Codage efficace de contour avec perte utilisant la consistance spatio-temporelle

Marc CHAUMONT, Stéphane PATEUX, Henri NICOLAS,

IRISA/INRIA Campus de Beaulieu, 35042 Rennes, France
Marc.Chaumont@irisa.fr,
Stephane.Pateux@irisa.fr, Henri.Nicolas@irisa.fr

Résumé – Ce papier traite du problème de codage vidéo de forme avec perte. Ce problème appartient au domaine du codage vidéo basé objets. Notre approche propose d'exploiter la corrélation spatio-temporelle, en considérant l'évolution du contour et en traitant le phénomène d'occultation. La nouveauté réside dans une mise en correspondance fine des contours ainsi que dans le prolongement des parties occultées. Une fois que les contours sont traités, une transformation ondelette avec un schéma IPB est utilisée pour les encoder. Les résultats expérimentaux montrent une amélioration significative à très faible débit. De plus, le train binaire est totalement progressif.

Abstract – This paper deals with the problem of lossy video object shape coding. This problem belongs to the area of video object-based coder. Our approach proposes to exploit spatio temporal correlation. This is done by considering the contour evolution and dealing with occultation phenomenon. The novelty lies in a fine mapping of consecutive contours and in the contour padding in occulted parts. Once a group of contours is processed, wavelet transformations with a predictive IPB (Intra, simple Prediction, Bidirectional prediction) scheme is used to encode it. Experimental results show significant improvement at very low bit rate. Moreover, the bitstream is fully progressive.

1 Introduction

Dans les codeurs vidéo basés objets tels que MPEG4 [1], la forme des objets vidéo est à coder. Plusieurs techniques ont été proposées pour diminuer le coût de ce codage [2][3]. Généralement, on classe les techniques de codage de forme en deux catégories, le codage basé pixels et le codage basé contours.

Les approches de codage basées pixels (MMR, CAE, quad-arbre...) reposent sur un codage d'information binaire. Par exemple, la technique Context Arithmetic Encoder (CAE) [4], met en œuvre un codeur arithmétique pour coder chaque pixel sachant le contexte voisin. Certaines de ces approches prennent en compte la redondance temporelle. Par exemple, MPEG4 CAE a développé une solution basée BABs (Binary Alpha Blocks), CAE et prédiction temporelle. MPEG4 CAE est actuellement la solution de référence pour le codage de forme avec ou sans perte. Toutefois, le problème de cette solution est l'effet de bloc à bas débit, inhérent à la modélisation adoptée.

Les approches basées contours (Freeman, polygonal, spline ...) décrivent uniquement le contour. La technique de Freeman code une suite de déplacements. Les approches polygonale et spline définissent les contours au moyen de segments de ligne ou respectivement au moyen de points de contrôle.

Yoshida et al. [5] ont proposé une solution avec perte basée contour qui égale les performances de MPEG4 CAE à faible débit. Ils prennent en compte l'information temporelle en codant plusieurs contours consécutifs simultanément. Notre approche est inspirée par leur solution. La principale nouveauté est l'amélioration de la stabilité temporelle en travaillant uniquement sur les "vrais" contours des objets, c'est-à-dire en ne prenant en compte que les parties du contour qui appartiennent à l'objet (voir figure 1). De plus, une meilleure mise en correspondance temporelle exploitant la compensation de mouvement est proposée.

(a) Segmentation initiale

(b) Contour extérieur

FIG. 1: Extraction du contour apparent d'un objet vidéo

2 Alignement des contours et prolongement spatio-temporel

Notre objectif est d'obtenir une mise en correspondance des points du contour au cours du temps. Cette mise en correspondance permet alors de représenter l'évolution du contour par deux plans spatio-temporels donnant la position (x, y) d'un point du contour sachant l'indice s sur ce contour et le numéro de l'image t (voir figure 5).

Les différentes étapes pour obtenir les deux plans spatio-temporels sont : (1) extraction des contours, (2) mise en correspondance des contours consécutifs et alignement de ceux-ci et (3) prolongement spatio-temporel dans le cas de contours ouverts.

2.1 L'extraction du contour sur plusieurs images

Une forme est composée de contours internes (« trou » à l'intérieur de l'objet) et externes (forme globale de l'objet).

