LOCAL BACKGROUND SPRITE GENERATION

Andreas Krutz[1]*, Alexander Glantz*[1]*, Michael Frater*[2]*, and Thomas Sikora*[1]

[1]Communication Systems Group, Technische Universität Berlin,
EN-1, Einsteinufer 17, 10587 Berlin, Germany, {krutz,glantz,sikora}@nue.tu-berlin.de
[2] School of Information Technology and Electrical Engineering, University of New South Wales,
Canberra ACT 2600, Australia, m.frater@adfa.edu.au

ABSTRACT

Background modeling of video sequences can be used in many different applications. For video object segmentation it is often applied in a background subtraction method. When conventional sprites like single or multiple sprites are used a background sequence has to be reconstructed from the model. The double mapping into the coordinate system of the sprite and back can lead to severe distortion of the background model and therefore to erroneous segmentation masks. We present a novel background modeling approach that lessens distortion. These so-called local background sprites are built for every reference frame independently and fit its original size. Experimental results show that this new approach clearly outperforms conventional background sprites in terms of PSNR.

1. INTRODUCTION

Background modeling – meaning the description of the background of a video sequence – is an important research field. Application scenarios range from video surveillance systems to video coding.

A so-called background sprite or single sprite generated over a certain number of frames can be such a background model. A background sprite is an image that typically is of large size and contains only the background pixels of a video sequence. This approach can be extended to so-called multiple sprites or super-resolution sprites. In case of multiple sprites the video sequence is divided into parts and for each part one background sprite is generated independently. In case of super-resolution a higher quality background sprite is generated for a sequence.

The potential for using background sprites for object-based video coding has been summarized in [1]. Furthermore, background modeling is an efficient means for video object segmentation. Various approaches using single or multiple background sprites in a background subtraction method have demonstrated that this kind of background model is very promising [2], [3]. However, the mapping of pixel content from various frames in a scene into a single sprite or a collection of multiple sprites may cause severe geometrical distorion of the background. For reconstruction of the background of a single frame a second mapping needs to be performed which causes additional distortion. Object segmentation is one application that is degraded by such distortion.

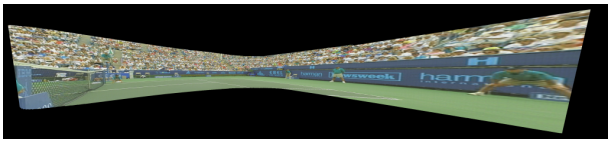


Figure 1. Single sprite, sequence "Stefan", reference frame 253

In our proposed local background sprite algorithm a mapping of content from many frames in a scene is performed for each individual frame for background construction. In other words, global motion estimation (registration) is performed from many adjacent frames into the frame where the background needs to be reconstructed. No backward mapping is required. Thus, our background sprites are local and there are as many individual sprites generated as frames exist in a sequence. This will result in a more precise background reconstruction compared to conventional global sprites.

This paper is organized as follows. Section 2 provides a short introduction to conventional background sprite techniques. Section 3 describes the new background modeling approach. The experimental evaluation in Section 4 shows the excellent performance of this new approach and Section 5 concludes the paper.

2. GENERAL BACKGROUND SPRITES

2.1. Single Sprites

A single sprite models the background of a given sequence in one single image. This image usually is of large size and contains only the pixels from the background of the sequence. An example of a single sprite is depicted in Figure 1.

For the creation of a single sprite a reference frame is chosen and all other frames of the sequence are warped into the coordinate system of the reference. For that long-term higher-order motion parameters are computed that describe this transformation [4]. First, short-term parameters are calculated using global motion estimation (GME) with the well-known 8-parameter motion model. These short-term parameters are then accumulated by simple matrix multiplication as shown in Figure 2.

