# Segmentation temporelle de séquences d'images en couleurs compressées et non compressées en temps réel

Sébastien Lefèvre[1,2], Jérôme Holler[1], Nicole Vincent[1]
lefevre@univ-tours.fr, vincent@univ-tours.fr

[1]Laboratoire d'Informatique
Université de Tours / E3i
64, Avenue Jean Portalis
37200 Tours

[2]Atos Services
19, rue de la Vallée Maillard
BP 1311 - 41013 Blois Cedex

**Résumé.** Nous présentons ici une nouvelle méthode pour segmenter dans le domaine temporel des séquences d'images compressées et non compressées. Les séquences d'images traitées contiennent des images en couleurs et ont été acquises à l'aide d'une caméra en mouvement. La détection de changements de plan joue le rôle d'une étape de prétraitement pour le suivi d'objet et doit donc pouvoir s'exécuter en temps réel. Afin de satisfaire cette contrainte, des images à faible résolution sont utilisées. Les effets dus aux changements d'illumination sont limités grâce à un changement d'espace couleur, depuis RGB vers HSV. On utilise alors les composantes H et S pour comparer deux images successives. La détection des *cuts* et des autres effets est basée sur l'étude de l'évolution temporelle de la différence entre deux images successives et la comparaison de cette différence avec un seuil adaptatif.

**Mots-clés :** indexation multimédia, détection des changements de plan, temps réel.

# 1 Introduction

Les données multimédia prennent une part de plus en plus importante dans les applications actuelles. La plupart de ces applications nécessitent des outils pour segmenter des séquences d'images dans le domaine temporel. Cette segmentation est souvent effectuée par une détection des *cuts* (transitions brusques) ou d'autres effets plus complexes (fondu, volet...). La segmentation est une étape cruciale dans des applications telles que la gestion de bases de données multimédia ou la création automatique de résumés de séquences télévisées ou de films. Elle peut être aussi utilisée comme prétraitement pour le suivi d'objet en temps réel dans des scènes dynamiques (acquises avec une caméra en mouvement).

Nous nous intéressons particulièrement au dernier type d'application, c'est-à-dire le suivi d'objet en temps réel. La détection des changements de plan peut être considérée comme un prétraitement ou une aide au suivi d'objet. En effet, si l'objet suivi par l'algorithme est perdu, il est nécessaire de déterminer si la perte est momentanée ou non. La détection d'un changement de plan se traduira par la perte définitive de l'objet suivi. Dans le cas contraire, la situation de perte peut être liée à des phénomènes d'occultation temporaire. La contrainte de temps réel imposée à l'algorithme de suivi concerne donc aussi l'algorithme de segmentation temporelle.

Nous rappellerons tout d'abord quelques méthodes classiques de détection des changements de plan. La méthode proposée sera ensuite présentée, et les principales étapes de l'algorithme seront détaillées. Enfin, la robustesse et l'efficacité de notre méthode seront évaluées sur la base de résultats obtenus à partir de séquences d'images issues de retransmissions de matchs de football.

## 2 Travaux antérieurs

Les méthodes présentes dans la littérature peuvent être principalement regroupées en trois catégories utilisant respectivement les différences pixel à pixel, la comparaison d'histogrammes, et l'estimation de mouvement. Ces différentes catégories sont présentées ci-dessous. Pour un état de l'art plus complet, le lecteur pourra se référer à [1], [2], [3]. Il est important de noter que les méthodes peuvent traiter des séquences vidéo non compressées [4] ou compressées [5].

Les méthodes basées sur les différences pixel à pixel détectent un changement de plan en calculant une différence entre les pixels de l'image à l'instant $t$ et ceux de l'image à l'instant $t+1$. Si le nombre de pixels différents est supérieur à un seuil, alors on considère que l'on est en présence d'un *cut* ou d'un autre effet. Le défaut principal de ces méthodes est leur manque de robustesse au bruit et aux forts mouvements pouvant être présents dans la scène.

