Camera Motion-constraint Video Codec Selection

Andreas Krutz [#1], Sebastian Knorr [*2], Matthias Kunter [*3] , and Thomas Sikora [#4]

[#] *Communication Systems Group, TU Berlin*
Einsteinufer 17, Berlin, Germany
[1] krutz@nue.tu-berlin.de
[4] sikora@nue.tu-berlin.de

[*] *imcube Media*
Einsteinufer 17, Berlin, Germany
[2] knorr@imcube.de
[3] kunter@imcube.de

***Abstract*—In recent years advanced video codecs have been developed, such as standardized in MPEG-4. The latest video codec H.264/AVC provides compression performance superior to previous standards, but is based on the same basic motion-compensated-DCT architecture. However, for certain types of video, it has been shown that it is possible to outperform the H.264/AVC using an object-based video codec. Towards a general-purpose object-based video coding system we present an automated approach to separate a video sequences into sub-sequences regarding its camera motion type. Then, the sub-sequences are coded either with an object-based codec or the common H.264/AVC. Applying different video codecs for different kinds of camera motion, we achieve a higher overall coding gain for the video sequence. In first experimental evaluations, we demonstrate the excellence performance of this approach on two test sequences.**

I. Introduction

During the last two decades, efficient video codecs have been developed and standardized (e.g. MPEG-1,2 and 4). Recent video coding research has led to the latest video coding standard H.264/AVC [1], whose compression performance significantly exceeds previous standards. All these codecs rely on the well-known hybrid video coding scheme standardized first in MPEG-1.

Object-based video coding using video mosaics provides an alternative approach [2], [3], [4]. In such systems, the video content is segmented into foreground and background objects. A background mosaic is built over a group of frames that have similar background. The mosaic and the segmented foreground objects are coded separately. At the decoder, both are merged together in order to reconstruct the original video. For a certain kind of video content this approach can significantely outperform the common hybrid video coding scheme. There are two critical points applying this kind of coding, the foreground/background segmentation and the generation of the mosaic. A lot of work has been done in this area (e.g. [5], [6]). For coding the separated video data, several coding approaches have been developed and standardized in MPEG-4 Video, too.

The first drawback of the object-based video coding using background mosaics is the computational cost of the sophisticated pre-processing steps in segmentation and generating a video mosaic at the encoder. Furthermore, the use of different coding algorithms for the mosaic and segmented foreground objects also increases the complexity at the decoder. To make this object-based codec more applicable, a new object-based video codec has been proposed recently for single- and multi-view video [7],[8]. Here, the video mosaic is generated first. An object segmentation algorithm is then applied using the recontructed frames from the video mosaic, integrating the foreground/background segmentation into the mosaic generation. We emphasize that foreground/background segmentation is performed fully automatically. Thus, no user-assisted segmentation is required. Experimental results have shown that the object-based approach outperforms the common hybrid coding techniques for the considered test sequences. The second drawback of the object-based approach is the generic usability. It has been shown that object-based representation results in coding gain for sequences with a rotating camera. Therefore, work is underway to select the video codec depending on the content [9].

In this paper we will show that the codec selection only relies on the camera motion of the sequence. We consider the object-based video codec (OBVC) in combination wirh H.264/AVC for camera pans and the common H.264/AVC for camera motion types where the OBVC does not work, e.g. camera track. Two algorithms are presented and compared for recognizing the camera motion type. The sequence is then segmented into sub-segments relying on the several motion types. The first technique uses the frame-by-frame global motion compensated error values (RMSE - root mean square error) as the input data. After applying a median filter for noise reduction, a variance-based threshold defines the video segments. The global motion estimation algorithm uses the well-known 8-parameter perspective motion model [11]. Using this model, camera motion, such as pan, tilt, zoom, rotation, can be estimated very good. This leads to small $RMSE$-values in the global motion compensated error frames. Having the threshold calculated based on the variance on the $RMSE$-curve, the video is segmented into two segments for sprite coding mode ($RMSE$-values below the threshold) and H.264/AVC mode ($RMSE$-values above the threshold). The second technique relies on feature tracking and motion model selection. Here, the camera track is recognized using the *Geometric Robust*

information Criterion (GRIC) [12].

