H.264/SVC SCENE MOTION ANALYSIS

Christian Käs, Henri Nicolas

LaBRI, University of Bordeaux 1
351, Cours de la Libération, 33405 Talence Cedex
{kaes,nicolas}@labri.fr

ABSTRACT

We present a simple and lightweight approach to scene analysis in the H.264/SVC domain. The method is entirely based on the motion vectors found in the compressed stream. Motion segmentation and object detection is performed after the estimation of the camera motion. Important object properties are calculated, which are used for object matching and trajectory estimation. The relative distance to the camera is estimated, resulting in a pseudo 3-D representation of the object trajectories.

Index Terms— H.264/SVC, compressed domain, scene analysis, trajectory estimation

1. INTRODUCTION

In scene analysis, global camera motion and local object motion are important features. Multiple applications, like video surveillance or video summaries, can profit from automatically obtained local and global motion information.

The main focus of this article is the trajectory estimation of moving objects in a video scene. We are working on scalable, compressed video streams encoded by H.264/SVC, the scalable extension of H.264/AVC. The coded stream already contains block-based motion vectors (MVs) for the variable sized macro-blocks (MBs) inherent to SVC, so the expensive estimation process was already performed by the encoder. However, the resulting motion vector field is sparse and noisy, so pre-processing has to applied.

Since we work with scenes that have been shot by a single uncalibrated camera, we first estimate the objects' trajectories in the image plane seen by the camera. Combining some extracted object properties, we finally estimate the relative distance to the camera over time, resulting in pseudo 3-D representation of local object motion during a video scene.

We assume that we have separated video scenes without any cuts or transitions. This can be achieved by first applying a compressed domain shot boundary detector, one of which was proposed by Bruyne [1] specifically for H.264 streams.

This work has been carried out in the context of the french national project ICOS-HD (ANR-06-MDCA-010-03) funded by the ANR (Agence Nationale de la Recherche).

2. RELATED WORK

A large number of compressed domain object segmentation and tracking algorithms appeared over the years. The proposed tracking approaches in the compressed domain rely either on MVs, residual information, or both. A lot of these works exploit the information found in MPEG-1/2 streams, where MVs and DCT coefficients are easily accessible. Hesseler et al. [2] perform the tracking initialization on decoded I-frames and use histograms of MVs of the MPEG-2 stream to perform tracking. Other MPEG-2 based methods have been proposed in [3, 4, 5, 6, 7, 8, 9].

Though most of the mentioned work can generally be ported to the H.264-AVC/SVC domain, some basic assumptions are no longer valid. The often used AC and DC coefficients (e.g., [3, 7, 8, 9]) of intra-coded blocks in H.264/AVC are transformed from spatially intra-predicted values instead of the original pixel values, so full decoding is necessary. Concerning our goal of unsupervised, compressed domain scene analysis, other shortcomings of former approaches include manual tracking initialization (e.g., [6, 9]), no support for camera motion (e.g., [9]) and no support for multiple, occluding objects (e.g., [4]). None of the approaches addresses the estimation of the object distance to the camera.

3. OBJECT DETECTION

The extraction of moving objects is based on the estimation of the global camera motion. We apply a robust algorithm similar to the one presented in [8], which estimates the well-known 6-parameter affine motion model using the MVs present in the stream. The algorithm follows an iterative weighted least squares scheme with outlier rejection after each iteration. For H.264/SVC, the entropy coding has to be reversed as the only necessary decoding step. The displacement values dx and dy are stored in quarter-pel precision for each MB and sub-MB partition.

The result of the global motion estimation (GME) is the vector ϕ, which contains the six parameters $a_1..a_6$ of the assumed motion model. During the GME, outlier masks in sub-MB resolution are created from vectors that do not follow the global motion. Outliers mainly originate from moving objects

and also from low-textured areas, where the block-matching algorithm of MPEG-based video codecs delivers very noisy results that do not reflect the real motion.

In order to alleviate the impact of these miss-detections, spatio-temporal filtering along the MV trajectories is performed. We apply morphological filtering, followed by a median and low-pass filter over a temporal window of 8 frames, which corresponds to one GOP in our test videos. The resulting outlier masks give a rough segmentation of the scene in static background and moving foreground objects.

We now construct a *Motion History Image* (MHI) [10] from the resulting masks to enhance the representation of small and poorly detected objects. After each update step, the MHI is segmented into its connected regions. For the time being, each separate region in the MHI is considered as a single moving object, if it is larger than a threshold size. As a final step of the detection stage, we assign a label to each object and calculate and memorize certain properties, namely the

- mask and its size in sub-MBs,
- the centroid and
- the local object motion.

The local object motion is estimated similar to the global motion, except that only the MVs covered by the mask are considered as active estimation support. Figure 1d-e gives an example for a raw and a filtered mask.

