# Sprite-Based Video Coding Using On-line Segmentation

*Regis Crinon and Ibrahim Sezan*

SHARP Laboratories of America
5755 NW Pacific Rim Blvd.
Camas, Washington 98607, USA

## ABSTRACT

We address the problem of on-line sprite-based video coding in cases where scene segmentation is not available a priori or transmission of such segmentation information cannot be afforded due to low bit rate requirements. We propose an on-line segmentation method that can be integrated into an MPEG4 on-line sprite based video codec. The proposed method uses macroblock types as well as motion compensated residuals to perform the on-line segmentation. It produces a backround mosaic without requiring a priori foreground-background segmentation information. Our results demonstrate the coding efficiency and functionality benefits of the proposed approach.

## 1. INTRODUCTION

Sprite (mosaic) based video compression refers to building a mosaic image of a particular object within the scene (e.g., background) and using it as reference in predictive coding to increase the resulting compression efficiency. This is an added new dimension to classical video coding where predictive coding is limited to inter-frame coding only. The benefit of sprite-based coding using a background mosaic is realized when a part of the background that has been visible in the past (hence included in the mosaic) is uncovered at a current time instant. In this case, prediction from the mosaic increases the coding efficiency. Both off-line and on-line sprite-based video coding are currently being considered in the MPEG4 standardization [1,2]. In off-line coding, the background mosaic is constructed and transmitted to the decoder prior to coding and then used by directly warping it to form the background. In subsequent steps only warping parameters are transmitted. In on-line sprite-based coding, the background mosaic is generated during transmission using a priori background-foreground segmentation information, both at the encoder and the decoder at the same time. A fundamental difference in on-line mosaic based coding is the fact that predictive coding in reference to the dynamic mosaic is performed rather than direct warping and copying the contents of a static mosaic. Image mosaics were also used in [3] in an MPEG2-like video codec. The current methods in MPEG4 do utilize explicit foreground-background segmentation information that is assumed to be known a priori. This information is transmitted via binary shape mask coding. In this paper, we consider the cases where (i) segmentation information is not available a priori; or (ii) transmission of shape information may consume a significant amount of bits in lower bit rate environments.

We propose an on-line mosaic building technique that is fully integrated to an MPEG4 encoder and decoder, and is also capable of implicitly performing foreground rejection in creating the background mosaic. The advantages of the proposed technique are (i) ability of using sprite-based coding even when a priori segmentation information is not available (i.e., increased versatility); (i) creating a background mosaic only (without foreground) in the absence of explicit a priori segmentation information (i.e., increased functionality), and (ii) increasing visual image quality in low bit rates by not requiring explicit coding of shape information. Resulting background mosaics created as byproduct of compression can be stored and subsequently used as representative images of the video bitstream. In the proposed codec, the background video object plane (VOP) is assumed to move with the dominant image motion. On-line segmentation of the background and building of the associated mosaic is performed by making use of the dominant motion in the scene, and the macroblock type chosen by the encoder. Each macroblock can be coded 'intra', 'non-intra' or "mosaic" type, as specified in [1], where the choice by the encoder is made on the basis of motion-compensated prediction error.

In the following, we first describe the proposed method of on-line segmentation in a sprite-based MPEG4 video codec. We then present the simulation results reflecting the compression efficiency and added functionality benefits of the proposed approach.

## 2. ON-LINE SEGMENTATION

On-line segmentation is achieved by distinguishing the background and foreground pixels in a video frame — a rectangular VOP. The background mosaic is reconstructed using only the background pixels as they become visible. Automatic discrimination between background and foreground pixels is performed on the basis of a macroblock (16x16 luminance pixels) using its motion and coding type, called the "macroblock type" determined by the encoder. We now explain this process in some detail. We define the following quantities. We assume that the current frame is at time $t$ and the decoded version of the previous frame at time $t-1$ is available at both encoder and the decoder.

$(j,k)$: The position of the macroblock in the VOP being encoded. The coordinates $(j,k)$ represent the upper left corner of the macroblock. The size of a macroblock is $B_h \times B_v$ pixels, where $B_h$ is the horizontal dimension and $B_v$ is the vertical dimension of the macroblock, respectively.

