Joint Global Motion Estimation and Coding for Scalable H.264/SVC High-Definition Video Streams

Christian Kaes, Henri Nicolas

JOINT GLOBAL MOTION ESTIMATION AND CODING FOR SCALABLE H.264/SVC HIGH-DEFINITION VIDEO STREAMS

Christian Käs, Henri Nicolas

LaBRI UMR CNRS, Université de Bordeaux
Domaine Universitaire, 351 cours de la Libération, 33405 Talence Cedex, France
{kaes,nicolas}@labri.fr

ABSTRACT

This paper presents a joint indexing-coding approach applied to global camera motion detection in the scalable H.264/SVC compressed domain. Our goal is to facilitate and to improve indexing in the MPEG compressed domain, if necessary by modifying the original coded stream, without losing compatibility with the standard. In frames with very noisy motion vector fields, we use the global motion information obtained by analyzing the raw video to modify the vectors of chosen macroblocks. This results in slightly decreased coding efficiency, but improved and more reliable indexing results in the compressed domain.

Index Terms— Global Motion Estimation, H.264/SVC, Compressed Domain, Indexing, Scalability

1. INTRODUCTION AND MOTIVATION

The growing amount of available video content necessitates efficient and reliable indexing techniques. Since most video content is stored in compressed form, it is desirable to perform indexing tasks directly in the compressed domain, so no decoding is necessary and fast processing is possible.

The steadily increasing availability of High-Definition (HD) video content with resolutions of up to 1920x1080 is pushing the deployment of the more complex and efficient MPEG-4/AVC (H.264) standard. The heterogeneity of available end-user terminals used to watch videos, ranging from high-end TV sets to small portable devices with low computational power and limited bandwidth connections, led to the introduction of scalable video coding. Various scalable video codecs have been presented in the literature (e.g., [5], [7]).

In this article, we present a robust estimation technique for global motion estimation (GME) of HD videos encoded with H.264 / Scalable Video Coding (SVC), the scalable extension of H.264/AVC [7]. The estimation is based on the motion vectors (MVs) of the macroblocks (MBs) of the SVC base layer. Since the MV fields found in the stream are optimized in terms of coding efficiency, they do not necessarily represent the real motion present in the scene, resulting in inaccurate global motion estimates for certain parts of the video. Numerous indexing tasks rely on accurate global motion estimates, among them the construction of mosaics, camera motion characterization and moving object detection. To overcome the problem of inaccurate estimates, we propose a method to optimize the motion vector field with respect to indexing tasks. The resulting loss in coding efficiency is weighted against the gain regarding indexing tasks.

A lot of work concerning video indexing in the MPEG-1/2 compressed domain can be found in the literature. Most of the presented methods concerning GME are based on the analysis of MVs of forward predicted P-frames in the MPEG-2 compressed domain ([1], [2], [3], [9]), with MBs of unit size, mainly dealing with video content of standard definition (SD) or smaller.

Wang et al [9], Bouthemy et al [1] and Durik et al [2] use similar motion estimation and outlier rejection algorithms with some form of weighted least squares estimation applied to the MVs. Hesseler et al [3] apply two-dimensional MV histograms and the AC-coefficients for outlier detection.

The existing methods do not incorporate scalable video streams, exact frame wise GME, HD video, varying MB sizes and bi-predicted B-frame MVs, which are present in H.264/-SVC or AVC streams due to flexible MB partitioning and hierarchical picture prediction (see [7]). Furthermore, the global motion estimate may be inaccurate or wrong due to noisy MVs and MVs not representing the real motion.

The suite of this article is organized as follows: In section 2 we briefly present some properties of the upcoming standard H.264/SVC. Section 3 explains the GME method working with the original streams, Section 4 shows how the original MV fields are corrected and results and perspectives are provided in Section 5.

