

# Réduction du débit des séquences bruitées dans un codeur

## MPEG4

K.Hachicha<sup>1</sup> P.Garda<sup>1</sup>

<sup>1</sup> Laboratoire des Instruments et Systèmes en Ile de France, Groupe SYEL

Université Pierre et Marie Curie  
4 place Jussieu 75252 Paris Cedex 05

[khalil.hachicha, patrick.garda@lis.jussieu.fr](mailto:khalil.hachicha,patrick.garda@lis.jussieu.fr)

### Résumé

*Dans cet article, nous nous proposons de créer un filtre pour le résidu résultant de la différence entre les matching macro blocs se trouvant dans l'image prédite et l'image de référence. Le filtre, modelé par un algorithme de détection de mouvement basé sur les champs aléatoires de Markov, réduit la variation de la luminance qui résulte plus souvent du bruit que d'un mouvement réel. Ainsi, le débit binaire diminue et la qualité de la séquence est maintenue. Nous évaluons par la suite la complexité algorithmique introduite par cette technique en l'implémentant sur le DSP TMX320C5510.*

### Mots clefs

Compression vidéo, détection de mouvement, champs markoviens

## 1 Introduction

Dans le standard MPEG4, le codage des images en mode P utilise la prédiction compensée en mouvement basé sur une image de référence. Pour chaque macro bloc de l'image actuelle, nous cherchons le macro bloc le plus semblable dans l'image de référence et nous codons la différence.

Quand le bruit devient important dans la séquence, cette différence augmente considérablement. Ceci atténue la performance du codage et entraîne l'augmentation du débit sans amélioration de la qualité.

Pour remédier à ce problème, nous proposons la création d'un masque de mouvement pour chaque macro block de différence. Il joue le rôle d'un filtre et permet la réduction des variations de luminance ayant une plus grande probabilité de résulter d'un bruit que d'un mouvement réel. Le débit est ainsi réduit tout en maintenant la qualité de la séquence.

Dans cet article, nous présentons le principe du codeur MPEG4 que nous avons développé et nous expliquons les principes fondamentaux de l'algorithme de détection de

mouvement basé sur les champs de Markov. Nous décrivons également les différentes étapes suivies pour intégrer cet algorithme dans le codeur MPEG4. Enfin, nous évaluons cette nouvelle technique sur différents types de séquences.

## 2 Description du codeur MPEG4

En s'aidant du codeur de MoMusys proposé par l'ISO [1], la première étape de ce travail était de développer un codeur MPEG4 basique. Notre codeur comporte 2 modes de codage pour les images : Intra (I) et prédictif (P). Chaque image est divisée en macro blocs (4 blocs de Pixels de luminance et 2 blocs de chrominance). Les images I sont codées sans référence. Nous appliquons pour chaque bloc la DCT suivie d'une quantification, puis d'une prédiction DC/AC et enfin d'un codage entropique. Pour les images de type P, le mouvement entre l'image actuelle et la référence est étudié. Les macro blocs sont alors pris un par un pour déterminer le plus ressemblant dans l'image de référence (la zone de recherche est +/- 16 Pixels). L'algorithme aboutit alors à l'une des deux conclusions suivantes : si le résidu résultant de la différence des matching des deux macro blocs dépasse un certain seuil, l'algorithme conclut que le macro bloc est nouveau pour la séquence. Il est dans ce cas codé en mode Intra. Si la différence est en dessous du seuil, nous améliorons le processus d'estimation de mouvement en passant au niveau du 1/2 pixel. Nous interpolons l'image de référence et nous essayons de trouver dedans le meilleur macro bloc correspondant au macro bloc à coder. Nous calculons leur différence qu'on appellera D1. Nous faisons la même opération pour les 4 sous blocs constituant le macro bloc à coder et nous calculons la somme de leur différence par rapport à leur meilleur correspondant. Cette somme sera désignée par D2. La comparaison entre D1 et D2 nous permet de décider si l'on doit transmettre 1 Mv (Motion Vector) ou bien 4 Mv au codeur.

