera les informations disponibles dans le fichier texte pour guider ses choix et aura un comportement par défaut lorsqu'aucune instruction ne sera disponible.

Ci-dessous sont présentées les valeurs numériques associées aux modes de codage et aux tailles de partitions et de sous-partitions qui seront testées:

```
#define MODE_CODEUR  0
#define MODE_INTER   1
#define MODE_INTRA   2

#define I_16x16     0
#define I_4x4       1
#define I_16x16_4x4 2

#define P_16x16           0
#define P_16x8            1
#define P_8x16            2
#define P_8x8             3
#define P_16x16_16x8      4
#define P_16x16_8x16      5
#define P_16x16_8x8       6
#define P_16x8_8x16       7
#define P_16x8_8x8        8
#define P_8x16_8x8        9
#define P_16x16_16x8_8x16 10
#define P_16x16_16x8_8x8  11
#define P_16x16_8x16_8x8  12
#define P_16x8_8x16_8x8   13
#define P_ALL_PARTITION   14

#define P_SUB_8x8            0
#define P_SUB_8x4            1
#define P_SUB_4x8            2
#define P_SUB_4x4            3
#define P_SUB_8x8_8x4        4
#define P_SUB_8x8_4x8        5
#define P_SUB_8x8_4x4        6
#define P_SUB_8x4_4x8        7
#define P_SUB_8x4_4x4        8
#define P_SUB_4x8_4x4        9
#define P_SUB_8x8_8x4_4x8    10
#define P_SUB_8x8_8x4_4x4    11
#define P_SUB_8x8_4x8_4x4    12
#define P_SUB_8x4_4x8_4x4    13
#define P_SUB_ALL_PARTITION  14
```

Il est également possible de générer des fichiers après chaque étape de notre outil de pré-analyse et de conditionnement de flux vidéo :

- fichier texte contenant les vecteurs de mouvement de chaque tube spatio-temporel ;
- fichier texte contenant les paramètres du mouvement global pour chaque segment temporel ;
- vidéo au format YUV contenant les cartes de la segmentation basée mouvement de chaque segment temporel ;

- vidéo au format YUV contenant les cartes de la segmentation spatio-temporelle par approche markovienne de chaque segment temporel ;
- vidéo au format YUV contenant les cartes de saillance spatio-temporelle de chaque segment temporel.

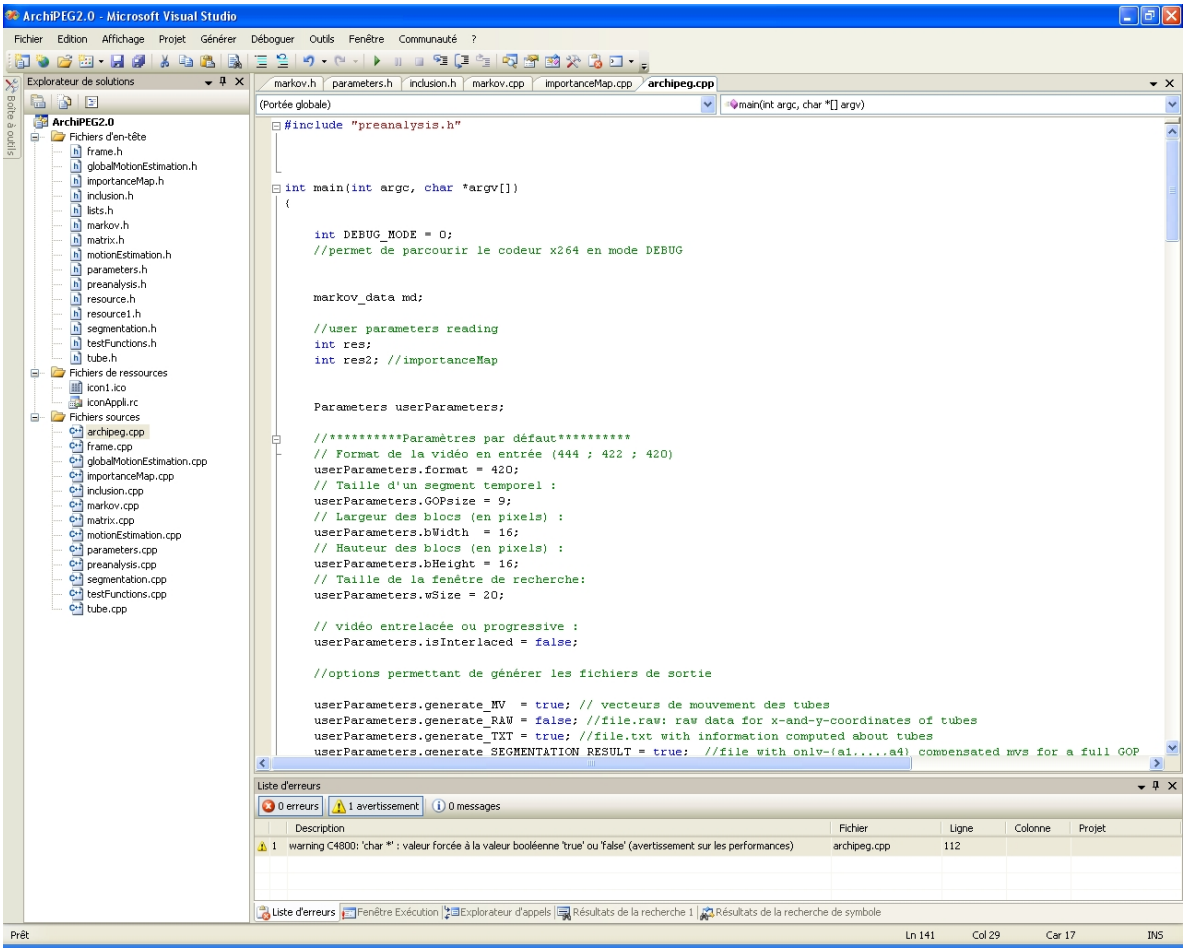

FIG. 1.1 – Solution ArchiPEG2.0 avec MVS 2005

Chapter 2

Le logiciel gui4x264

2.1 Introduction

Le besoin d'un outil tel que le logiciel gui4x264 (*Graphical User Interface for x264*) est apparu au cours de l'avancement du projet RIAM ArchiPEG. De fait, dans ce projet, il est question de guider les choix d'un encodeur H.264 AVC pour le codage d'objets spatio-temporels contenus dans une séquence vidéo. Le but de l'outil gui4x264 est donc de fournir un moyen simple et interactif de créer des jeux de paramètres interprétables par un encodeur AVC. Les paramètres à piloter dans le codeur ont été fixés dans le cadre du projet, il s'agit:

- du mode de codage (Intra, Inter P, Inter B);
- de la taille de partition des macroblocs;
- du paramètre de quantification (QP);
- de l'ordre de codage des images.

Á partir de la décomposition d'une séquence en objets spatio-temporels, l'outil développé doit donc permettre de choisir ces paramètres pour chaque objet. Le jeu de paramètres sera alors formaté et stocké dans un fichier texte. Ce fichier sera donné en entrée d'un encodeur H.264 modifié et permettra de guider les choix de codage. Notons que la décomposition de la séquence en objets spatio-temporels est fournie par l'outil ArchiPEG2.0, qui se base sur une estimation de mouvement par tubes spatio-temporel et une approche markovienne pour segmenter la vidéo originale.

2.2 Présentation de l'outil

2.2.1 Création d'un projet

Pour utiliser le logiciel gui4x264, il faut disposer en entrée de la décomposition d'une séquence vidéo en objets spatio-temporels. Cette décomposition est fournie par le logiciel ArchiPEG2.0 sous la forme d'une suite de cartes de segmentation. Chacune des cartes représente la segmentation spatio-temporelle de l'image centrale d'un segment temporel de neuf images. Pour obtenir la carte de segmentation de chacune des neufs images du segment courant, il faut compenser en mouvement la carte centrale. Pour cela, un fichier contenant les informations de mouvement pour chaque segment temporel (mouvement global et mouvements locaux) est disponible et également fourni par le logiciel ArchiPEG2.0. Les informations passées en entrée du logiciel gui4x264 sont donc:

- le fichier qui contient la décomposition de la séquence vidéo en cartes de segmentation;
- la dimension de chaque carte de segmentation (hauteur, largeur et format d'échantillonnage de la couleur);

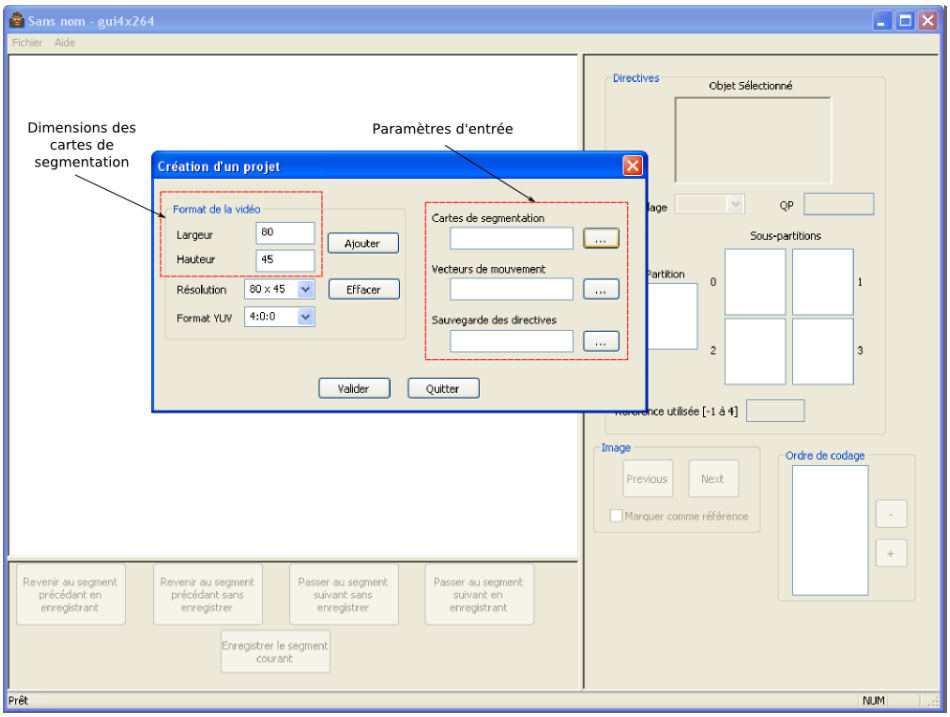

Figure 2.1: Entrées du logiciel gui4x264

- le fichier qui contient les informations de mouvement;
- le fichier dans lequel sera sauvegardé le jeu de paramètres que doit utiliser l'encodeur H.264.

La figure 2.1 présente la boîte de dialogue qui permet de renseigner chacune de ces informations. Notons que le nom de fichier pour sauvegarder les directives est en réalité un nom générique. Pour accéder rapidement aux informations contenues dans un fichier, ce dernier doit avoir une taille en octets relativement faible. Nous créons donc un fichier texte pour chaque segment temporel. Si le nom passé en entrée de la boîte de dialogue est par exemple *directives.txt*, les fichiers créés seront alors nommés *directives0.txt*, *directives1.txt*, ...

Le logiciel vérifie que les fichiers passés en entrée sont correctement formatés, si c'est le cas, l'utilisateur peut alors visualiser les cartes de segmentation et utiliser l'interface pour fixer interactivement le codage de chaque objet.

2.2.2 Choix du jeu de paramètres de chaque objet

Une fois le projet créé, le logiciel permet de naviguer d'un segment temporel au segment suivant (ou précédant) et de parcourir les neufs images du segment courant. Pour cela, le logiciel compense en mouvement la carte de segmentation centrale du segment courant et en déduit les cartes de segmentation associées aux huit autres images. L'utilisateur peut alors choisir le jeu de paramètres pour chaque objet de chacune des images de la séquence vidéo.

Le choix de l'objet courant s'effectue de façon interactive en cliquant directement sur sa représentation dans l'image courante. L'objet sélectionné s'affiche alors en blanc dans le cadre "Objet Sélectionné" relatif aux directives. L'utilisateur peut alors choisir de coder cet objet en mode Inter (seul le mode Inter P est pris en compte), en mode Intra ou de laisser le codeur agir par défaut.

Dans le cas où l'utilisateur choisit le mode "Codeur", les champs relatifs au QP, aux tailles de partitions et aux tailles de sous-partitions ne sont pas disponibles: le codeur se comporte par défaut.

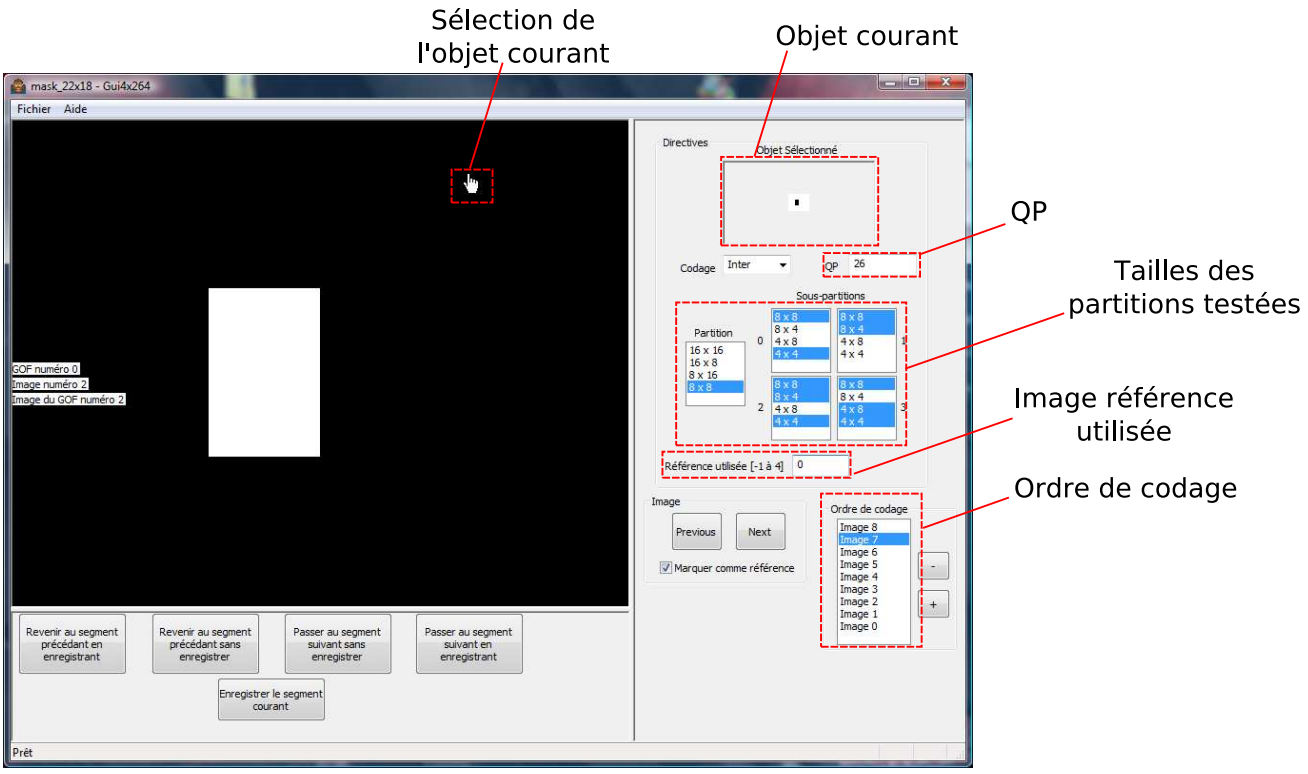

Figure 2.2: Choix du jeu de paramètres

Dans le cas contraire, l'utilisateur peut fixer la valeur du QP (comprise entre 0 et 51), les tailles de partitions et de sous-partitions à tester et l'image de référence à utiliser (la valeur -1 indique que toutes les images disponibles dans le buffer sont utilisées). Parallèlement, l'utilisateur fixe l'ordre de codage des images qui peut être différent de l'ordre naturel d'affichage.