MBType $(j,k)$: The macroblock type. This quantity can take the value *INTRA*, *INTER1V* (one motion vector for the whole macroblock), *INTER4V* (four motion vectors for each of the 8x8 blocks in the macroblock) and *MOSAIC*. The *INTER1V* and

*INTER4V* modes correspond to a prediction from the previously decoded VOP. In this case, prediction signal is computed using one or four motion vectors to align the current macroblock at $(j,k)$ with the corresponding 16x16 pixel array in the previous VOP. The *MOSAIC* type corresponds to prediction from the mosaic which has been updated last at the previous time. The prediction is obtained by backward warping the current macroblock to the mosaic using the parameters defining the global motion model between the current VOP and the previously decoded VOP at the previous time. This is because the mosaic used for prediction is always referenced at the previous time. The warping is performed on the basis of a particular global motion model, such as affine or perspective.

$e(j,k)$: The residual at the macroblock $(j,k)$. This residual results from computing the motion-compensated difference between a predictor (reference) image area and the data in the macroblock once the macroblock type has been selected. Depending on the macroblock type, the residual may have been obtained by predicting from the most recent mosaic using global motion vectors, or most recently decoded VOP using local motion vectors.

$f(j,k)$: The residual at the macroblock $(j,k)$ resulting from motion-compensating the macroblock by backward warping it onto the previously decoded VOP. The warping is performed according to the global motion model.

$q$: The current value of the quantizer step used by the encoder.

$\theta$ : A pre-defined threshold value greater or equal to 1. This threshold value is a function of the quantizer step $q$.

$\mathbb{W}_f$: Forward warping operator

$\mathbb{W}_b$ : Backward warping operator

$\underline{w}$ : Vector of warping parameters specifying the mappings $\mathbb{W}_f$ and $\mathbb{W}_b$. The vector $\underline{w}$ has zero, two, six or eight entries depending whether the warping is an identity, a translational, an affine or a perspective transform, respectively.

$\alpha$ : A pre-defined blending factor.

*Step 1 : Initialization of the mosaic:* The content of the mosaic is initialized with the content of the first VOP at time $t = t_0$. The shape map of the mosaic is initialized to 1 over a region corresponding to the rectangular VOP shape. The value 1 indicates that texture content has been loaded at this location in the mosaic.

$$S_t(\underline{R},t_0) = \begin{cases} VO_t(\underline{r},t_0) \ \ if \ \ VO_s(\underline{r},t_0) == 1 \\ 0 \qquad\quad otherwise \end{cases}$$

$$S_s(\underline{R},t_0) = \begin{cases} 1 \ if \ \ VO_s(\underline{r},t_0) == 1 \\ 0 \qquad\quad otherwise \end{cases}$$

where $S_s$ , $S_t$ , $VO_s$ , $VO_t$ represent the mosaic shape map, the mosaic texture, the decoded VOP shape map (rectangular here) and the decoded VOP texture fields, respectively. The mosaic shape $S_s$ and the decoded VOP shape $VO_s$ are binary alpha maps. In the mosaic shape map, the values 0 and 1 mean that the mosaic content is not determined and determined at this location, respectively. The position vectors $\underline{R}$ and $\underline{r}$ represent the pixel position in the mosaic and in the VOP, respectively. These spatial coordinates differ by an offset $\underline{\Delta}$ as described in [1].

*Step 2: Acquire the next VOP (time $t+1$) and select macroblock types:* Assuming that the current time is $t+1$, the mosaic that has been updated last at time $t$ exists. Residuals from previous VOP prediction are obtained by using conventional block matching. The global backward mapping $\mathbb{W}_b$ is used on every macroblock to get residuals resulting from mosaic prediction. Comparing the residuals, the encoder selects the macroblock type resulting in the least amount of residual or INTRA type when prediction is poor. At this time, the global backward mapping is also used to compute and record the residuals $f(j,k)$ obtained by predicting from the previously decoded VOP.

*Step 3: Encode and decode the VOP:* The encoder encodes and decodes the VOP at time $t+1$. The decoder decodes the bitstream to generate the same decoded VOP.

*Step 4: Create binary map to detect macroblocks belonging to foreground:* For every macroblock $(j,k)$ in the current rectangular-shaped VOP, build an initial object segmentation map $g(j,k)$, defined as follows, given the macroblock types and coded residuals of macroblocks at time t+1.

```
/* Test whether macroblock is of type MOSAIC */
if( MBType(j,k) = = MOSAIC )
{
/* Set segmentation map to 0 */
                    g( j,k) = 0
}

/* Test whether macroblock is of type INTER4V */
else if( MBType(j,k) == INTER4V )
{
/* Set segmentation map to 1 */
                    g( j,k) = 1
}

/* For macroblock types other than INTER4V and MOSAIC
   (i.e., INTRA and, INTER1V */
else
{
/* Compare residual from Global Motion Estimation on previous
VOP against macroblock residual */
```

```
        if (    f(j,k) > θ(q) e(j,k)  )
        {
/* Set segmentation map to 1 */
                g(j,k) =1
        }
        else
        {
                /* Set segmentation map to 0 */
                g(j,k) =0
        }
}
```

The binary map $g(j,k)$ represents an initial foreground/background segmentation. Detected foreground texture is denoted by setting $g(j,k) = 1$. This is the case whenever the macroblock is an *INTER4V* macroblock since it corresponds to the situation where there are four distinct and local motion modes. For all other macroblocks types other than *MOSAIC*, foreground is detected whenever residual from global motion compensation from previous VOP is larger than the encoded residual. In this situation, the global motion model does not correspond to the local dynamics of the foreground object.

