Estimateur de mouvement temps réel multi-DSP pour l'encodage vidéo MPEG-4 AVC/H.264 haute définition

F. Urban[1] R. Poullaouec[1] O. Deforges[2] J-F. Nezan[2]

[1] THOMSON RD France VCL
1 av.belle fontaine, 35576 Cesson-Sévigné
{urbanf, poullaouecr}@thomson.net

[2] IETR - UMR CNRS 6164/groupe Image
20 av. des Buttes de Coësmes 35043 Rennes
{odeforge, jnezan}@insa-rennes.fr

Résumé

Le dernier standard vidéo MPEG-4 AVC/H.264 proposé en mars 2003 s'appuie sur des nouvelles techniques améliorant la compression. En contrepartie, MPEG-4 AVC introduit une complexité rendant problématique les solutions temps réel dans le domaine de l'embarqué. Les performances de ces codeurs vidéo dépendent en grande partie de celles de l'estimation de mouvement. C'est aussi la fonction qui requiert le plus de ressources de calcul et de bande passante mémoire. Le cadre de la vidéo haute définition amplifie cette difficulté d'une exécution temps-réel.

De nombreuses techniques d'estimation de mouvement ont été développées afin de réduire les temps de traitement en gardant la meilleure précision possible. L'objectif de cet article est tout d'abord de faire le point sur ces travaux et de mettre en évidence les meilleurs candidats pour un estimateur MPEG-4 AVC. Les algorithmes HME et EPZS sont ensuite étudiés et implantés sur un processeur de traitement du signal. Leurs performances en terme de qualité d'estimation et de vitesse d'exécution sont finalement comparées.

Mots clefs

Estimation de mouvement, Codeur MPEG-4 AVC temps réel, DSP

1 Introduction

Grâce au développement des techniques de compression vidéo et des systèmes de communications, la diffusion de séquences vidéo est de plus en plus répandue. Néanmoins la bande passante nécessaire pour transmettre une vidéo haute définition reste importante, malgrès le développement de schémas de compression toujours plus efficaces qui permettent de réduire les débits. Le standard de compression vidéo MPEG-4 AVC / H.264 issu de la collaboration entre ITU et MPEG (JVT) permet d'atteindre des débits 50% inférieurs à ceux offerts par MPEG-2.

La compression de donnée dans une vidéo est basée sur l'élimination des redondances spatiales et temporelles. Chaque image peut en effet être reconstruite en utilisant de la prédiction intra-image (I) ou inter-image (P, B). Les images P et B sont reconstruites à l'aide d'une ou plusieurs images précédemment encodées (référence) auxquelles un champ de vecteur est associé. L'opération d'estimation de mouvement consiste à rechercher pour tous les blocs de l'image courante leur mouvement respectif par rapport à une image de référence. Les performances d'un encodeur vidéo dépendent donc beaucoup de la précision de l'estimation de mouvement.

L'estimation de mouvement est une opération qui nécessite une puissance de calcul importante, notamment dans la norme H.264 où les images peuvent être découpées en blocs de tailles variables, les vecteurs de mouvements sont exprimés au quart de pixel près et plusieurs images de référence sont autorisées [1]. De 60 à 80% des ressources matérielles d'un encodeur vidéo sont occupées par l'estimation de mouvement.

L'objectif général de ces travaux est le prototypage d'un estimateur de mouvement temps réel pour un encodage H.264 format HD, et implanté sur une architecture embarquée multi-composants. Cet article présente les résultats de la première phase, à savoir le choix des algorithmes et leur implantation sur DSP (Digital Signal Processor)

La partie 2 dresse un état de l'art rapide des techniques d'estimation de mouvement existantes, la section 3 décrit les implantations embarquées réalisées. La section 4 donne des résultats de comparaison de deux algorithmes d'estimation de mouvement en terme de qualité d'estimation et de temps d'exécution.