La connaissance des parties valides (segment qui n'est pas dû à une occultation) de l'enveloppe externe d'un objet est déduite grâce à l'ordre de profondeur associé à chaque segment composant l'enveloppe [6].

Ainsi, nous décrivons une forme par une liste de positions

extraite à partir du contour externe. Cette représentation possède éventuellement des « ruptures » puisque le contour peut être partiellement occulté.

Une fois que les listes de position sont obtenues (une par image), les listes sont projetées vers un temps de référence en utilisant le mouvement de la texture. Cette projection facilite le processus de mise en correspondance des contours et renforce la stabilité temporelle de notre représentation. Le fait de considérer le mouvement de texture et non celui de contour permet de décorréler mouvement et forme; cela rend possible un codage pleinement progressif pour chacune de ces deux informations [7]. L'estimation du mouvement est réalisée par maillage actif [8].

2.2 Alignement de contour et sur-échantillonnage

Après mise en correspondance deux à deux des contours (par programmation dynamique par exemple), il est nécessaire de sur-échantillonner l'ensemble des contours pour pouvoir avoir une correspondance bijective entre tous les points de tous les contours. Pour cela, des points « virtuels » sont ajoutés sur chaque contour.

Pour résoudre le problème d'insertion de points, on introduit la notion d'abscisse universelle. C'est un nombre entier donné à chaque position d'un contour. Ce nombre est le même pour les points appartenant à une même trajectoire. Le parcours d'un contour voit ce nombre augmenter. Son domaine de définition est borné et totalement couvert.

Une fois que chaque contour possède une correspondance vers l'abscisse universelle, les points « virtuels » sont ajoutés partout où la valeur de l'abscisse universelle manque (voir figure 2).

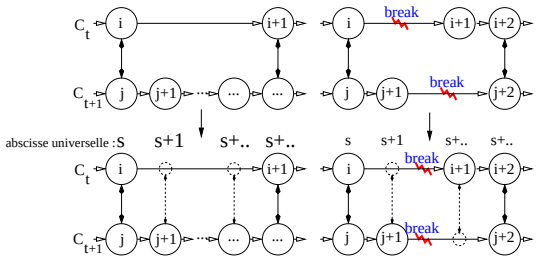


FIG. 2: Alignement du groupe de contour par le calcul d'une abscisse universelle. La notion d'abscisse universelle permet d'ajouter des points «virtuels» (sur-échantillonage). Les points «virtuels» sont représentés par les cercles en pointillés

2.3 Prolongement spatio-temporel de contour

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

Dans un premier temps, nous ajoutons des points « virtuels » pour fermer les contours (le nombre de points ajoutés est fonction de la distance entre les points extrémités de la « rupture »). La figure 3 montre l'ajout de points « virtuels » quand il y a une « rupture » de contour. Les points « virtuels » seront représentés par l'ensemble Ω_{Out} tandis que les points originaux seront représentés par l'ensemble Ω_{In}.

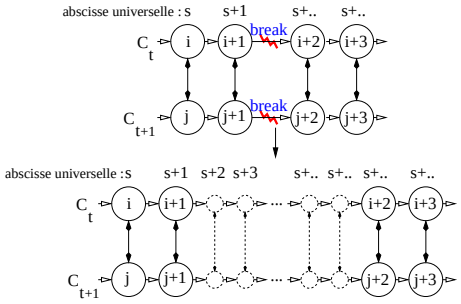


FIG. 3: Ajout de point «virtuels» pour fermer les contours ouverts

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

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

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

On exploite la représentation hiérarchique de $\overline{C}^L(s)$ en proposant un calcul itératif consistant à raffiner le signal par une succession de phases d'analyse et de synthèse aux différentes résolutions. La figure 4 montre l'évolution du signal moyen $\overline{C}^L(s)$ lors des itérations succesives.