Assuming that the camera is tracked, this sub-sequence is coded with the H.264/AVC mode. All other sub-sequences are coded with the sprite coding mode. We will show that this codec selection based on the camera motion outperforms the coding performance of using the H.264/AVC for the whole sequence.

The paper is organized as follows. Both camera motion analysis techniques are detailed in the next Section. The complete camera motion contraint video codec (CMCVC) using the H.264/AVC is outlined in Section III. In Section IV experimental results are shown and the last Section concludes the paper.

II. Algorithms for Video Codec Selection

The fundamental idea is to segment the input sequence depending on the camera motion of the scene. We consider two types of camera set up's, fix camera with pan, tilt, zoom or roll and camera track. It has been shown over the last decade that video seqences with a fix camera (pan, tilt, zoom, roll) can be coded very efficient with an object-based coding approach, i.e. sprite coding. If the camera tracks during the sequence sprite coding is not possible. Therefore, we will automatically segment the input sequence into sub-sequences and code the sub-sequences with different video codecs, that are sprite coding (e.g. pan) and H.264/AVC (track). The next two sub-sections describe two approaches for analysing and segmenting a video sequence regarding its camera motion.

A. Segmenting the Sequence using Global Motion Estimation

To find a criterion for the decision of the video codec, the $RMSE$-curve over the whole input sequence is considered. We first apply a frame-by-frame global motion estimation algorithm [10]. Then, for each frame a global motion-compensated error frame is computed. The $RMSE$-value based on this error frame is taken as quality criterion of the estimation. After applying a temporal median filter on the $RMSE$-values of the whole sequence for noise reduction, we calculate a threshold which defines the two types of camera motion. If the camera is panning the global motion estimation with the 8-parameter perspective camera model performs quite well for this motion type. The $RMSE$-values will be low. If the camera tracks it is not possible to have a stable estimation of the global motion because the camera model does not fit anymore. The $RMSE$-values will increase in this sub-sequence. We calculate the variance of the whole $RMSE$-curve defining the threshold. We apply this algorithm on one synthetic test sequence "Room3D" with two pans and one camera track. Figure 1 shows the result for first test sequence. It can be seen that the threshold seperated the sequence in three sub-seqeunces. The first and the third are sub-sequences with camera pans ($RMSE$ below the threshold) and the second sub-sequence has a camera track ($RMSE$ above the threshold). That means, the first and the third sub-sequence can be coded with the sprite-based approach and the second sub-sequence is coded with the common H.264/AVC.

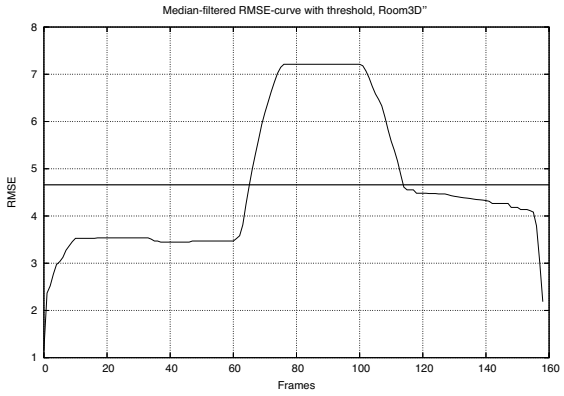
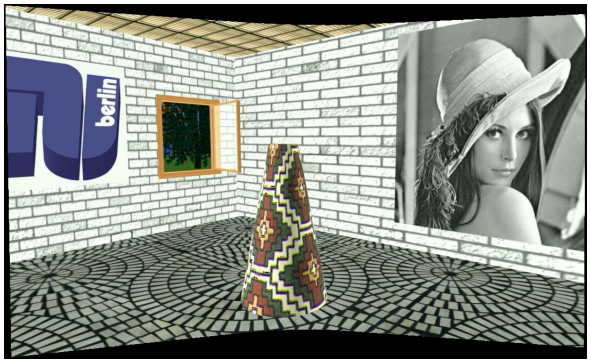
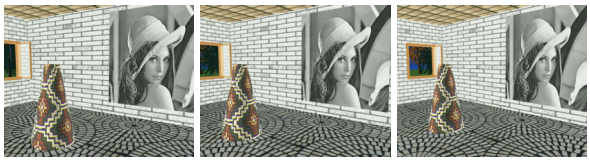