2 Méthodes d'estimation de mouvement - état de l'art

L'opération d'estimation de mouvement permet de retrouver les mouvements relatifs entre deux images afin d'éliminer la redondance temporelle. Il existe différentes techniques d'estimation de mouvement et plusieurs façons d'exprimer le résultat. Les algorithmes pixels-récursifs basés gradient [2] produisent un champ de vecteurs dense, c'est à dire qu'une description du mouvement sera donnée

pour chaque pixel de l'image. Une telle description est utile pour des traitements vidéo comme le suivi d'objet, le désentrelacement ou encore la conversion de standard, mais est inadaptée à la compression vidéo où l'image est classiquement divisée en blocs pour être compressée. Les techniques basées sur des transformations fréquentielles comme la corrélation de phase [3, 4] s'appuient sur la théorie mathématique de la transformée de Fourier. Le champ de vecteur résultant a alors des propriétés intéressantes : il correspond aux mouvements réels et est insensible aux variations de luminance. Cependant, la quantité de calculs engendrée est très importante. De plus l'estimation de mouvement a pour but de trouver le bloc le plus ressemblant dans une image de référence, et donc le mouvement réel n'est pas forcément le mieux adapté. Pour un estimateur de mouvement dédié à l'encodage vidéo, un algorithme de mise en correspondance de blocs (BMA : Block Matching Algorithm) est préféré.

2.1 Mise en correspondance de blocs

Les BMA consistent à rechercher pour chaque bloc de l'image courante le bloc qui lui ressemble le plus dans une image de référence. L'image courante est découpée en blocs de taille $M \times N$, puis pour chacun d'entre eux une mise en correspondance est effectuée. Une mesure de distance est alors calculée entre le bloc courant et un certain nombre de candidats. Cette mesure peut par exemple être la somme quadratique des différences (SSE : Sum of Square Errors), la somme des valeurs absolues des différences (SAD : Sum of Absolute Differences) ou un calcul prenant en compte plus précisément le coût de codage du résidu grâce à une transformée comme la SATD (Sum of Absolute Transformed Differences). Pour des raisons de facilité d'implantation et de coût de calcul, la SAD est souvent retenue. Le coût de codage des vecteurs peut aussi être pris en compte dans cette mesure par le biais d'un coefficient de lagrange pour résoudre le problème d'optimisation débit/distorsion [5].

L'algorithme de mise en correspondance le plus simple est la recherche exhaustive. Chaque déplacement possible d'amplitude maximale p est considéré comme candidat. C'est l'algorithme qui demande le plus de puissance de calcul. En effet, $(2p+1)^2$candidats sont considérés par bloc, induisant pour chacun une SAD $M \times N$ à calculer. Par exemple pour une image 720p (1280×720), des blocs de taille 8×8 et $p = 16$, cela abouti à 15,7 millions de SAD par image. La puissance de calcul nécessaire étant démesurée, plusieurs algorithmes rapides ont été proposés en utilisant principalement trois techniques d'optimisation.

La première consiste à calculer la SAD complète le moins souvent possible [6, 7]. En effet il est possible d'éliminer rapidement certains candidats avant même d'avoir effectué tous les calculs. Ces algorithmes réduisent énormément le nombre de calculs, néanmoins ils impliquent des opérations de conditionnement, coûteuses en temps de calcul et qui rend difficile l'optimisation de l'implantation. De plus, la contrainte temps réel nous conduit à considérer le pire cas d'exécution qui est alors moins bon que la technique de base.

La deuxième technique consiste à réduire le nombre de candidats et à orienter la recherche le plus tôt possible vers les candidats les plus probables. L'hypothèse principale est que la SAD croît lorsque l'on s'éloigne de l'optimum. Cet hypothèse n'est pas toujours vérifiée et conduit certains algorithmes à tomber dans des minima locaux. Chen et Al [8] proposent de rechercher dans une direction à la fois, alors que l'algorithme logarithmique proposé par Jain et Jain [9] et l'algorithme à trois pas de Koga et Linuma [10] font d'abord une estimation grossière puis raffinent le résultat. Dans [11] le mouvement est estimé grâce à une méthode récursive. A chaque étape, les SAD sont évaluées pour quelques candidats suivant un motif en diamant autour de la position courante. Le mouvement est raffiné successivement en suivant la direction de descente.