2. H.264/SVC

We coded the original, raw YUV test sequences with the reference implementation of H.264/SVC, version 9.8 (JSVM 9.8, available at [8]), with three spatial layers and an AVC-compatible base layer. The Group of Picture (GOP) size and the intra period were set to 8. For predicted pictures, the maximum number of reference frames was limited to 2 for the sake of simplicity. Each GOP is made up by one intra-coded key frame, which contains no motion information, and (GOP_Size -1) predicted frames, illustrated in Figure 1.
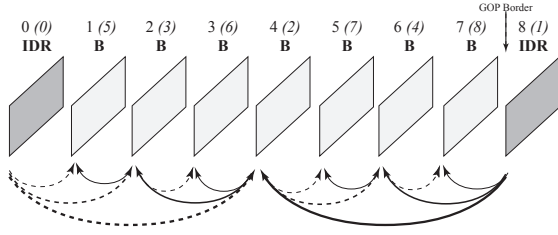

Fig. 1. Hierarchical picture prediction. First number on top is display order, the number in brackets the coding order. Dashed arrows denote LIST_0 prediction, solid lines stand for LIST_1 prediction

For each macroblock of a bi-predicted B-frame, multiple MVs may be encoded in the stream, organized in two lists, LIST_0 and LIST_1. LIST_0 MVs reference previous frames in display order, whereas LIST_1 MVs are pointing from future frames to the current one. Different to previous standards like MPEG-1/2, 16x16 MBs may be divided into smaller sub-MB partitions. Six different sub-partition sizes are possible, ranging from 16x16 down to 4x4 pixels. A SVC stream usually contains several spatial layers. To assure downward compatibility, the base layer may be coded in AVC mode, i.e., any decoder conforming to the H.264/AVC standard is able to decode the video in base layer resolution. All higher spatial layers employ independent motion-prediction structures that incorporate inter-layer prediction between the different layers. The reader is referred to [7] for a more detailed overview of H.264/SVC.

The test sequences have Full-HD resolution (1920x1080) and were shot in the 1080p mode at a frame rate of 25 fps. Since the current version of the JSVM needs the dimensions of all spatial layers to be a multiple of 16, we resized the highest layer to 1920x1088, so the lower layers at half resolutions have the dimensions 960x544 and 480x272, respectively.

3. GLOBAL MOTION ESTIMATION ON ORIGINAL STREAMS

The processing of high definition video streams is computationally very expensive. The figures 8 (a) and 9 (a)-(c) show

that we do not need to process the video at Full HD resolution to obtain reliable global motion estimation (GME) results. The plots show the estimation results of the raw test sequence *camMotion* (see also Section 5) at different resolutions, from full HD down to 60x34 pixels. The estimation, most notably concerning the variables *zoom* and *rotation*, remain stable until a resolution of 120x68 pixels. To save computing time, we take advantage of the spatial scalability of the compressed stream and extract only the AVC base layer with an original resolution of 480x272. No stream decoding is necessary for this step, higher layer packets are just discarded. To obtain the MV values of the base layer MBs, the entropy coding has to be reversed.

For each B-frame MB, depending on the prediction mode (LIST_0, LIST_1, direct or bi-prediction), we get a MV from LIST_0 and one from LIST_1. The choice between LIST_0 or LIST_1 MVs as active estimation support has shown to be arbitrary. Further on, only LIST_1 MVs are processed, scaled by the distance to its reference picture. The limitation to only one list delivered better results than an averaging of both MVs (with those from LIST_0 mirrored), because averaging smoothens abrupt changes in motion that occur within a GOP.

To obtain an estimate for intra-coded frames without MVs, we take the mirrored LIST_0 vectors from the subsequent B-frame in display order as an estimation basis. Figure 2 shows an example of the macroblock partitions of different sizes and the sparse motion vector field at a moment with camera panning.
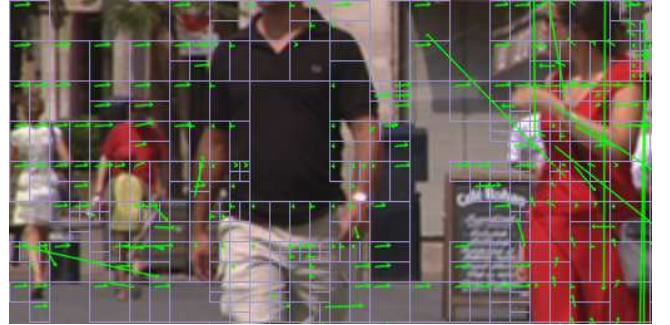

Fig. 2. Example of noisy MVs and MB partitions at a camera pan. MBs in *SKIP* mode are not drawn. Image was magnified and cropped for better visibility (Sequence *redDress*, thankfully provided by Pierre Lasserre).