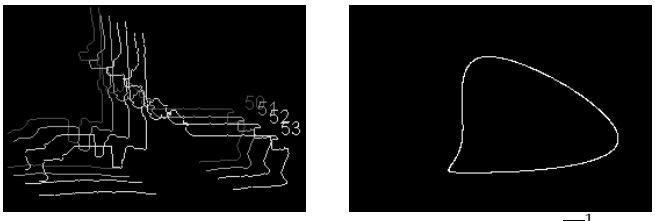
La phase de synthèse permet de mettre à jour le signal moyen $\overline{C}^l(s)$ pour un niveau l grâce au signal de raffinement $\Delta C^l(s)$. Le signal de raffinement $\Delta C^l(s)$ est trouvé durant la phase d'analyse par calcul des coefficients incrémentaux δc_k^l. La mise à jour du signal moyen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

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

et on définit un résidu sur Ω_{In} :

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

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



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

(c) Signal moyen $\overline{C}^2(s)$ (d) Signal moyen $\overline{C}^3(s)$

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

FIG. 4: Signal moyen $\overline{C}^L(s)$ à différents niveaux L

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

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

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

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

3 Codage spatio-temporel de forme d'objet

3.1 Le schéma IPB

Une fois que les contours ont été alignés et prolongés, nous possédons une surface d'évolution du contour (les deux plans spatio-temporels) paramétrés par s (l'abscisse universelle) et t (le temps) (voir figure 5).

Deux méthodes de paramétrisation sont comparées pour encoder la surface spatio-temporelle du contour. La première est basée sur les B-splines et la seconde sur une décomposition

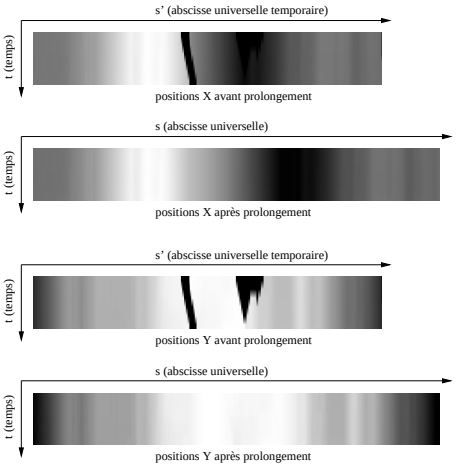


FIG. 5: Plans spatio-temporels pour les positions X et Y. Les zones noires représentent les «ruptures» de contour présentes avant le prolongement de contour

en ondelettes. De plus, les contours sont codés en utilisant un schéma de type IPB. Le premier contour sera codé en intra (I) et les autres seront codés en utilisant une simple prédiction (P) ou une prédiction bidirectionnelle (B). Une seule image B est insérée entre deux images I ou P. Les coefficients sont codés avec un codeur arithmétique en plans de bits. La quantification avec perte est obtenue en choisissant le nombre de plans de bits encodés.

3.2 La représentation en B-spline

Un contour peut être exprimé par :

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

où les P_k sont les points de contrôle, les ϕ sont les noyaux d'une B-spline, s est l'abscisse universelle et s_k est l'abscisse universelle correspondant au P_k.

Les points de contrôle sont calculés de sorte que la B-spline représente au mieux le contour original. On les obtient par une approche par minimisation :

$$\min_{P_k} \sum_{i=1}^{i=I} \|B(s_i) - C(s)\|^2.$$

Cette minimisation mène à un système linéaire qui peut être résolu de manière simple par un outil classique d'algèbre linéaire (par exemple par gradient conjugué).

3.3 La représentation en ondelette

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

4 Résultats

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

La figure 6 montre la distortion en fonction du débit par élément de contour pour la séquence Foreman. Il faut remarquer tout d'abord que les résultats, pour un codage avec perte, ne présentent un intérêt que pour des débits inférieure à 1,2 bits/élément de contour. En effet, un codage de Freeman avec codeur arithmétique permet d'atteindre ce débit sans perte.

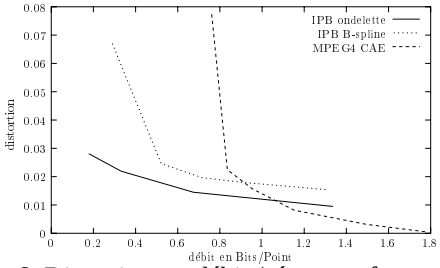
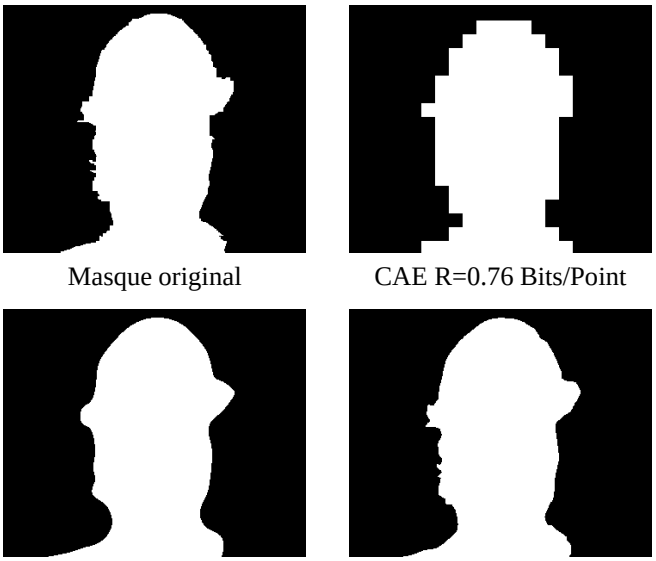