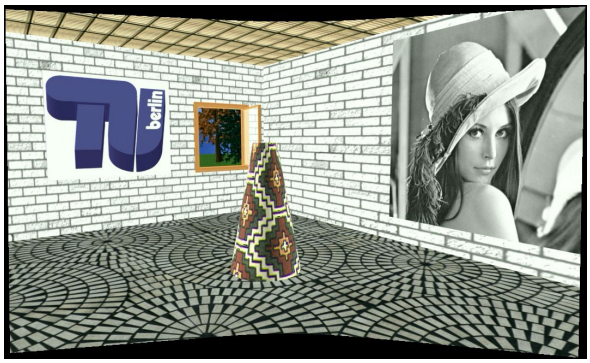
Fig. 1. RMSE (median filtered) "Room3D" of frame-by-frame global motion estimation with threshold for sequence segmentation



(a) pan, 1-67



(b) camera track (c) camera track (d) camera track

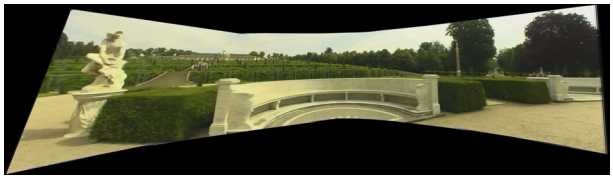


(e) pan, 116-160

Fig. 2. Video segments using GME of the synthetic sequence "Room3D"

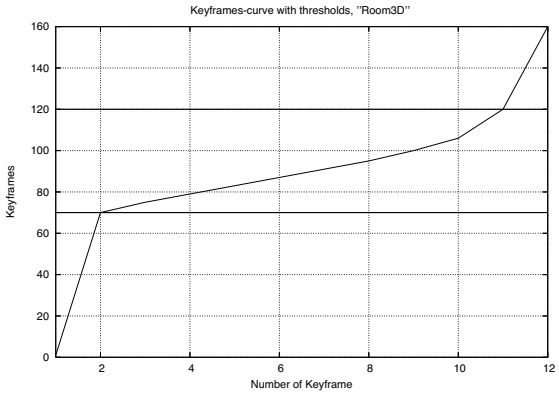
Figure 2 shows the three sub-sequences. The first and the third sequences are already transformed into background sprites [6]. For the second sub-sequence, the first, the middle, and the last frame are depicted. The second test sequence was captured with a hand-held camera and thus it is comparable

(a) camera track (b) camera track (c) camera track



(d) pan, 199-400

Fig. 3. Video segments using GME of the real sequence "Castle"



(a) Key frame curve "Room3D"

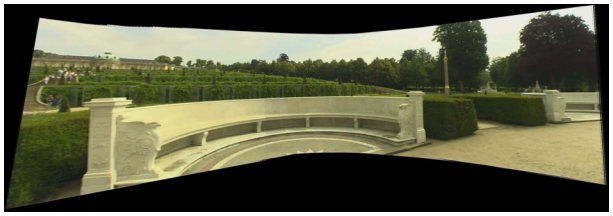
Fig. 4. Key frame curves of feature tracker using GRIC-score with turning points

with comon home videos. The first part of the sequence has a camera track and the second part is a camera pan. The way of capturing this sequence is lean on for example home videos. The result for the second video sequence can be seen in Fig. 3. For the first sub-sequence the first, middle, and last frame are shown again for the camera track. The second sub-sequence (pan) is also shown in background sprite-based representation.

B. Segmenting the Sequence using Feature Tracking and GRIC - Score

The second algorithm for analyzing the camera motion is based on a motion model selection approach. A slightly modified version of the well-known Kanade-Lucas tracker [**?**] is used to track feature points throughout the video sequence. Since the baseline between consecutive frames is small or the camera rotates about its center, a 2D perspective motion model, H (homography), can be used to transfer features from one frame to their corresponding positions in the second frame [11]. If the baseline increases during the tracking process and if the features belong to a 3D scene structure, the transfer error increases as well, i.e., the 2D motion model must be upgraded to a 3D motion model, F (epipolar geometry). Hence, the current frame is the intersection between both motion models