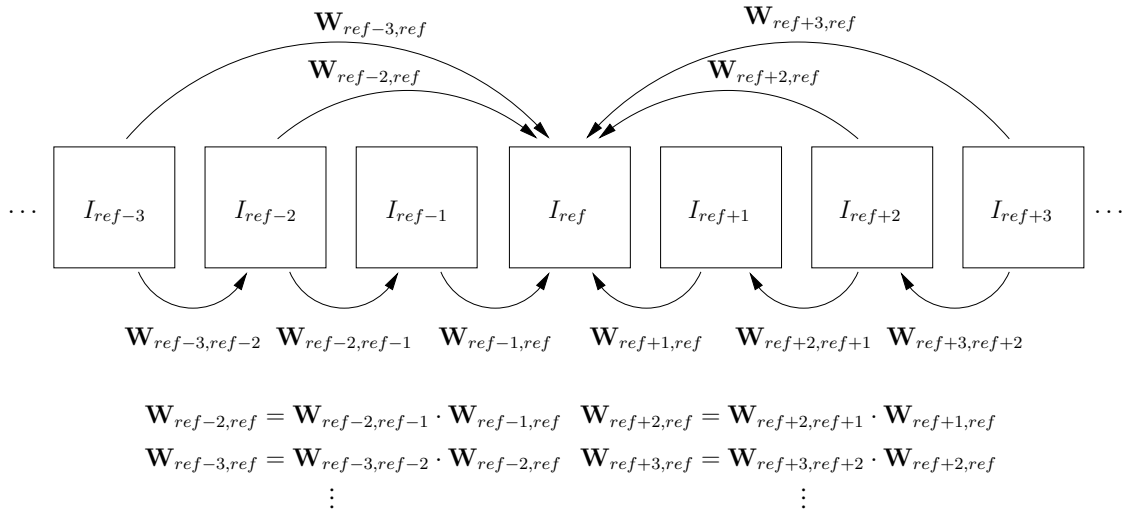


Figure 2. Creation of long-term motion parameters

(a) Frames 0-244

(b) Frames 245-261

(c) Frames 262-297

Figure 3. Multiple sprites, sequence "Stefan"

By transforming all frames using the long-term parameters into the coordinate system of the background sprite a stack of size $M \times N \times S$ is created where M and N are the dimensions of the final background sprite and S is the number of frames in the sequence. The images in this stack are then blended together to generate the background sprite. The intention is to eliminate the foreground objects in the background sprite. Blending filters normally used are the mean or the median of all pixel values lying in dimension S on top of each other.

2.2. Multiple Sprites

Depending on the camera motion the generation of a single sprite leads to severe distortion in the border area. These cause decreased quality of the background model and additionally increase coding costs which can be seen in Figure 1. In some cases where the camera pan exceeds an angle of 90 degrees generation of one single sprite is not even possible. These drawbacks have lead to so-called multiple sprites [2], [5].

For the generation of multiple sprites the camera motion is analyzed first. The next step is to split the video sequence into parts depending on the analyzed motion. Afterwards a background sprite is generated independently for every part as shown previously. An example of a multiple sprite is shown in Figure 3.

2.3. Super-resolution Sprites

Super-resolution is a technique that aims on increasing the quality of an image. A high-resolution counterpart is built from several images with lower resolution. These images can originate from one camera taking multiple images of a scene in time, from multiple cameras each taking one image in time or from the frames of a moving video camera. The idea of this approach is that an arbitrary point is visible several times.

This method can easily be extended to background sprite generation. When building a background sprite a video sequence is used. After global motion estimation and transformation into the coordinate system of the reference frame the pixel locations are rarely integer values. This feature can be used to generate a background sprite of higher resolution. For more details see [6] or [7].

3. LOCAL BACKGROUND SPRITES

A local background sprite specifies a model of the background. Other than general background sprites one model is built for every frame and not one model for the whole video sequence. Only the local temporal neighborhood of each reference frame is taken into account for sprite generation. The dimensions of a local background sprite match those of the reference frame. The idea is to minimize distortion in background regions. When a background frame is reconstructed from a general background sprite distortion can be severe. This is due to accumulated errors in the global motion estimation, non-ideal interpolation and the double mapping into the coordinate system of the background sprite and back.

The algorithm for modeling local background sprites for a given video sequence is depicted in Figure 4. Its