La combinaison des vecteurs de mouvement et de l'image de référence permet de créer l'image compensée. La

différence entre l'image à coder et l'image compensée est convertie par la DCT, quantifiée et codée avec les vecteurs de mouvement. Nous avons vérifié la compatibilité de notre codeur en utilisant un outil fourni par IBM. Il nous a permis d'ajouter la couche système à notre flux et d'afficher ainsi les séquences avec les différents players MPEG4 disponibles (Quicktime, Philips,...).

### 3 Algorithme Markovien de détection de mouvement

Nous avons repris un modèle reposant sur la théorie de Markov qui segmente robustement les pixels en mouvement et qui permet de résister aux variations légères de luminance provenant d'un bruit [2]. Cet algorithme fournit en sortie des images de masques modélisées suivant la théorie mathématique des champs de Markov. Il « nettoie » les images de différences et conduit à une augmentation du taux de compression. Il est constitué de deux blocs : un premier bloc de détection de mouvement qui effectue la différence entre une image de référence et l'image suivante et fournit en sortie une image de variation de la luminance. Le deuxième bloc crée la carte binaire du mouvement : il prend la valeur absolue du signal de différence et la binarise en choisissant un seuil qui élimine les détections parasites. Cette mise en forme a pour objectif de fournir les entrées nécessaires à l'algorithme de l'ICM [3]. Ce dernier est une méthode d'optimisation qui garantit la convergence vers le premier minimum de la fonction d'énergie.

Le calcul d'énergie nécessite la connaissance des états des pixels voisins appartenant à un voisinage défini par 8 voisins spatiaux et deux voisins temporels. Il est composé de :

- une énergie associée aux données (1):

$$U_d(s) = \frac{1}{2\sigma^2} (O(s) - \psi(s))^2 \begin{cases} \psi(s) = 0 \text{ si } s = 0 \\ \psi(s) = \alpha \text{ si } s = 1 \end{cases} \quad (1)$$

$$\begin{cases} O(s) = \psi(s) + b \\ \Psi \text{ modélise les observations.} \\ b \text{ bruit supposé blanc de variance } \sigma^2 \end{cases}$$

- Une énergie attachée au modèle (2)

Elle est composée de l'énergie spatiale (censée modéliser la cohérence et la compacité d'un objet en mouvement) (3) et de l'énergie temporelle, qui représente la variation de la fonction intensité lors du passage d'une image à l'autre (4).

$$U_m(s, rf, rp) = U_s(s) + U_t(s, rp, rf) \quad (2)$$

$$U_s(s) = \sum_s V_s(s) \quad \text{Avec} \begin{cases} V_s = -\beta_s \text{ si } s = r \\ V_s = +\beta_s \text{ si } s \neq r \end{cases} \quad (3)$$

$$U_t(s, rf, rp) = V_p(s, rp) + V_f(s, rf) \quad (4)$$

$$\begin{cases} V_p = -\beta_p \text{ si } s = rp \\ V_p = +\beta_p \text{ si } s \neq rp \end{cases} \quad \begin{cases} V_f = -\beta_f \text{ si } s = rf \\ V_f = +\beta_f \text{ si } s \neq rf \end{cases}$$

Pour tous les sites de l'image, on calcule l'énergie locale relative à l'état immobile, ainsi que l'énergie locale relative à l'état mobile. Ensuite, on affecte au site en cours de traitement l'état qui minimise l'énergie. La minimisation de l'énergie a un effet de filtrage du bruit. En sortie de l'ICM, nous obtenons une image d'énergie minimale qui représente la carte binaire du mouvement.

|           |    |
|-----------|----|
| $\beta_p$ | 10 |
| $\beta_s$ | 20 |
| $\beta_f$ | 30 |
| $\alpha$  | 15 |