Un exemple est présenté en figure 2.2. Il s'agit d'une carte de segmentation extraite d'une séquence synthétique, dans laquelle un objet indéformable suit une translation horizontale de gauche à droite sur un fond texturé et immobile. L'objet sélectionné est le fond (en blanc dans le cadre "Objet Sélectionné"), le QP est fixé à 26, le mode choisi est le mode Inter, les tailles de partitions testées sont en surbrillance et la prédiction s'effectuera à partir de la première image[1] du buffer des images de référence. Ici, l'ordre de codage des images est inversé par rapport à l'ordre d'affichage (indices de 8 à 0).

Une fois que tous les paramètres liés au codage du segment courant sont entrés, l'utilisateur peut cliquer sur le bouton "enregistrer le segment courant". Les informations de codage sont alors formatées et écrites dans le fichier texte correspondant.

2.3 Évolutions possibles

Ce chapitre a présenté le logiciel le logiciel gui4x264 qui a été réalisé de manière à fournir les outils de bases absolument nécessaires à la conception de tests lors du projet ArchiPEG. Cependant, d'autres fonctionnalités supplémentaires pourraient être ajoutées, la liste suivante propose quelques évolutions envisageables sur l'outil gui4x264:

[1] l'image k du buffer est indexée par l'indice $k-1$

- L'outil ne traite que le cas du codage Inter P. Nous pouvons donc souhaiter ajouter la gestion du codage Inter B. Cependant, cet ajout modifiera la gestion des images références: l'utilisateur devra choisir une image référence passée et une image référence future pour prédire l'image courante.
- L'outil ne permet pas de choisir une référence pour chaque sous-partition d'un macrobloc. Cependant, une telle option présente un intérêt très limité dans le cadre du projet ArchiPEG. En effet, nous fixons une image de référence pour chaque objet et non pour chacune des partitions des macroblocs.
- Un bouton supplémentaire pourrait être ajouté pour permettre à l'utilisateur d'appliquer un jeu de paramètres à un objet sur toutes les images du segment courant et de sauvergarder ses préférences.

Chapitre 3

Modifications du codeur

VideoLAN fournit de manière libre le projet x264 sous la forme d'une solution (fichier x264.sln) pour le logiciel *Microsoft Visual Studio 2005* (MVS 2005). Il est évidemment possible d'utiliser x264 sans cette interface de développement, mais dans ce cas l'architecture du projet n'est plus disponible de façon aussi simple et structurée. La figure 3.1 présente la vue du codeur x264 obtenue avec MVS 2005. Cette solution permet de "naviguer" dans le code de façon simplifiée, nous allons donc l'utiliser pour modifier le code de l'encodeur x264.

3.1 Les parties du code à modifier

Les parties à modifier dans le code du codeur x264 sont déterminées par le cahier des charges du sous-projet "*Pré-analyse et conditionnement d'un flux vidéo HD*" du projet RIAM *ArchiPEG*. Ce sous-projet va fournir pour chaque image de la séquence vidéo à encoder, un jeu de paramètres :

- les valeurs de QP (*Quantization Parameter*) pour chaque macrobloc ;
- le mode de chaque macrobloc (I-P-B) ;
- les tailles de partition de chaque macrobloc ;
- l'image référence à utiliser pour un macrobloc de type Inter-P ou Inter-B.

Par défaut, le codeur ne donne pas la possibilité d'intégrer ce jeu de paramètres, nous devons donc cibler et modifier certaines parties du code. Afin de naviguer dans le code en mode DEBUG, une modification doit également être apportée au codeur afin de l'empêcher de fonctionner avec une CLI.

3.2 Les directives de codage

L'objectif est de fournir au codeur x264 un jeu de paramètres relatif à la quantification, au choix des modes de prédiction et au choix des images de référence. Ce jeu de paramètres est fourni en entrée du codeur sous la forme d'un fichier texte à analyser. En réalité, il y a un fichier texte descriptif de chaque segment temporel à encoder[1]. Pour que le codeur utilise les jeux de paramètres définis, plusieurs étapes sont nécessaires :

- ajout d'une option en entrée du codeur pour lire les directives,
- ajout de données dans le codeur pour stocker ces directives,
- lecture des directives.

Enfin, l'étape d'analyse du codeur pourra être modifiée en tenant compte des jeux de paramètres lus en entrée.

[1] Un segment temporel est un ensemble de 9 images successives

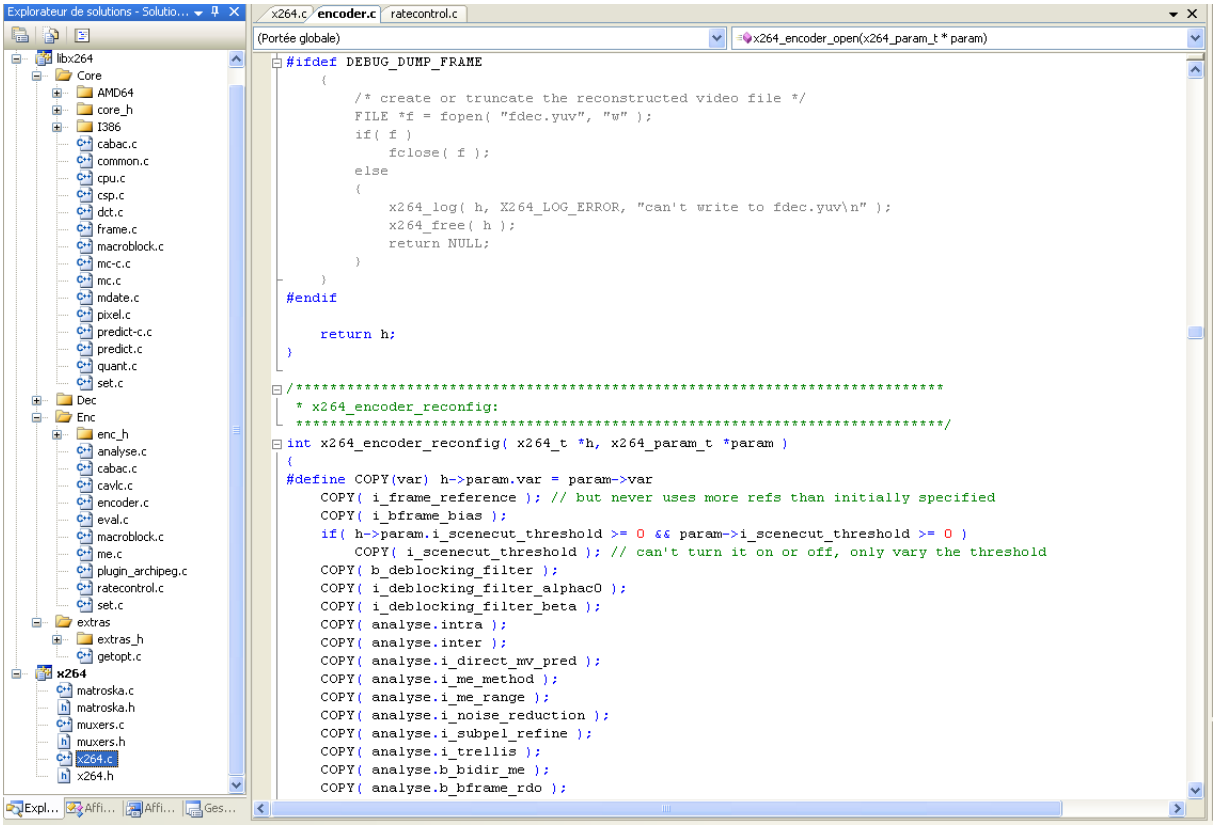

FIG. 3.1 – Solution x264 avec MVS 2005

3.2.1 Ajout d'une option pour lire les fichiers de directives de codage

Vu le grand nombre de segments temporels qui peuvent constituer une séquence vidéo (une séquence de 500 images représente 55 segments), il n'est pas envisageable de passer le nom de chacun des fichiers de directives en paramètre d'entrée du codeur. Nous choisissons donc d'ajouter en entrée du codeur un unique paramètre, qui est le nom générique associé aux fichiers de directives. L'option sera précisée lors de l'utilisation de la CLI en écrivant :

"x264.exe ***-d generic_filename.txt*** ..."

Dans ce cas, les modifications que nous allons apporter au codeur permettront de lire les fichiers : generic_filename0.txt, generic_filename1.txt, ... Pour intégrer cette nouvelle option au codeur x264, nous modifions tout d'abord la fonction ***Help()*** du fichier x264.c, afin de faire figurer dans l'aide les options relatives aux fichiers de directives :

```
H0( "\n" );
H0( "Forced Analysis:\n" );
H0( "\n" );
H0( "-d,--directive <string>  Specify the file which contains
    the directives\n" );
```

De même, la structure ***x264_param_t*** définie dans le fichier d'en tête x264.h est enrichie d'un champ supplémentaire. Ce champ permet de conserver en mémoire le nom générique des fichiers contenant les jeux de paramètres pour chaque segment temporel :

```
typedef struct
{
  ...
  /* Video Properties */
  char *s_generic_filename_dir;
  ...
} x264_param_t;
```

Ce champ sera initialisé et renseigné dans la fonction ***Parse()*** du fichier x264.c, qui permet de lire et de stocker les entrées passées dans la CLI de x264 :

```
static int Parse( int argc, char **argv, x264_param_t *param,
    cli_opt_t *opt )
{
  ...
  param->s_generic_filename_dir = NULL;
  ...
  /* Parse command line options */
  for( ;; ) {
    ...
    static struct option long_options[] =
    {
      { "help",no_argument,NULL, 'h' }, ...
      {"directive",required_argument,NULL,'d'}, ...
    };
    int c = getopt_long(argc,argv,"8A:B:b:d:f:hI:i:m:o:p:q:r:t:
       Vvw",long_options,&long_options_index);
    ...
    switch(c)
    {
      ...
      case 'd':
```

```
20          if(optarg == NULL){
21            fprintf( stderr, "x264 [error]: Intern Error\n" );
22            return -1;
23          }
24          param->s_generic_filename_dir = (char *)malloc( (strlen(
                optarg)+5)*sizeof(char) );
25          _snprintf(param->s_generic_filename_dir,(strlen(optarg)
                +5)*sizeof(char),"%s",optarg);
26          break;
27         ...
28       }
29     }
30   }
```

Le nom générique des fichiers de directives étant maintenant connu, il faut analyser chacun de ces fichiers, en récupérer les informations et les stocker dans des structures adaptées. L'objet de la prochaine étape est la création de ces structures.

3.2.2 Structures de stockage des directives

Les structures que nous allons définir vont permettre de stocker pour un segment temporel complet, l'ensemble des paramètres de chaque macrobloc : QP, mode de prédiction, taille de partition, tailles des sous-partitions, références utilisées. Nous définissons donc une structure contenant ces informations élémentaires relatives à chaque macrobloc :

```
1  typedef struct {
2    int mode;  //mode de prédiction
3    int qp;    //QP
4    int partition; //taille de partition
5    int souspartition[4];
6    int iRef; //références utilisées
7  } codage;
```

Á partir de cette structure, il est possible de créer une structure englobante, de plus haut-niveau, pour définir entièrement un segment temporel :

```
1  typedef struct {
2    int i_gof;
3    int ordre_codage[9];
4    int i_real_frame_num;
5    int i_gof_frame_num;
6    int i_mb_xy_max;
7    codage* mbs[9];
8  } segment;
```

où :
- i_gof désigne l'indice du segment temporel courant ;
- ordre_codage indique l'ordre temporel dans lequel sont codées les images du segment courant ;
- i_real_frame_num désigne l'indice de l'image courante ;
- i_gof_frame_num désigne l'indice de l'image courante dans le segment courant ;
- i_mb_xy_max désigne le nombre total de macroblocs dans une image ;
- mbs désigne le jeu de paramètres pour chaque macrobloc du segment temporel courant.

Lorsqu'un segment temporel est traité, cette structure doit être accessible à n'importe qu'elle étape. Pour s'en assurer, nous créons la variable ***currentSegment*** de type ***segment*** et l'intégrons à la structure ***x264_t***, définie dans le fichier common.h, et dont l'instantiation ***h*** a un comportement pseudo-global (variable accessible dans la quasi-totalité des fonctions du projet x264) :

```
struct x264_t {
  ...
  char *gofDirectives;
  segment currentSegment;
};
```

Notons qu'en plus de la variable ***currentSegment***, nous ajoutons également la variable ***gofDirectives*** qui permettra de garder en mémoire le contenu du fichier texte relatif au jeu de paramètres du segment temporel courant.

3.2.3 Lecture des directives

Comme nous l'avons précisé, les directives contenues dans un fichier texte vont renseigner le jeu de paramètres pour chaque macrobloc d'un segment temporel de neuf images. Nous allons donc modifier la fonction ***Encode()*** du fichier x264.c pour lire le jeu de paramètres qui permettra d'encoder les images d'un segment temporel :

```
/* Lit le fichier de directives et le met dans h->gofDirectives
   */
if ( h->param.s_generic_filename_dir != NULL )
{
  ...
  sprintf(s_fileDir,"%s%d.txt",h->param.s_generic_filename_dir,
     i_gof);
  fid = fopen(s_fileDir,"rb");
  if ( fid != NULL )
  {
    fseek(fid,0,SEEK_END);
    fileSize = ftell(fid);
    fseek(fid,0,SEEK_SET);
    h->gofDirectives = (char *)malloc(sizeof(char)*fileSize);
    fread(h->gofDirectives,sizeof(char),fileSize,fid);
    fclose(fid);
  }

  /* Lit dans h->gofDirectives les directives pour le segment
     courant */
  sprintf_s(ligneCherchee,100,"GOF = %d",i_gof);
  debut_gof = strstr(h->gofDirectives,ligneCherchee);
  ...
```

La modification de cette fonction permet de stocker les informations contenues dans le fichier texte, relatif au segment courant, dans la variable ***currentSegment*** définie précédemment. Notons que si aucun fichier texte n'est trouvé où que si ce fichier texte est mal formaté, le codeur gardera son comportement par défaut. Un exemple de fichier texte contenant les directives de codage et formaté correctement est présenté ci-dessous :

```
GOF = 0 ;
ordre = 0,1,2,3,4,5,6,7,8 ;
frame = 0 ; mbx = 0 ; mby = 0 ; mode = 2 ; QP = 26 ; partition =
    0 ;
frame = 0 ; mbx = 1 ; mby = 0 ; mode = 2 ; QP = 26 ; partition =
    0 ;
...
```

```
6 frame = 4 ; mbx = 9 ; mby = 10 ; mode = 1 ; QP = 19 ; partition
      = 3 ; souspartition = 0,0,0,0 ; ref = -1 ;
7 ...
```

Le formatage renseigne tout d'abord le numéro du segment temporel courant (ici GOF = 0), puis l'ordre de codage des images de ce segment (ici l'ordre naturel) et enfin le mode de codage de chacun des macroblocs de ce segment.