The applied global motion estimation algorithm is based on the work of [1] and [2]. It consists of the following steps: 1) Least-squares (LS) estimation of the 6-parameter affine model, 2) Outlier rejection, 3) Iterative re-weighted LS estimation and 4) Camera motion characterization.

Usually not all MVs in the stream represent the global camera motion. Outliers appear due to moving objects in the scene and due to falsely detected motion during the encod-

ing process. Small MB partitions are more likely to represent moving objects or object borders, so we only consider MBs with each dimension being greater than 4. After an initial LS estimation, outliers are excluded from the estimation support, if their magnitude is larger than the standard deviation (max. rejection of 50 percent of all MVs).

The remaining MVs $(d_x, d_y)^T$ are fitted to the 2-D 6-parameter affine model, given by

$$
\begin{aligned}
d_x &= a_1 + a_2(x - x_0) + a_3(y - y_0) \\
d_y &= a_4 + a_5(x - x_0) + a_6(y - y_0),
\end{aligned}
\tag{1}
$$

where $(x_0, y_0)^T$ denotes the reference point in the image (e.g., the image center) and $(x, y)^T$ the center of the MB. We then obtain the motion parameters by an iterative re-weighted least squares estimation, given by

$$
\hat{\phi} = (H^T W H)^{-1} H^T W Z,
\tag{2}
$$

where $\hat{\phi}$ is a vector containing the estimated motion parameters $(a_1, \ldots, a_6)^T$, H is the observation matrix containing the MB centers, Z is a column vector with the extracted motion vectors and W a diagonal matrix containing the weights. In the first iteration, all weights are set equal to 1, resulting in the standard LS solution.

After the first iteration, the weights w_i for each MV are obtained by the following Gaussian function:

$$
w_i = \frac{\rho^2}{\sqrt{\pi}} e^{\left(-\frac{r_i^2}{\rho}\right)},
\tag{3}
$$

where ρ was set to the standard deviation σ (or to 1, if $\sigma < 1$) and r_i denotes the estimation residual after the last iteration. To obtain a more intuitive understanding of the motion parameters $a_1 \ldots a_6$, they are finally transformed into another basis of elementary camera motion descriptors $\theta = (pan, tilt, zoom, rot)$, with [1]

$$
\begin{aligned}
pan &= a_1, & tilt &= a_4, \\
zoom &= \gamma \cdot (a_2 + a_6), & rot &= \gamma \cdot (a_5 - a_3),
\end{aligned}
\tag{4}
$$

with $\gamma = \sqrt{height^2 + width^2}/4$ being a resolution dependent scaling factor, to project the $zoom$ and rot value in the same dimension as pan and $tilt$, since they represent ratios and not pixel values like the two translational motion parameters. Detected motion that lasts no longer than 3 frames is regarded as noise or jitter motion. The proposed compressed domain GME algorithm works about 10 times faster on HD streams as a GME solution working on uncompressed videos, like Motion2D [6], which is based on a robust multiresolution scheme where an error functional is minimized using an iterative re-weighted LS method.

Although we achieve to detect the significant and dominant global motion with the described algorithm (see Section 5), some missed and some false detections occur. Under certain circumstances, the majority of MVs does not represent the global motion or is very noisy due to moving objects or great, low-textured areas [3]. Since we aim to provide precise and reliable motion information, we propose a correction of the motion vector fields to overcome this problem. We use a modified version of the JSVM encoder.

4. MOTION VECTOR CORRECTION

Our idea is to provide scalable SVC streams that are still nearly as efficiently coded as the original ones, but with MV fields optimized for precise indexing tasks, while retaining conformity to the standard. Figure 3 shows an overview of the experimental setup.
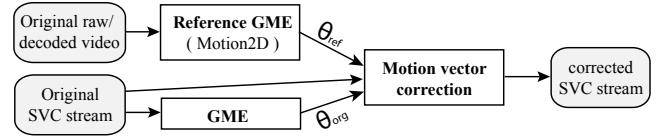

Fig. 3. Correction schema

The reference GME was performed with Motion2D [6], which analyzes the uncompressed original or the decoded sequence at base layer resolution. Output of this step are the motion parameters θ_{ref}.