La troisième technique prend en considération le contenu des séquences vidéos à traiter. En effet il existe une certaine continuité du mouvement (spatialement et temporellement), c'est à dire que le mouvement d'un bloc a une grande chance d'être proche de celui d'un bloc voisin, ou du bloc colocalisé dans l'image précédente. Il est alors possible d'avoir un ensemble de prédicteurs du mouvement recherché grâce aux résultats déjà obtenus. La pertinence des différents prédicteurs est évaluée (en calculant la SAD avec le bloc courant) puis une recherche locale autour du (ou des) meilleur(s) prédicteur(s) est effectuée pour raffiner le résultat. De nombreux algorithmes utilisant cette technique ont été développés [12, 13, 14, 15, 16]. Ils diffèrent par le choix des prédicteurs et la technique de recherche locale.

Les algorithmes EPZS [12] et hiérarchique [17] sont particulièrement intéressants pour une implantation DSP. Ceux sont eux qui ont finalement été retenus, et qui sont détaillés plus précisément ci-dessous.

2.2 EPZS

L'algorithme EPZS (Enhanced Predictive Zonal Search) est une amélioration de l'algorithme PMVFAST [13] grâce à des nouveaux prédicteurs. La phase de prédiction est alors rendue plus précise et la recherche locale (figure 1-a)) est par conséquent réduite. Le raffinement grossier suivant un grand motif en diamant réalisé dans PMVFAST est devenu inutile. Le meilleur prédicteur est directement raffiné finement suivant un motif en diamant à 4 ou 8 connexités (figure 1-b)). L'amélioration de l'étape de prédiction réduit sensiblement le temps d'exécution.

Le seuil adaptatif, déjà utilisé dans PMVFAST, permet d'accélérer davantage le traitement en éliminant les calculs inutiles. La recherche est stoppée prématurément si le résultat est "assez bon", c'est à dire inférieur au seuil adaptatif. Le peu de calcul à réaliser font de cet estimateur un bon candidat pour une implantation logicielle temps réel.

Cet algorithme est utilisé actuellement dans les logiciels d'encodage vidéo tel que XviD et dans l'encodeur de

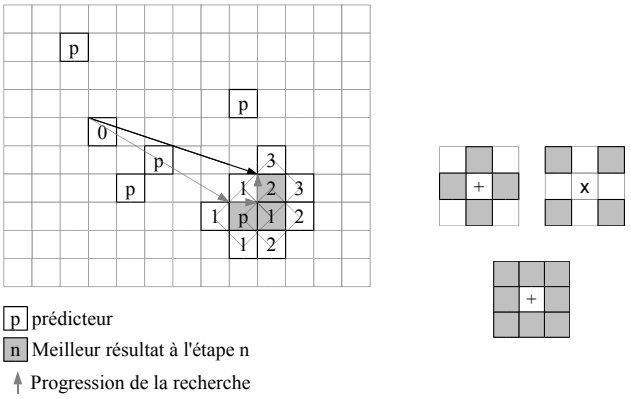

FIG. 1 – *EPZS*

référence H.264 JM pour sa vitesse d'exécution et la qualité de l'estimation.

2.3 HME

La méthode d'Estimation du Mouvement Hiérarchique (HME) [17] est basée sur un raffinement successif des vecteurs déplacement d'abord estimés grossièrement en sous-échantillonnant l'image.
L'algorithme débute par la construction du couple de pyramides d'images. Le niveau 0 correspond à l'image pleine résolution, l'image de niveau $n+1$ est obtenu en sous-échantillonnant l'image de niveau n d'un facteur 2 après l'application d'un filtre passe bas gaussien (figure 2).