Les méthodes à base d'histogramme comparent deux images successives en s'appuyant sur leurs histogrammes respectifs. Une différence des deux histogrammes est calculée et comparée à un seuil. Comme dans le cas des méthodes basées sur les différences pixel à pixel, un changement de plan est détecté si la différence obtenue est supérieure au seuil. Du fait de l'utilisation d'histogrammes, il est possible de ne pas détecter un *cut* si les deux images concernées ont un histogramme similaire mais un contenu différent.

Les méthodes basées sur une estimation du mouvement utilisent l'information de mouvement comme critère principal pour la détection des changements de plan [6]. Les mouvements sont estimés pour chaque pixel d'une image obtenue à l'instant $t$, et sont comparés avec ceux de l'image correspondant à l'instant $t+1$. Un nombre trop important de mouvements incohérents entre les deux images successives implique alors la détection d'un changement de plan. Ces méthodes ne sont souvent pas capables de traiter des séquences vidéo en temps réel, du fait des ressources nécessaires pour les estimations de mouvement.

Même si les différentes méthodes présentées précédemment ont leurs propres avantages et inconvénients, elles utilisent toutes des seuils fixés la plupart du temps de manière empirique. Ces seuils nécessitent des paramétrages spécifiques au type de séquence vidéo analysée, et ne permettent donc pas une acquisition de données non contrainte ni une méthode générique, c'est-à-dire indépendante du domaine des séquences vidéo traitées.

## 3 Méthode proposée

La méthode présentée ici a pour but d'analyser en temps réel des séquences dynamiques d'images en couleurs. Nous avons élaboré une méthode capable de traiter aussi bien des séquences vidéo compressées que non compressées. Dans une première section sera décrite l'étape qui nous permet d'obtenir des images à faible résolution. Celles-ci peuvent être créées en utilisant les coefficients DC dans le cas de données compressées, ou grâce à une représentation par blocs pour les données non compressées. La diminution de la résolution spatiale des images joue un rôle particulièrement important dans la réduction du temps de calcul.

Nous détaillerons ensuite les trois étapes de notre méthode. Tout d'abord, le problème du choix de l'espace couleur sera résolu afin de limiter les effets d'illumination, puis la

mesure de distance utilisée dans notre méthode pour comparer deux images successives sera présentée. La dernière étape correspond à la détection des *cuts* et des autres effets. Elle est basée sur l'évolution temporelle des différences entre images successives et l'utilisation d'un seuil adaptatif.

## 3.1 Images à faible résolution

Les méthodes travaillant au niveau du pixel sont sensibles au bruit et sont souvent caractérisées par des temps de calcul importants. Afin d'éviter ces inconvénients, nous construisons des images à faible résolution. Ces images sont obtenues de deux façons, selon le type des données d'entrée (compressées ou non).

Les séquences vidéo compressées telles que MPEG ou MJPEG utilisent des coefficients DC. Chaque bloc obtenu après utilisation de la transformée en cosinus discrète (DCT) est caractérisé par un coefficient DC et plusieurs coefficients AC. Nous utilisons ici les coefficients DC pour calculer l'intensité moyenne de chaque bloc. Il est donc possible de reconstruire des images à faible résolution, selon la méthode présentée dans [7]. Les images sont reconstruites en utilisant une valeur pour chaque bloc de taille $8 \times 8$ pixels.

Pour les séquences vidéo non compressées, on définit $n$ blocs de taille $8 \times 8$ pixels. La valeur moyenne des pixels composant chaque bloc est alors calculée et utilisée pour créer l'image à faible résolution.

Dans les deux cas, nous obtenons finalement des images réduites dans le domaine spatial. Ces images sont en fait des ensembles de trois images, une pour chaque composante couleur de l'espace RGB. Après avoir réduit la résolution spatiale pour améliorer la robustesse au bruit et diminuer la taille des données, nous réduisons la résolution sur la couleur.