FIG. 6: Distortion vs. débit (séquence foreman)

En utilisant l'approche IPB B-spline ou ondelette, il est possible de descendre à des débits très bas (0,4 bits/élément de contour) alors que MPEG4 CAE est borné environ à 0,8 bits/élément de contour. De plus, comme illustré sur la figure 7, à très faible débit (autour de 0,7 bits/élément de contour), la qualité est toujours acceptable, ce qui n'est pas le cas pour MPEG4 CAE.



Masque original — CAE R=0.76 Bits/Point

B-spline R=0.72 Bits/Point — Ondelette R=0.67 Bits/Point

FIG. 7: Comparaison des techniques de codage pour un débit d'environ 0.7 bits par élément de contour

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

Le tableau 1 illustre le gain obtenu sur MPEG4, à faible débit, en utilisant notre schéma. 30% à 60% peuvent être gagnés sur le débit, à un niveau de distortion acceptable tout en offrant une représentation progressive.

TAB. 1: Gain de l'approche IPB ondelette sur MPEG4

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

5 Conclusion

Dans ce papier, nous proposons une technique efficace de codage de forme. Celle-ci tire bénéfice d'une mise en correspondance améliorée, de l'alignement de contours ainsi que du prolongement spatio-temporel. En utilisant un codage ondelette et un schéma IPB nous obtenons de meilleurs résultats que MPEG4 CAE à faible débit et grâce aux propriétés spatio-temporelles intrinsèques, l'aspect visuel est acceptable. Nos futurs travaux traiterons de l'intégration de notre codeur de forme dans un codeur objet pleinement progressif (forme, mouvement, texture) [7].

Références

[1] ISO. Draft MPEG-4, "Video verification model version 8.0.," *ISO/IEC JTC1/SC29/WG11*, 1997.

[2] C. Le Buhan Jordan, F. Bossan, and T. Ebrahimi, "Scalable shape representation for content-based visual data compression," *International Conference on Image Processing, ICIP'1997*, vol. 1, pp. 512–515, Oct. 1997.

[3] A. Katsaggelos, L. Kondi, F. Meier, J. Ostermann, and G. Schuster, "Mpeg-4 and rate-distortion-based shape-coding techniques," *IEEE Proc., special issue on Multimedia Signal Processing*, vol. 86, no. 6, pp. 1126–1154, Juin 1998.

[4] N. Brady, F. Bossen, and N. Murphy, "Context-based arithmetic encoding of 2d shape sequence," *International Conference on Image Processing, ICIP'1997*, vol. 1, pp. 29–32, Oct. 1997.

[5] T. Yoshida, T. Asami, and Y. Sakai, "Compression of moving contours on spatio-temporal 2-d plane," *International Conference on Image Processing, ICIP'1998*, vol. 1, pp. 276–280, Oct. 1998.

[6] L. Bonnaud and C. Labit, "Multiple occluding objects tracking using a non-redundant boundary-based representation for image sequence interpolation after decoding," *International Conference on Image Processing, ICIP'1997*, vol. 2, pp. 426–429, Oct. 1997.

[7] M. Chaumont, N. Cammas, and S. Pateux, "Fully scalable object based video coder based on analysis-synthesis scheme," *International Conference on Image Processing, ICIP'2003*, Sept. 2003.

[8] G. Marquant, S. Pateux, and C. Labit, "Mesh and "crack lines": Application to object-based motion estimation and higher scalability," *International Conference on Image Processing, ICIP'2000*, vol. 2, pp. 554–557, Sept. 2000.

[9] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, "Image coding using the wavelet transform," *IEEE Trans. Image Processing*, vol. 1, no. 2, pp. 205–220, Fev. 1992.