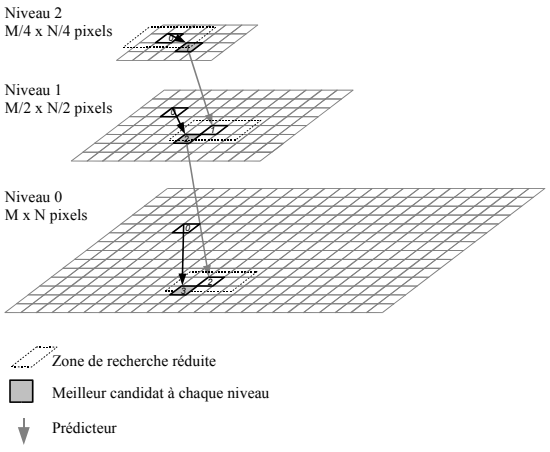

FIG. 2 – *Décomposition pyramidale d'une image*

L'estimation de mouvement est d'abord réalisée sur l'image basse résolution (niveau le plus élevé) puis le champs de vecteur est raffiné successivement. La taille des blocs à travers la pyramide reste constante de sorte que les petits détails n'influent pas sur les mouvements des niveaux supérieurs. Les détails et les mouvements des petits objets sont détectés à mesure que la résolution est augmentée.
Pour chaque niveau, l'estimation de mouvement est réalisée. On retrouve comme dans le cas de EPZS une phase de prédiction et une phase de raffinement. Les différents prédicteurs sont : le déplacement nul, les prédicteurs spatiaux, temporels et hiérarchiques. La phase de raffinement est une recherche exhaustive très réduite centrée autour du meilleur prédicteur. Cet algorithme est robuste grâce aux prédicteurs hiérarchiques qui permettent de détecter les mouvements de grande amplitude, bien que la recherche locale soit très réduite. Cet estimateur offre des performances comparables à un estimateur exhaustif en réduisant énormément le nombre de calculs.
Grâce au mécanisme prédictif et à une zone de recherche locale réduite, le champ de vecteur résultant des estimateurs HME et EPZS sont naturellement homogènes et donc de faible entropie. Ceci est un avantage non négligeable pour un encodeur vidéo.

2.4 Tailles de blocs variables

La norme H.264 réduit le débit des vidéo en ajoutant des modes de codage par rapport aux anciens standards. La compensation de mouvement peut être faite sur des blocs de taille variable. Les macroblocs 16×16 peuvent être divisé en 16×8, 8×16 ou 8×8 et les sous-partitions 8×8 peut être à nouveau divisées en 8×4, 4×8 ou 4×4. Le choix de petits blocs améliore la précision de la compensation de mouvement mais nécessite la transmission d'un plus grand nombre de vecteurs. Un algorithme de décision efficace doit donc être mis en place. Il peut faire partie intégrante de l'estimateur ou bien être séparément exécuté. Dans ce dernier cas l'estimateur de mouvement fourni les vecteurs pour toutes les tailles de blocs.
L'estimation de mouvement à taille de blocs variables peut être réalisée avec différentes approches. La première consiste à recopier le schéma d'estimation pour chaque taille de bloc. Cela permet de aisément de paralléliser les traitements mais ne permet pas d'exploiter la redondance des calculs.
La deuxième approche calcule un champ de vecteur pour une taille de bloc, ce qui permet d'initialiser une recherche beaucoup plus rapide pour les autres tailles [18].
Dans la suite du document, l'accent est porté sur la technique d'estimation pour une taille de bloc fixe. Les mouvement pour les autres tailles sont considérés déduits (cas de la deuxième approche). Les perfomances de l'estimateur global dépendent donc directement de celles de la première passe. De plus, comme l'intérêt est porté sur la qualité intrinsèque des champs de vecteurs, on cherche à s'affranchir de l'influence d'un algorithme de décision.

2.5 Précision quart de pixel

Dans H.264, la précision des vecteurs de mouvement atteint le quart de pixel. Un gain de compression significatif est apporté au prix d'une complexité plus élevée. La définition de l'image est augmentée en interpolant successivement au demi-pixel avec un filtre à 6 coefficients, puis

au quart de pixel avec un filtre linéaire.