## 3.2 Changement de l'espace couleur et définition de la mesure de distance

Lorsque qu'une méthode générale est élaborée, il est nécessaire de se préoccuper de sa sensibilité à l'éclairement de la scène, pouvant être plus ou moins intense selon le facteur d'illumination. Nous cherchons donc un espace de représentation couleur plus robuste aux changements d'illumination que l'espace d'origine RGB. Nous avons sélectionné l'espace HSV (*Hue, Saturation, Value* ou Teinte, Saturation, Valeur). Une fois la conversion depuis l'espace RGB vers l'espace HSV effectuée, nous ne conservons que les composantes $H$ et $S$, ce qui nous permet d'éliminer l'effet de l'illumination représenté par la composante $V$. De plus, nous avons de nouveau diminué la taille des données à traiter. Finalement, pour chaque pixel $i$ d'une image obtenue à l'instant $t$, nous calculons le vecteur $P$ à l'aide de $H$ et $S$ comme le montre l'équation (1) :

$$P_i(t) = \begin{pmatrix} H_i(t) \\ S_i(t) \end{pmatrix} \tag{1}$$

On peut donc considérer $P$ comme un vecteur caractéristique de l'état d'un pixel avec la particularité d'invariance à l'illumination. Ce calcul nous permet de diminuer la dimension de l'espace de représentation des pixels, diminuant ainsi la taille des données à traiter, toujours dans un but de respect des contraintes de temps réel.

Les changements de plan sont détectés en calculant une différence entre deux images. Dans notre cas nous devons comparer deux images à faible résolution. Afin d'estimer la dissimilarité entre deux images, qui est liée à la probabilité de détection d'un changement de plan, nous définissons la mesure de distance suivante :

$$d(t_1, t_2) = \sum_i \left( (P_i(t_1) \ominus P_i(t_2))^n \right) \tag{2}$$

avec $n \in [0, 2]$ et la convention $0^0 = 0$. La valeur de $n$ nous permet de donner plus ou moins d'importance à la différence $P_i(t_1) \ominus P_i(t_2)$ : si on choisit $n = 0$, une faible différence entre deux pixels consécutifs dans le domaine temporel aura la même influence qu'une forte différence, contrairement au cas où $n = 2$ qui augmente l'influence de la différence entre deux pixels.

L'opérateur $\ominus$ utilisé pour comparer deux vecteurs $P$ est défini comme :

$$P_i(t_1) \ominus P_i(t_2) = \alpha \ (H_i(t_1) - H_i(t_2)) \pmod{2\pi} + (1 - \alpha) \ |S_i(t_1) - S_i(t_2)| \tag{3}$$

où $\alpha$ doit être fixé empiriquement. Si l'on souhaite minimiser la perte d'information, il est possible d'utiliser des critères tels que celui d'Akaike [8].

Cette approche est bien adaptée aux contraintes temps réel, où il est intéressant de ne traiter qu'un volume minimal de données. Nous obtenons ici une seule valeur pour chaque différence entre deux images, ce qui implique de faibles temps de calcul. L'utilisation d'une mesure de distance plus complexe n'apporterait pas une quantité d'information supplémentaire permettant de compenser le surcoût en terme de temps de calcul.

Il est maintenant possible d'étudier l'évolution de la différence entre deux images successives tout au long de la séquence vidéo afin de détecter et discriminer les changements de plan.

## 3.3 Evolution temporelle des différences pour la détection des changements de plan

Nous avons vu précédemment qu'il était possible de représenter une différence entre deux images successives par une seule valeur. Afin de détecter des changements de plan, nous suivons cette valeur dans le temps comme dans [4] et [9]. Nous présenterons tout d'abord la détection des *cuts*, puis nous appliquerons notre méthode à la détection d'effets plus complexes.