θ_{ref} and θ_{org} (obtained like described in Section 3) are input to the correction module, which also reads the original compressed stream. If the summed, element wise mean squared error (MSE) between θ_{ref} and θ_{org} is below a threshold λ_{MSE}, the original stream remains untouched. λ_{MSE} controls the number of frames to be corrected.

If it exceeds the threshold, a subset of all LIST_1 MVs of LIST_1- or bi-predicted MBs (no MBs smaller than 8x8) are corrected according to the reference GME, if their angle and/or magnitude differs from the reference values more than a second threshold, λ_{diff}, and if they are considered as background MBs. λ_{diff} determines the number of changed MBs per corrected frame.

The distinction between background and foreground MBs is made by the GME module, which marks the outliers during the estimation process. The obtained outlier masks are buffered for the current GOP and spatiotemporally filtered along the motion trajectories of the MBs. This filtering process consists of morphological operations, median- and lowpass filtering and is followed by a thresholding of the filtered mask values. This filters out noisy motion vectors in background areas, so the probability that the resulting mask covers only objects in motion is increased. An example of a raw outlier mask and its spatiotemporally filtered version is given in Figure 4. In order to limit the impact on the coding efficiency, all MBs in *DIRECT* or *SKIP* mode are passed through and are not modified.
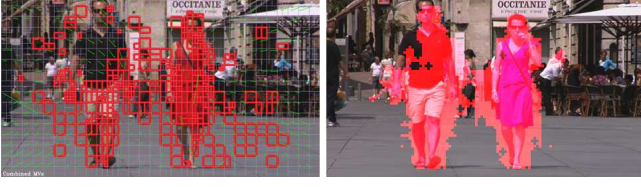
The new values for the corrected MVs are constructed

Fig. 4. Raw and filtered outlier masks of frame 134 of the sequence *redDress*.

Fig. 5. Frames 1 and 900 of the test sequence *camMotion*. Problematic are areas like the homogeneous, low-textured wall.

from the motion parameters θ_{ref}. In situations of pure panning or tilting, the new MV $(dx_{new}, dy_{new})^T$ is simply $(pan, tilt)^T$. For zooming and rotation, the new MV is taken from the respectively reconstructed, ideal MV field for zoom or rotation at the appropriate MB position, given by Equations 5 (based on [4]):

$$
\begin{aligned}
&\alpha_{zoom}(x_i, y_j) = \arctan(\tfrac{\text{sgn}(zoom_{ref})(-y_j+y_0)}{x_i - x_0}), \\
&l_{zoom}(x_i, y_j) = c \cdot zoom_{ref} \cdot \sqrt{(y_j - y_0)^2 + (x_i - x_0)^2}, \\
&\alpha_{rot}(x_i, y_j) = \arctan(\tfrac{(-y_j+y_0)}{x_i - x_0}) - \text{sgn}(rot_{ref}) * \pi/2, \\
&l_{rot}(x_i, y_j) = c \cdot rot_{ref} \cdot \sqrt{(y_j - y_0)^2 + (x_i - x_0)^2},
\end{aligned}
\tag{5}
$$

with angle α, vector length l, image center $(x_0, y_0)^T$, MB center $(x_i, y_i)^T$, $\text{sgn}(x)$ being the signum funtion and the constant factor $c = 0.14$, which was estimated from extracted motion vector fields at moments of pure zooming.

If the MV of a MB is changed, the prediction residual has also to be changed in the SVC stream. The new residual is usually greater than the original, so the coding efficiency is decreased. The mentioned thresholds have to be chosen so that a minimal decrease in efficiency leads to indexing gains, which depends on the targeted application. Here, we consider the detection of the presence of camera movement. Besides the loss in efficiency, the second disadvantage of the approach is the increased complexity at the encoder side. After the correction, the same GME algorithm as described in Section 3 is applied on the modified stream to obtain GME results.