La précision quart de pixel peut être obtenue de différentes manières ; la première est de rechercher directement dans l'image interpolée. Le nombre de candidats à prendre en compte se trouve donc augmenté, ou autrement dit, les dimensions horizontales et verticales de l'image sont quadruplées. La deuxième manière consiste à réaliser l'estimation de mouvement en plusieurs étapes [18] : dans un premier temps la recherche est effectuée à la précision pixel entier, puis le vecteur est raffiné au quart de pixel. Cette deuxième solution permet une interpolation de l'image "à la volée", c'est à dire que les positions demi et quart de pixels sont calculées uniquement lorsqu'elles sont utiles. Cela réduit la bande passante mémoire nécessaire mais augmente légèrement les calculs à effectuer. Cette dernière solution semble être un bon compromis pour une implantation DSP. De plus il est alors très aisé de rendre le raffinement subpixel facultatif et donc d'avoir une implantation évolutive.

Les vecteurs sont donc estimés avec un des algorithmes décrits précédemment au pixel entier près (avec HME ou EPZS), puis ils sont raffinés successivement au demi puis au quart de pixel.

3 Implantation temps réel sur DSP

L'implantation d'un estimateur de mouvement pour l'encodage MPEG-4 AVC/H.264 [19] haute définition représente d'énormes contraintes temps réel. Avec des spécificités telles que des tailles de blocs variables ou la précision quart de pixel des vecteurs de mouvement, l'opération d'estimation de mouvement à elle seule pose des problèmes de bande passante mémoire et de puissance de calcul. Dans cette section une implantation sur DSP d'un estimateur de mouvement pour des blocs de taille 8×8 est étudiée. Les algorithmes HME et EPZS sont comparés entre eux en termes de qualité et de temps d'exécution.

Nous ne détaillerons pas dans cet article la manière dont les optimisations d'implantation ont été effectuées. Nous pouvons toutefois préciser que les temps de traitement ont pu être divisés par 5 entre les versions originales et optimisées sur DSP.

3.1 Implantation de EPZS

L'algorithme EPZS implanté possède les caractéristiques suivantes. Les prédicteurs utilisés sont le vecteur nul, 1 prédicteur temporel et 4 prédicteurs spatiaux. L'arrêt anticipé permettant de stopper prématurément l'algorithme n'a pas été implanté de façon à obtenir un temps d'exécution plus constant, ce qui est indenspensable pour une implantation temps réel. De plus la précision des vecteurs s'en trouve améliorée.

La recherche locale est effectuée avec un motif carré (8 connexités). La qualité des vecteurs est alors meilleure au prix d'un temps d'exécution légèrement plus important par rapport à un motif en diamant à 4 connexités.

3.2 Implantation de HME

En plus des opérations d'estimation de mouvement proprement dites, l'implantation de l'algorithme hiérarchique prend en compte la construction de la pyramide d'images sous-échantillonnées. Chaque niveau est obtenu en appliquant un filtre gaussien et en sous-échantillonnant d'un facteur 2 l'image de niveau inférieur.

Le mécanisme de hiérarchie permet d'ajouter des prédicteurs (hiérarchiques) fiables par rapport à EPZS mais avec un coûp de calcul plus important.

3.3 Raffinement quart de pixel

Le point le plus contraignant du point de vue de l'implantation est sans doute le raffinement quart de pixel des vecteurs de mouvement. En effet les filtres d'interpolation sont très coûteux en terme de temps de calcul. Néanmoins le gain de compression apporté est significatif. Son implantation est donc nécessaire.

Le raffinement subpixel est implanté de la même façon, que l'algorithme d'estimation de mouvement choisi soit EPZS ou HME. Ce dernier fournit la meilleure position pixel qui est ensuite raffinée au demi-pixel en interpolant la zone utile de l'image grâce au filtre à 6 coefficients utilisé dans la norme H.264. Comme l'opération d'estimation de mouvement n'est pas normée (contrairement à la compensation) un filtre plus simple permettrait d'accélérer les calculs, cependant la qualité serait inférieure. Lorsque les valeurs demi-pixel sont disponiles, la meilleure des 8 positions demi-pixel adjacentes à la position pixel est retenue en évaluant les SAD. Ensuite le vecteur est raffiné de même au quart de pixel en appliquant un filtre d'interpolation linéaire à l'image demi-pixel.

