OBJECT-BASED MULTIPLE SPRITE CODING OF UNSEGMENTED VIDEOS USING H.264/AVC

Matthias Kunter, Andreas Krutz*, Michael Dröse*, Michael Frater**, and Thomas Sikora**

*Communication Systems Group
TU Berlin
Berlin, Germany

**School of IT and EE
University of New South Wales
Canberra, Australia

ABSTRACT

In spite of recent progress in the development of hybrid block-based video codecs, it has been shown that for low-bitrate scenarios there is still coding gain applying object-based techniques. We present a sprite-based codec, based on latest H.264 features using an inbuilt segmentation approach for scenes recorded by a rotating camera. The segmentation itself is built up on reliable background estimation from the sprite and short-term image registration. Moreover, we generate multiple sprites based on physical camera parameter estimation that overcome three of the main drawbacks of sprite coding techniques. First, the coding cost for the sprite image is minimized. Second, multiple sprites allow temporal background refresh and finally, registration error accumulation is kept very small. Experimental results show that this coding approach significantly outperforms latest H.264 extensions applying hierarchical B pictures.

Index Terms— Object-based video coding, sprite coding, multiple sprites, H.264/AVC

1. INTRODUCTION

The latest developments of advanced hybrid video coding techniques have resulted in several extensions of the well-known H.264/AVC video coding standard [1]. Due to its increased flexibility compared to earlier standards (MPEG-1,2,4), new coding strategies, e.g. the employment of hierarchical B pictures, can easily be applied [2]. However, it has been shown that redundancy reduction based on preceding video analysis outperforms classical hybrid coding for several types of videos. Especially for low bit-rates object-based video coding (OBVC), the use of sprite techniques is very promising [3].

Sprite-based coding has been successfully standardized as part of MPEG-4 visual. Nevertheless, there exist only a few publications evaluating the performance and comparing it to other techniques [3]. The reasons for this are manifold. One main problem is the generation of different VOPs, i.e. the object segmentation. It is unlikely that commonly recorded video carries the segmentation information. Therefore, applicable sprite codecs have to inherently segment the independently moving foreground objects. Recently some authors have presented their segmentation methods for sprite coding, e.g. [4]. We approach the segmentation problem by an extended background subtraction, where the background is estimated from the blended sprite itself. To increase robustness, we also exploit background estimation of the short-term image registration, which has less errors in the background region. A combination of both object masks yields very precise segmentation results.

A second problem for sprite-based coding techniques is the limited objective reconstruction quality of the movie background. Due to error accumulation and spatial distortions in the construction process the re-projected frame is degraded. Subjectively those errors are often irrelevant, but for SNR-based assessment the distortions increase with the temporal distance to the reference frame. Also the size and thus the coding cost of the sprite grows quickly due to geometrical properties of the underlying projection model. Moreover, if the camera rotates over 90 degree with respect to the reference frame a single sprite would be impossible to construct. In contrast to some heuristical approaches, Farin [4] proposes an image size-based algorithm for the construction of multiple sprites to overcome these problems. In our approach the sequences are parted accordingly to a rough estimation of rotation angles and focal lengths of the recording camera [5]. Thus, the assignment of frames to a certain sprite is not necessarily done in a sequential order. This fact is especially important for recurrent video background content. Inherently the multiple sprite approach enables a refresh of the decoded background which is helpful in scenes with slightly changing background objects.

We utilize H.264/AVC for coding of the objects. The texture of the multiple sprites is coded as intra frame and the segmented foreground is adapted to the macro block structure and coded in an $IBBB...$ frame structure with $GOP = 15$. The binary mask coding utilizes the binary arithmetic coder "M-Coder" of H.246/AVC and generates the side information together with the transformation parameters. With this work we provide an extended coding mode being chosen for low-bitrate ranges for certain types of videos with rotational camera motion.

The remainder of this paper is organized as follows. The camera calibration and the subsequent multiple sprite construction is described in Section 2. Section 3 presents the H.264/AVC-based sprite coding scheme which includes the devised segmentation algorithm. Experimental results are given in Section 4 while Section 5 concludes the paper and outlines future work.

2. MULTIPLE SPRITE CONSTRUCTION

In this section, we give an overview of the techniques applied to construct multiple sprites for optimal coding. First, global motion estimation, i.e. the short-term registration, is presented. It is the base for our coarse calibration technique and is also used for sprite construction and foreground segmentation (see next Section).