Un macrobloc est indexé par :
- le numéro d'image à laquelle il appartient,
- sa position (x,y) dans cette image (mbx = 1 ; mby = 0).

Pour chaque macrobloc, le mode de codage est précisé. Trois valeurs sont possibles :
- 0 : comportement du codeur par défaut,
- 1 : codage en mode inter,
- 2 : codage en mode intra.

Toute autre valeur conduira le codeur à garder son comportement par défaut (mode = 0). Dans le cas des modes intra ou inter, le paramètre de quantification est précisé (QP = 26) et les tailles de partition testées également. Les valeurs numériques associées aux tailles des partitions testées sont toutes disponibles dans le fichier inclusion.h. Pour certaines partitions (partition = 3, qui correspond à une partition 8×8), les tailles de sous-partitions testées sont également précisées. Les valeurs numériques des sous-partitions testées sont également disponibles dans le fichier inclusion.h.

Le jeu de paramètres d'un segment temporel en mémoire, il convient à présent de modifier le code de x264 relatif à l'analyse pour tenir compte des directives imposées.

3.3 Modification de l'analyse

La modification de l'analyse vise ici trois objectifs :
- choix des modes de prédiction,
- choix des images de référence,
- choix du paramètre de quantification QP.

Nous allons présenter pour chacune de ces modifications, les parties du code original de x264 qui ont été modifiées.

3.3.1 Choix des modes de prédiction

Pour forcer le codeur à utiliser un jeu de paramètres imposé pour un macrobloc, il faut modifier la fonction d'analyse ***x264_macroblock_analyse()*** du fichier analyse.c. Par défaut, cette fonction calcule, pour chaque macrobloc, un coût associé à chaque mode de codage (inter ou intra) et pour chaque taille de partition. Nous allons donc modifier cette fonction pour qu'elle ne calcule que les coûts associés aux modes et aux partitions que le fichier texte de directives autorise à tester.

Notons qu'en pratique il est complexe de modifier le codeur de sorte qu'il ne teste pas tous les modes de prédiction. Par exemple, ce dernier ne peut mener l'analyse d'un macrobloc avec deux partitions 16×8 qu'après avoir fini l'analyse avec une partition 16×16. Nous allons donc tester tous les modes de prédiction, les modes qui n'étaient pas prévus par le fichier de directives se voient alors attribuer un coût maximal, afin que le codeur ne les retienne pas. Notons qu'avec cette méthode, les choix de codage seront respectés mais que les temps de calcul ne seront pas diminués.

La modification de cette fonction est complexe et assez longue, les extraits de code ne sont donc pas présentés ici. Cependant les modifications de cette fonction sont toutes commentées dans le code. Notons tout de même que les informations relatives au mode choisi après l'analyse sont stockées dans les variables : ***h->mb.i_type*** et ***h->mb.i_partition***.

3.3.2 Gestion des images de référence

Plusieurs fonctions interviennent dans la gestion des images de référence. La fonction ***x264_reference_update()*** du fichier encoder.c permet de gérer les images qui rentrent et sortent du DPB (*Decoded Picture Buf-*

fer). Par défaut, ce buffer est géré comme une pile FIFO (*First In First Out*). La modification de cette fonction permettrait par exemple de gérer des images références à long-terme. Le buffer contenant les images de référence est défini dans la structure ***x264_t*** du fichier common.h :

```
1  struct x264_t
2  {
3     ...
4     /* frames used for reference + sentinels */
5     x264_frame_t *reference[16+2];
6     ...
7  }
```

Le codeur permet donc de stocker 16 images de référence au maximum. Les références contenues dans ce buffer sont alors organisées en deux listes L0 et L1 comme indiqué dans la norme H.264. La liste L0 ne contient que des images de référence passées par rapport à l'image courante codée, et la liste L1 ne contient que des images futures. Notons néanmoins que, contrairement aux spécifications données par la norme, la liste L1 ne peut être utilisée que dans le cas d'images codées avec le mode Inter-B. Une image Inter-P ne peut donc pas utiliser d'image référence future. Dans le cadre du projet ArchiPEG, ce point est gênant, et une solution simple à été envisagée : modifier l'ordre des images de la séquence originale avant d'encoder avec le codeur x264. Ces listes sont construites dans la fonction ***x264_reference_build_list()*** du fichier encoder.c. Ces deux listes sont également définies dans la structure ***x264*** _t du fichier common.h :

```
1  struct x264_t
2  {
3     ...
4     /* reference lists */
5     int           i_ref0;        /* nombre d'images dans la liste
         L0 */
6     x264_frame_t *fref0[16+3];   /* ref list 0 */
7     int           i_ref1;        /* nombre d'images dans la liste
         L1 */
8     x264_frame_t *fref1[16+3];   /* ref list 1 */
9     ...
10 }
```

Pour forcer le choix d'un image de référence, nous modifions dans le fichier analyse.c les fonctions d'analyse :

- ***x264_mb_analyse_inter_p16x16()***,
- ***x264_mb_analyse_inter_p16x8()***,
- ***x264_mb_analyse_inter_p8x16()***,
- ***x264_mb_analyse_inter_p8x8().***

Les fonctions équivalentes doivent être modifiées pour le cas des images B. En pratique, seule la fonction ***x264_mb_analyse_inter_p16x16()*** a été modifiée. En effet, le pré-traitement indique une image référence pour un objet constitué de plusieurs macroblocs. Les partitions des macroblocs d'un même objet utiliseront donc toutes la même référence, les fonctions d'analyse pour des partitions inférieures utiliserons donc la même image de référence que celle utilisée lors de l'analyse 16×16. La modification du code source est simple : nous fixons l'indice de l'image référence utilisée dans la liste L0, au lieu de laisser le codeur tester toutes les images de référence.

3.3.3 Modification du QP

Le QP d'un macrobloc est fixé dans la fonction ***x264_macroblock_analyse()*** du fichier analyse.c et passé en paramètre de la fonction ***x264_mb_analyse_init()*** :

```
1  void x264_macroblock_analyse( x264_t *h )
```

```
{
    ...
```

```
    ...
    if ( mode == MODE_INTRA || mode == MODE_INTER ){
       i_qp = h->currentSegment.mbs[i_f][i_mb_xy].qp;
       weightedQP =  (int)(x264_ratecontrol_qp(h) + (-1*i_qp));
       x264_mb_analyse_init( h, &analysis, weightedQP );
    }
    else
       x264_mb_analyse_init(h,&analysis,x264_ratecontrol_qp(h));
    ...
}
```

Conclusion

Ce document a présenté le manuel d'utilisation du logiciel de filtrage et pré-analyse du flux vidéo. Le deuxième chapitre étant consacré à la description du logiciel *gui4x264*. Ce logiciel est un moyen simple et interactif de créer des jeux de paramètres interprétables par un encodeur AVC. Le dernier chapitre de ce rapport a présenté les modifications apportées au codeur *x264*, afin de prendre en compte les directives de codage générées par notre logiciel de filtrage et de pré-analyse de flux vidéo.

Dans le cadre du sous-projet 4 : "Pré-analyse et conditionnement du flux vidéo en haute définition" du projet ArchiPEG, les derniers travaux à réaliser, seront l'optimisation des algorithmes de pré-traitement du flux vidéo ainsi que les phases de test des algorithmes sur prototypes et sur la plate-forme d'accueil du projet.

Annexe A

Segmentation par approche markovienne

A.1 Introduction

Comme cela a été présenté précédemment, une segmentation au sens du mouvement uniquement peut ne pas suffire pour créer une décomposition satisfaisante d'un segment temporel[1] en objets spatio-temporels. En effet, dans certaines scènes présentant des mouvements de caméra complexes (zoom, rotation) et des contenus spatiaux uniformes, les vecteurs déplacement calculés ne reflètent pas suffisamment bien les mouvements réels des objets et ne sont pas assez précis pour être rattachés efficacement à l'un des objets détectés avec des critères basés mouvement.

Pour obtenir une segmentation cohérente dans de tels cas de figure, des critères spatiaux et temporels supplémentaires vont être calculés, et intégrés afin d'affiner la carte de segmentation initiale basée sur le mouvement. Imaginons par exemple, une séquence dans laquelle une zone de découvrement ne permettrait pas d'estimer correctement les mouvements, l'ajout de critères purement spatiaux permettra d'appareiller chaque élément de cette zone découverte, dont le mouvement n'est pas fiable, à l'objet spatio-temporel qui lui correspond.

Les critères supplémentaires choisis sont :

- la connexité spatio-temporelle intra-segment
- la couleur
- la texture
- le voisinage temporel

Le calcul et l'intégration de ces critères à l'outil de pré-analyse permettront non seulement de corriger la carte de segmentation initiale, mais aussi d'assurer le suivi des objets d'un segment temporel à l'autre.

En effet, le calcul de ces critères spatiaux-temporels va fournir une description très précise des objets détectés. Le niveau de détail atteint quant à la caractérisation des objets permettra donc d'assurer le suivi des objets entre plusieurs segments temporels successifs.

Les spécifications présentées ci-dessus nous ont menés à décomposer le système de raffinement de la segmentation en un ensemble de fonctions, agencées les unes avec les autres selon le schéma bloc présenté en figureA.1. Les approches statistiques étant couramment utilisées pour aborder la construction des masques des objets, et comme le type de connaissances *a priori* que l'on veut inclure s'exprime principalement en termes de contextes spatial et temporel, les critères spatiaux-temporels seront intégrés au système initial avec une approche markovienne.

[1] Un segment temporel est constituée d'une suite de 9 images, soit un intervalle temporel de sensiblement 180ms (temps de fixation de l'œil humain)

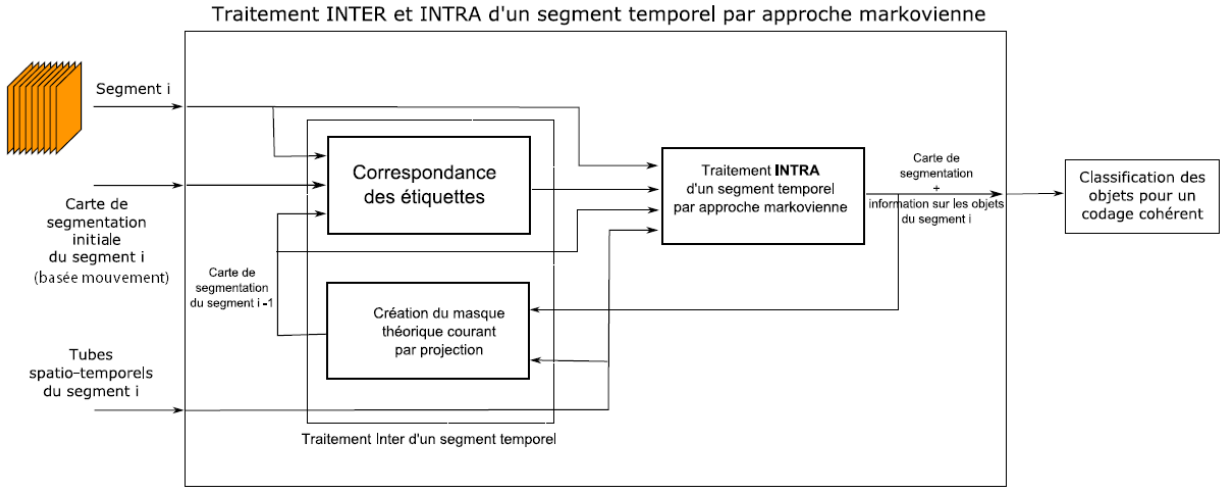

FIG. A.1 – Bloc de traitement d'un segment temporel par approche markovienne

A.2 Modélisation par champ markovien

Un champ de Markov est caractérisé par sa propriété locale, tandis qu'un champ de Gibbs est caractérisé par sa propriété globale (distribution de Gibbs). Besag, [1], a reformulé la relation entre champs markoviens et distributions de Gibbs initialement démontrée par Hammersley et Clifford en 1971. La possibilité d'exprimer par une distribution explicite les propriétés markoviennes d'un champ a permis l'essor du développement de modèles markoviens.

Nous allons dans un premier temps reprendre les principaux aspects mathématiques de ce type de modélisation. Les notations suivantes sont adoptées pour la résolution de notre problème :

- $E = \{e_s,\ s \in S\}$ est le champ d'étiquettes sur l'ensemble S des sites s. Dans notre cas, un site est un tube spatio-temporel, et les sites d'une région segmentée (correspondant à un objet en mouvement à travers les segments temporels) ont le même label ;
- $O = \{O_s,\ s \in S\}$ est le champ d'observations. Les réalisations des champs O seront notées $o = \{o_s,\ s \in S\}$;
- Λ (respectivement Ω) est l'ensemble des réalisations possibles de E (respectivement toutes les configurations d'étiquettes possibles de e) ;
- $\eta = \{\eta_s,\ s \in S\}$ est une structure de voisinage définie sur s.

$(E,\ O)$ est modélisé par un champ de Markov aléatoire. Dans ce cas, le champ d'étiquettes optimal $\hat{e}$ est obtenu selon un critère MAP (*Maximum A Posteriori*). Le théorème de Hammersley et Clifford établit l'équivalence entre les champs markoviens et les distributions de Gibbs [1], la configuration optimale du champ des étiquettes est alors obtenue en minimisant une fonction d'énergie globale $U(o, e)$:

$$\hat{e} = arg \min_{e \in \Omega} U(o, e) \tag{A.1}$$

Les propriétés markoviennes du champ d'étiquettes permettent d'écrire cette fonction d'énergie comme étant la somme de fonctions de potentiel élémentaires. Ces fonctions de potentiel sont définies localement sur des structures appelées cliques [3] :

$$U(o, e) = \sum_{c \in C} V_c(o, e) \tag{A.2}$$

où C est l'ensemble des cliques c de S relatives au voisinage η. Une clique est un sous-ensemble de sites de S tel que, si s_i et s_j sont deux sites quelconques de cette clique, s_i et s_j sont voisins au sens de η. Des exemples de systèmes de voisinages et de cliques associées sont présentés en figure A.2.