5. RESULTS

In this section, we present the results of our GME method working on the corrected streams in comparison to those obtained by the original streams and to those obtained by Motion2D [6]. We show the results for two Full HD test sequences, *redDress* (25 sec.) and *camMotion* (39 sec.), which are part of the HD video corpus of the LaBRI and contain neither cuts nor transitions. Exemplary screenshots are given in Figure 4 and Figure 5.

Figure 6 shows the GME results for the sequence *redDress*, obtained by (a) analyzing the raw video at full resolution (1920x1088) with Motion2D, and (b) analyzing the base layer of the original SVC stream. For this sequence, the proposed GME algorithm performs very well, so no correction

of the MV fields was necessary. For better comparability, the curves in (a) have been scaled down by factor 4, which corresponds to the decrease in resolution between the base layer and the full resolution. The results also reconfirm the scalability of the low-level descriptor global motion. This circumstance allows to extract and analyze only lower video layers, which speeds up the processing time, most notably when analyzing high definition sequences. For the regarded clips, the processing time was reduced by a factor of 10 when working at the four times smaller base layer resolution.

The results for the sequence *camMotion* are depicted in Figure 8. Here, the detection results obtained with the original SVC stream are degraded, mainly due to the presence of large, flat zones in the image, resulting in very noisy and arbitrary MV fields for certain frames. To enhance indexing results, the MVs of these frames have been corrected. The threshold λ_{diff} was fixed to $10°$ and λ_{MSE} was varied between (A) 2 and (B) 1. For $\lambda_{MSE} = 2$, this caused the modification of 5 percent of all frames (48 frames), and per modified frame an average number of 46 MBs, representing on average 30 percent of the estimation support and affecting about 23 percent of all pixels. For $\lambda_{MSE} = 1$, the motion vector fields of 9 percent of all frames had been corrected. The average luminance PSNR of both decoded, corrected sequences stayed nearly the same (-0.2 dB), at an increase in bit rate of 0.7 percent for $\lambda_{MSE} = 2$ and of 1.1 percent for $\lambda_{MSE} = 1$.

Figure 7 shows the qualitative global motion detection results of all three approaches. The detection threshold λ_{det} for significant global motion was empirically determined and set to ±0.7 for all 3 approaches. λ_{det} is also resolution dependent and has to be multiplied by the dimension scaling factor for sequences smaller or larger in size. The recall, i.e., the percentage of all frames that really contain camera movement and that were found and correctly classified by the system could be improved by 6 percent. More notably, the proposed correction mechanism reduced the number of false detections, i.e., improved the precision by 11 percent. Values for precision and recall are summarized in Table 1, considering the camera motion classification for each frame.

The obtained results nearly reach the performance of Motion2D. In particular, the indexing results of frames with large, low-textured areas could be improved, which are considered
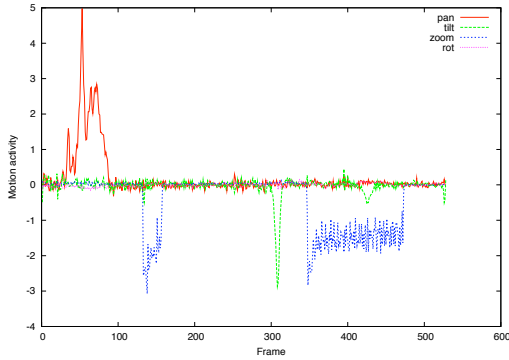
(a) Motion2D
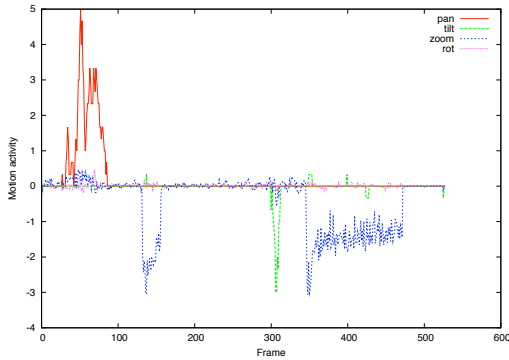
(b) Original SVC

Fig. 6. Motion Analysis of sequence *redDress* with (a) Motion2D [6] of raw video at base layer resolution, (b) of the original SVC stream.

as the most problematic ones [3], [4].