(a) camera track (b) camera track (c) camera track



(d) pan, 249-400

Fig. 5. Video segments using GRIC, "Castle"

and can be selected as a keyframe. The *Geometric Robust Information Criterion* (GRIC) [12] is a robust model selection criterion to extract such keyframes and is defined as:

$$GRIC = \sum \rho(e_i^2) + \lambda_1 dn + \lambda_2 k \qquad (1)$$

where $\rho(e_i^2)$ is a function of residuals:

$$\rho(e_i^2) = min\left(\frac{e_i^2}{\sigma^2}, \lambda_3(r-d)\right) \qquad (2)$$

The parameters are defined as follows: d is the dimension of the selected motion model (H has the dimension 2,whereas F has 3 dimensions), r is the dimension of the data (i.e. equal to 4 for two views), k is the number of the estimated model parameters (7 for F and 8 for H), n is the number of tracked features, σ is the standard deviation of the error on each coordinate and e_i is the distance between a feature point transferred through H and the corresponding point in the target image, or the Euclidian distance between the epipolar line of a feature point and its corresponding point in the target image, dependent on the selected model. λ_1, λ_2, and λ_3 are tuning parameters.

Initializing the first frame of the sequence as key-frame and proceeding frame by frame, the next key-frame is selected, if the GRIC value of the motion model F is below the GRIC value of H, i.e. a 2D motion model is no longer an accurate representation of the camera motion with respect to the 3D structure.

Having the set of keyframes of the considered sequence, we have to find a decision criterion for segmenting the sequence. For this, we draw the keyframe-curve over all frames of the sequence. Figure 4 shows the diagram for test sequence "Room3D". Defining the criterion for the sequence segmentation, we need to take a look at the characteristic of this curve. We can see that in the first part of the sequence, where the camera pans, no keyframes are selected by the GRIC-score algorithm except the initial first frame of the sequence. When the camera turns to a track keyframes are selected along this

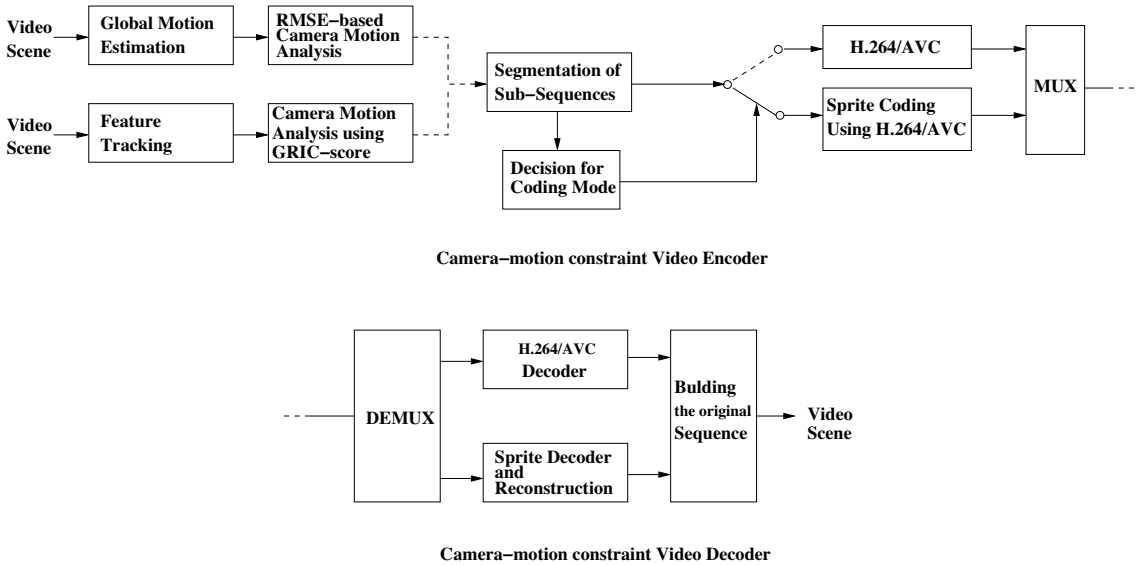


Fig. 6. Camera-motion constraint Video Codec (CMCVC)

sub-sequence. In the third part, the camera motion turns back to pan and only the last frame is selected for completion. We now have to find the turning points of the keyframe-curve to define borders of the camera motion types by calculating the maximum of the second derivative of this curve. It can be seen in Fig. 4 that the detected turning points segment the video sequence regarding to the camera motion types. For sequence "Room3D" the sub-sequences are slightly different to the GME-based sequence segmentation as described above (first part frame 1-70, second 71-120, third 121-160). The segmentation of the second test sequence "Castle" using the GRIC-score is more different to the first GME-based approach. Figure 5 shows the results. The performance of both algorithms are evaluated within a camera-motion constraint video codec. We will see the impact of using different kinds of video codecs and the role of the sub-sequence segmetation.