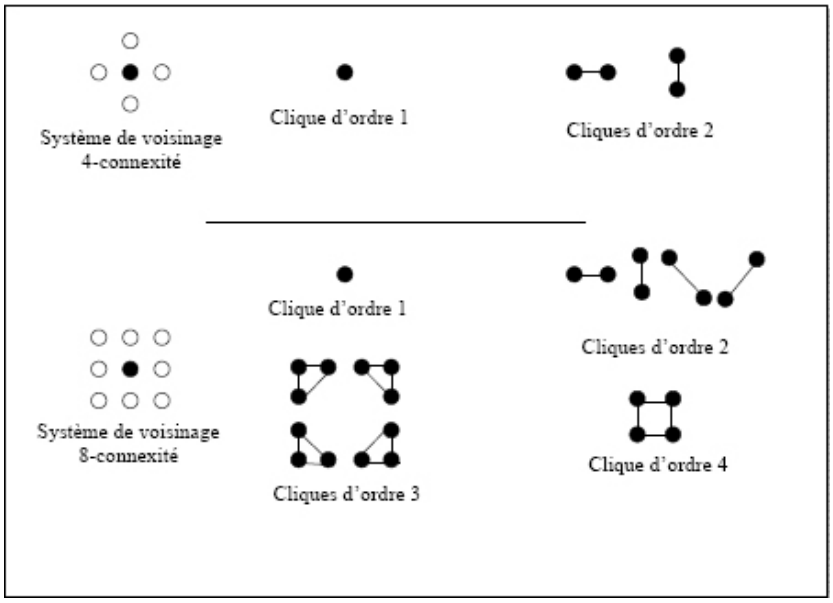

FIG. A.2 – Cliques associées à des systèmes de voisinage en 4-connexité et 8-connexité

La fonction de potentiel V_c est définie localement sur la clique c et donne les interactions locales entre les différents sites qui composent la clique. L'expression analytique de la fonction V_c est dépendante du problème posé et des résultats souhaités, elle définit les propriétés locales et globales du problème.

A.3 Fonctions de potentiel

Les fonctions de potentiel vont permettre de définir, en fonction de chaque nouveau critère (couleur, texture, ...), la probabilité pour un site donné s d'être étiqueté avec l'étiquette e. D'après l'équation A.1, le champ d'étiquettes le plus probable sera celui qui minimisera l'énergie globale $U(o, e)$. Chaque critère va contribuer à la valeur de cette énergie, qui s'exprime donc sous la forme :

$$U(o, e) = \alpha_1.W_1 + \alpha_2.W_2 + \alpha_3.W_3 + \alpha_4.W_4 + \alpha_5.W_5$$

où α_1, α_2, α_3, α_4 et α_5 sont les poids des énergies élémentaires W_1, W_2, W_3, W_4 et W_5 qui représentent respectivement les critères de voisinage spatial, de couleur, de texture, de mouvement et de voisinage temporel. Les énergies élémentaires $\{W_i\}_{i=1..5}$ sont calculées comme la somme de fonctions de potentiel élémentaires (cf. équation A.2). Afin de pouvoir comparer ces énergies et d'obtenir des ordres de grandeur homogènes, nous allons normaliser toutes les fonctions de potentiel sur l'intervalle centré $[-1; 1]$. Ainsi, seuls les poids $\{\alpha_i\}_{i=1..5}$ attachés à ces énergies, permettront de pondérer l'importance de chaque critère dans le calcul de l'énergie globale $U(o, e)$.

A.3.1 Connexité spatio-temporelles

Pour un segment temporel donné de neuf images, une région segmentée doit respecter une cohérence spatiale, c'est-à-dire que la région segmentée (constituée d'une fusion de tubes spatio-temporels) doit être localement homogène et compacte. L'énergie à minimiser $U(o, e)$ sera donc composée d'une énergie élémentaire chargée d'assurer l'homogénéité des labels pour des sites voisins. Dans notre cas, le système de voisinage choisi est un voisinage 8-connexe, dont les cliques retenues sont les cliques d'ordre 2. Ce système de voisinage est représenté en figure A.3.

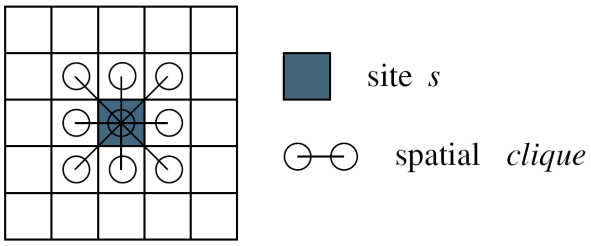

FIG. A.3 – Ensemble des cliques spatiales d'ordre 2 associées à un voisinage 8-connexe

Le modèle choisi pour favoriser la création de régions homogènes est tel que sa fonction de potentiel s'écrit :

$$\forall t \in \eta_s \{ \begin{array}{l} V_{c_s} = \beta_s \quad \text{si } e_t \neq e_s \\ V_{c_s} = -\beta_s \text{ si } e_t = e_s \end{array}$$

avec $\beta_s > 0$. Dans notre cas, chaque clique correspond à une paire de tubes spatio-temporels voisins et connectés au sens d'un voisinage 8-connexe. Afin de normaliser cette fonction de potentiel sur l'intervalle $[-1; 1]$, le paramètre β_s sera fixé à $1/8$ pour un voisinage 8-connexe. L'énergie élémentaire $W_1(e_s)$ liée au voisinage spatial, s'exprime donc sous la forme :

$$W_1(e_s) = \sum_{c_s \in C_s} V_{c_s}(e_s, e_t)$$

où C_s représente l'ensemble de toutes les cliques spatiales de S.

A.3.2 Caractéristiques de couleur

Afin de savoir si un site est étiqueté de manière cohérente dans un segment temporel, nous souhaitons pouvoir comparer les distributions de couleur de ce site avec celles des différentes régions existantes. Plusieurs méthodes sont adaptées au cas discret (intersection, L_2, χ_2, ...), nous avons opté pour l'utilisation du coefficient de Bhattacharyya qui permet de mesurer la similarité entre deux distributions :

- soit $\hat{s} = \{\hat{s_u}\}_{u=1..m}$ la densité de probabilité discrète de couleur du site courant s ;
- soit $\widehat{R(e_s)} = \{\widehat{R(e_s)}_u\}_{u=1..m}$ la densité de probabilité discrète de couleur de la région $R(e_s)$ constituée des sites étiquetés e_s.

Le coefficient de Bhattacharyya qui permet de comparer ces densités est défini par :

$$\rho_{couleur} = \rho_{couleur}(\widehat{R(e_s)}, \hat{s}) = \sum_{u=1}^{m} \sqrt{\widehat{R(e_s)}_u \times \hat{s_u}}$$

Les densités de probabilité discrètes de couleur sont calculées à partir des histogrammes de couleur correspondants. Pour diminuer la complexité des calculs et regrouper les couleurs proches, chaque composante couleur est uniformément quantifiée sur 16 niveaux (donc $m = 16^3 = 4096$ couleurs possibles). Les histogrammes de couleur sont ensuite calculés en considérant uniquement l'image centrale du segment temporel courant : l'histogramme couleur d'un site est donc calculé à partir du macrobloc central du tube correspondant et non des neuf macroblocs qui constituent ce tube. Ces histogrammes sont alors normalisés par le nombre d'éléments qui ont y contribué, afin d'obtenir les densités de probabilité discrètes de couleur.

Le coefficient de Bhattacharyya varie de 0 (distributions totalement différentes) à 1 (distributions identiques). Afin de normaliser la fonction de potentiel associée à la couleur sur l'intervalle $[-1; 1]$, nous utilisons la transformation linéaire : $V_{couleur} = 1 - 2 \times \rho_{couleur}(\widehat{R(e_s)}, \hat{s})$. Notons que la fonction

utilisée inverse le signe initial du coefficient de Bhattacharyya, afin que deux distributions proches (coefficient de Bhattacharyya fort) aient une énergie faible. L'énergie élémentaire W_2 pour le critère de couleur est donc définie par :

$$W_2(e_s, o_s, o(R(e_s))) = \sum_{s \in S} V_{couleur}$$

A.3.3 Caractéristiques de texture

Il s'agit ici de comparer la similarité entre les textures d'un site et celles des différentes régions existantes. Comme dans le cas de la fonction de potentiel associée à la couleur, l'information de texture va être représentée sous la forme d'une distribution. Le même système de notation est conservé :

- $\hat{s} = \{\hat{s_v}\}_{v=1..n}$ est la densité de probabilité discrète de texture du site courant s ;
- $\widehat{R(e_s)} = \{\widehat{R(e_s)}_v\}_{v=1..n}$ est la densité de probabilité discrète de texture de la région $R(e_s)$ constituée des sites étiquetés e_s.

Par soucis de simplification, les distributions pour la texture seront calculées en ne considérant que l'image centrale du segment temporel courant.

Chaque pixel de l'image centrale va donner une information de texture représentée sous la forme d'un couple de gradients (ΔH,ΔV) (respectivement le gradient spatial horizontal et le gradient spatial vertical). Afin de réduire l'importance du bruit d'acquisition dans le calcul des textures, le gradient ΔH (respectivement ΔV) de chaque pixel est obtenu en filtrant l'image centrale du segment temporel avec un filtre de Sobel horizontal (respectivement vertical)[2]. Les gradients correspondent alors aux valeurs filtrées (en valeurs absolues) et quantifiées selon la loi suivante :

$$\begin{cases} \Delta H = \lfloor |x|/4 \rfloor & \text{si } x < 64 \\ \Delta H = 15 & \text{sinon} \end{cases}$$

où x représente la valeur filtrée d'un pixel de l'image avec un noyau de Sobel horizontal (la même loi est utilisée dans le cas vertical). D'après cette équation, chacun des deux gradients qui composent la texture est quantifié sur 16 niveaux, il y a donc $n = 16^2 = 256$ "valeurs" possibles de texture. Une fois les distributions de texture calculées pour un site et pour une région, nous les comparons en utilisant de nouveau le coefficient de Bhattacharyya :

$$\rho_{texture} = \rho_{texture}(\widehat{R(e_s)}, \hat{s}) = \sum_{v=1}^{n} \sqrt{\widehat{R(e_s)}_v \times \hat{s_v}}$$

Comme dans le cas de la couleur, une fonction de potentiel à valeurs dans l'intervalle $[-1; 1]$ est déduite de ce coefficient : $V_{texture} = 1 - 2 \times \rho_{texture}(\widehat{R(e_s)}, \hat{s})$. L'énergie élémentaire W_3 pour le critère de texture est donc définie par :

$$W_3(e_s, o_s, o(R(e_s))) = \sum_{s \in S} V_{texture}$$

[2] noyau de Sobel du filtre horizontal $\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$, du filtre vertical $\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$

A.3.4 Caractéristiques de mouvement

Dans un segment temporel, nous avons vu que le critère principal pour la segmentation est le mouvement : pour une région donnée, les vecteurs mouvement associés à des tubes spatio-temporels doivent avoir des valeurs proches. C'est le critère qui permet de créer un premier champ d'étiquettes avant son raffinement par modèles markoviens. Ce critère doit donc conserver une importance dans le calcul de l'énergie globale $U(o, e)$. Une énergie élémentaire liée au mouvement est alors définie, de manière à mesurer la ressemblance entre le mouvement d'un site et celui d'une région. La fonction de potentiel qui mesure cette ressemblance est définie par :

$$V_{mouvement} = -\frac{\overrightarrow{MV_s} \times \overrightarrow{MV_{R(e_s)}}}{max(\left\|\overrightarrow{MV_s}\right\|, \left\|\overrightarrow{MV_{R(e_s)}}\right\|)^2}$$

Où $\overrightarrow{MV_s}$ et $\overrightarrow{MV_{R(e_s)}}$ sont respectivement les vecteurs de mouvement associés au site s, et à la région $R(e_s)$ formée des sites étiquetés e_s.

Le produit scalaire normalisé, présenté dans l'équation ci-dessus, fournit une valeur de ressemblance pour les vecteurs qui varie entre - 1 et 1. L'inversion du signe de ce produit scalaire permet d'attribuer un potentiel faible lorsque les mouvements sont proches et plus important lorsqu'ils sont différents. L'énergie élémentaire W_4 pour le critère de mouvement est donc définie par :

$$W_4(e_s, o_s, o(R(e_s)) = \sum_{s \in S} V_{mouvement}$$

A.3.5 Caractéristiques temporelles

La durée de vie d'un objet spatio-temporel est généralement plus grande que la durée d'un segment temporel (180ms) : le cycle de vie d'un objet s'étend typiquement sur plusieurs segments successifs. Une région segmentée dans un segment temporel t doit donc respecter une cohérence temporelle avec la région correspondante dans le segment $t-1$ (si elle existe), c'est-à-dire que la forme d'une région segmentée doit rester temporellement homogène et compacte. Afin d'assurer cette propriété entre deux segments successifs, nous utilisons la projection temporelle du segment $t-1$ à l'instant t. Cette projection tient compte du mouvement global de caméra et des mouvements locaux des objets segmentés. Une clique temporelle peut alors être définie entre le segment courant et le segment précédent projeté pour maintenir l'homogénéité de la forme de la région segmentée. Ce système est présenté en figure A.4. Sur cette représentation, une région, représentée en couleur, est suivie d'un segment à un autre et change sensiblement de forme au cours du temps.

La fonction de potentiel associée au critère temporel est définie par :

$$\begin{cases} V_{c_t} = \beta_t & \text{si } e_s(t) \neq e_s(t-1) \\ V_{c_t} = -\beta_t & \text{si } e_s(t) = e_s(t-1) \end{cases}$$

avec $\beta_t = 1$, et où $e_s(t)$ et $e_s(t-1)$ sont respectivement les étiquettes du site du segment courant et du site du segment précédent projeté. L'énergie élémentaire W_5 pour le critère temporel est alors définie par :

$$W_5(e_s(t)) = \sum_{c_t \in C_t} V_{c_t}(e_s(t), e_s(t-1))$$

où C_t est l'ensemble de toutes les cliques temporelles de S. Notons que pour utiliser cette fonction de potentiel, il faut qu'un même objet garde la même étiquette d'un segment temporel à l'autre, il faut donc assurer le suivi temporel des objets à travers les segments successifs. C'est l'objet de la prochaine section.