The proposed indexing enhanced system can be included directly in the source encoder, where the module delivering the reference indexing information may be replaced by any indexing method working on uncompressed videos. Further ideas include the pre-segmentation of the source video in order to influence the choice of the MB partition sizes on object borders to facilitate and improve moving object detection and fast compressed domain segmentation.

6. SUMMARY

We presented an approach to efficiently determine the frame wise global motion of HD videos in the scalable H.264/SVC compressed domain. Furthermore, we proposed the correction of the motion vector fields of the compressed stream, if global motion estimation fails due to noisy vector fields. The correction leads to accurate detection results, at only slightly
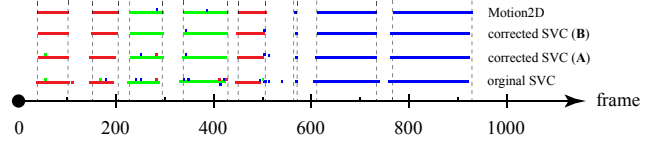

Fig. 7. Detection results for fixed $\lambda_{det} = \pm 0.7$ and $\lambda_{diff} = 10°$. (A) $\lambda_{MSE} = 2$ (B) $\lambda_{MSE} = 1$. The dotted lines depict the limits of the manually determined ground truth. Sequence *camMotion*
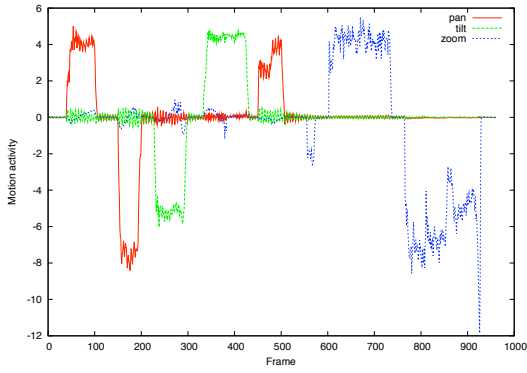
	Orig.	Corr. (A)	Corr. (B)	Motion2D
Recall	0.90	0.93	0.96	0.98
Precision	0.83	0.92	0.94	0.97

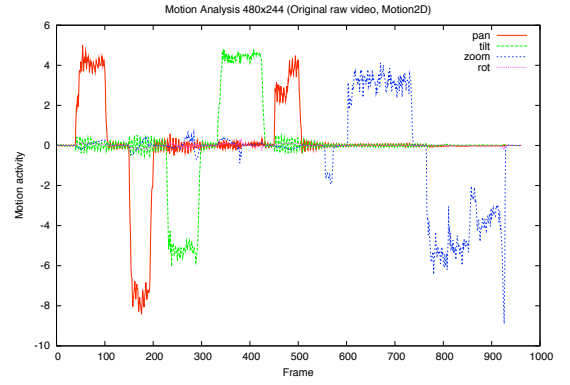**Table 1**. Results for recall and precision at $\lambda_{det} = \pm 0.7$. Sequence *camMotion*.
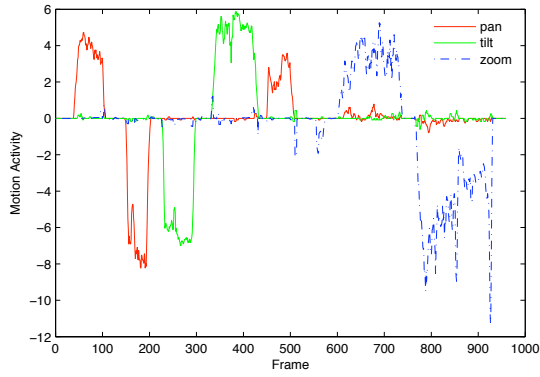
decreased coding efficiency.

7. REFERENCES

[1] P. Bouthemy, M. Gelgon, and F.Ganansia. A unified approach to shot change detection and camera motion characterization. In *IEEE Transactions on Circuits and Systems for Video Technology*, October 1999.