*Step 5: Process segmentation map to make regions more homogeneous:* The purpose of this processing stage is to remove any isolated 1s or 0s in the binary map $g$ This can be achieved by means of a two-dimensional separable or non-separable rank filter. The operation of such a filter can be defined as follows. Consider a neighborhood of macroblocks $\Omega$ around the macroblock of interest at location $(j,k)$. Let M be the number of macroblocks in this neighborhood. Take the values of the segmentation map $g$ at each of the macroblocks belonging to the neighborhood system $\Omega$ and rank them in increasing order in an array A with M entries. Since $g$ can only take the value 0 or 1, A is an array of M bits where there are K zeros followed by (M-K) ones, K being the number of times the map $g$ takes the value 0 in the neighborhood $\Omega$. Given a pre-fixed rank $\rho$, $1 \le \rho \le M$, select the output of the filter as the $\rho$ th entry in the array A, that is A[ $\rho$ ]. The output of the filter at each macroblock location $(j,k)$ can then be used to generate a second segmentation map $h$, such that $h(j,k) = A$ [ $\rho$ ]. The result of applying this filter to the segmentation map is to remove spurious 1's or 0's in the initial segmentation, thereby making it more spatially homogeneous. If the filter is separable, the filtering operation above must be repeated along each dimension (horizontally then vertically or vice versa). At the end of the first pass, the output map $h$ must be copied to the map $g$ before the second pass is started.

*Step 6: Update mosaic according to new segmentation map to form the mosaic at time $t+1$:*
For every macroblock $(j,k)$ in the current VOP at time $t+1$, update the mosaic according to the following algorithm:

Given a macroblock position $(j,k)$, let $\underline{r} = \begin{bmatrix} j+l \\ k+p \end{bmatrix}$ where the variables $l$ and $p$ are such that $0 \le l \le B_h - 1$ and $0 \le p \le B_v - 1$. The variables $j+l$ and $k+p$ are used to denote the position of each pixel within the macroblock $(j,k)$.

For every value $l$ and $p$ in the range specified above, do

```
/*  Test  whether pixel belongs to VOP and whether mosaic content is already determined at this location */
if(  (VO_s( r, t+1) == 1) && ( S_s(R, t) == 1)  )
{
        /* Test whether macroblock has been classified as foreground macroblock */
        if  ( h(j,k) == 1 )
        {
                /* Warp mosaic forward but do not change its content */
                S_t(R,t+1) = W_f( S_t(R,t) , w)
        }
        else
        {
                /* Warp mosaic forward and update it by blending current VOP content in */
                S_t(R,t+1) = (1 − α )
                W_f( S_t(R, t), w)+α VO_t(r,t+1)   }
        /* Set mosaic shape to 1 to signal that content has been determined */
        S_s(R,t+1) = 1
}
/*  Test  whether pixel belongs to VOP and whether mosaic content is undetermined at this location */
elseif(  (VO_s( r, t+1) == 1) && ( S_s(R, t) == 0)  )
{
        /*  Set mosaic content to current VOP content */
        S_t(R,t+1) =  VO_t(r,t+1)
        /* Set mosaic shape to 1 to signal that content has been determined */
        S_s(R,t+1) = 1
}
```

*Step 7: Acquire the next VOP; Go to Step 2 and repeat the process till all VOPs are processed.*

## 3. RESULTS

All simulations were performed using the MPEG4 test sequence STEFAN, considering the following cases. We have used Momusys Video VM8.0 (Verification Model) MPEG4 software.

- Case 1: The object segmentation is known. Sprite-based coding is not invoked.
- Case 2: The object segmentation is known. On-line sprite-based coding is used for the background Video Object.
- Case 3: The object segmentation is not known. On-line sprite-based coding is used with the proposed on-line segmentation method.