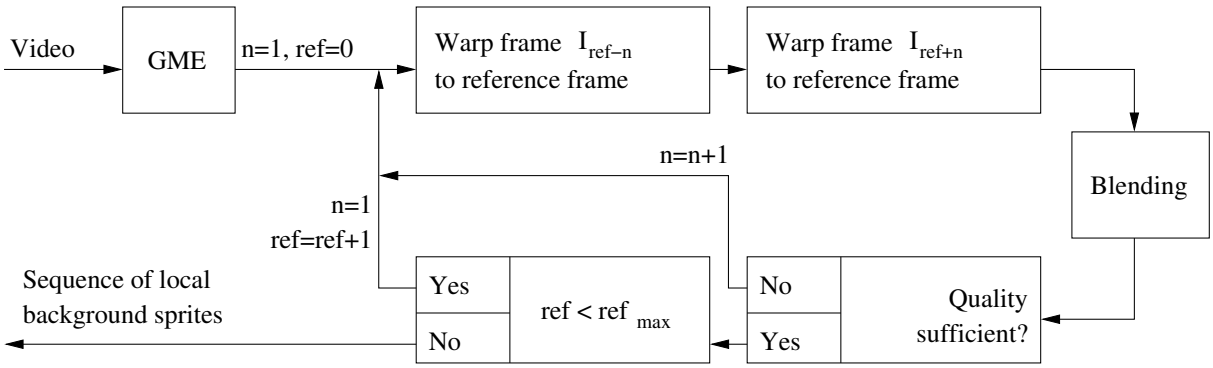


Figure 4. Modeling the background by means of local background sprites

different parts are explained in this section.

3.1. Global Motion Estimation

For global motion estimation a hierarchical gradient descent approach based on the Gauss-Newton method is applied as used in [8] or [9]. A block chart of the approach can be seen in Figure 5. The algorithm estimates the displacement between two temporally adjacent frames I_p and I_q of a sequence using the 8-parametric higher-order perspective motion model which is described by the following equations

$$x_q = \frac{m_0 x_p + m_1 y_p + m_2}{m_6 x_p + m_7 y_p + 1} \tag{1}$$

$$y_q = \frac{m_3 x_p + m_4 y_p + m_5}{m_6 x_p + m_7 y_p + 1} \tag{2}$$

where $(x_p\ y_p)^{\mathrm{T}}$ is the location of a pixel in frame I_p and $(x_q\ y_q)^{\mathrm{T}}$ its corresponding position in frame I_q. The parameters m_0 to m_7 describe the motion by means of translation, scaling, rotation, sheering and perspective transformation.

The algorithm first generates a 4-step image pyramid for the two frames to register. The image pyramid contains the original frames, two downsampled versions and one in the upsampled domain. For downsampling a 5-tap Le-Gall wavelet filter is used and for upsampling a 7-tap Daubechies wavelet filter. The first gradient descent step is performed on the coarsest resolution and is initialized with a translational motion model using the feature tracker presented by Kanade et al. [10]. The algorithm then performs a gradient descent step in every other layer of the image pyramid using the motion parameters from the step before as initialization.

Since the short-term displacement between two frames I_p and I_q is used several times while creating all local background sprites for a video sequence the motion parameters are computed in a preprocessing step. This means for a sequence with n frames the set T of transformation matrices

$$T = \{\mathbf{W}_{0,1}, \mathbf{W}_{1,2}, \ldots, \mathbf{W}_{n-2,n-1}\} \tag{3}$$

and its inverted correspondences

$$T_{inv} = \{\mathbf{W}_{0,1}^{-1}, \mathbf{W}_{1,2}^{-1}, \ldots, \mathbf{W}_{n-2,n-1}^{-1}\} \tag{4}$$

are computed where $|T| = |T_{inv}| = n-1$, $\mathbf{W}_{p,q}^{-1} = \mathbf{W}_{q,p}$ and

$$\mathbf{W}_{p,q} = \begin{bmatrix} m_{0,p,q} & m_{1,p,q} & m_{2,p,q} \\ m_{3,p,q} & m_{4,p,q} & m_{5,p,q} \\ m_{6,p,q} & m_{7,p,q} & 1 \end{bmatrix} \tag{5}$$

is the transformation matrix between frames I_p and I_q.

3.2. Warping and Blending

For every reference frame a local background sprite is built. The algorithm iteratively transforms neighboring frames into the coordinate system of the reference. This produces a dynamically growing image stack of size $M \times N \times S_t$ where M and N are the dimensions of the reference frame and $S_t = 2t + 1$ is the depth of the stack in step t. In step $t = 0$ the stack only contains the reference frame. This approach can be seen in Figure 6.

For the transformation of an arbitrary frame into the reference's coordinate system the short-term motion parameters from the preprocessing step are accumulated to generate long-term parameters which can be seen in Figure 2. The global motion estimation can only compute the displacement between two frames by approximation. Due to existing small errors and the accumulation the error in the long-term parameters grows larger with increasing temporal distance to the reference frame. Hence, the long-term parameters are used as initialization for another gradient descent step to reduce this error.