[2] M. Durik and J. Benois-Pineau. Robust motion characterisation for video indexing based on mpeg2 optical flow. In *Proceedings of International Workshop on Content-Based Multimedia Indexing (CBMI'01)*, 2001.

[3] W. Hesseler and S. Eickeler. Mpeg-2 compressed-domain algorithms for video analysis. *EURASIP Journal on Applied Signal Processing, Issue 2*, 2006.

[4] R. Jin, Y. Qi, and A. Hauptmann. A probabilistic model for camera zoom detection. In *16th Conference of the International Association for Pattern Recognition (ICPR'02), Quebec City, Canada*, August 2002.

[5] M.H. Lee, R.-Y. Qiao, and K. Bengston. Error-resilient scalable video over the internet. *Australian Telecommunication Networks and Applications Conference (ATNAC2006), Melbourne, Australia*, pages 420–424, December 2006.

[6] Motion2D. A software to estimate 2d parametric motion models. http://www.irisa.fr/vista/Motion2D/.

[7] H. Schwarz, D. Marpe, and T. Wiegand. Overview of the scalable h.264/mpeg4-avc extension. In *IEEE International Conference on Image Processing (ICIP'06), Atlanta, USA*, October 2006.

[8] JSVM Reference Software. Reference software for h.264/svc. http://ftp3.itu.ch/av-arch/jvt-site/.

[9] R. Wang and T. Huang. Fast camera motion analysis in mpeg domain. In *Proceedings of IEEE International Conference on Image Processing (ICIP'99), Kobe, Japan*, volume 3, pages 691–694, October 2001.
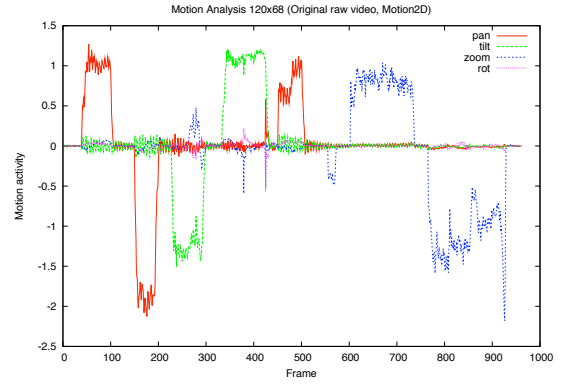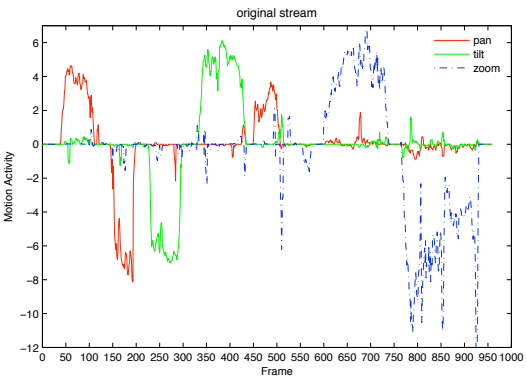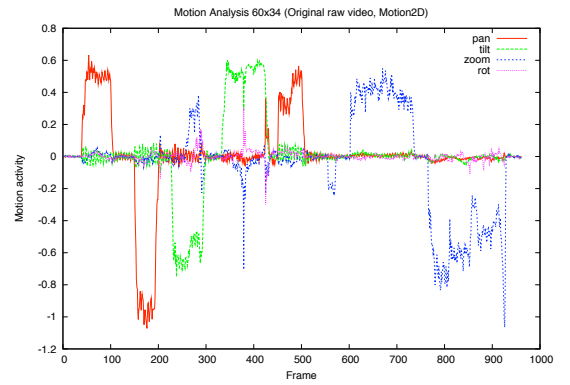
(a) Motion2D

(b) Corrected SVC

(c) Original SVC

Fig. 8. Motion Analysis with (a) Motion2D [6] of raw video at base layer resolution, (b) of the corrected SVC stream with $\lambda_{diff} = 10°$, $\lambda_{MSE} = 1$ and (c) of the original SVC stream.

(a) Resolution 480x272

(b) Resolution 120x68

(c) Resolution 60x34

Fig. 9. Motion Analysis at different resolutions.