Recherche de contenu appliquée à la surveillance vidéo

André Caron Pierre-Marc Jodoin
Centre de recherche MOIVRE
Université de Sherbrooke
Sherbrooke, Canada

***Abstract*— Cet article porte sur une méthode de recherche par contenu adaptée au domaine de la surveillance vidéo. Notre méthode s'appuie sur une représentation hiérarchique spatio-temporelle de la forme, de la taille et de la direction des objets en mouvement. À l'aide de cette représentation, une table de hachage de type *locality-sensitive hashing* (LSH) est construite. Légère en mémoire, cette table non seulement résume le contenu utile de la vidéo, mais elle facilite également la navigation à travers de longues séquences (des heures voire des jours de vidéo). La navigation se fait via des requêtes entrées par un utilisateur. Ces requêtes portent sur la forme, la direction et la position des événements recherchés. Une fonction de hachage transforme la requête en une clé permettant de retrouver en temps réel les sections de la vidéo dont le contenu s'apparente aux critères recherchés. De par la nature d'une table de hashage, la complexité de la recherche est $O(1)$. Notons que la table est trois ordres de magnitude plus légère sur disque que la vidéo d'origine. Cela correspond à environ 55Mo pour une vidéo 320x240 d'une durée de 12 heures dont le contenu brut occupe plus de 20Go.**

I. Introduction

Avec l'avènement des caméras IP à bas prix [1], le nombre d'applications en surveillance vidéo a connu une augmentation soutenue au cours des dernières années [2]. Parallèlement, l'augmentation fulgurante des capacités de stockage a eu pour effet d'augmenter le nombre d'heures de vidéo stockées sur disque [3]. Bien que ces vidéos soient rarement visualisées, leur stockage permet un retour sur des événements passés tels un vandalisme, une agression ou un accident routier. Or, la recherche d'un événement contenu dans une très longue séquence vidéo est un problème fondamental ne pouvant être résolu efficacement par une méthode triviale. De fait, la simple navigation à travers des semaines de vidéo s'avère un défi en soi pour lequel peu (voire pas) de solutions viables ont été proposée à ce jour. Bien que des méthodes permettant de résumer et d'organiser des vidéos aient déjà été publiées [4]–[6] la plupart ne fonctionnent que pour des oeuvres cinématographiques classiques (films, séries télé, clips, etc.) dont les transitions de scènes sont facilement détectables par des méthodes de type ≪images clés≫ (*key framing*) [7].

Malheureusement, ces techniques ne peuvent être appliquées à des vidéos de surveillance filmées par une caméra fixe et ne contenant aucune transition de scène. En fait, on ne peut utiliser des informations de texture et de couleur pour décrire le contenu de la scène, ces dernières changeant peu ou pas au cours de la vidéo. De fait, seule la dynamique et la forme des objets en mouvement permettent l'analyse et la description du contenu utile des vidéos de surveillance.

Récemment, quelques articles portant sur l'annotation et la réorganisation de longues vidéos de surveillance ont été publiés. C'est le cas de certaines méthodes [8]–[11] dont l'objectif est de résumer de très longues vidéos en éliminant le contenu redondant. Ces méthodes produisent ainsi une vidéo ≪condensée≫ contenant uniquement les objets en mouvement. Malheureusement, ce type d'approches est inapplicable au problème de recherche par le contenu. D'autres auteurs [12]–[15] ont exploré la possibilité d'indexer de façon spatio-temporelle des informations basées sur la dynamique et la forme des objets en mouvement. Or, certaines de ces méthodes requièrent une indexation manuelle [15], d'autres n'ont été validées que sur une ou deux séquences vidéo [12], [13] et d'autres portent exclusivement sur l'aspect ≪indexation≫ sans porter attention à l'aspect ≪recherche≫ [14].