La détection des changements de plan nécessite généralement un ajustement de paramètres. Ceux-ci sont souvent spécifiques au domaine vidéo étudié ou au type de plans présents dans la séquence. Par exemple, certains paramètres seront adaptés à des plans éloignés (où les objets en mouvement sont petits) et d'autres à des plans proches ou serrés (où les objets en mouvement occupent une portion importante de l'image). Cependant certaines séquences vidéo (comme les retransmissions télévisées de matchs de football) contiennent des plans éloignées et des plans proches. Il est donc nécessaire d'utiliser une méthode générale qui peut s'adapter à ces différents types de plans. La méthode proposée ici est capable de traiter différents domaines vidéo et différents types de plans, grâce à l'utilisation d'un seuil adaptatif.

Afin de détecter les *cuts*, nous calculons tout d'abord les différences $D$ définies comme les dérivées des distances $d$ :

$$D(t) = |d(t, t-1) - d(t-1, t-2)| \tag{4}$$

On compare alors ces différences avec un seuil $T_D$ défini en fonction de la mesure de distance choisie précédemment. Ce seuil est lié à la valeur moyenne des vecteurs caractéristiques $P$ des blocs. Par exemple, si on considère le cas $n = 1$, le seuil $T_D$ est calculé comme :

$$T_D = 0.05\ K\ \mu \tag{5}$$

avec $K$ le nombre de blocs et $\mu$ la moyenne des valeurs $P$. Si la valeur $D(t)$ est supérieure au seuil $T_D$, on considère qu'un changement de plan a eu lieu, comme le montre la figure 1. En effet, si la différence entre $d(t, t-1)$ et $d(t-1, t-2)$ est significative, alors la différence entre les images obtenues aux instants $t$ et $t-1$ n'est pas la même que celle entre les images obtenues aux instants $t-1$ et $t-2$. Il existe donc un changement brusque dans la séquence à l'instant $t-1$. Ce type de changement s'oppose à un changement progressif pouvant être observé lors d'un mouvement de caméra ou d'un effet de transition progressif (volet, fondu...).

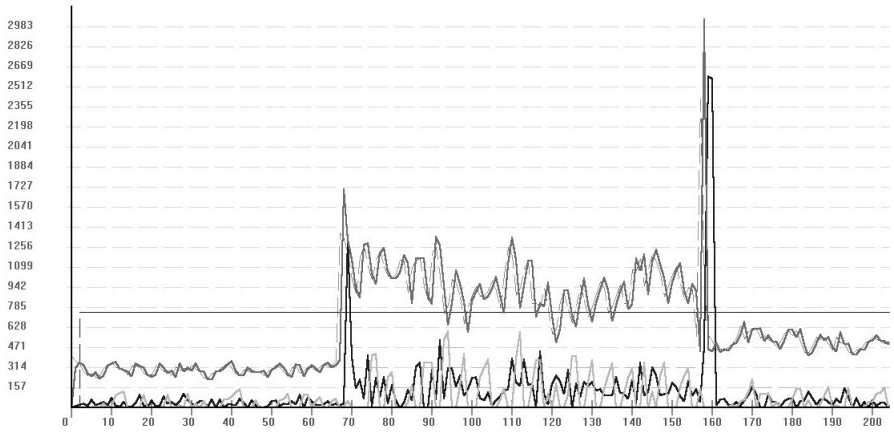


FIG. 1 – Evolution de $d(t)$ et $D(t)$ pour une séquence contenant deux *cuts* aux instants 69 et 160. Les valeurs de $D(t)$, $d(t)$ et $D^c(t)$ sont représentées par les courbes en trait continu, du plus foncé au plus clair.