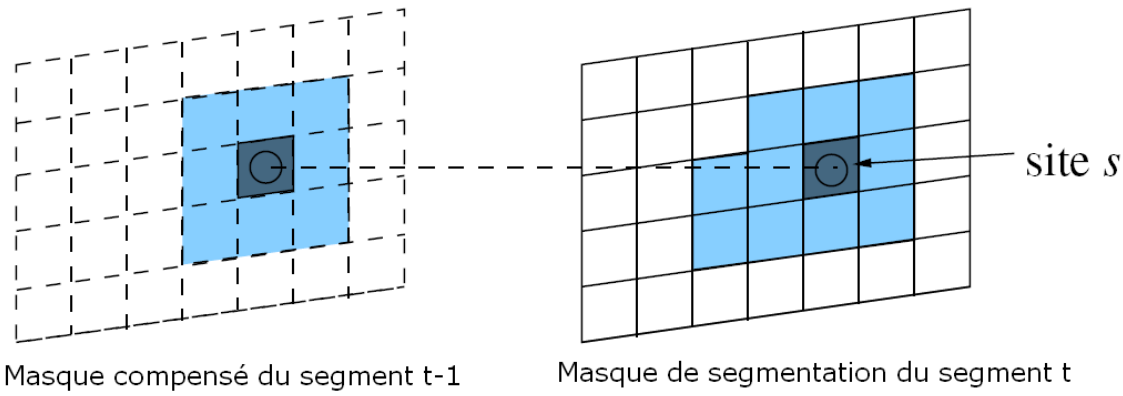

FIG. A.4 – Clique temporelle entre deux segments successifs

A.4 Suivi d'objets

Comme nous l'avons mentionné dans la section précédente, un même objet peut avoir un cycle de vie qui s'étend sur plusieurs segments temporels successifs. Dans une perspective de codage cohérente d'un même objet, il est donc intéressant de réussir à suivre un objet sur plusieurs segments successifs. Pour réaliser ce suivi entre un objet du segment t-1 et un objet du segment t, nous compensons le mouvement de la carte de segmentation du segment t-1 à l'instant t, puis nous appareillons cette carte compensée avec la carte de segmentation du segment courant à l'instant t.

A.4.1 Compensation en mouvement de la carte de segmentation du segment t-1

Considérons une carte de segmentation disponible pour le segment temporel t-1, pour pouvoir la comparer spatialement avec le segment courant t, il faut la projeter temporellement à l'instant t. Une nouvelle carte de segmentation projetée est alors disponible. Cette projection est décomposée en plusieurs étapes :

- détection de l'objet "fond",
- initialisation de la carte de segmentation projetée par l'étiquette du fond,
- projection des objets avec leurs vecteurs de déplacement.

Dans le cas de recouvrement d'objets, on considère que l'objet le plus petit est mis au premier plan. Ainsi, si ce n'est pas le cas dans la réalité, il ne sera simplement appareillé avec aucun objet du segment courant t. Un exemple de projection est présenté en figure A.5.

A.4.2 Étiquetage et suivi

Pour appareiller les objets d'un segment au segment suivant, une métrique basée sur la similarité des couleurs, des textures, et sur le taux de recouvrement est utilisée. La similarité des couleurs et des textures est mesurée de nouveau à l'aide du coefficient de Bhattacharyya. Chaque objet du segment courant est comparé aux objets présents dans la carte compensée du segment précédent. L'objet du segment courant prend alors l'étiquette de l'objet le plus proche, en accord avec la métrique utilisée, à condition que leur similarité soit "assez forte". En pratique, on fixe des seuils expérimentaux pour le coefficient de Bhattacharyya sur la couleur, le coefficient de Bhattacharyya sur la texture, et le taux de recouvrement. Si les deux objets les plus proches présentent, pour chacun de ces trois seuils, une similarité assez forte, alors on considère qu'il s'agit du même objet sur les deux segments successifs.

D'autre part, il peut arriver que la méthode de segmentation au sens du mouvement ne distingue aucun objet (tous les objets ont des mouvements proches ou trop difficiles à estimer). Dans ce cas, la

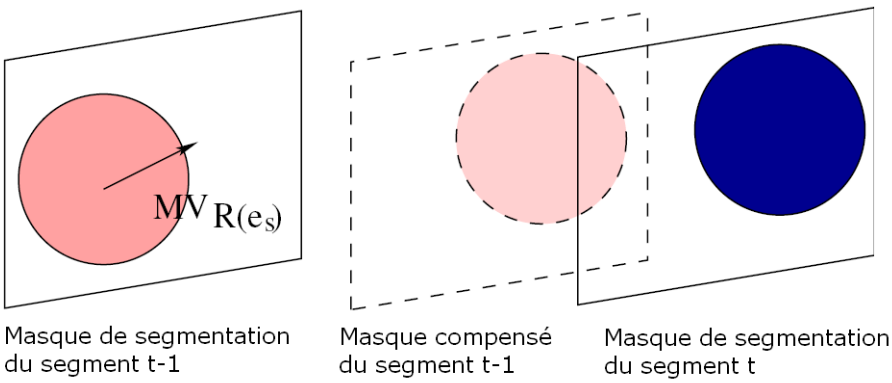

FIG. A.5 – Suivi d'objets entre des segments temporels successifs

carte de segmentation initiale du segment courant pour le traitement avec approche markovienne est vide (en fait elle ne comprend qu'un seul objet : le fond), et donc notre approche markovienne n'aurait aucun effet. Nous choisissons alors, pour ces configurations particulières, d'initialiser la méthode de segmentation markovienne avec la carte de segmentation projetée du segment précédent. Cette technique permet de rester efficace dans le suivi des objets à travers les segments temporels successifs.

A.5 Minimisation de l'énergie

Nous avons montré que pour obtenir le champ d'étiquettes optimal, il faut minimiser la fonction d'énergie $U(o, e)$ donnée par :

$$U(o, e) = \alpha_1.W_1 + \alpha_2.W_2 + \alpha_3.W_3 + \alpha_4.W_4 + \alpha_5.W_5$$

La carte de segmentation issue de la segmentation basée mouvement va servir d'initialisation pour la segmentation par approche markovienne. Nous calculons pour chaque site un degré de stabilité $\Delta U(s)$, qui correspond à la variation entre l'énergie associée au site pour l'étiquette courante e_c et l'énergie minimale qu'aurait ce site avec une étiquette optimale e_s :

$$\Delta U(s) = U(s, e_c) - U(s, e_s)$$

si $\Delta U(s)$ est non nul alors le site est instable. Nous mettons en œuvre une pile d'instabilité : le site le plus instable est traité en premier et ainsi de suite de façon itérative, jusqu'à ce que tous les sites soient stables. Chaque site instable est traité de la façon suivante :

- si le site a déjà été traité plusieurs fois, il empêche la solution de converger, il est donc retiré des sites à traiter ;
- le site courant prend l'étiquette qui minimise son énergie ;
- l'énergie des sites voisins est modifiée en fonction de la nouvelle étiquette du site courant.

Typiquement, les sites les plus instables sont ceux situés sur les bords de la carte de segmentation et ceux situés sur les bords des objets segmentés. Le traitement est terminé lorsque tous les sites sont stables, ou que les seuls sites non-stables restants sont ceux qui empêchent la solution de converger.

Notons, que la méthode utilisée ici vise à minimiser l'énergie globale $U(o, e)$ du champ des étiquettes, en minimisant successivement et localement les énergies de chaque site. Cette méthode simple est une méthode de relaxation déterministe. Elle assure la convergence vers le premier minimum d'énergie trouvé qui n'est pas forcément le minimum global. Á l'inverse, les méthodes de relaxation stochastiques autorisent des configurations qui augmentent provisoirement l'énergie du système, afin de converger vers un minimum global[2]. Cependant ces méthodes sont complexes et peu adaptées à notre contexte d'utilisation.

Séquence	Segmentation mouvement seul	Segmentation complète
Tractor (690 images)	33%	84%
New Mobile and Calendar (500 images)	85%	92%
Shields (500 images)	94%	100%

TAB. A.1 – Ratio des objets en mouvement détectés

A.6 Facteur d'importance des critères ajoutés

Un objet vidéo est une forme spatio-temporelle caractérisée par sa texture, sa couleur et son mouvement qui, souvent, diffère du mouvement global de la scène. Nous avons choisi de poser l'hypothèse selon laquelle le mouvement est le critère le plus déterminant pour segmenter les objets d'un segment temporel. Cependant, les informations de mouvement sont obtenues à partir d'une méthode d'estimation dont la précision dépend fortement des contenus vidéos. Ainsi, pour certaines séquences, le mouvement sera un critère fiable, alors que pour d'autres séquences la segmentation devra s'appuyer plus fortement sur les critères de couleur, de texture ou de voisinage.

Cette constatation nous a amenés à créer deux jeux de paramètres $\{\alpha_i\}_{i=1..5}$ pour calculer l'énergie globale $U(o,e)$ selon que l'estimation des mouvements soit considérée fiable ou non. Dans le cas ou le mouvement sera fiable, le paramètre α_4 qui représente l'importance de l'énergie liée au mouvement dans l'énergie globale $U(o,e)$ sera augmenté, tandis que les poids liés aux autres énergies seront plus faibles. Inversement, si l'estimation de mouvement est jugée trop peu précise, ce poids sera diminué et les autres poids augmentés.

Pour caractériser la précision de l'estimation de mouvement, nous calculons pour chaque objet spatio-temporel une pseudo-variance des vecteurs de mouvement associés aux tubes qui constituent cet objet. Soit $\overrightarrow{MV_{objet}}$ le vecteur mouvement représentant le déplacement d'un objet, et $\{\overrightarrow{MV_i}\}_{i=1..N}$ l'ensemble des vecteurs déplacement rattachés à cet objet, la pseudo-variance $\overline{\sigma}^2$ sera alors définie par :

$$\overline{\sigma}^2 = \frac{1}{N}\sum_{i=1}^{N}(\overrightarrow{MV_{objet}} - \overrightarrow{MV_i})^2$$

où N représente le nombre de tubes qui constituent l'objet. Si cette pseudo-variance est inférieure à un certain seuil, nous considérons que la segmentation au sens du mouvement est assez précise : les mouvements associés aux tubes qui composent un objet sont proches, les poids liés aux autres critères (couleur, texture, ...) seront donc moins importants :

$$\begin{cases} \alpha_1 = \alpha_2 = \alpha_3 = 0.4,\ \alpha_4 = 1,\ \alpha_5 = 0.6 & \text{si } \overline{\sigma}^2 < \textit{seuil} \\ \alpha_1 = \alpha_2 = 1,\ \alpha_3 = 0.4,\ \alpha_4 = \alpha_5 = 0.6 & \text{si } \overline{\sigma}^2 \geqslant \textit{seuil} \end{cases}$$

A.7 Résultats

Les résultats obtenus après une segmentation basée mouvement seule et ceux obtenus en couplant cette segmentation à l'approche markovienne présentée ici montrent que les objets spatio-temporels sont plus fidèlement détectés avec l'approche markovienne. Le tableau A.1 présente, pour chacune des deux méthodes, le ratio entre le nombre d'objets en mouvement détectés et le nombre réel d'objets en mouvement au sein de séquences HD. Bien que le taux de détection soit augmenté, nous remarquons que l'approche markovienne n'assure la détection de tous les objets. Par exemple, à la fin de la séquence *Tractor*, le tracteur est trop petit pour être détecté à cause du zoom sortant de la caméra.

Les figures A.6 et A.7 présentent, pour quatre segments temporels successifs des séquences *Tractor* et *New Mobile and Calendar*, les cartes de segmentation obtenues avec une segmentation basée mouvement uniquement (ligne du milieu) et une approche markovienne (ligne du bas). Les objets en mouvement sont correctement détectés avec la segmentation basée mouvement, mais le suivi des objets en mouvement entre les segments n'est pas assuré (un même objet peut avoir des étiquettes

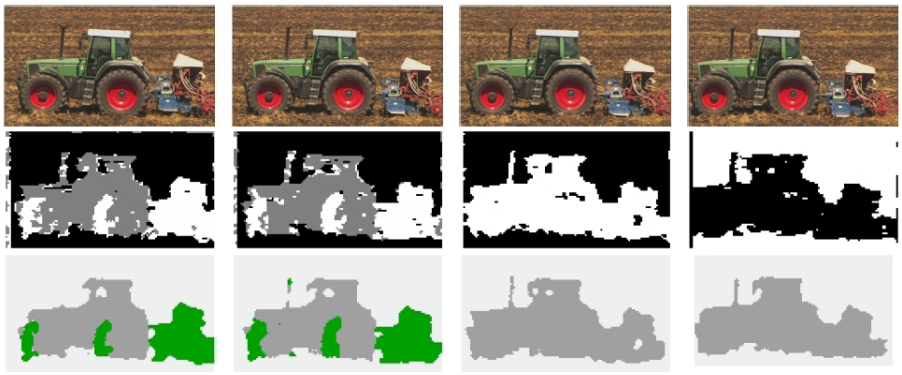

FIG. A.6 – Cartes de segmentation pour *Tractor* (segments 13 à 16) : segmentation mouvement (ligne du milieu) et approche markovienne (ligne du bas)

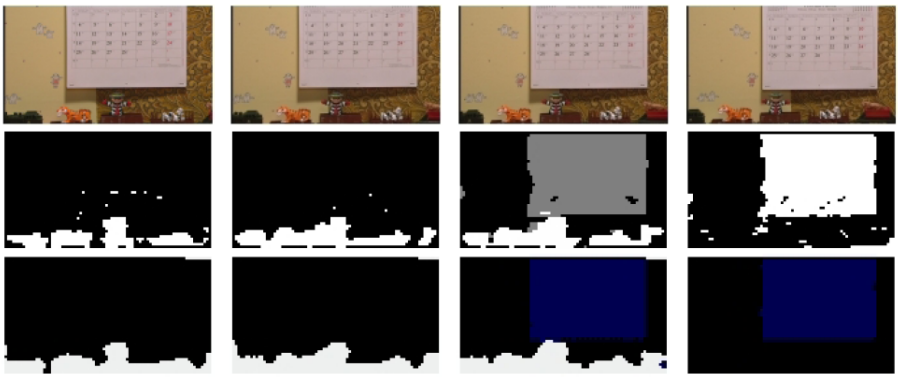

FIG. A.7 – Cartes de segmentation pour *New Mobile and Calendar* (segments 50 à 53) : segmentation mouvement (ligne du milieu) et approche markovienne (ligne du bas)

différentes d'un segment à l'autre). Avec l'approche markovienne, les bords des objets en mouvement sont plus réguliers et les contenus plus homogènes, de plus le suivi entre segments est assuré : par exemple, l'étiquette du tracteur reste la même sur les quatre segments temporels. L'ajout de critères spatiaux-temporels par approche markovienne a donc permis, d'une part, d'améliorer la qualité de la segmentation basée mouvement initiale et, d'autre part, d'assurer le suivi des objets sur plusieurs segments temporels successifs.

A.8 Conclusion

Ce chapitre a présenté notre méthode de segmentation spatio-temporelle basé sur les champs aléatoires de Markov. Celle-ci combine des informations de mouvement (issues de notre estimation de mouvement basée sur des tubes spatio-temporels), de couleur, de texture et de connexité spatiale et temporelle. Les résultats obtenus montrent que l'ajout de critères spatiaux-temporels par approche markovienne permet, d'une part, d'améliorer la qualité de la segmentation basée mouvement initiale et, d'autre part, d'assurer le suivi des objets sur plusieurs segments temporels successifs.