La méthode proposée dans cet article possède un double objectif : (1) résumer de façon concise le contenu dynamique de la vidéo afin de (2) permettre la recherche en temps réel d'événements correspondant à une requête utilisateur. Pour ce faire, la vidéo est résumée à l'aide d'une table de hachage de type *locality-sensitive hashing*(LSH) [16]. Les requêtes utilisateur sont ensuite transformées en une *clé de hachage* via une *fonction de hachage*. Cette clé permet d'accéder en temps réel aux sections de la vidéo dont le contenu s'apparente aux critères recherchés.

II. Méthode Proposée

a) Document et représentation hiérarchique: Une vidéo est un volume spatio-temporel de taille $H \times W \times T$ où $H \times W$ est la taille en pixels des images et T le nombre d'images dans la vidéo. Afin d'en structurer le contenu, l'axe temporel est divisé en *documents* contenant chacun A images. Tel qu'illustré à la figure 1, les images sont divisées en tuiles de taille $B \times B$. Ainsi subdivisé, chaque *atome* couvre un volume de $A \times B^2$ pixels.

Lorsqu'un document est créé (généralement au fur et à mesure que la vidéo est enregistrée) ses atomes sont associés à un ensemble de caractéristiques liées aux objets en mouvement. Les caractéristiques ici retenues portent sur la direction du mouvement [17], la quantité d'activité enregistrée

Cette recherche est subventionnée par la subvention à la découverte 371951 du CRSNG ainsi que par le Ministère du Développement économique, innovation et exportation du Québec.

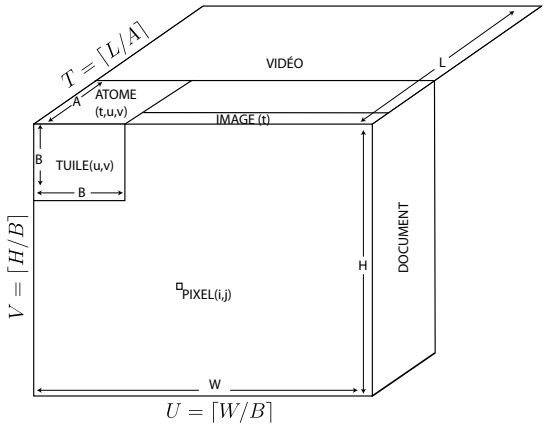

Fig. 1. **Représentation spatio-temporelle d'une vidéo.**

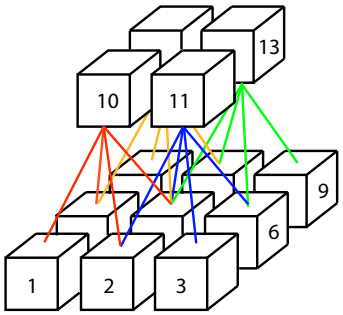

Fig. 2. **Représentation schématique d'une structure pyramidale à $h = 2$ niveaux comprenant 4 sous-arbres composés de 9 atomes et 4 parents. Le contenu des 4 noeuds parents est obtenu en fusionnant le contenu de leurs enfants respectifs. À noter qu'un troisième niveau peut être créé en fusionnant les noeuds 10,11,12 et 13.**

à l'intérieur de l'atome [18] ainsi que la taille médiane des objets en mouvement. Une fois les atomes traités, le document est converti en une structure pyramidale similaire à celle de la figure 2. Le contenu de chaque noeud dans la pyramide est obtenu en fusionnant les caractéristiques de ses enfants. Considérant qu'un sous-arbre à k niveaux couvre une surface de $h \times h$ atomes, chaque sous-arbre contient le résumé des objets en mouvement détectés dans cette zone.

b) Table de hachage: Au début de l'acquisition vidéo, une table de hachage $\mathcal{H}$ est initialisée puis enrichie à chaque fois qu'un document est créé. L'objectif de cette table est de contenir l'indice de tous les sous-arbres contenant au moins un objet en mouvement. Pour y arriver, nous avons adopté une table de hachage de type LSH [16]. Contrairement aux tables de hachage conventionnelles dont l'objectif est de minimiser les collisions [19], les tables LSH cherchent à maximiser les collisions entre les entrées dont le contenu est similaire. Autrement dit, une table LSH a pour objectif de regrouper ensemble l'indice des sous-arbres dont les caractéristiques d'activité sont similaires.