This work was developed within 3DTV (FP6-PLT-511568-3DTV), a European Network of Excellence funded under the European Commission IST FP6 programme.

M. R. Frater was supported in this work by the Australian Research Council under project DP0667074

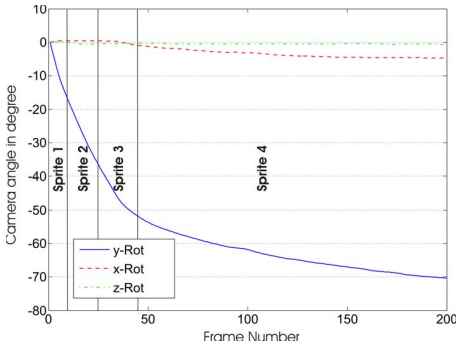


Fig. 1. Rotation angles for test sequence "Biathlon" and sequence partitioning according to y-rotation for multiple sprite construction

2.1. Image Registration

The performance of the whole sprite coder critically depends on the estimation of the background object motion. Therefore, it is very important to apply an image registration technique with very accurate estimation of the higher-order motion parameters, in our case of the 8-parameter perspective motion model. For this, a gradient-based approach is applied using additional techniques, such as windowed phase correlation based initialization, image pyramid decomposition, and image up-sampling applying wavelet filter. A detailed description of the utilized image registration algorithm can be found in [6].

2.2. Physical Parameter Estimation

Concatenating the short-term perspective camera parameters (homographies $\mathbf{H}_{n-1,n}$) in a recursive way yields non-exact long-term parameters representing the transformation $\mathbf{H}_{0,n}$ between any frame and the reference frame. These homographies are the base for an robust but coarse camera calibration technique, published in [5]. Here we exploit the fact that for common camera setups the homographies can be decomposed in a product of intrinsic and extrinsic camera parameter matrices

$$\begin{aligned} \mathbf{H}_{0,n} &= \mathbf{F}_n \mathbf{R}_{0,n} \mathbf{F}_0^{-1} \qquad (1) \\ &= \frac{1}{\alpha_{0,n}} \begin{pmatrix} r_{00} & r_{01} & f_0 r_{02} \\ r_{10} & r_{11} & f_0 r_{12} \\ r_{20}\alpha_{0,n}/f_0 & r_{21}\alpha_{0,n}/f_0 & r_{22}\alpha_{0,n} \end{pmatrix}, \end{aligned}$$

where $\mathbf{R}_{0,n}$ is the rotation matrix between frame 0 and n and $\mathbf{F}_n$ and $\mathbf{F}_0$ contain focal length values of both views. After computing the focal length ratio $\alpha_{0,n} = f_0/f_n$ we calculate the focal length of the reference frame as median of all solutions resulting from Equ. 1. This is done by exploiting orthogonality and constant vector norm constraints for the matrices $\mathbf{H}_{0,n}$. Knowing all focal lengths the rotation angles can finally be computed using trigonometrically properties of the center points of every image [5]. Figure 1 shows the rotation angles for sequence "Biathlon". The main motion is a left pan of the camera.

2.3. Sprite Generation

To split a video shot into several sequences, we first compute angle division for the rotation angle (φ_y or φ_x) with the maximum overall rotation $\Delta\varphi_{max}$. We minimize cost function C with respect to the number of angles M in order to find the number of sprites to be generated for one rotation plane.

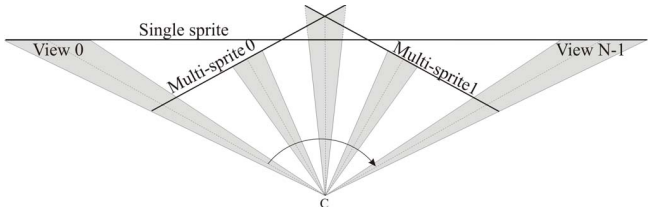


Fig. 2. Partition of a sequence into multi-sprites for panning camera with constant focal length

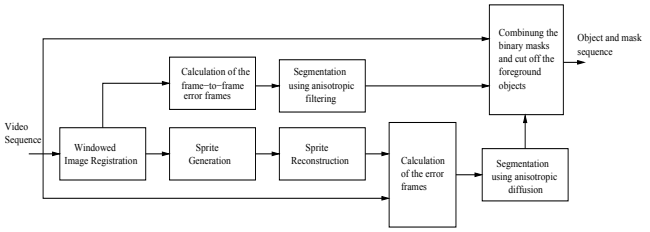


Fig. 3. Flowchart of the motion-based object segmentation using anisotropic filtering

$$C\left(\Delta\varphi_{max}, M\right) = \sum_{i=0}^{M-1} f_{max,i} \cdot 2\tan\left(\frac{\Delta\varphi_{max}}{2} + \frac{FOV}{2}\right) \qquad (2)$$