Tableau 1 : Paramétrage de l'algorithme

Le produit entre la carte binaire du mouvement et l'image de différence génère une image de variation de la luminance filtrée. La figure suivante illustre un exemple de résultat que nous avons obtenu par le biais de cet algorithme :

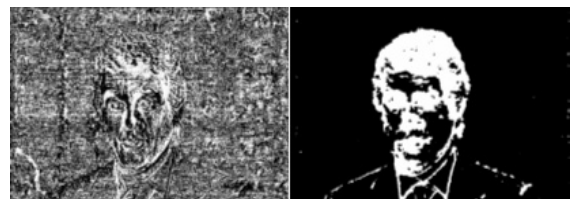


Figure 1 : différence Carte binaire de mouvement

### 4 Intégration de l'algorithme markovien dans le codeur MPEG4

L'algorithme que nous venons de présenter est basé sur l'hypothèse d'un capteur fixe. Si le capteur est mobile, il est évident que le mouvement propre du capteur engendre une variation de la luminance qui n'a pas de lien direct avec le mouvement. La création du masque de mouvement devient alors impossible, et la technique montre ses limites. Cependant, avec le processus d'estimation/compensation de mouvement dans MPEG4, la différence entre les matching macro blocs représente

plutôt la variation de la luminance directement liée au mouvement et non pas au mouvement de la caméra. Partant de ce constat, nous avons appliqué cette technique au niveau de chaque macro bloc de différence pour n'importe quelle type de séquence. Le schéma de codage complet est représenté ci dessous :

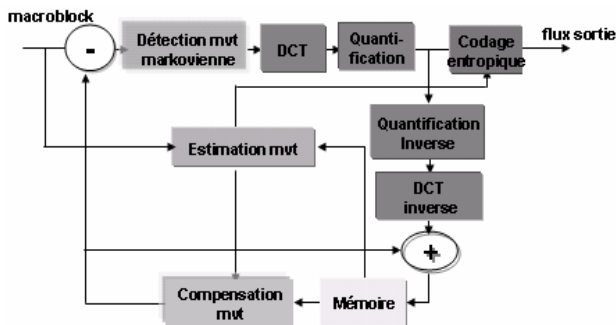


Figure2 : Markov MPEG4

L'algorithme code l'erreur de prédiction seulement pour les pixels dont l'information de masque de mouvement est placée à 1. La création du masque jouant le rôle d'un filtre, dépend du type de codage du macro block. S'il est du type intra (I) alors toute l'information contenue dans le macro bloc est transmise sans aucune opération de filtrage. S'il est du type Inter, nous créons :

- Un masque pour tout le macro block dans le cas où le codeur décide d'envoyer 1MV/MB,
- 4 masques (1 pour chaque block dans le macro block) dans le cas où 4 MV/MB sont envoyés.

Le synopsis de la méthodologie est décrit en détail par le diagramme suivant :

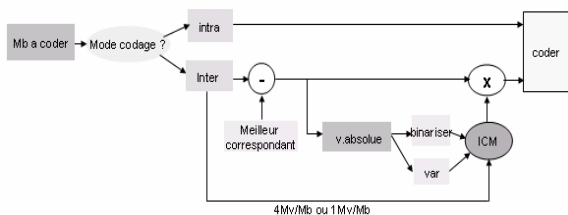


Figure 3 : Création des masques

## 5 Résultats et interprétation

Nous avons testé notre algorithme sur plusieurs types de séquences. Pour les séquences bruitées, l'algorithme se montre très performant par rapport à un codeur classique : le débit binaire est réduit d'une façon intéressante, et malgré une réduction du PSNR due au filtrage, la qualité de l'image reste tout à fait comparable à celle résultant d'un codage classique MPEG4 sans recours à l'algorithme

de détection de mouvement de Markov. Nous présentons ci-dessous un exemple de gain en débit obtenu pour une séquence bruitée d'une émission de France 3 et d'une deuxième extraite d'un reportage où nous trouvons beaucoup de mouvement et de bruit. Pour chacun des tests, les mêmes paramètres de codage sont utilisés et d'un test à l'autre, nous augmentons la valeur de quantification.