Soient $\vec{p}, \vec{q} \in \mathcal{U}$ les vecteurs de caractéristiques de deux sous-arbres et $h : \mathcal{U} \rightarrow \mathbb{Z}$, une fonction de hachage LSH permettant de convertir un vecteur de caractéristiques en une clé de hachage, *i.e.* un indice dans la table $\mathcal{H}$. L'objectif de la fonction h est de produire la même clé pour tout sous-arbre dont le contenu est similaire. En d'autres mots, $h(\vec{p}) = h(\vec{q})$ lorsque $\vec{p}$ et $\vec{q}$ sont similaires et $h(\vec{p}) \neq h(\vec{q})$ dans le cas contraire (se référer à l'article [16] pour plus de détails). La fonction de hachage utilisée par notre méthode est basée sur la méthode des distributions *p-stable* [20].

Ainsi, l'indice c_p de chaque sous-arbre $\vec{p}$ nouvellement construit est ajouté à la table : $\mathcal{H}(h(\vec{p})) = \mathcal{H}(h(\vec{p})) \cup c_p$. De cette façon, les sous-arbres au contenu similaire verront leur indice stocké au même endroit dans la table. Notons que les sous-arbres correspondant à des zones d'activité nulle sont ignorés. Une fois la vidéo terminée, la table $\mathcal{H}$ contient le résumé tous les sous-arbres (et donc de tous les événements) répertoriés dans la vidéo. À noter que seul $\mathcal{H}$ est stockée en mémoire, les tables pyramidales étant réinitialisées à chaque fois qu'un nouveau document est créé.

c) Recherche: L'objectif de la recherche est de retrouver les sections de la vidéo correspondant à une requête utilisateur. Au vu de notre méthode, une requête est une question du type ≪objet abandonné de forme carrée≫, ≪objets de taille Y ayant traversés la scène de gauche à droite≫ou encore ≪voitures ayant tournées à gauche au coin de la rue≫. En fait, comme l'illustre la figure 3, une requête contient une région d'intérêt sélectionnée par un utilisateur ainsi que les caractéristiques de l'événement recherché (taille, forme et direction).

Une fois la requête entrée, un nouveau document est créé (c.f. figure 1) dans lequel les atomes sélectionnés (en vert dans la figure 3) voient leurs caractéristiques associées à celles entrées par l'utilisateur. Les caractéristiques des atomes non-sélectionnés sont forcées à zéro. Une structure pyramidale (c.f. figure 2) est ensuite extraite de ce document et une recherche est effectuée pour tous les sous-arbres dont le vecteur de caractéristiques $\vec{p}$ est non nul à savoir :

$$C = \bigcup_{\vec{p}} \mathcal{H}(h(\vec{p})) \tag{1}$$

où C est l'ensemble des sous-arbres dans la vidéo dont le contenu correspond à la requête.

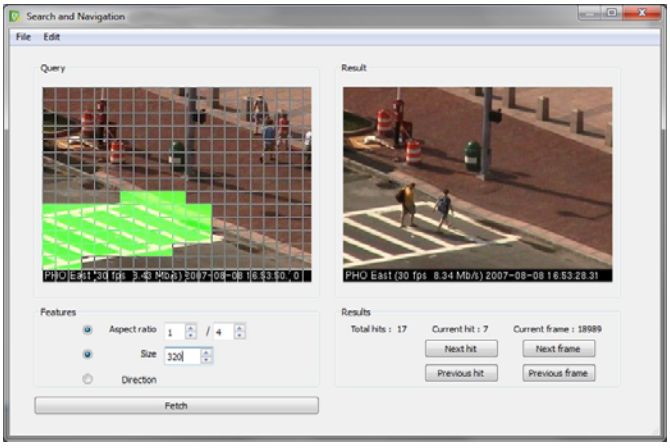

Fig. 3. **Une requête comprend une région d'intérêt sélectionnée par l'utilisateur et les caractéristiques approximatives des objets recherchés. Dans cet exemple, la recherche retrouve les piétons (et non les voitures) passant sur le passage piétonnier. À noter que le quadrillage de gauche indique la position des atomes.**