III. Camera motion-constraint Video Coding using Sprite Coding and H.264/AVC (CMCVC)

The described analysis algorithms are now embedded in a video coding system. A simplified block chart of the system is depicted in Fig. 6. For the analysis part, bot algorithms can be used as the pre-processing step. Having the borders calculated,

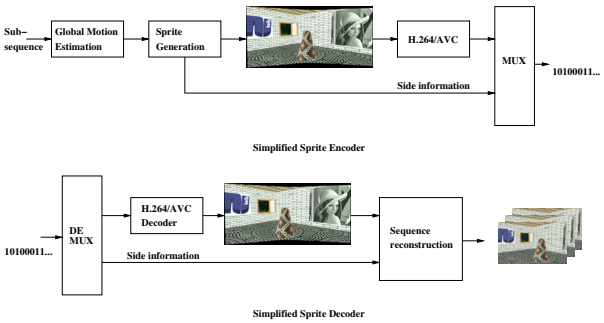


Fig. 7. Simplified Sprite Codec using H.264

the input sequence is segmented into a sub-sequence where the camera is on a fixed point and the movements are pan, tilt, zoom, rotation and a sub-sequence where the camera is tracked. Both labels go into the decider which switches the both possible video codecs. The sub-sequences with a fixed camera (motion types : pan, tilt, zoom, rotation) are coded with a sprite codec using the H.264/AVC. The two test sequences considered contain no moving foreground objects. Therefore, no object segmentation is needed. We use a simplified version of the object-based video codec proposed in [8]. If the GRIC-based analysis is used before the whole processing chain including short-term global motion estimation is applied. Otherwise, the GME-part can be left behind and the global motion parameters already calculated during the analysis part are used. This is the main advantage of the GME-based analysis algorithm. Then, a sprite generation algorithm is used to store all the frames into one image. In this work, we use a single background sprite. The background sprite image is coded using the H.264/AVC as an Intra-frame. The long-term motion parameters which control the background sprite generation and reconstruction are not coded. We assume that providing three byte per parameter is accurate enough for transmission. The 8-parameter perspective motion model is used. That means, we have 24 byte per frame as side information. Figure 7 illustrates this simplified sprite codec using H.264/AVC.

If the camera motion of the sub-sequence is track it is treated like a general video sequence, because is not possible to generate a background sprite for this type of camera motion. For this, we use the common H.264/AVC video codec.

IV. Experimental Results

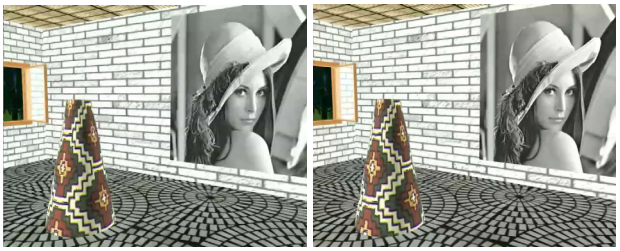
As already mentioned in Section II, we consider two test sequences for the experimental evaluation. The first sequence is a synthetic video called "Room3D" (720x576 pixel, 25 fps, 160 frames, progressive) with camera pan, track and pan again. The second test sequence is a real video captured with a