Si un *cut* n'a pas été détecté, il est encore possible de se trouver en présence d'un autre type d'effet, pouvant être considéré comme un changement de plan étalé dans le temps, c'est-à-dire sur plusieurs images. Dans ce cas, il est plus difficile de détecter le changement de plan et une méthode basée sur un seuil fixe serait limitée. Nous utilisons donc un seuil adaptatif qui présente deux principaux avantages. Le premier est évidemment les possibilités d'adaptation de la méthode aux différents domaines vidéo (sport, bulletin d'informations, etc.). Le second avantage concerne la capacité de la méthode à traiter différents types de plans (proches ou éloignés) dans une même séquence vidéo. Ceci est très utile dans le cas de retransmissions d'événements sportifs où la vidéo peut être composée de plans éloignées (les objets en mouvement sont alors petits, ce qui nécessite une valeur de seuil faible) et de plans rapprochés (les objets en mouvements sont alors plus gros du fait d'un zoom, la valeur de seuil devant être importante afin d'éviter les fausses détections).

Afin de détecter ces effets progressifs, nous analysons les valeurs successives $d(t, t-1)$. Contrairement à $D(t)$, la valeur $d(t, t-1)$ est directement liée à la séquence vidéo analysée et ne sera pas la même dans le cas d'un plan proche ou d'un plan éloi-

gné. Comme nous l'avons vu précédemment, l'utilisation d'un seuil adaptatif va nous permettre d'analyser correctement la valeur $d(t, t-1)$, indépendamment de la situation.

Nous utilisons une technique proche de celle présentée dans [4] afin de détecter ces effets progressifs : nous fixons un second seuil $T_d$ et le comparons avec la valeur courante $d(t, t-1)$ représentant la différence entre les images obtenues aux instants $t$ et $t-1$. Si $d(t, t-1) > T_d$, alors il est possible qu'il y ait un changement de plan. Nous effectuons alors un calcul cumulé $D^c(t)$ de $D(t)$ tant que la différence $d(t, t-1)$ est supérieure au seuil $T_d$ :

$$D^c(t) = \sum_{t \in [t_1, t_2]} D(t) \quad \text{avec } t_1, t_2 \text{ définis tels que } \forall t \in [t_1, t_2] \;\; d(t, t-1) > T_d \qquad (6)$$

Finalement nous comparons la valeur cumulée $D^c(t)$ obtenue avec le seuil $T_D$ décrit précédemment : un changement de plan est détecté si $D^c(t)$ est supérieur au seuil, comme le montre la figure 2.

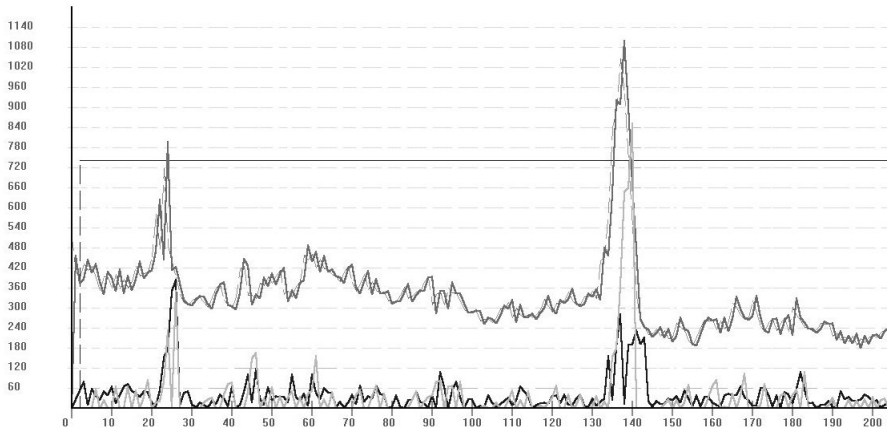


FIG. 2 – Evolution de $d(t)$ et $D(t)$ pour une séquence contenant un effet progressif (fondu) entre les instants 134 et 141. Les valeurs de $D(t)$, $d(t)$ et $D^c(t)$ sont représentées par les courbes en trait continu, du plus foncé au plus clair.