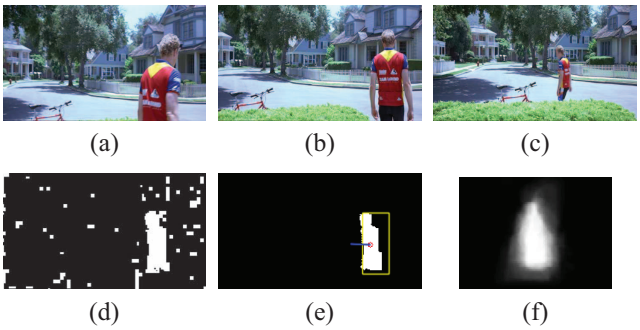


Fig. 1. a-c) Screenshots d) Raw outliers e) Filtered mask with detected object f) OHI of man. Sequence "street"©*Warner Bros. Adv. Media Services Inc.*

Table 1 shows the results of the object detection stage for multiple test sequences. The method performs well for various kinds of sequences.

4. OBJECT MATCHING

The frame-to-frame object correspondence is solved by a simple matching process which takes the object position and its properties over time into account. The first detection of moving objects in image I_0 initializes the object matcher. All object properties are calculated and the local motion estimation is used to predict the position in the successive frame. The label of the closest match in the successive frame is assigned to the new object.

Table 1. Object Detection Results

Sequence	Dur. in frames (sec)	Corr. det. objects	Missed objects	False pos.
street	270 (10.8s)	268/270 (99%)	2/270	22
parkrun	100 (4.0s)	95/100 (95%)	5/100	3
surveillance	118 (4.7s)	224/236 (95%)	12/236	3
kung fu	180 (7.2s)	291/303 (96%)	14/303	0
hall monitor	300 (12.0s)	404/455 (89%)	51/455	0
restaurant	310 (12.4s)	288/310 (93%)	22/310	7
train	228 (9.1s)	223/228 (97%)	5/228	2

Merge situations are detected if the predicted positions of multiple objects coincide in one single object in the following frame. Similarly, *split* situations are resolved when two objects emerge out of one. In case multiple separate objects get initialized as one merged object due to overlapping silhouettes, we know only after a split that it actually contained multiple objects. Therefore, we update the object labels of the past merged object.

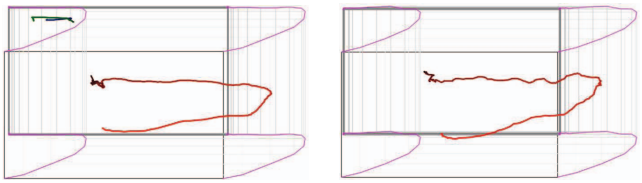
Besides, a set of simple and general rules are applied to cope with common types of miss-detections and problematic cases. For example, a small separate object that is covered by the projected mask from the previous image is marked as a "child" object under the same label than the main object. Partially occluded object parts (due to tables, lampposts, etc.) often fall under this category.

5. TRAJECTORY CONSTRUCTION

At this stage, we have a series of labeled objects together with their properties for each video frame. Since the appearance of a moving object usually changes over time due to the changing perspective, the position of the mask centroid may be subject to severe jitter. This is true most notably for non-rigid objects like human beings. We construct so-called *Object History Images* (OHIs) to stabilize the trajectory control point, one per object. The initial OHI is represented by the firstly detected object mask at t_0. If the object matcher finds a corresponding object at $t_0 + 1$, we increment the values in the OHI where the projected mask from t_0 overlaps with the current mask. An exemplary OHI for the *street* sequence is shown in Fig. 1f. As the trajectory control point, we calculate the center of gravity of the OHI, which assigns more importance to high values in the OHI, corresponding to the most stable parts of the object.

At this point, we can draw the trajectories over time in the image plane as seen by the camera. The complete, global

motion compensated trajectory for the man in the *street* sequence is shown in Fig. 2a. Figure 2b shows the ground truth, obtained by a user who was demanded to click on the middle of the mans' waistline in each frame. The rectangles represent the viewport of the camera over time. One rectangle per second is drawn. The curves connecting the rectangle corners represent the estimated global camera translation over time. It can be observed that the camera is following the moving object. It has to be noticed that the "third" dimension in this representation is time and not space. The estimated trajectory deviates only slightly from the ground truth. The short lines in the top left corner of image 2a were caused by the moving branch of a tree.



(a) (b)

Fig. 2. a) Estimated trajectory b) Manually obtained trajectory. Sequence "street".