hand camera. It is called "Castle" (720x576 pixel, 25fps, 400 frames, progressive). Here we have a typical "home-made" scenario where the camera is tracked in the first part and panned in the second part of the sequence. Of course, this camera movement can occur on a large number of types of video content. The application for home videos is only an example. We code the sequences using our camera motion-constraint coding approach with the GME-based analysis (CMCVC - GME) and feature tracking and the GRIC-score (CMCVC - GRIC). We compare both approaches to each other and against the common H.264/AVC objectively using the $PSNR$-value for the objective quality. A GOP-size of 15 frames is used for the common H.264/AVC. For the prediction structure, we use the latest approved hierarchical B-frames. Additionally, we apply all new features inlcuding CABAC to have best coding performance for both compared codecs. Rate-distortion curves are shown in Fig. 10 ("Room3D") and Fig. 11 ("Castle"). It can be seen that we achieve a higher coding gain over a large bit rate range for the first test sequence. We achieve bit rate savings up to 50% against the common H.264/AVC with both approaches. However, for this sequence, the GME-based algorithm slightly outperforms the GRIC-based approach. For the second test sequence, we increase the coding performance in comparison to the common H.264/AVC in the lower bit rate range. The last point of the H.264-curve is the lowest coding limit for this sequence. We achieve up to 20% bit rate savings. Furthermore, we can extend the bit rate range in the lower direction using the CMCVC. The GRIC-based approach is here much better than the GME-based CMCVC.

Overall we can say that it is possible to increase the coding performance of the common hybrid video codecs by applying a pre-processing step where the camera motion type is automatically recognized. Based on this, we add a sprite coding mode for sequences where camera is fix and the movements are pan, tilt, zoom, rotation, etc..We can interpretate these first experimental results regarding two issues.

The first one is the ratio of the frames of the whole input sequence which are coded using the common H.264/AVC mode and the frames which are coded using the sprite coding mode (coding mode frame ratio (CMFR)). For the first sequence, the CMFR is 60:100 (ground truth). That means, the subsequence where the camera tracks contains 60 frames (coded with H.264 mode) and the two other sub-sequences where the camera pans contain 100 frames (coded with the sprite mode). The automatically segmented sub-sequences lead to a CMFR of $48{:}112 = 0.43$ for the GME-based algorithm and $50{:}110 = 0.45$ for the GRIC-based approach. These results are very similar, however, the GME-based algorithm achieves a slightly higher coding gain which means that less frames of the second sub-sequence (track) are coded with the sprite mode. For the test sequence "Castle", no exact ground truth is available because of the smooth crossover of the camera track and pan. It can be seen that here the GRIC-based approach segments the sequence better than the GME-based approach. Compared to sequence one, a coding gain can only be achieved for some parts of the bit rate range. The CMFR is 248:152 = 1.63 for GRIC-based and $198{:}202 = 0.98$ for GME-based analysis. The GME-based algorithm includes too many frames from the camera track for the sprite coding mode. This leads to distortions during the sprite generation process because for the frames taken from the camera track sub-sequence the assumed camera model is not valid anymore. The CMCVC using GRIC-based analysis produce a much better result. If we compare the best CMCVC result for sequence one and two we can see that the coding gain against the common H.264/AVC is not that high like for the first sequence for bit rate savings and coding gain over bit rate range. The reason is the much higher CMFR (1.63) for the second sequence.

The second issue for interpretation of the results is the



(a) H.264 (common use), 465.98 kbits/s, frame 73 (b) CMCVC (H.264 mode), 235.93 kbits/s (GME), frame 73

(c) Original (cut) (d) H.264/AVC (cut) (e) CMCVC (cut)

Fig. 8. Example for subjective comparison, CMCVC and H.264/AVC only, "Room3D" frame 73

(a) H.264 (common use), 223.85 kbits/s, frame 100 (b) CMCVC (H.264 mode), 172.84 kbits/s (GME), frame 100

(c) Original (d) H.264/AVC (e) CMCVC

Fig. 9. Example for subjective comparison, CMCVC and H.264/AVC only, "Castle" frame 100

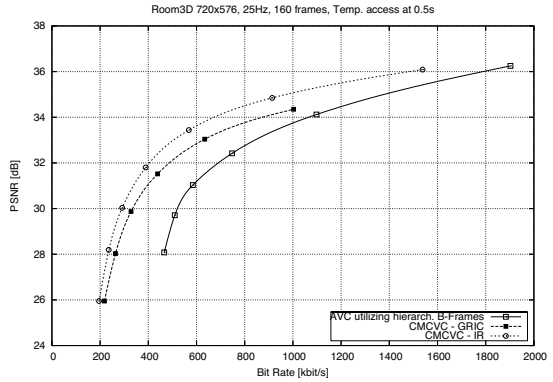


Fig. 10. Rate-distortion curves comparing the CMCVC and H.264, "Room3D"