Annexe B

Cartes de saillance visuelle

B.1 Introduction

Pour faire face à l'énorme quantité d'informations visuelles de notre environnement visuel, le système visuel possède la faculté de sélectionner une information pertinente localisée spatialement dans le champ visuel parmi toutes celles qui lui parviennent : on parle d'attention visuelle. Du fait de la grande complexité des mécanismes et des inter actions, des inter dépendances existants entre les mécanismes du système visuel humain, modéliser l'attention visuelle dans son ensemble reste trop complexe. Une voie réaliste est de modéliser l'attention visuelle pré-attentive. Le modèle proposé doit être capable de déterminer les zones visuellement importantes d'une image et dans notre cas d'une séquence vidéo. Ce chapitre présente notre modèle d'attention visuelle pré-attentive permettant d'obtenir des cartes de saillance spatio-temporelle.

B.2 Calcul des cartes de saillance visuelle

De nombreux facteurs influençant l'attention visuelle ont été identifiés[5] et sont regroupées en deux catégories. La première regroupe toutes les informations spatiales dont les plus susceptibles de stimuler notre attention visuelle regroupent la couleur, l'orientation et la taille. La deuxième concerne les informations temporelles. Or, une séquence vidéo contient ces deux types d'informations susceptibles de stimuler notre attention visuelle. C'est pourquoi notre modèle d'attention visuelle pré-attentive doit les prendre en compte. Celui-ci se décompose en deux parties, l'une modélise l'attention visuelle à partir des informations spatiales et l'autre à partir des données temporelles. La dernière étape combine ces deux parties afin d'obtenir une carte de saillance spatio-temporelle.

B.2.1 Saillance spatiale basée sur le contraste de couleur

Des informations importantes peuvent être trouvées dans la littérature sur la théorie des couleurs et plus particulièrement sur les attributs des couleurs qui contribuent à rendre un objet visuellement saillant ou non. En terme de saillance de couleur, d'autres méthodes d'attention visuelle artificielle se sont concentrés uniquement sur les attributs de couleurs qui ont été signalés en psychologie et de nombreux aspects importants décrits à cet effet dans la théorie des couleurs ont été négligés. Les artistes utilisent ses aspects pour créer des effets de contraste, des mises en avant visuelles et de mobilité dans leurs illustrations. Dans leurs travaux Aziz et Mertsching[4] combinent ces concepts et formulent un ensemble de points possible à mettre en oeuvre. Il reste ensuite à décider quelles couleurs vont bénéficier de la saillance en présence d'un contraste. Les différents points avec la mention de la couleur saillante gagnante dans chaque situation sont énumérés ci-dessous :

1. *Contraste de Saturation* : Un contraste est produit par des couleurs faiblement et fortement saturées. La valeur du contraste est directement proportionnelle à la magnitude de la différence de

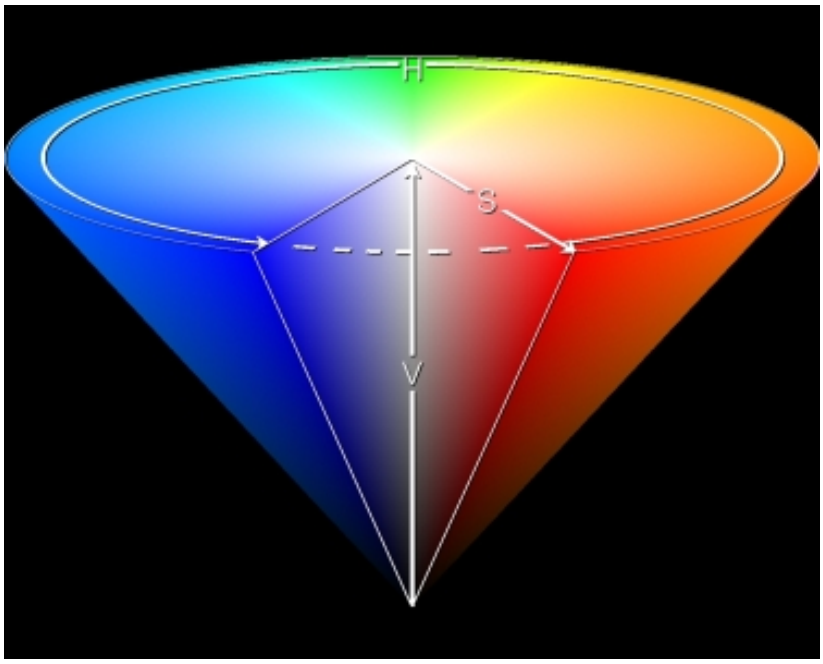

FIG. B.1 – Représentation conique de l'espace TSV (*HSV*).

saturation. Des couleurs fortement saturées tendent à attirer l'attention dans de telles situations à moins qu'une région faiblement saturée soit entourée par une région fortement saturée.

2. *Contraste d'Intensité* : Un contraste sera visible lorsque des couleurs sombres et lumineuses coexistent. Plus la différence d'intensité est importante, plus l'effet de contraste augmente. Les couleurs lumineuses attirent l'attention dans cette situation à moins que la région sombre soit entourée par une région lumineuse.
3. *Contraste de Teinte* : La différence des angles de teinte sur le disque des couleurs (cf figure B.1) contribue à la création d'un contraste. Une différence importante va manifestement produire un contraste réel. Du fait de la nature circulaire de la teinte, la plus grande différence entre deux valeurs de teinte est de 180°.
4. *Contraste d'Opposants* : Les couleurs situées sur les côtés opposés du disque de teinte produisent une importante valeur de contraste. Cela signifie naturellement que la différence des angles des valeurs de teinte doit être proche de 180°. Les couleurs situées dans la première moitié du disque de teinte, connues comme la gamme de couleur active, domineront sur le reste des couleurs passives.
5. *Contraste des couleurs Chaude et Froide* : Les couleurs chaudes, c'est-à-dire rouge, jaune et orange sont visuellement plus saillantes. Ces couleurs sont situées dans les premiers 45° du disque de teinte. Les couleurs chaudes et froides créent un contraste dans lequel les couleurs chaudes restent dominantes.
6. *Dominance des Couleurs Chaudes* : Les couleurs chaudes dominent leur environnement, même si un contraste existe dans cet environnement.
7. *Dominance de la Luminosité et de la Saturation* : Les couleurs fortement lumineuses et saturées sont considérées comme étant attractives sans tenir compte de leurs valeurs de teinte. De telles couleurs ont plus de chances d'attirer l'attention.

L'effet de contraste est contrôlée par la valeur de saturation des deux couleurs impliquées dans les situations mentionnées aux points 2 à 5. Les couleurs fortement saturées impliquent des contrastes

importants. Notre modèle de saillance des couleurs, basé sur les travaux de Aziz et Mertsching[4], combinent tous les points mentionnés ci-dessus. Nous divisons cette procédure en sept étapes dont chacune contribue à la valeur de saillance d'un site s. Les valeurs des différentes composantes de couleur utilisées dans nos calculs pour un site s sont les valeurs moyennes du bloc situé au centre du tube considéré.

B.2.1.1 Transformation de l'espace de couleur

Avant de calculer la saillance spatiale, nous réalisons une transformation des couleurs. En effet, les séquences vidéos originales sont au format YUV. Le modèle YUV définit un espace colorimétrique en trois composantes. Le premier représente la luminance et les deux autres représentent la chrominance. Nous utilisons deux transformations pour obtenir des données dans l'espace de couleur HSV. La première est une transformation de l'espace YUV vers l'espace de couleur RGB et s'écrit :

$$RGB = \begin{pmatrix} 1 & 0 & 1.5701 \\ 1 & -0.187 & -0.4664 \\ 1 & 1.8556 & 0 \end{pmatrix} .YUV, \tag{B.1}$$

$$\begin{aligned} R &= Y + 1.5701 \times (V - 128), \\ G &= Y - 0.187 \times (U - 128) - 0.4664 \times (V - 128), \\ B &= Y + 1.8556 \times (U - 128). \end{aligned}$$

La deuxième transformation permet d'obtenir des données dans l'espace de couleur HSV à partir de données issues de l'espace RGB. Les équations de cette transformation sont les suivantes :

$$\begin{aligned} H &= \begin{cases} \frac{60}{360} \times \frac{G-B}{max_{RGB}-min_{RGB}} & si \quad max_{RGB} = R \\ \frac{60}{360} \times \frac{B-R}{max_{RGB}-min_{RGB}} & si \quad max_{RGB} = G \\ \frac{60}{360} \times \frac{R-G}{max_{RGB}-min_{RGB}} & si \quad max_{RGB} = B \end{cases} \\ S &= \frac{max_{RGB}-min_{RGB}}{max_{RGB}} \\ V &= max_{RGB} \end{aligned} \tag{B.2}$$

B.2.1.2 Calcul de la saillance spatiale

Les cinq premières étapes de l'algorithme utilisent un ou les deux facteurs de saturation f_{ij}^{sat}et d'intensité f_{ij}^{int} dans leurs calculs. Les indices i et j représentent respectivement la position du site courant et d'un site voisin (voisinage 8-connexe). La première partie du facteur de saturation f_{ij}^{sat} est obtenue en calculant la moyenne des valeurs de saturation entre le site s_i et le site s_j, de sorte l'effet de ce facteur soit plus important lorsque les deux blocs ont une valeur élevée de saturation et vice versa. La deuxième partie dépend seulement de la saturation du site s_i et détient une valeur minimale égale à k_{min}, afin de ne pas supprimer l'interaction des blocs avec une saturation proche de zéro. Le reste de la seconde partie est obtenue à partir de la saturation du site s_i et est pondéré par $(1 - k_{min})$. Le facteur pour l'intensité est calculé de la même façon en utilisant la valeur de l'intensité de la couleur du bloc et non la valeur de la saturation. Soient $S(s_i)$et $I(s_i)$, respectivement les valeurs de saturation et d'intensité du site s et la valeur maximale pour la saturation et l'intensité étant égale à 1, les deux facteurs de saturation et d'intensité sont définis par :

$$f_{ij}^{sat} = \frac{S(s_i) + S(s_j)}{2} \times (k_{min} + (1 - k_{min}).S(s_i)), \tag{B.3}$$

$$f_{ij}^{int} = \frac{I(s_i) + I(s_j)}{2} \times (k_{min} + (1 - k_{min}).I(s_i)), \tag{B.4}$$

où $k_{min} = 0,21$.

La contribution de la première étape en terme de saillance pour un site s_i est obtenue à partir des deux facteurs de saturation et d'intensité :

$$X_1(s_i) = \sum_{j=1}^{j=p_i} f_{ij}^{sat}.f_{ij}^{int} \ \ \forall s_j \in \eta_i, \tag{B.5}$$

où p_i est la taille du voisinage (8-connexe) et η_i représente l'ensemble des sites voisins de s_i.

La seconde étape collecte les contributions des sites qui ont une valeur de teinte éloignée de celle du site s_i. Le calcul de X_2^i est réalisé de la manière suivante :

$$X_2(s_i) = \sum_{j=1}^{j=p_i} f_{ij}^{sat}.f_{ij}^{int}.\triangle_{ij}^{teinte} \ \ \forall s_j \in \eta_i, \tag{B.6}$$

où $\triangle_{ij}^{teinte}$ représente la différence de teinte entre le site s_i et le site voisin s_j. Du fait de la nature circulaire de la teinte, nous calculons la différence de teinte entre deux sites s_i et s_j de la façon suivante :

$$\triangle_{ij}^{teinte} = \begin{cases} \triangle_{ij}^{\mu} & \text{pour } \triangle_{ij}^{\mu} \leq 0,5 \\ 1 - \triangle_{ij}^{\mu} & \text{sinon} \end{cases}$$

où $\triangle_{ij}^{\mu} = |H(s_i) - H(s_j)|$, $H(s_i)$ étant la valeur de teinte du site s, celle-ci étant comprise entre 0 et 1. Une valeur de teinte égale à 1 représente un angle de 360° et donc une valeur de $0,5$ représente un angle de 180°. Les sites voisins ayant un contraste important en terme de teinte avec le site s_i vont augmenter le poids de cette seconde contribution à la saillance finale.

Dans la troisième étape, nous étendons le principe de contraste entre couleurs chaudes et froides au contraste entre couleurs passives et actives. Une couleur est considérée comme étant active si sa valeur de teinte est comprise dans la première moitié du disque de représentation de la teinte, c'est-à-dire, une valeur inférieure à $0,5$ (180°). Ainsi, si la couleur d'un site s_i est active, alors un site s_j avec une couleur passive va contribuer à la saillance du site s_i. Une différence importante en terme de teinte va rendre ce contraste plus saillant. Cette contribution à la saillance du site s_i s'écrit sous la forme :

$$X_3(s_i) = \sum_{j=1}^{j=p_i} f_{ij}^{sat}.f_{ij}^{int}.\triangle_{ij}^{teinte} \ \ \forall s_j \in \eta_i, \ \ si \ \ H(s_i) < 0,5 \ \ et \ \ H(s_j) \geq 0,5 \tag{B.7}$$

La quatrième étape constitue la contribution liée au contraste de saturation. Les sites possédant des différences de saturation importantes dans leur voisinage contribuent à la saillance du site s_i de la façon suivante :

$$X_4(s_i) = \sum_{j=1}^{j=p_i} f_{ij}^{sat}.f_{ij}^{int}.\triangle_{ij}^{sat} \ \ \forall s_j \in \eta_i, \tag{B.8}$$

où $\triangle_{ij}^{sat}$ est la différence de saturation entre les sites s_i et s_j.

La cinquième étape regroupe les contributions pour le site s_i, à partir des blocs voisins ayant une différence importante en terme d'intensité (contraste d'intensité). Le formule utilisée est similaire à celui de la quatrième étape et s'écrit :

$$X_5(s_i) = \sum_{j=1}^{j=p_i} f_{ij}^{sat}.f_{ij}^{int}.\triangle_{ij}^{int} \ \ \forall s_j \in \eta_i, \tag{B.9}$$

où $\triangle_{ij}^{int}$ est la différence d'intensité entre les sites s_i et s_j.

Pour chaque site s_i, p_i sites voisins ont contribué à la saillance dans les cinq premières étapes. Les contributions finales sont obtenues en fonction du nombre de voisins pour chaque site s_i :

$$V_\varsigma(s_i) = \frac{X_\varsigma^i}{p_i} \ \ \forall \varsigma \in \{1..5\}, \tag{B.10}$$

où p_i est le nombre de voisins disponible dans le voisinage 8-connexe.

Les couleurs chaudes constituées de l'intervalle de couleurs rouge, orange et jaune produisent une contribution supplémentaire afin de renforcer leur saillance dans la sixième étape. Cet intervalle de couleur est situé dans les premiers 45° du disque de représentation de la teinte. Cette contribution se formule de la façon suivante :

$$V_6(s_i) = \begin{cases} S(s_i).I(s)_i & \text{pour } 0 \leq H(s_i) < 0,125 \\ 0 & \text{sinon} \end{cases} \tag{B.11}$$

la valeur de la teinte variant entre 0 et 1, un angle de 45° correspond à une valeur de $0,125$.