6. RELATIVE DEPTH ESTIMATION

Since we work with a single, uncalibrated camera and compressed domain information, the distance of the moving object to the camera can only be observed indirectly through its position and size. Furthermore, no absolute distance values can be obtained due to the lack of scene geometry knowledge and camera parameters. We aim at estimating a relative distance measure that reflects if an object is approaching or moving away from the camera. We consider the following object properties as distance indicators:

- visible object area (mask size $a(t)$),
- total mask height ($h(t)$),
- top and bottom point of mask ($topY(t)$, $botY(t)$).

We assume that the relation between the visible object surface and its relative distance $d_{rel}(t)$ is of quadratic nature, which in theory is only true for observed objects that do not change their appearance/silhouette over time. Due to occlusions, noise, perspective distortions and non-rigid objects, this assumption is violated in nearly all real world scenarios. Nevertheless, the application of a very strong temporal moving average filter (window $>$ 2 sec.) levels out most of the mentioned effects and leads to a more confident measure $a_f(t)$ of the object area. It relates to the distance as $a_f \sim d_{rel}^2$. We also filter the object height and assume the linear relation $h_f(t) \sim d_{rel}$. The third indicator we take into account is the top and/or bottom Y-coordinate of the object over time, assuming the object moves on a flat surface and the camera angle to this surface is less than 90°. Preferably, $botY(t)$ is used instead of $topY(t)$ because of the direct connection to the ground, but we switch to $topY(t)$ if the objects' lowest point is outside the viewable image, detected by the mask touching the lower image border. We interpret positive ΔY values as approaching and negative values as distancing.

The three indicator vectors are finally stacked in the matrix $\mathbf{I} = [a_f(t)^T h_f(t)^T posY_f(t)^T]$, which is multiplied by the weight vector $\mathbf{w} = [w_0 w_1 w_3]^T$, where $w_i = \frac{1}{3}$ in order to account similar relevance to all three indices. The final distance estimation result is then obtained by $d_{rel}(t) = \mathbf{I} \cdot \mathbf{w}$.

Figure 3 shows the filtered indices that are used for the distance estimation over time and the final result for the *street* sequence. All indices have been normalized and the resulting measure is relative, where the chosen scale factor 100 is arbitrary and has no physical meaning. The man in the sequence steps down the sidewalk, so the flat surface assumption is not respected, leading to the deviation of the $topY$ index. In addition, the object is not entirely visible all the time. Nevertheless, the relative distance is well approximated (see also Sec. 7).

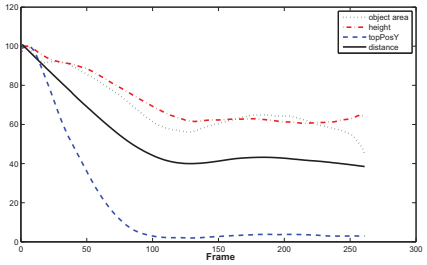


Fig. 3. Normalized distance indices and final, relative distance estimation result. Sequence "street".

Camera zoom is detected by the global motion estimator. In moments of zooming, all indicators have to be compensated. Though this is possible, we only tested our method on videos with pure translational camera motion or fix cameras.

7. FINAL RESULTS

Figure 4 shows the pseudo 3-D trajectory of the man in the *street* sequence (for screenshots see Fig. 1). The relative movement away from the camera and around the shrub is well detected. The motion of the model railroad in sequence *train* is depicted in Fig. 5. Though the estimation result is not perfect, it gives a good idea of the real setup. Screenshots and the trajectory for the waiter in the sequence *restaurant* are given in Fig. 5. This trajectory also reflects the waiters' real path, except for a moment where he stops to clean a table.

With only his arm in motion, a sharp movement away from the camera and back is estimated.

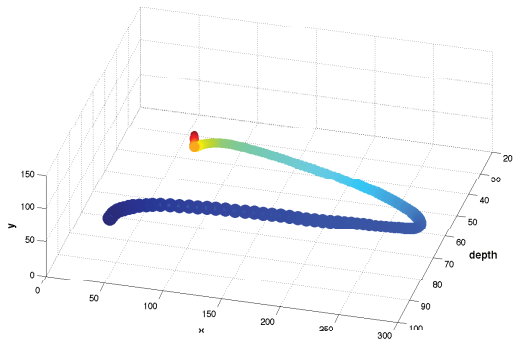


Fig. 4. Trajectory with relative object distance (depth) to the camera. Colors: blue $\hat{=}\, t_0$, red $\hat{=}\, t_{end}$. Sequence "street".