4 Résultats

Les deux estimateurs de mouvement EPZS et HME ont été implémentés sur un DSP et intégrés dans un encodeur vidéo H.264. L'estimation de mouvement est limitée à des tailles de blocs 8×8 pour pouvoir comparer la qualité des champs de vecteurs fournis.

4.1 Temps d'exécution

Les algorithmes EPZS et HME ont été portés et optimisés sur un DSP Texas Instrument TMSC6416 à 1Ghz. Le tableau 1 donne les différents temps d'exécution obtenus pour des images progressives SD (720x576) et HD (1280x720). Pour chaque algorithme, deux versions ont été envisagées : avec ou sans raffinement quart de pixel.

Dans la version quart de pixel, EPZS est deux fois plus rapide que HME. Ceci est principalement dû à la création de la pyramide d'images et à l'estimation de mouvement des niveaux hiérarchiques, inexistants dans EPZS. Le plus grand nombre de prédicteur pour HME et une recherche locale plus intensive contribuent également à augmenter légèrement le temps d'exécution.

Une implantation de EPZS au quart de pixel est donc temps réel (30 images par secondes) sur un DSP pour de la

Algorithme	SD 720x576	HD 1280x720
HME $\frac{1}{4}$-pel	30 ms	60 ms
EPZS $\frac{1}{4}$-pel	16 ms	32 ms
HME pel	21 ms	41 ms
EPZS pel	7 ms	13 ms

TAB. 1 – *Temps d'exécution des différents algorithmes*

haute définition. Pour implémenter HME en temps réel, on peut envisager un pipeline de 2 DSP (un pour les niveaux hiérarchiques, et un pour la pleine résolution) capable de traiter la vidéo HD à 25 images par secondes.

4.2 Qualité d'estimation

Pour évaluer la qualité des champs de vecteurs fournis par les estimateurs de mouvement, ceux-ci ont été implantés dans l'encodeur vidéo H.264 développé à THOMSON Corporate Research. Seuls les blocs de taille 8×8 et le mode de codage inter sont considérés. Ceci permet de comparer uniquement la qualité des champs de vecteurs.
Les figures 3, 4 et 5 donnent les courbes débit/distorsion correspondant à l'encodage des 200 premières images de différents types de séquences. Pour chaque séquence les courbes de qualité (donnée en PSNR moyen) correspondant à chaque estimateur sont tracées en fonction du débit moyen.

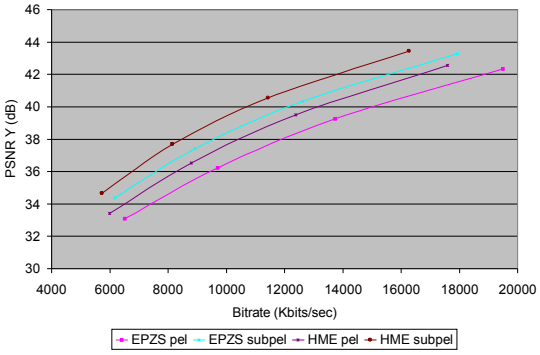

FIG. 3 – *Séquence HD*

La première séquence de test (figure 3) est une succession rapide de différente scènes en haute définition (1280x720). L'algorithme hiérarchique atteint une qualité largement supérieure du fait de la détections des mouvements de grande amplitude offert par les prédicteurs hiérarchiques. Les prédicteurs temporels de EPZS, naturellement inadaptés à chaque début de scène sont une autre explication de cette qualité inférieure. En effet lors d'un changement de scène les vecteurs ne sont pas fiables. Ils ne le sont pas non plus en tant que prédicteurs temporels pour l'image suivante. On voit aussi clairement que le raffinement $\frac{1}{4}$-pixel des vecteurs augmente les performances de l'encodeur.