Thus, we optimize the sprite memory cost with respect to the sprite image size along one dimension. Figure 2 shows exemplarily the partition of a horizontal pan for the generation of two sprites. Note that the horizontal latitudes of the multi-sprites together is much smaller than the single sprite latitude. The reference frame is chosen to be the middle frame of a sub-sequence with respect to the rotation angle. Sprites are finally constructed by applying direct frame-to-mosaic registration and advanced median blending to remove artefacts from the foreground objects.

3. OBJECT-BASED CODING USING H.264/AVC

The sprite coding approach using H.264/AVC is presented in the following. An overview of the whole codec is shown in Fig. 4. After the multiple sprite generation, sprite-based robust foreground segmentation is used. Sprites, foreground textures, mask information and transformation parameters are then coded independently.

3.1. Object Segmentation

The segmentation approach is based on pre-computed error frames. For improving robustness and reliability, we compute error-frames for the re-projected sprite images and for the compensated adjacent frames using short-term transformation parameters. The former one is more sensitive in image regions where foreground objects appear while the latter is almost errorless in the background region. Combining the results yields very precise segmentation masks. A flowchart of the segmentation approach is shown in Figure 3. After generating the error image on the luminance channel, anisotropic Gaussian filtering is performed on the absolute values. This is achieved by locally

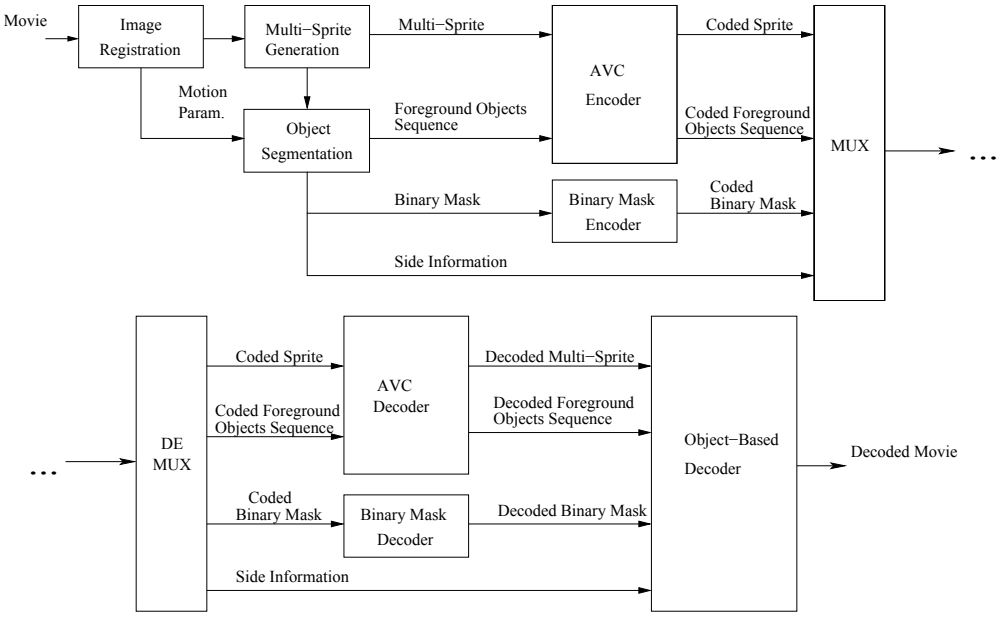
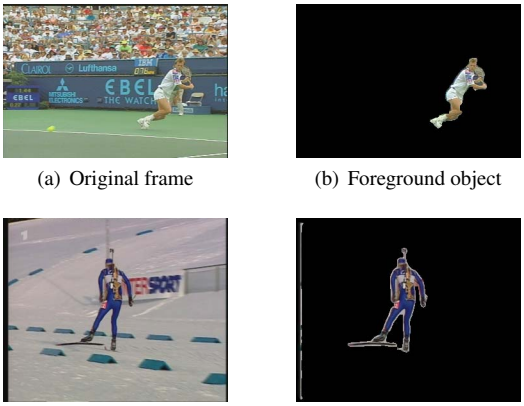


Fig. 4. Object-based video codec (OBVC) using multiple sprites and inherent segmentation