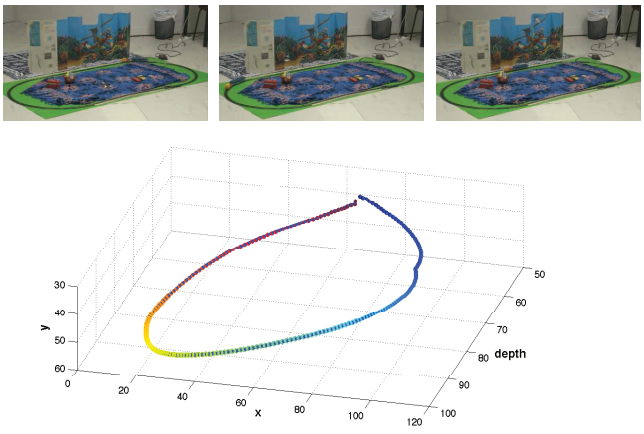


Fig. 5. Screenshots and trajectory. Sequence "train".

The processing time for all tested sequences at a resolution of 480x272 was near real-time to real-time (23-26.5 fps) on an Intel 2.16 GHz Core2Duo with 1 GB RAM.

8. CONCLUSIONS

We presented an efficient method to analyze scene motion in the compressed domain that can cope with camera motion. Two still-image representations are proposed to summarize the local and/or global motion within scene. The simple architecture delivers very promising results and enables fast processing. Future directions include the classification in rigid/non-rigid objects.

9. REFERENCES

[1] Sarah De Bruyne, Wesley De Neve, Davy De Schrijver, Peter Lambert, Piet Verhoeve, and Rik Van de Walle, "Shot boundary detection for h.264/avc bitstreams with frames containing multiple types of slices," in *PCM*, 2007, pp. 177–186.

[2] W. Hesseler and S. Eickeler, "Mpeg-2 compressed-domain algorithms for video analysis," *EURASIP Journal on Applied Signal Processing*, vol. 2, pp. 1–11, 2006.

[3] A.S.V. Radhakrishna, M.S. Kankanhalli, and P. Mulhem, "Compressed domain object tracking for automatic indexing of objects in mpeg home video," in *IEEE International Conference in Multimedia and Expo (ICME 2002)*, Lausanne, Switzerland, August 2002.

[4] Sung-Mo Park and Joonwhoan Lee, "Compressed domain object tracking for automatic indexing of objects in mpeg home video," in *4th Pacific Rim Conference on Multimedia*, Singapore, December 2003, vol. 2, pp. 748–752.

[5] Wen-Nung Lie and Wei-Chuan Hsiao, "Content-based video retrieval based on object motion trajectory," in *IEEE Workshop on Multimedia Signal Proc.*, December 2002, pp. 237–240.

[6] L. Favalli, A. Mecocci, and F. Moschetti, "Object tracking for retrieval applications in mpeg-2," in *IEEE Trans. on Circuits and Sys. for Video Tech.*, April 2000, vol. 10, pp. 427–432.

[7] Hanfeng Chen, Yiqiang Zhan, and Feihu Qi, "Rapid object tracking on compressed video," in *PCM '01: Proceedings of the Second IEEE Pacific Rim Conference on Multimedia*, London, UK, 2001, pp. 1066–1071, Springer-Verlag.

[8] F. Manerba, J. Benois-Pineau, R. Leonardi, and B. Mansencal, "Multiple moving object detection for fast video content description in compressed domain," *EURASIP J. Adv. Signal Process*, vol. 2008, no. 1, pp. 1–13, 2008.

[9] A. Aggarwal, S. Biswas, S. Singh, S. Sural, and A.K. Majumdar, "Object tracking using background subtraction and motion estimation in mpeg videos," in *ACCV06*, 2006, pp. II:121–130.

[10] Gary R. Bradski and James W. Davis, "Motion segmentation and pose recognition with motion history gradients," *Mach. Vision Appl.*, vol. 13, no. 3, pp. 174–184, 2002.

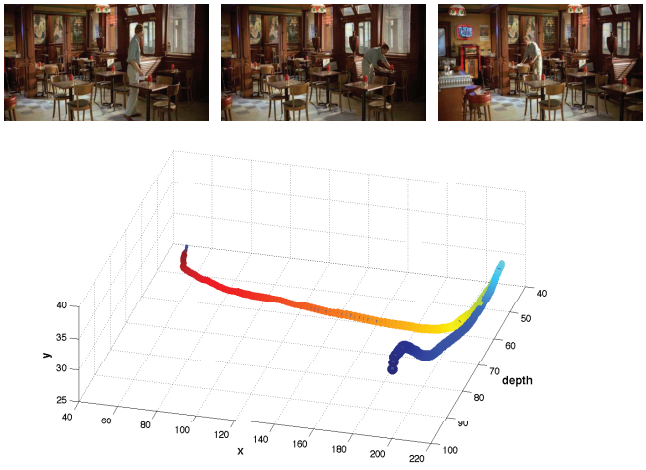


Fig. 6. Screenshots and trajectory. Sequence "restaurant"©*Warner Bros. Adv. Media Services Inc.*