Le seuil adaptatif $T_d$ est défini comme :

$$T_d(t) = 0.25\, T_d(t-1) + 0.75\, d(t, t-1) \qquad (7)$$

Le seuil $T_d$ est mis à jour pour chaque nouvelle image de la séquence. De cette façon, il s'adapte automatiquement avec une certaine inertie au contenu de la vidéo étudiée, sa valeur étant modifiée selon l'évolution des valeurs de $d(t, t-1)$ afin d'éviter les fausses détections et les détections manquées.

Une description générale de l'algorithme permettant de détecter les *cuts* et les effets progressifs est donnée dans le tableau 1.

# 4 Résultats

Notre méthode a été testée sur des séquences vidéo de différents domaines. Afin de souligner les capacités de notre méthode, nous utilisons des résultats obtenus à partir de

Si $D(t) > T_D$
  Détection d'un *cut*
  $D^c(t) \leftarrow 0$
Sinon si $d(t, t-1) > T_d$
  $D^c(t) \leftarrow D^c(t-1) + D(t)$
Sinon si $D^c(t) > T_D$
  Détection d'un fondu
  $D^c(t) \leftarrow 0$
Sinon
  Fausse alerte
  $D^c(t) \leftarrow 0$
Réitérer avec $t \leftarrow t + 1$

TAB. 1 – Algorithme pour la détection des transitions.

retransmissions de matchs de football. Nous considérons que la détection des changements de plan est plus complexe pour ce type de vidéo que pour d'autres types de vidéo comme les bulletins d'information télévisés. En effet, les résultats obtenus sur d'autres types de vidéo étaient toujours meilleurs que ceux obtenus sur des séquences vidéo de football.

Les images présentées ici sont issues de séquences télévisées acquises à une fréquence de 25 Hz et leur taille est de $160 \times 120$ pixels. Après l'étape de réduction, les images à traiter contiennent $20 \times 15$ pixels, comme le montre la figure 3. Les séquences vidéo contiennent différents effets (*cut*, volet, fondu, mais aussi certains effets combinant volet et fondu). Elle incluent aussi du bruit dû au mouvement global de la caméra, aux objets en mouvement, ainsi qu'aux effets d'illumination.

FIG. 3 – Réduction spatiale d'une image.

Des exemples de changements de plan sont présentés sur les figures 4 à 6. Ils montrent les différents types de transitions décrits précédemment : *cut* (figure 4), fondu (figure 5) et volet (figure 6).

La qualité d'une méthode de détection de changements de plan peut être évaluée grâce à différentes mesures [10]. Nous utilisons les mesures suivantes :

$$Q_R = \frac{N_d}{N_d + N_m} \tag{8}$$

$$Q_P = \frac{N_d}{N_d + N_f} \tag{9}$$

$$Q_G = \frac{N_d - N_f}{N_d + N_m} \tag{10}$$

où $Q_R$, $Q_P$, et $Q_G$ représentent respectivement les qualités de robustesse, de précision, et la qualité générale, et où $N_d$, $N_f$, et $N_m$ sont respectivement les nombres de changements de plan détectés, manqués, et le nombre de fausses détections.

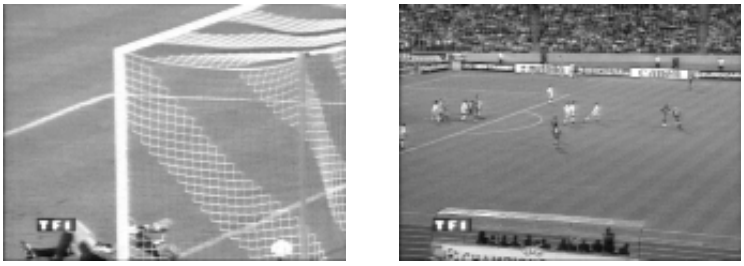

FIG. 4 – Images successives représentant un *cut*.

FIG. 5 – Images successives représentant un fondu.

FIG. 6 – Images successives représentant un volet.