Finalement, la septième étape est constituée de la contribution liée aux sites ayant une couleur fortement saturée et une intensité lumineuse importante. Ces composantes de couleurs du site s_i sont combinées afin de déterminer la contribution pour la dernière étape :

$$V_7(s_i) = S(s_i).I(s_i) \tag{B.12}$$

La saillance spatiale finale est obtenue en combinant les contributions des sept étapes :

$$S^{SP}(s_i) = \frac{1}{7}\sum_{\varsigma=1}^{7} V_\varsigma(s_i) \tag{B.13}$$

Cette carte est ensuite normalisée en fonction de la saillance maximale globale obtenue pour chaque image :

$$S^{SP'}(s_i) = S^{SP}(s_i)/S_{max},$$

où S_{max} est la valeur de saillance maximale obtenue pour l'un des blocs de l'image.

B.2.2 Saillance temporelle

L'aspect temporel est primordial dans la modélisation de l'attention visuelle. Dans un contexte de recherche visuelle, J. Wolfe[6] a clairement identifié le mouvement comme un attracteur visuel. Une cible en mouvement enfouie dans un ensemble de distracteurs fixe attire l'attention. En outre, une cible fixe enfouie dans un ensemble de distracteurs en mouvement attire l'attention mais dans une moindre mesure. Dans ce contexte d'études, le contraste en mouvement est l'élément déterminant qui attire notre attention visuelle. La cible en contraste de mouvement saute littéralement aux yeux.

De plus, pour la détection de zones saillantes d'une séquence d'images projetées sur un écran, il est intéressant d'avoir à l'esprit les règles en vigueur dans la façon de filmer. Les mouvements de caméra influencent clairement la stratégie visuelle de l'observateur. La présence ou non de mouvement permet de hiérarchiser les différents évènements. Par ailleurs, la prise de vue est significative du message que le metteur en scène souhaite faire passer. Elle incite inconsciemment le téléspectateur à regarder quelque chose à un endroit particulier.

En conclusion, l'objectif est de déterminer les zones présentant un contraste de mouvement. À Partir des données issues de l'estimation du mouvement global et de la segmentation spatio-temporelle, il est possible de déterminer le contraste de mouvement pour chaque objet et plus particulièrement pour chaque tube. Ce contraste de mouvement étant la base de la construction de la saillance temporelle.

B.2.2.1 Mouvement dominant

Afin de réaliser la segmentation spatio-temporelle, nous avons premièrement estimé le mouvement global à l'aide d'un modèle affine à six paramètres :

$$\begin{pmatrix} V_x \\ V_y \end{pmatrix} = \begin{pmatrix} a_1 & a_2 \\ a_3 & a_4 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix} \tag{B.14}$$

L'équation B.14 donne le déplacement (V_x,V_y) d'un point à la position (x,y) en fonction de six paramètres liés au mouvement global. Le modèle affine réduit le nombre de mouvements de la caméra à trois types : les translations (t_x et t_y), les rotations (a_2, a_3) et les zooms (a_1, a_4). Nous avons adapter la méthode de Coudray [7, 8] pour estimer ces six paramètres.

Lors de l'estimation du mouvement global, nous avons déterminé les paramètres de translation en localisant le maximum de l'histogramme d'accumulation des vecteurs compensés par les paramètres de déformation. Après étude de tous les pics, une segmentation au sens du mouvement, en plus de l'estimation du mouvement global, est effectuée avec l'hypothèse que chaque pic représente le mouvement d'un objet.

Cette méthode d'estimation du mouvement global possède cependant un léger défaut. En effet, les paramètres de translation du mouvement global sont détectés à l'aide du pic principal dans l'histogramme d'accumulation des vecteurs compensés par les paramètres de déformation. Si la séquence vidéo traitée contient un objet uniforme de taille importante, c'est-à-dire, recouvrant plus de la moitié de l'image, les vecteurs de mouvement de cet objet vont alors être identifiés comme le pic principal dans l'histogramme d'accumulation. Afin de résoudre ce problème et d'identifier correctement le mouvement apparent dominant de la séquence, les blocs de chaque objet segmenté situés sur le bord de l'image sont comptabilisés. L'objet possédant le plus grand nombre de blocs situés sur le bord de l'image sera identifié comme le fond de la scène. Le vecteur de translation associé à cet objet sera donc identifié comme le mouvement apparent dominant.

B.2.2.2 Mouvement relatif et saillance temporelle

À Partir de la connaissance du mouvement apparent dominant $\overrightarrow{V}_{\Theta}$ et du déplacement local $\overrightarrow{V}_{local}$ pour chaque site (macrobloc du tube situé sur l'image centrale du segment temporel de neuf images), le mouvement relatif $\overrightarrow{V}_{relatif}$, exprimé dans le référentiel rétinien est obtenu simplement par la relation suivante :

$$\overrightarrow{V}_{relatif}(s) = \overrightarrow{V}_{\Theta}(s) - \overrightarrow{V}_{local}(s) \tag{B.15}$$

Le mouvement relatif est nécessaire pour estimer le contraste de mouvement inhérent à un site particulier. Mais ce n'est pas suffisant de le considérer de cette façon. En effet l'œil est capable de poursuivre des objets en déplacement. Cette faculté liée au mouvement oculaire de poursuite, permet de conserver l'objet suivi dans la fovéa, partie de la rétine présentant la sensibilité spatiale la plus élevée. Par conséquent, considérer directement le mouvement relatif donné par la relation B.15 serait réducteur. Il n'est pas correct de dire que plus le mouvement relatif est important, plus la saillance est forte. Il faut prendre en compte la capacité maximale de poursuite de l'œil. S. Daly[9] a montré que la vitesse de poursuite maximale de l'œil pouvait aller jusqu'à 80 deg/sec. Si la vélocité du mouvement relatif est supérieure est supérieure à la vélocité maximale de poursuite, alors la saillance temporelle est nulle. De plus, celle-ci sera maximale entre $\overrightarrow{v}_1 = \frac{20}{100} * 30deg/sec$ et $\overrightarrow{v}_2 = 30deg/sec$. Pour les vélocités inférieures à $\overrightarrow{v}_1$ et supérieures à $\overrightarrow{v}_2$, la saillance sera obtenue en fonction d'une droite affine, définie ci-dessous :

$$S^T(s) = \begin{cases} \frac{1}{7}\overrightarrow{V}_{relatif}(s) & \text{pour} \quad 0 \leq \overrightarrow{V}_{relatif}(s) < \overrightarrow{v}_1, \\ 1 & \text{pour} \quad \overrightarrow{v}_1 \leq \overrightarrow{V}_{relatif}(s) < \overrightarrow{v}_2, \\ \frac{1}{60}\overrightarrow{V}_{relatif}(s) + \frac{8}{5} & \text{pour} \quad \overrightarrow{v}_2 \leq \overrightarrow{V}_{relatif}(s) < \overrightarrow{v}_{,max} \\ 0 & \text{sinon,} \end{cases} \tag{B.16}$$

où $\overrightarrow{v}_1 = \frac{20}{100} * 30deg/sec$, $\overrightarrow{v}_2 = 30deg/sec$ et $\overrightarrow{v}_{,max} = 80deg/sec$. L'indice de saillance temporelle obtenu en fonction de la vélocité temporelle est illustré dans la figure B.2.

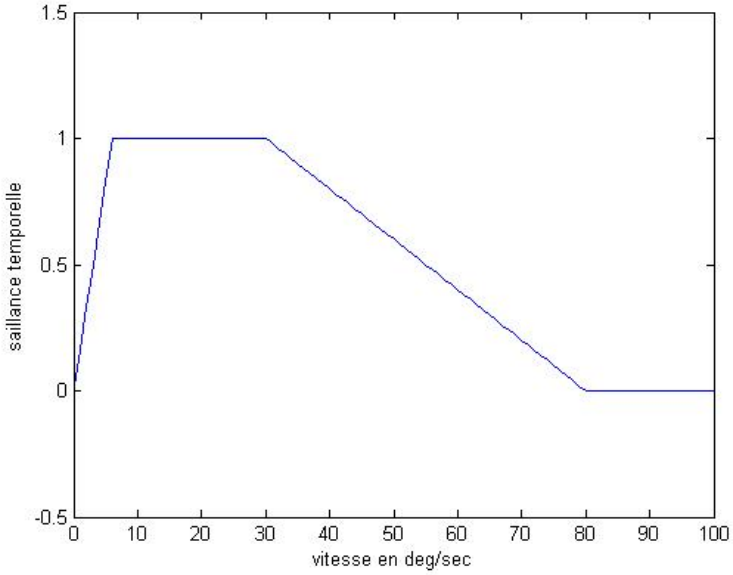

FIG. B.2 – Fonction représentant la saillance temporelle en fonction de la vitesse.

B.2.3 Saillance finale

À Partir de la saillance spatiale et de la saillance temporelle, la saillance spatio-temporelle est à déterminer. Les études réalisées par O. Lemeur[10] montrent que les observateurs ont tendance à favoriser le centre de l'écran. C'est pourquoi il pondère son modèle de saillance spatiale par une gaussienne bi-dimensionnelle centrée sur l'image. Son étendue spatiale a été optimisée sur une base d'images et sa valeur est de $2,5$ degrés visuel. Lors de nos tests, nous utilisons des séquences vidéos Haute Définition. Celles-ci ont une définition maximale de 1080 lignes par 1920 colonnes. C'est pourquoi, nous avons décidé d'utiliser une gaussienne bi-dimensionnelle centrée sur l'image, dont l'étendue spatiale est égale à 5 degrés visuel. La saillance spatio-temporelle est obtenue en combinant la saillance temporelle et la saillance spatiale pondérées par une gaussienne bi-dimensionnelle de la façon suivante :

$$S^{SP-T}(s) = (S^{T}(s) + \frac{1}{2}S^{SP'}(s)) \times gauss2D(s), \tag{B.17}$$

où gauss2D est la gaussienne bi-dimensionnelle d'étendue spatiale égale à 5 degré visuel. Le mouvement étant l'un des paramètres qui influence le plus l'attention visuelle[11], la pondération de la saillance temporelle est deux fois plus importante que celle de la saillance spatiale.

Finalement, nous obtenons une carte de saillance par groupe de neuf images. Ensuite, on projette cette carte pour les images précédentes et suivantes au sein du segment temporel (neuf images) à l'aide des informations issues de l'estimation du mouvement et de la segmentation spatio-temporelle.

B.3 Résultats qualitatifs

Les figures B.3, B.4 et B.5 présentent, pour quatre segments temporels successifs des séquences *Tractor, New Mobile and Calendar* et *Knightshields*, les différentes cartes de saillance obtenues. Concernant les résultats de la figure B.3, on constate que la zone la plus saillante est le tracteur. En effet, le mouvement réel de celui-ci est détecté par notre méthode et ainsi, le tracteur devient la zone la plus saillante. On observe cependant que la saillance du tracteur n'est pas uniforme. Les caractéristiques spatiales du tracteur sont très hétérogènes en terme de couleur et ce sont les roues de couleur rouge (couleur

chaude) qui sont les plus saillantes. Concernant les résultats de la figure B.4 relatif à la séquence *New Mobile and Calendar*, les zones les plus saillantes sont également les objets en mouvement. Dans les trois premiers segments temporels présentés (43, 44, et 45), le calendrier est une zone saillante alors que pour le segment temporel 46, il n'est plus saillant. En effet, on constate que la saillance temporelle basée sur le mouvement ne détecte pas le calendrier comme une zone saillante. Le mouvement de translation du calendrier dans le segment temporel 46 s'estompe et devient quasiment nul, de ce fait, sa saillance temporelle basée sur le mouvement de celui-ci est nulle. Dans les trois premiers segments temporels, la saillance spatio-temporelle du calendrier n'est pas uniforme. En effet, la gaussienne bi-dimensionnelle utilisée pour reproduire l'effet de favorisation du centre de l'écran par les observateurs diminue progressivement la saillance des zones éloignées du centre de l'image. De plus, les chiffres écrits en rouge sur le calendrier sont plus saillantes que les zones voisines. Les figurines disposées sur le train en mouvement sont détectées dans les quatre segments temporels. La figurine orange représentant un tigre est détectée comme la zone la plus saillante, du fait de sa position (au centre en bas) et de sa couleur (orange). Pour la dernière séquence testée, les résultats présentés dans la figure B.5, semblent corrects également. En effet, la zone la plus saillante est l'homme se déplaçant. Le fond, bien qu'étant immobile (réellement), très riche en informations spatiales est saillant par endroit. Les différents blasons ("*shields*") sont plus ou moins saillants en fonction de leur couleur. Les blasons possédant des couleurs chaudes (rouge, orange jaune) sont des zones saillantes.

Il est difficile de conclure définitivement sur la qualité de la modélisation proposée. Elle semble toutefois permettre de détecter relativement fiablement les zones les plus saillantes d'une séquence vidéo.

FIG. B.3 – Cartes de saillance pour la séquence *Tractor* (segments 13 à 16), avec de haut en bas : images originales, cartes de saillance spatiale, cartes de saillance temporelle et cartes de saillance spatio-temporelle.

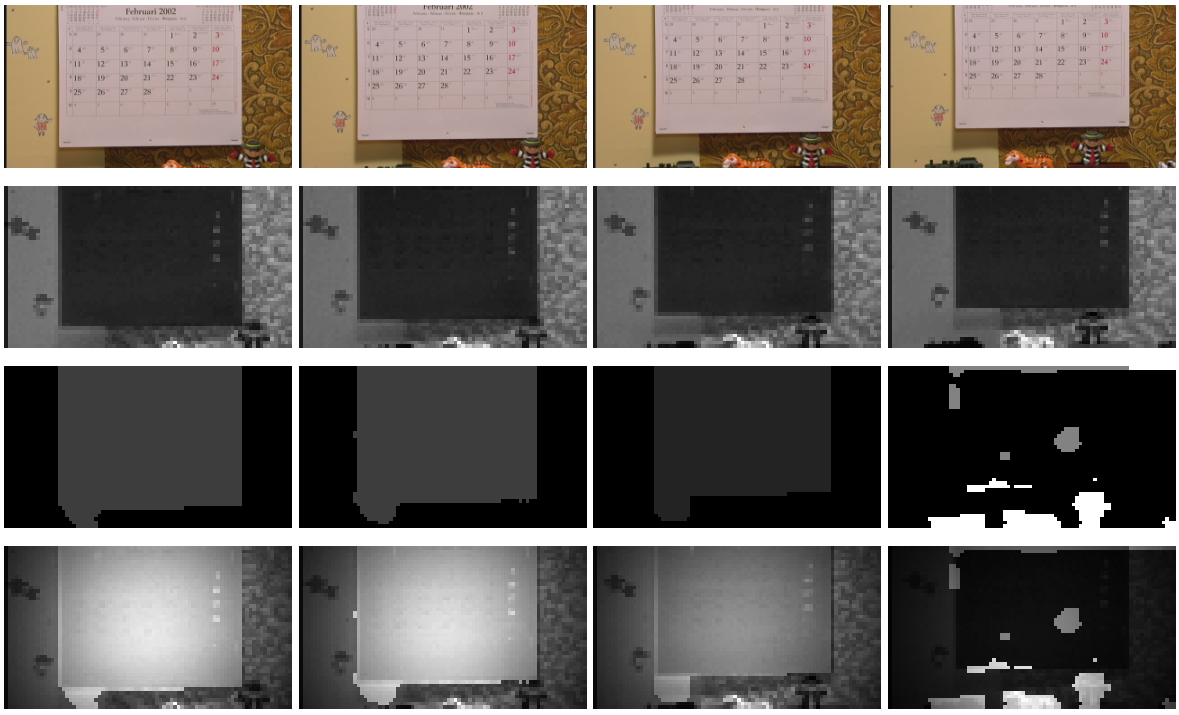

FIG. B.4 – Cartes de saillance pour la séquence *New Mobile and Calendar* (segments 43 à 46), avec de haut en bas : images originales, cartes de saillance spatiale, cartes de saillance temporelle et cartes de saillance spatio-temporelle.