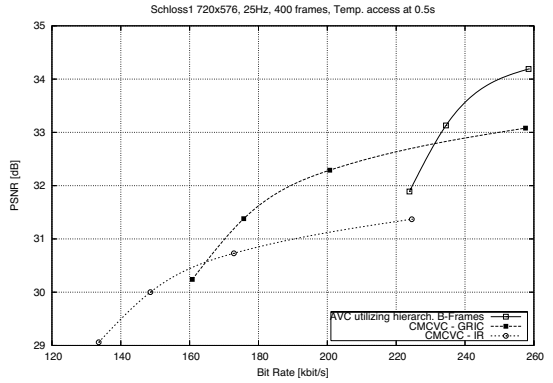


Fig. 11. Rate-distortion curves comparing the CMCVC and H.264, "Castle"

video content itself. Test sequence "Room3D" has high-frequent textures which is really hard to predict for a DCT-based motion-compensated codec. The use of a sprite codec improves the objective as well as the subjective quality. Figure 8 and 9 shows examples for the two test sequences. Example frames and close-ups are shown for H.264/AVC and CMCVC with H.264 mode. It can be seen that due to the bit rate savings from the sprite coding mode the sub-sequence part with the camera track can also be coded with higher subjective quality in comparison to the H.264 only. Thus, the sequences coded with CMCVC come along with bit rate savings together with increasing the subjective quality of the video.

V. Conclusions

We have proposed a camera motion-constraint video codec which combines recent developments in sprite coding and hybrid motion-compensated video coding. A pre-segmentation step of the input video into sub-sequence regarding to the camera motion has been introduced. Two apporaches are used and compared. Depending on the camera motion type, the optimal video codec is selected. We consider two options, a simplified sprite codec and the common H.264. This approach brings high coding performance in comparison to the use of the H.264 only.

VI. Acknowledgement

The work presented was developed within VISNET2, a European Network of Excellence (http://www.visnet-noe.org), funded under the European Commission IST FP6 programme.

We would like to thank Dr. Heiko Schwarz for the technical support.

References

[1] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the h.264/avc video coding standard," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 302–311, July 2003.

[2] T. Sikora, "Trends and perspectives in image and video coding," *Proceedings of the IEEE*, vol. 93, pp. 6–17, January 2005.

[3] ——, "The mpeg-4 video standard verification model," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 7, no. 8, pp. 19–31, February 1997.

[4] A. Smolic, T. Sikora, and J.-R. Ohm, "Long-term global motion estimation and its application for sprite coding," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 9, no. 8, pp. 1227–1242, December 1998.

[5] D. Farin, P. H. N. de With, and W. Effelsberg, "Video object segmentation using multi-sprite background subtraction," in *Int. Conf. on Multimedia and Expo (ICME)*, Taipei, Taiwan, Jun. 2004.

[6] M. Kunter, J. Kim, and T. Sikora, "Super-resolution mosaicing using embedded hybrid recursive flow-based segmentation," in *IEEE Int. Conf. on Information, Communication and Signal Processing (ICICS'05)*, Bangkok, Thailand, Dec. 2005.

[7] A. Krutz, M. Droese, M. Kunter, M. Mandal, M. Frater, and T. Sikora, "Low bit-rate object-based multi-view video coding using mvc," in *First International 3DTV-Conference*, Kos, Greece, May 2007.

[8] M. Kunter, A. Krutz, M. Droese, M. Frater, and T. Sikora, "Object-based multiple sprite coding of unsegmented videos using h.264/avc," in *IEEE Int. Conf. on Image Processing (ICIP2007)*, San Antonio, USA, Sep. 2007.

[9] A. Krutz, M. Kunter, M. Droese, M. Frater, and T. Sikora, "Content-adaptive video coding combining object-based coding and h.264/avc," in *Picture Coding Symposium (PCS)*, Lisbon, Portugal, Nov. 2007.

[10] A. Krutz, M. Frater, M. Kunter, and T. Sikora, "Windowed image registration for robust mosaicing of scenes with large background occlusions," in *Int. Conf. on Image Processing (ICIP06)*, Atlanta, USA, Oct. 2006.

[11] A. Z. R. Hartley, *Multiple view geometry*. Cambridge University Press, UK, 2003.

[12] P. S. Torr, "Geometric motion segmentation and model selection," *Phil. Trans. Royal Society of London*, pp. 1321–1340, 1998.