| France 3       | Test 1 | Test 2 | Test 3 | Test 4 |
|----------------|--------|--------|--------|--------|
| Bit rate       | 291    | 181    | 130    | 102    |
| Psnr luminance | 38.83  | 36.26  | 34.51  | 33.22  |
| France3markov  |        |        |        |        |
| Bit rate       | 244    | 159    | 118    | 95     |
| Psnr luminance | 36.13  | 34.83  | 33.69  | 32.73  |

Tableau 2 : France 3

|                     | Test  | Test2 | Test 3 | Test 4 |
|---------------------|-------|-------|--------|--------|
| MPEG 4              |       |       |        |        |
| Bit rate (kb/s)     | 570   | 365   | 265    | 210    |
| Psnr luminance (db) | 38.1  | 35.64 | 33.97  | 32.73  |
| MPEG 4 markov       |       |       |        |        |
| Bit rate (kb/s)     | 480   | 310   | 235    | 190    |
| Psnr luminance (db) | 35.71 | 33.57 | 32.62  | 31.85  |

Tableau 3 : Reportage

## 6 Evaluation de l'algorithme de détection de mouvement sur le C5510

Les systèmes d'imagerie embarqués disposent de ressources limitées de calcul et d'énergie. L'intégration de l'algorithme de détection de mouvement dans le codeur MPEG4, nous oblige à évaluer la complexité algorithmique introduite surtout si la durée de codage est un critère important. Actuellement, les deux grandes classes d'architectures de traitement sont les architectures programmables — DSP ou microprocesseurs — et les architectures spécialisées — câblées, configurables ou reconfigurables dynamiquement. Certes, jusqu'à présent les architectures spécialisées présentaient les meilleurs compromis calcul/consommation, cependant, les architectures programmables ont connu récemment des progrès très importants. En conséquence, nous avons privilégié ces dernières. Nous avons étudié plus particulièrement le cœur du DSP C5510 de Texas Instruments, parce qu'il présente des caractéristiques très attractives et parce qu'il est un élément essentiel dans deux plates-formes embarquées adaptées à l'image, OMAP510 et DSC24.

## 6.1 Le TMS320VC5510

Le TMX320VC5510 possède un CPU de fréquence d'horloge 160 MHz. Il est composé de 4 unités. L'unité I de décodage d'instructions qui reçoit 4 octets de code dans un tampon d'instructions et décode de 1 à 6 octets par cycle d'horloge. L'unité P génère toutes les adresses de l'espace mémoire programme et contrôle la séquence des instructions en dirigeant les opérations de saut et d'exécution conditionnelle. L'unité A contient une ALU de 16 bits et toutes les logiques nécessaires pour générer les adresses de l'espace mémoire de données. Enfin, l'unité D, essentielle pour le traitement, comporte une ALU de 40 bits, un décaleur qui assure en un seul cycle d'horloge des décalages de 32 bits vers la droite ou la gauche et 2 MAC qui caractérisent typiquement le noyau DSP. Dans un seul cycle d'horloge, chaque MAC effectue des multiplications 17 bits x 17 bits et des additions/soustractions de 40 bits.

Le DSP possède 320 Ko de mémoire interne et 1 Mo de mémoire externe extensible à 16 Mo. Cet espace mémoire est unifié pour les données et le programme, l'accès à chacune des deux mémoires se faisant via un chemin distinct. Cette organisation permet de transférer une instruction et des données simultanément, ce qui améliore les performances. Toutes les unités communiquent bidirectionnellement avec la mémoire à travers différents bus. Pour lire les données, 3 bus d'adresses de 24 bits et 3 bus de lecture de 16 bits sont utilisés. Pour lire le programme, le CPU utilise un bus d'adresses 24 bits et un bus de lecture 32 bits.