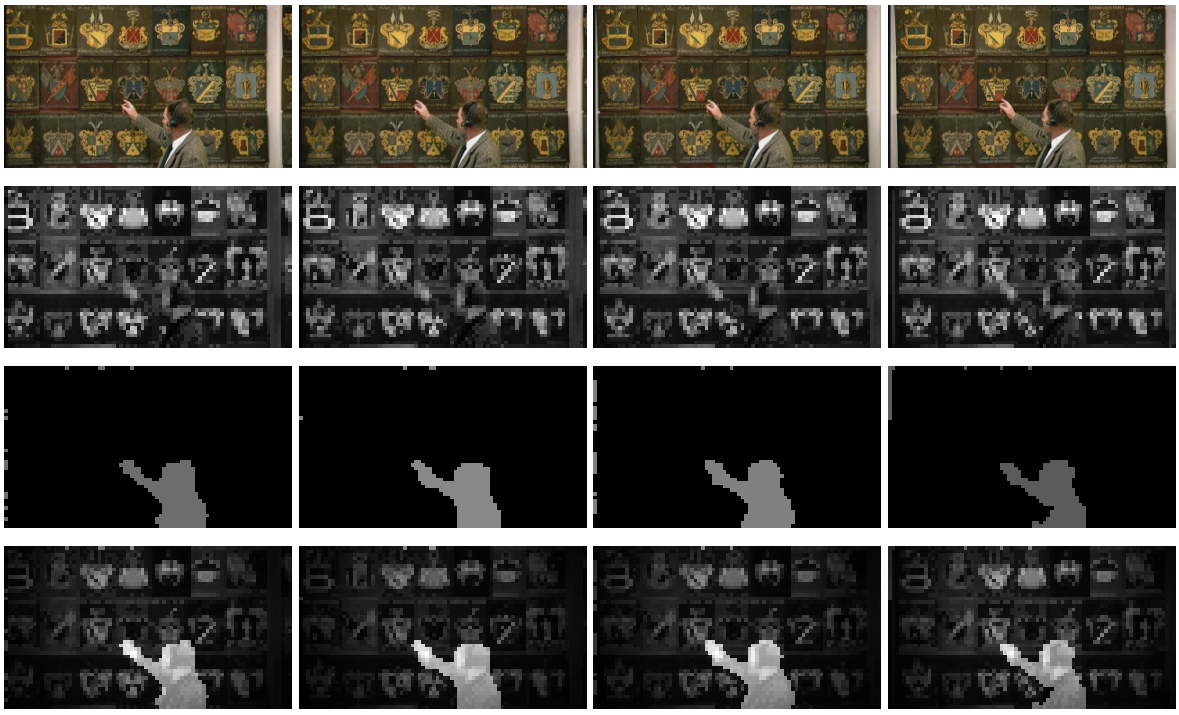

FIG. B.5 – Cartes de saillance pour la séquence *Knightshields* (segments 28 à 31), avec de haut en bas : images originales, cartes de saillance spatiale, cartes de saillance temporelle et cartes de saillance spatio-temporelle.

B.4 Conclusion

Ce chapitre a présenté une méthode de modélisation de saillance spatio-temporelle. Le calcul de la saillance spatiale est basé sur les couleurs et plus particulièrement sur les contrastes de couleur. Comparativement à la détermination de la saillance spatiale, la saillance temporelle est plus facile à calculer car le concept sous-jacent est relativement simple. Une zone temporellement saillante est une zone en contraste de mouvement. Dans l'approche proposée, le contraste de mouvement est déterminé à partir d'une estimation locale (vecteur de mouvement par tube) et d'une estimation globale du mouvement ; la différence, appelée mouvement relatif, indique les zones temporellement saillantes. La saillance finale est obtenue en combinant les saillances spatiale et temporelle. Le centre de l'image en lui-même étant une zone particulière qui attire l'attention visuelle, nous avons utilisé une gaussienne bi-dimensionnelle afin de modéliser ce phénomène lors du calcul de notre saillance spatio-temporelle. Les résultats obtenus en terme de saillance visuelle semblent fiables.

Afin d'évaluer quantitativement les résultats de notre modèle, il serait intéressant de posséder des données réelles en terme de saillance visuelle. Pour cela, on pourra réaliser des expérimentations oculométriques sur les séquences vidéos testées et collecter les données pour construire une référence en terme de saillance visuelle.

Annexe C

Codage vidéo à qualité différenciée basé sur la saillance visuelle

C.1 Introduction

L'objet de ce chapitre concerne l'application de compression de la vidéo avec une qualité visuelle différenciée pilotée par les cartes de saillance. Ce type de compression est communément appelée compression sélective ou compression avec régions d'intérêt. Contrairement aux approches conventionnelles de compression d'images distribuant de façon homogène les ressources de codage, la compression sélective répartit les ressources de codage de façon adaptée directement ou indirectement. Dans un contexte de compression avec pertes, la distribution adaptée des ressources de codage peut permettre d'accroître substantiellement la qualité globale perçue. L'idée est simple puisqu'elle consiste à favoriser la qualité des zones les plus importantes visuellement. Bien évidemment, cela nécessite de disposer d'informations à priori sur la scène à coder. Dans ce contexte d'études, nous couplons un schéma de compression sélective directe avec notre modèle de saillance visuelle décrit dans le chapitre précédent.

C.2 Compression sélective directe

L'objectif de la compression sélective directe est de contrôler la distribution des ressources de codage ne fonction de l'intérêt visuel de chaque macrobloc afin d'accroître la qualité visuelle perçue. Il a été montré qu'une compression sélective sur images fixes permettait d'améliorer la qualité subjective d'une part lorsque les zones d'intérêt sont de tailles relativement faibles et d'autre part, lorsque pour une approche classique de codage, le débit de consigne provoque l'apparition d'artefacts de codage sur les zones saillantes. La plupart du temps, le paramètre sur lequel on agit est la consigne de quantification. En d'autres termes, un macrobloc présentant un intérêt visuel faible sera quantifié plus grossièrement qu'un macrobloc ayant un intérêt visuel important.

C.3 Modification du cœur de codage

La modification du coeur de codage consiste à déterminer l'index de quantification de chaque macrobloc. L'index de quantification est déterminer en fonction de la carte de saillance spatio-temporelle. Plus une zone est saillante et plus elle sera quantifiée finement et inversement. La carte de saillance spatio-temporelle nous indique la saillance de chaque macrobloc situé sur l'image centrale d'un segment temporel (neuf images). Les cartes de saillance des autres images du segment temporel (les quatre images précédentes et les quatre images successives) sont déduites à partir des informations de mouvement (projection de la carte de saillance de l'image centrale).

L'indice de saillance calculé pour un macrobloc varie entre 0 (saillance nulle) et 1 (très saillant). Afin de quantifier les macroblocs en fonction de leur indice de saillance, le pas de quantification doit

être modifié par rapport à une stratégie classique de codage. Pour ce faire, on modifie la valeur du pas de quantification calculé par le codeur. Si le macrobloc est saillant, on diminue la valeur du pas de quantification et dans le cas contraire on augmente celui-ci.

La modification du pas de quantification au sein du codeur est réalisé de la façon suivante :

$$QP_{saillance}(i) = QP_{codeur}(i) - I_{qp}(i), \tag{C.1}$$

où $QP_{codeur}(i)$ est l'index de quantification calculé par le codeur pour le macrobloc i et $I_{qp}(i)$ est l'indice de quantification calculé en fonction de la saillance du macrobloc i et est calculé de la manière suivante :

$$I_{qp}(i) = 2 \times \frac{S^{SP-T}(i) - \overline{S^{SP-T}}}{1 - \overline{S^{SP-T}}},$$

où $\overline{S^{SP-T}}$ est la valeur moyenne de la saillance de l'image à laquelle appartient le macrobloc i.

C.4 Conclusion

L'objectif de ce chapitre était de décliner un cadre général de compression vidéo utilisant une carte de saillance. Après avoir défini la compression sélective directe, la carte de saillance a été couplée à un schéma de codage. Cette méthode concerne la compression sélective directe. Pour ce type de compression, c'est le cœur et la stratégie de codage qui sont modifiés. L'objectif est d'améliorer la qualité perçue comparativement à une approche classique de codage.

Annexe D

Présentation des séquences vidéo utilisées lors des tests

Les séquences utilisées lors des tests réalisés pour les besoins de ce rapport sont disponibles via le serveur ftp `ftp://ftp.ldv.e-technik.tu-muenchen.de/pub/test_sequences/`. Ces séquences ont été filmées à une fréquence de 50 images par seconde avec l'équipement du SVT en octobre 2004. La plus grande attention a été donnée à la conversion des films vers un format numérique. Les détails concernant les conditions de prise de vue et les post-traitements sont présentés dans la documentation fournie par le SVT [**??**].

D.1 Les séquences 720p

Les séquences 720p utilisées ici sont des vidéos progressives de 720 lignes par 1280 colonnes, cadencées à 50 images par seconde, la structure d'échantillonnage couleur des composantes YUV est 4 :2 :0.

D.1.1 New Mobile and Calendar

La séquence comporte 500 images filmées en plan rapproché. La caméra, qui subit un mouvement translationnel puis de zoom arrière, filme un calendrier avec du texte et une photo détaillée du Vasa[1]. Á partir de la 355ème image apparaît un train en mouvement translationnel avec des jouets très colorés. Le fond est composé de deux types de papiers peints, le premier est jaune, uniforme avec quelques figures dessinées et le second est très texturé. La figure D.1 présente une image extraite de la séquence *New Mobile and Calendar*.

D.1.2 Knightshields

La séquence comporte 500 images filmées en plan rapproché. Un homme avec une barbe et une veste très texturée marche devant un mur composé de boucliers de chevaliers détaillés. Á la fin de la séquence, le capteur effectue un zoom avant de la scène. La figure D.2 présente une image extraite de la séquence *Knightshields*.

D.1.3 Parkrun

La séquence comporte 500 images filmées en plan éloigné. La scène représente un homme, avec un parapluie dans sa main, qui court dans un parc puis s'arrête et reste immobile vers la 340ème image. L'arrière plan est composé d'arbres, de neige et d'une source d'eau. Le contenu est très détaillé. La figure D.3 présente une image extraite de la séquence *Parkrun*.

[1] Le Vasa est un vaisseau de guerre scandinave du 17ème siècle.

D.2 La séquence 1080p

Les séquences 1080p utilisées ici sont des vidéos progressives de 1080 lignes par 1920 colonnes, cadencées à 25 images par seconde, la structure d'échantillonnage couleur des composantes YUV est également 4 :2 :0.

D.2.1 Tractor

La séquence comporte 690 images qui présentent un tracteur dans un champ. La séquence entière contient des zones sur lesquelles un très fort zoom avant est appliqué de manière à en obtenir une vue totale. La caméra suit le tracteur, avec un mouvement chaotique, sur la structure du champ de récolte. La figure D.4 présente une image extraite de la séquence *Tractor*.

FIG. D.1 – Image 478 de la séquence *New Mobile and Calendar*.

FIG. D.3 – Image 160 de la séquence *Parkrun*.

FIG. D.2 – Image 1 de la séquence *Knightshields*.

FIG. D.4 – Image 60 de la séquence *Tractor*.

Bibliographie

[1] E. Besag. "*Spatial interaction and the statistical analysis of lattice systems (with discussion)*". *Journal of the Royal Statistical Society*, Series B, 36 :196-236, 1974.

[2] P. Lalande. *"Détection du mouvement dans les séquences d'images selon une approche markovienne ; application à la robotique sous-marine*". Thèse de doctorat, Université de Rennes I, 1990.

[3] S. Geman, and D. Geman, "*Stochastic relaxation, Gibbs distribution and the Bayesian restoration of images," IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, no. 6, pp. 721 - 741, 1984.

[4] M. Z. Aziz, and B. Mertsching, *"Fast and Robust Generation of Feature Maps for Region-Based Visual Attention."* *IEEE Transactions on Image Processing*, vol. 17 , no. 5 , pp. 633-644 , 2008.

[5] J.M. Wolfe, and T.S. Horowitz, *"What attributes guide the deployment of visual attention and how do they do it ?" Nature Rev. Neuroscience,* vol. 5, pp. 1-7, 2004.

[6] J.M. Wolfe, K.R. Cave, and S.L. Franzel, *"Guided search : an alternative to the feature integration model for visual search." Journal of experimental psychology. Human perception and performance*, vol. 15, no3, pp. 419-433, 1989

[7] R. Coudray, and B. Besserer, *"Global motion estimation for MPEG-encoded streams," in Proc. IEEE International Conference on Image Processing,* ICIP 2004, Singapore, Republic of Singapore, October 2004.

[8] O. Brouard, F. Delannay, V. Ricordel, and D. Barba, *"Robust Motion Segmentation for High Definition Sequences using a Fast Multi-Resolution Motion Estimation based on Spatio-Temporal Tubes" in Proc. Picture Coding Symposium,* PCS 2007, Lisbonne, Portugal, novembre 2007.

[9] S. Daly *"Engineering Observations from Spatiovelocity and Spatiotemporal Visual Models." In IS&T/SPIE Conference on Human Vision and Electronic Imaging III.*, SPIE vol. 3299, pp. 180-191, janvier 1998.

[10] O. Lemeur *"Attention sélective en visualisation d'images fixes et animées affichées sur écran : Modèles et évaluation de performances - Applications"* Université de Nantes, PhD. Thesis, Ecole polytechnique de l'université de Nantes, 2005.

[11] W. Osberger, A.J. Maeder, and N. Bergmann "A Perceptually Based quantization Technique for MPEG Encoding"*In IS&T/SPIE Conference on Human Vision and Electronic Imaging III.*, SPIE vol. 3299, pp. 148-1591, janvier 1998.

[12] http ://www.acceptv.com/