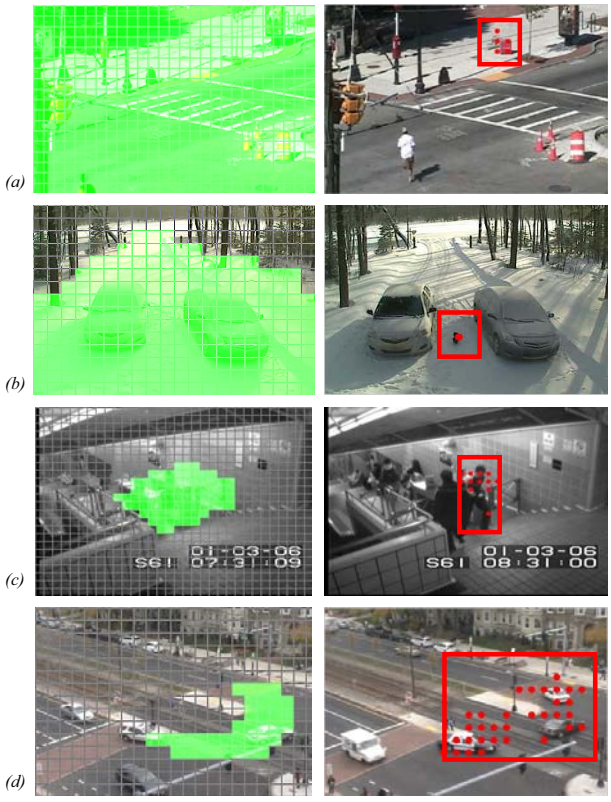

Fig. 4. **Résultats obtenus sur 4 vidéos avec (à gauche) la région d'intérêt sélectionnée par l'utilisateur et (à droite) la région couverte par les sous-arbres sélectionnés par la recherche.**

III. Résultats

Nous avons effectué plusieurs requêtes sur différentes vidéos afin de mesurer la robustesse de notre méthode. Les séquences utilisées contiennent entre 10 minutes et 4 heures de vidéo. Tel qu'illustré à la figure 4, nous avons recherché (a) un objet abandonné, (b) des animaux de petite taille, (c) les passagers d'un métro franchissant le guichet en sens inverse et (d) toutes les voitures effectuant un virage en U à une intersection. En plus de la région d'intérêt, une combinaison des caractéristiques liées à la persistance de l'activité, la taille des objets et la direction du mouvement fut retenue.

Dans tous les cas de figure, les délais de recherche sont quasi négligeables en raison non seulement de la complexité O(1) des arbres de hachage, mais également de la faible taille de la table $\mathcal{H}$. En effet, la petite taille de $\mathcal{H}$ nous permet de la stocker en mémoire vive et donc de réduire au minimum les accès disque. Considérant que les indices stockés dans $\mathcal{H}$ sont stockés sur 8 octets et que seul l'indice des sous-arbres ayant une activité non nulle sont conservés, la taille en octets de la table de hachage est la suivante

$$taille(\mathcal{H}) = \frac{H}{B} \times \frac{W}{B} \times \frac{L}{A} \times 8 \times f \qquad (2)$$

où $\frac{H}{B} \times \frac{W}{B} \times \frac{L}{A}$ correspond au nombre total de sous-arbres dans la vidéo et f est le taux d'activité dans la vidéo. Considérant que $B = 16$, $A = 30$, $H = 240$ $W = 320$ et que le taux d'activité moyen dans les vidéos utilisées dans cette recherche est de 2.5% en moyenne, la taille de $\mathcal{H}$ pour une vidéo longue de 12 heures est d'à peine 55Mo alors que la vidéo d'origine occupe plus de 20Go d'espace disque.