(a) Original frame (b) Foreground object

(c) Original frame (d) Foreground object

Fig. 5. Segmented foreground objects, (a)-(b): frame 176, "Stefan"; (c)-(d): frame 135, "Biathlon"

solving the anisotropic diffusion function

$$I_t = \mathrm{div}\left(\rho(|\nabla I|)\nabla I\right). \tag{3}$$

The images are then binarized applying an adaptive threshold depending on the overall error frame energy. After morphological filtering of the binary images and removing small objects the masks are combined using a logical AND-operator. Figure 5 shows segmentation results for sequences "Stefan" and "Biathlon". Since for "Stefan" a pre-generated mask was available we compared our segmentation results and measured an average Precision value $P = 81\%$. The average Recall is $R = 85\%$. Due to expansion of the masks to fit the macro block coding structure the whole object is segmented.

3.2. Texture and Binary Mask Coding

The textures of the sprite and the foreground objects are independently coded using H.264/AVC (see Figure 4). As coding software we used the JSVM version 6.5. To efficiently code the foreground the objects are expanded to fit to macro block structure. We applied coding in a hierarchical B picture scheme IBBB... with a GOP length of 15 frames, i.e. direct access every 0.5 seconds [2]. The sprite itself is coded as I picture.

The binary mask coding scheme uses the binary arithmetic coder "M-Coder", which is specified in the H.264/AVC standard. Eight contexts are initialized to model 8 possible spatial states for every pixel to code. In advance to the coding, the mask is divided into 16x16 blocks. The transformation parameters are coded as four frame corner point correspondences using 3 bytes per corner point and build the side information.

4. EXPERIMENTAL RESULTS

We conducted our experiments on sequences "Stefan" (352x240, 30 fps, 295 frames) and "Biathlon" (352x288, 30 fps, 195 frames). For comparison we also examined the single sprite case. As reference we used H.264/AVC applying hierarchical B pictures and the same structure as described in Section 3. Figures 6 and 7 show the resulted multiple sprites of both sequences. The costs for the binary mask and the transformation parameters were fixed at $9.54kbit/s$ for "Stefan" and $8.56kbit/s$ for "Biathlon". In Figures 9 and 10 the overall coding results for several combinations of coded sprite images and coded foreground objects plus side information are depicted. It can be observed that using multiple sprites yields much higher objective quality at same bit-rates than applying single sprites, which is up to $2.5dB$. Moreover, only the multiple sprite approach outperforms the reference coding scheme by up to $1.3dB$ at $78kbit/s$ for "Stefan" and more than $1.2dB$ at $60kbit/s$ for sequence "Biathlon". Thus, at bit-rates below $140kbit/s$ we achieve a coding gain for multiple sprites over latest hybrid coders. Comparing rate-distortion values to other sprite-based approaches is quite difficult since most works do concentrate only on the coding of the sprite image and do not code all objects, or if so, only code a small portion of the shot, even for the well-known "Stefan" sequence. A subjective comparison proving the excellent performance of the multiple sprite coder is shown

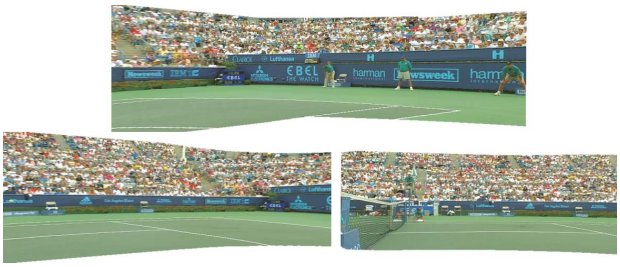


Fig. 6. Multiple sprites of sequence "Stefan", top: frames 0 to 244, bottom left: frames 245 to 261, bottom right: frames 262 to 299

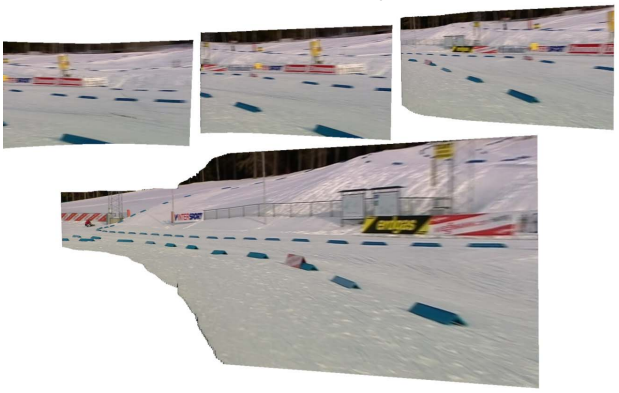
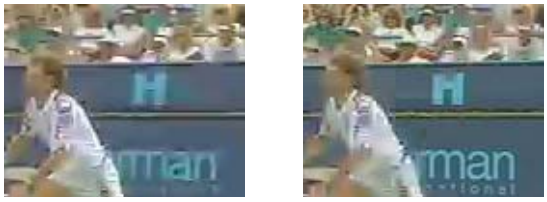