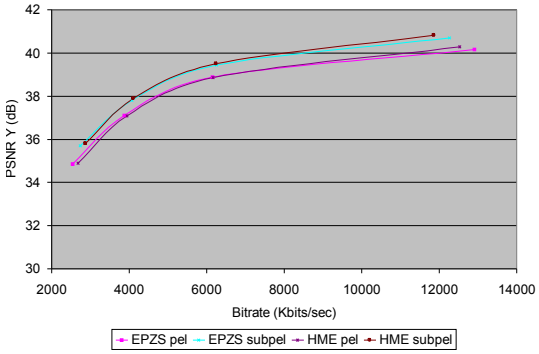

FIG. 4 – *Séquence Hockey HD*

La deuxième séquence de test (figure 4) représente un entrainement de hockey sur glace en haute définition (1280x720). Les mouvements sont assez homogènes et il y a peu de changements de scène. Les performances des estimateurs HME et EPZS sont donc comparables. Le raffinement $\frac{1}{4}$-pixel des vecteurs augmente toujours les performances de l'encodeur.

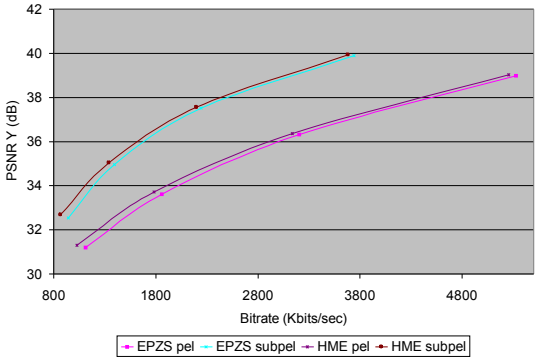

FIG. 5 – *Séquence Raid2Maroc SD*

La dernière séquence de test (figure 4) représente des scènes de sport en définition standard (720x576). Il y a quelques changements de scène mais la réduction de la définition par rapport à la première séquence permet de réduire l'écart des performances entre les deux estimateurs. Ici encore, le raffinement $\frac{1}{4}$-pixel des vecteurs semble indispensable pour offrir de bonnes performances.
Les deux estimateurs sont de qualité équivalentes pour de nombreuses séquences lorsque le mouvement reste faible ou homogène. EPZS fournit un champ de vecteur de qualité inférieure à HME lorsque la séquence contient beaucoup de mouvements. Son temps d'exécution largement inférieur en fait cependant un bon candidat pour une implantation temps réel peut coûteuse.

5 Conclusion

Un état de l'art des méthodes d'estimation de mouvement a été dressé. Les techniques HME et EPZS, intéressantes

en terme de qualité et de vitesse d'exécution ont été plus précisément décrites et implantées sur DSP Texas Instruments C64x à 1Ghz. Leurs performances dans un encodeur MPEG-4 AVC/H.264 ont été évaluées.
L'estimateur EPZS est temps réel (30 images/s) pour des images hautes définition 720 p (1280x720 progressif) au quart de pixel. HME est deux fois plus lent.
Les performances des deux estimateurs dans un encodeur vidéo sont équivalentes sur des séquences comportant peu de mouvement ou des mouvements homogènes (spatialement et temporellement). Pour des séquences plus complexes, HME est plus robuste. Le gain de codage apporté par le raffinement des vecteurs au quart de pixel semble justifier son implantation, malgré un coût de calcul beaucoup plus élevé.
Dans le cas d'un estimateur de mouvement à multiples tailles de bloc, le surcoût apporté par HME est réduit. En effet, celui-ci est dû principalement à la construction de la pyramide et à l'estimation de mouvement pour les niveaux hiérarchiques. Or leur dupliquation n'est pas nécessaire pour un estimateur à taille de blocs variable. De plus les résultats des niveaux hiérarchiques intermédiaires permettent d'initialiser une recherche rapide pour chaque taille de bloc.
On peut donc envisager le prototypage d'un estimateur de mouvement temps réel pour l'encodage MPEG-4 AVC/H.264 avec un DSP pour chaque taille de bloc dans le cas de EPZS et un DSP supplémentaire pour les niveaux hiérarchiques dans le cas de HME. Le nombre de DSP nécessaire peut être réduit en ajoutant de la décision dans l'estimateur de mouvement afin de choisir la taille de bloc et de calculer le vecteur associé.