IV. Travaux à venir

Dans les prochains mois, nous chercherons à tester notre logiciel sur des séquences longues de plusieurs jours et d'améliorer sa robustesse à des changement d'illumination brusques.

References

[1] Network camera reviews web site. [Online]. Available : http ://www.networkcamerareviews.com/

[2] Business video news. [Online]. Available : http ://www.tmcnet.com/usubmit/2006/07/19/1719384.htm

[3] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection : A survey," *ACM Computing Surveys*, vol. 41, no. 3, pp. 1–58, 2009.

[4] J. Gibson and A. Bovik, Eds., *Handbook of Image and Video Processing*, 2nd ed. Elsevier Academic Press, 2005.

[5] S. Shipman, A. Divakaran, and M. Flynn, "Highlight scene detection and video summarization for pvr-enabled high-definition television systems," in *Proc. IEEE Conf. Consumer Electronics*, 2007, pp. 1–2.

[6] I. Otsuka, R. Radharkishnan, M. Siracusa, A. Divakaran, and H. Mishima, "An enhanced video summarization system using audio features for a personal video recorder," *IEEE Trans. Consum. Electron.*, vol. 52, no. 1, pp. 168–172, 2006.

[7] X. Song and G. Fan, "Joint key-frame extraction and object segmentation for content-based video analysis," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 7, pp. 904–914, 2006.

[8] Y. Pritch, A. Rav-Ach, and S. Peleg, "Nonchronological video synopsis and indexing," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 30, no. 11, pp. 1971–1984, 2008.

[9] A. Rav-Acha, Y. Pritch, and S. Peleg, "Making a long video short : Dynamic video synopsis," in *Proc. IEEE Conf. Computer Vision Pattern Recognition*, 2009, pp. 435–441.

[10] M. Brand, "Image and video retargetting by darting," in *Proc. Int. Conf. Image Analysis and Recognition*, vol. 5627, 2009, pp. 33–42.

[11] Z.Zhuang, P.Ishwar, and J.Konrad, "Video condensation by ribbon carving," *IEEE Trans. Image Process.*, vol. 18, no. 11, pp. 2572–2583, 2009.

[12] X. Wang, X. Ma, and E.Grimson, "Unsupervised activity perception in crowded and complicated scenes using hierarchical bayesian models," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 31, no. 3, 2009.

[13] M. Breitenstein, H. Grabner, and L. V. Gool, "Hunting nessie – real-time abnormality detection from webcams," in *Proc. IEEE Int. Workshop on Visual Surveillance*, October 2009.

[14] N. Johnson and D. Hogg, "Learning the distribution of object trajectories for event recognition," in *Proc. British Machine Vision Conference*, vol. 14, 1996.

[15] D. Dailey, P. Harn, and P. Lin, "Its data fusion," Washington State Transportation Center - TRAC/WSDOT, Tech. Rep., 1996.

[16] A. Gionis, P. Indyk, and R. Motwani, "Similarity search in high dimensions via hashing," in *Proc. Inter. Conf. on Very Large Data Bases*. San Francisco, CA, USA : Morgan Kaufmann Publishers Inc., 1999, pp. 518–529.

[17] B.Horn and B. Schunck, "Determining optical flow," *Artif. Intell.*, vol. 17, no. 1-3, pp. 185–203, 1981.

[18] Y. Benezeth, P.-M. Jodoin, B. Emile, H. Laurent, and C. Rosenberger, "Comparative study of background subtraction algorithms," *J. of Elec. Imaging*, vol. 19, no. 3, pp. 1–12, 2010.

[19] C. Shaffer, *A Practical Introduction to Data Structures and Algorithm Analysis*. Upper Saddle River, NJ, USA : Prentice Hall PTR, 2001.

[20] M. Datar, N. Immorlica, P. Indyk, and V. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," in *Proc. of symp. on Computational geometry*, ser. SCG '04. New York, NY, USA : ACM, 2004, pp. 253–262.