Les figures 1 et 2 montrent les résultats obtenus avec deux séquences d'images différentes. Les paramètres $\alpha$ et $n$ étaient fixés respectivement à $0.3$ et $1$. Nous avons testé ces paramètres sur une vingtaine de séquences composées chacune de $500$ images et nous obtenons les valeurs $Q_R = 82\%$ , $Q_P = 100\%$, et $Q_G = 82\%$ qui montrent l'efficacité de notre méthode. Les différents taux de détection sont présentés dans le tableau 2.

| | $Q_R$ | $Q_P$ | $Q_G$ |
|---|---|---|---|
| *Cuts* | 100 % | 100 % | 100 % |
| Effets | 54 % | 100 % | 54 % |
| Total | 82 % | 100 % | 82 % |

TAB. 2 – Taux de détection des *cuts* et autres effets.

# 5 Conclusion

La méthode présentée ici est capable de détecter, en temps réel, des changements de plan dans des séquences d'images en couleurs acquises à l'aide d'une caméra en mouvement. Elle est robuste au bruit, aux mouvements importants, et aux effets d'illumination. Des résultats sur des séquences vidéo de matchs de football ont été présentés, et ont montré l'efficacité de la méthode.

Des tests doivent être effectués afin d'évaluer d'autres mesures de différence et d'autres espaces de couleurs. Une implémentation sur une machine multiprocesseurs est envisagée. Enfin, l'utilisation directe des blocs de différence doit être considérée comme un moyen de suivre des objets à faible résolution.

# Références


[1] F. Idris et S. Panchanathan, "Review of Image and Video Indexing Techniques", Journal of Visual Communication and Image Representation, Vol. 8, No. 2, pp. 146-166, juin 1997.

[2] G. Ahanger et T.D.C. Little, "A Survey of Technologies for Parsing and Indexing Digital Video", Journal of Visual Communication and Image Representation, Vol. 7, No. 1, pp. 28-43, mars 1996.

[3] R. Brunelli, O. Mich, et C.M. Modena, "A Survey on the Automatic Indexing of Video Data", Journal of Visual Communication and Image Representation, Vol. 10, No. 2, pp. 78-112, juin 1999.

[4] H.J. Zhang, A. Kankanhalli, et S.W. Smolliar, "Automatic Partitioning of Full-Motion Video", Multimedia Systems, Vol. 1, No. 1, pp. 10-28, 1993.

[5] J. Meng, Y. Juan, et S.F. Chang, "Scene Change Detection in a MPEG Compressed Video Sequence", Proceedings SPIE Conference on Digital Video Compression : Algorithms and Technologies, Vol. 2419, pp. 14-25, février 1995.

[6] P. Bouthemy, M. Gelgon, et F. Ganansia, "A Unified Approach to Shot Change Detection and Camera Motion Characterization", IEEE Transactions on Circuits and Systems for Video Technology, Vol. 9, No. 7, octobre 1999.

[7] J. Song et B.L. Yeo, "Spatially Reduced Image Extraction from MPEG-2 Video : Fast Algorithms and Applications", Proceedings SPIE Conference on Storage and Retrieval for Image and Video Database VI, pp. 93-107, janvier 1998.

[8] Y. Sakamoto, M. Ishiguro, et G. Kitagama, "Akaike Information Criterion Statistics", Kluwer Academic Publishers, novembre 1986.

[9] C.H. Demarty et S. Beucher, "Morphological tools for indexing video documents", Proceedings IEEE International Conference on Multimedia Computing and Systems, ICMCS'99, pp. 991-992, juin 1999.

[10] R. Ruiloba, P. Joly, S. Marchand-Maillet, et G. Quénot, "Towards a Standard Protocol for the Evaluation of Video-to-Shots Segmentation Algorithms", Proceedings European Workshop on Content Based Multimedia Indexing, CBMI'99, pp 41-48, octobre 1999.