Références

[1] Iain E.G.Richardson. *H.264 and MPEG-4 Video Compression : Video Coding for Next-generation Multimedia*. John Wiley and Sons, 2003.

[2] A. N. Netravali et J. D. Robbins. Motion-compensated television coding. I. *AT T Technical Journal*, 58 :631–670, mar 1979.

[3] G.A. Thomas. Television motion measurement for DATV and other applications. Rapport technique, BBC RD, 11 1987.

[4] R. Storey. HDTV Motion Adaptive Bandwidth Reduction using DATV. Rapport technique, BBC RD, 1986.

[5] G. Sullivan et T. Wiegand. Rate-Distortion Optimization for Video Compression. *IEEE Signal Processing Magazine*, pages 74–90, Nov 1998.

[6] Salari E. Li W.. Successive elimination algorithm for motion estimation. *IEEE Transactions on Image Processing*, 4 :107–110, 1995.

[7] Y-P Hung Y-S Chen et C-S Fuh. Fast Block Matching Algorithm Based on the Winner-Update Strategy. Dans *IEEE Transactions on Image Processing*, volume 10, August 2001.

[8] T.D. Chiueh M.J. Chen, L.G. Chen. One-dimensional full search motion estimation algorithm for video coding. *IEE Transactions on Circuits and Systems for Video Technology*, 4 :504–509, 1994.

[9] J. R. Jain et A. K. Jain. Displacement measurement and its application in interframe coding. *IEEE Transactions on Communications*, COM-29(12) :1799–1808, 1981.

[10] A.Hirano Y.Iijima T.Koga, K.Linuma et T.Ishiguro. Motion compensated interframe coding for video conferencing. *Proceedings of National Telecommunication Conference*, NTC81 :G5.3.1–G5.3.5, 1981.

[11] M. Ranganath J. Y. Tham, S ; Ranganath et A. A. Kassim. A Novel Unrestricted Center-Biased Diamond Search Algorithm for Block Motion Estimation. *IEEE Transactions on circuits and systemes for video technology*, 8, NO. 4, AUGUST 1998.

[12] Alexis Michael Tourapis. Enhanced Predictive Zonal Search for Single and Multiple Frame Motion Estimation. *proceedings of Visual Communications and Image Processing*, pages 1069–79, 2002.

[13] Alexis Michael Tourapis, Oscar C. Au, et Ming Lei Liou. Predictive Motion Vector Field Adaptive Search Technique (PMVFAST). Dans *Proceedings of Visual Communications and Image Processing 2001 (VCIP'01)*, 2001.

[14] P. Zhou Z. Chen et Y. He. Fast motion estimation for JVT. JVT-G016.doc, March 2003.

[15] Lappui Chau Ce Zhu, Xiao Lin et Lai-Man Po. Enhanced Hexagonal Search for Fast Block Motion Estimation. *IEEE Transactions on circuit and systems for video technology*, 14 :1210–1214, OCTOBER 2004.

[16] S. Masud F. Nasim S. Idris K. Virk, N. Khan. Low Complexity Recursive Search Based Motion Estimation Algorithm for Video Coding Applications. Dans *Proceedings of 13th European Signal Processing Conference*, Antalya, Turkey, 2005.

[17] B. Chupeau, P. Robert, M. Pecot, P. Guillotel. Multiscale motion estimation. *Workshop on Advanced Matching in Vision and Artificial Intelligence*, Munich, 5th, 6th June 1990.

[18] Jechang Jeong Woong IL Choi, Byeungwoo Jeon. Fast motion estimation with modified diamond search for variable motion block sizes. Dans *International Conference on Image Processing*, volume 2, pages 371–4, Sept. 2003.

[19] Joint Video Team of ITU-T et ISO/IEC 14496-10. Draft of version 4 of H.264/AVC. Rapport technique, Nov 2004.