## 7 Résultats

### 7.1 Cadence de traitement

| Labo  | Architecture | Frq Mhz | Type image  | Taille image | Nombre image/s |
|-------|--------------|---------|-------------|--------------|----------------|
| LISIF | TM320VC5510  | 160     | Niveau gris | 256x256      | 17             |
| LISIF | C80          | 60      | Niveau gris | 256x256      | 18             |
| Lis   | CNAPS        | 20      | Niveau gris | 256x 256     | 3              |
| Lis   | M 56002      | 60      | Niveau gris | 128x128      | 15             |

Tableau 4 : Cadence de traitement

### 7.2 Temps d'exécution pour créer un masque d'un macro bloc.

| Fonctions          | Temps d'exécution en us |
|--------------------|-------------------------|
| Différence         | 98                      |
| Valeur absolue     | 71                      |
| Binarisation       | 86                      |
| Calcul de variance | 164                     |
| ICM                | 229                     |
| Produit matrice    | 98                      |

Tableau 5 : Temps de création du masque

## 7.3 Consommation

Une des caractéristiques majeures du C5510 est sa faible consommation. De ce fait nous avons calculé le nombre de cartes binaires de mouvement générées par watt par seconde. Le graphe ci-dessus donne un comparatif entre notre résultat et ceux d'autres implémentations de l'algorithme effectuées sur d'autres plates-formes

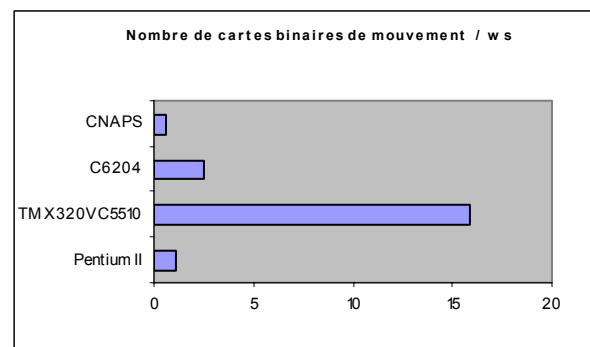


Figure 4 : Nombre de carte binaire /w/s

## 8 Conclusions

Notre travail est organisé en trois parties :

- 1) la compréhension et le développement d'un algorithme MPEG4 de base,
- 2) l'étude et le développement de l'algorithme de détection de mouvement basé sur les champs aléatoires de Markov
- 3) l'adaptation et l'intégration de l'algorithme de Markov dans le codeur MPEG4 et
- 4) l'évaluation de la complexité algorithmique introduite par cet algorithme en l'implémentant sur le TMX320VC5510.

Ce travail présente une solution efficace pour réduire l'information ayant une grande probabilité d'être du bruit. Les perspectives de notre travail sont d'intégrer l'algorithme de détection de mouvement dans la nouvelle norme H264 et d'estimer son apport.

## Références

- [1] ISO/IEC 14496-5:2001.Coding of Audio-Visual Objects – Part 5: Reference Software, 2nd Edition, 2001
- [2] A.Caplier, Modèle Markovien de détection de mouvement dans les séquences d'images: thèse institut national Polytechnique de Grenoble (INPG), France Décembre 1995
- [3] C.Dumontier, Etude et mise en oeuvre temps réel d'un algorithme de détection de mouvement par approche markovienne. Thèse de Doctorat de l'Institut National Polytechnique de Grenoble (INPG), France, Novembre 1996
- [4] ISO/IEC 14496-2:2001.Coding of Audio-Visual Objects Part2: Visual, 2nd Edition, 2001.
- [5] The MPEG4 Book, Fernando Pereira, Tourajdi Ebrahimi, p294-315, 2002 Pearson Education