Fig. 7. Multiple sprites of sequence "Biathlon", top left: frames 0 to 9, top center: frames 10 to 22, top right: frames 23 to 46, bottom: frames 47 to 199



(a) H.264 - 88.95$kbit/s$ (b) OBVC - 88.3$kbit/s$

Fig. 8. Subjective quality comparision for frame 226 of "Stefan" for (a) H.264/AVC and (b) multi-sprite OBVC (ctuout)

in Figure 8.

5. CONCLUSION

We have presented a new object-based approach using existing standard codecs fore object-based video coding (OBVC). To improve coding efficiency we applied multiple sprites, which are constructed upon a robust estimation of physical camera parameters. Thus, the coding cost for the sprite images was minimized. For low bit-rates the codec yields better performance in comparison to latest block-based hybrid codecs. For further work we will improve the sprite reconstruction quality to improve the limited re-projection quality and expand the segmentation method to apply temporal object tracking.

6. REFERENCES

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

[2] H. Schwarz, D. Marpe, and T. Wiegand, "Analysis of hierarchical b pictures and mctf," in *IEEE Int. Conference on Multimedia & Expo (ICME'06)*, Toronto, Canada, 2006.

[3] Y. Lu, W. Gao, and F. Wu, "Fast and robust sprite generation for mpeg-4 video coding," in *IEEE Pacific Rim Conference on Multimedia (PCM'01)*, Bejing, China, Oct. 2001.

[4] D. Farin and P. H. N. de With, "Enabling arbitrary rotational camera motion using multisprites with minimum coding cost," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 4, pp. 492–506, April 2006.

[5] M. Kunter, A. Krutz, M. Mandal, and T. Sikora, "Optimal multiple sprite generation based on physical camera parameter estimation," in *Visual Communications and Image Processing (VCIP'07)*, San Jose, USA, Jan. 2007.

[6] 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.

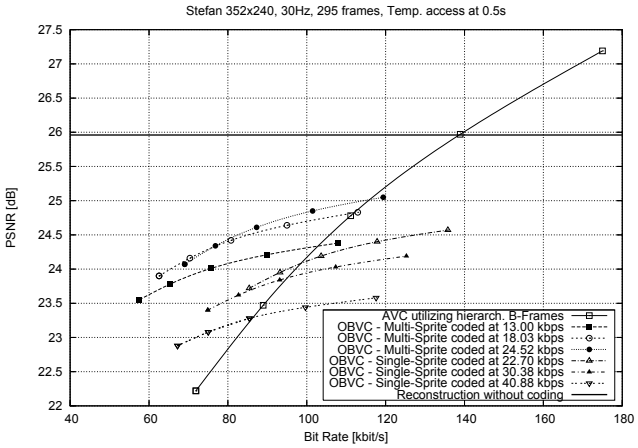


Fig. 9. Coding results for multi- and single sprite approach, sequence "Stefan"

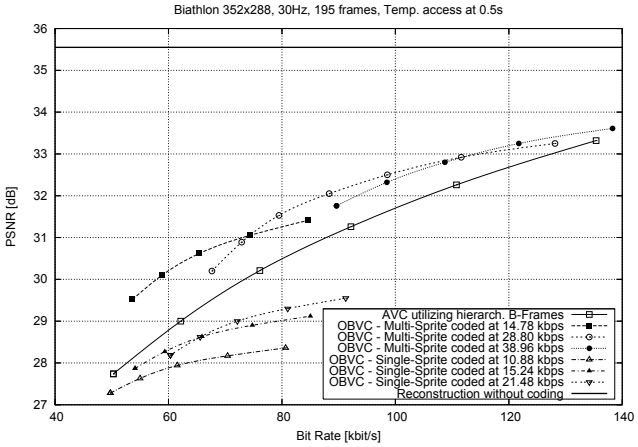


Fig. 10. Coding results for multi- and single sprite approach, sequence "Biathlon"