For Cases 2 and 3, the size of the mosaic is set to 720x300 pixels; motion is modeled as affine and estimated using the MPEG4 VM software; and the blending factor is set to unity. In Case 3, the threshold $\theta$ is set to 1.05 and the blending factor $\alpha$ is set to 0.5 . The macroblock neighborhood system $\Omega$ is a 3x3 region (M=3) centered about the macroblock of interest. The value of the rank $\rho$ is 6. The PSNR results (for luminance) for cases 2 and 3 are furnished in Table 1. In Case 2, where a priori segmentation map is used with 2 Video Objects, both foreground (fg) and background (bg) performance are specified. In case of on-line segmentation, the PSNR figure applies to the entire video frame.

**Table 1**: Summary of Results

| q | Given Segmentation | On-Line Segmentation |
|---|---|---|
| 12 | 1028 Kbps<br>29.60 dB (bg)<br>27.44 dB (fg) | 1032 Kbps<br>29.56 dB |
| 27 | | 383 Kbps<br>24.69 dB |
| 29 | 378 Kbps<br>24.35 dB (bg)<br>22.79 dB (fg) | |

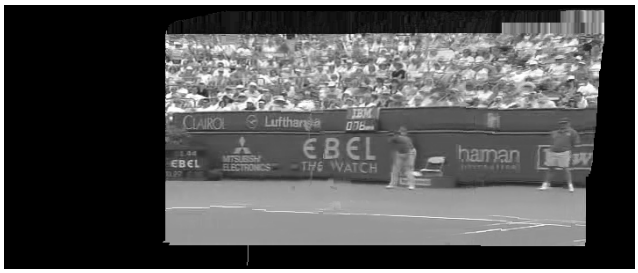
In Table 1 above, each entry shows the bit rates followed by the Luminance PSNR value obtained for the rectangular frames (cases 3 and 4) or for both the background and the foreground objects, respectively (cases 1 and 2).

The results indicate that coding efficiency provided by on-line sprite-based coding with automatic segmentation is comparable to the one obtained in the case of sprite-based coding applied to the background Video Object (Case 2) at the higher bit rate. On-line sprite-based coding with automatic segmentation becomes clearly favorable at low bit rates (384 Kbps) when bits spent to code the shape information (i.e., the segmentation map) becomes a relatively large overhead. At 1028 Kbps, the shape information represents about 5% of the transmitted data and the quantizer level is equal to q=12 in both cases. At low bit rates, the shape information represents about 12.5 % of the transmitted data. This relative increase in the amount of shape information penalizes the conventional on-line sprite-based coder and as a result, the quantizer level must be coarser (q=29) compared to the on-line segmentation-based coder where no shape information is transmitted (q=27). (Larger q implies coarser quantization and hence lower image quality).

We also like to note the benefits of sprite-based coding in general, with or without on-line segmentation. The comparable bit rate (1047 Kbps) was reached with MPEG4 VM without sprites at q=13 (compared to q=12). At the lower bit rate, the MPEG4 VM without sprites is coded at 404 Kbps at q=31 compared to sprite-based and on-line sprite based coding at comparable rates but at q=29 and q=27, respectively, where on-line segmentation is clearly superior to all.

In sprite-based compression, coding efficiency improvements come from scene re-visitation (when the camera pans back and forth), uncovering of background (when a foreground object covers and uncovers a portion of the background), and global motion estimation (no local motion vectors to transmit).

A sample mosaic generated on-line is shown in Figure 1. Further visual results will be shown during our presentation.




**Picture 1:** Background mosaic generated on line at a particular time instant in STEFAN.

## 4. SUMMARY

The proposed approach improves sprite-based coding further in terms of coding efficiency as well as functionality. In particular, it (i) improves coding efficiency, especially at low bit rates, by not requiring explicit shape coding; and (ii) does not require prior segmentation. In other words, on-line sprite-based coding with proposed on-line segmentation can be realized "entirely on line". From functionality point of view, the proposed approach also readily provides a background mosaic as a result of automatic, on-line segmentation. The background mosaic can be used as a representative image of the video sequence in indexing. Indeed, the foreground objects can be superimposed on the background mosaic in any desired fashion to express the scene dynamics.

## 5. REFERENCES

[1] MPEG-4 Video Verification Model Version 8.0, ISO/IEC JTC1/SC 29/WG11, document MPEG97/N1796, July 1997.

[2] F. Dufaux and F. Moscheni, "Background Mosaicking for Low Bit Rate Video Coding", Proc. ICIP96, Lausanne, Switzerland, Vol. I , pp. 673-676, 1996.

[3] M. Irani, S. Hsu and P. Anandan, "Video Compression using Mosaic Representation", *Signal Processing, Image Communication,* Vol. 7, pp. 529-552, 1995.