In every step t the images in the stack are merged together to build a preliminary local background sprite of size $M \times N$. For this purpose a so-called blending filter is used which here is a median filter. The median returns the middle value of an ordered dataset – in this case a set of luminance and chrominance values respectively. The advantage over using a mean filter is its robustness for outliers. Additionally, the median is always an element of the set itself and does not produce new values.

By successively adding temporally neighboring frames the foreground objects in the preliminary local background sprites are removed step by step. This is due to the area behind the foreground objects that is disclosed because of their movements. This can be seen in Figure 7. Depicted are the preliminary local background sprites for the "Stefan" sequence for various steps t. One can clearly see that

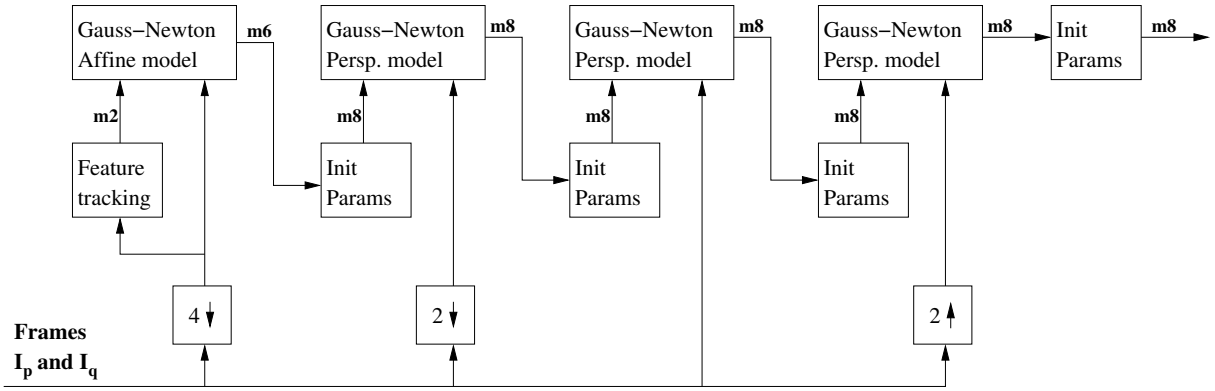


Figure 5. Global motion estimation algorithm

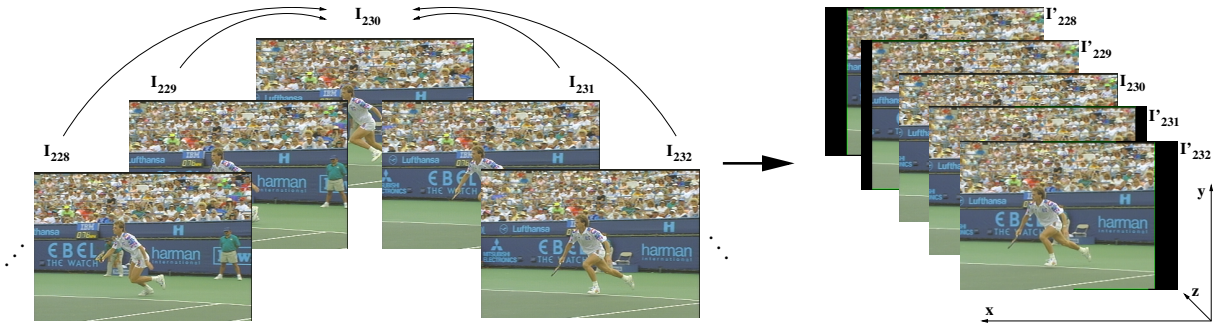


Figure 6. Creation of an image stack for the generation of a local background sprite, sequence "Stefan", reference frame 230

the foreground object has nearly completely vanished after eight blending steps.

It is possible to evaluate the quality of the background model in every step subjectively. However, an automatic evaluation criterion is desirable that stops the generation of the local background sprite when its quality is good enough and the foreground objects are removed sufficiently. An approach for automatic quality evaluation is presented next.

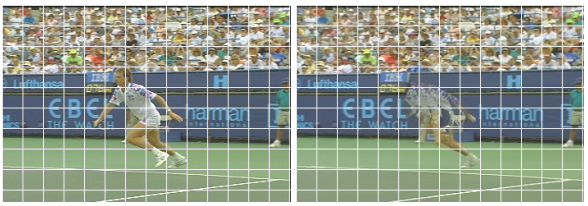
3.3. Quality Evaluation of Local Background Sprites

A possible measure for the difference between two images or frames is the root mean square error (RMSE). The RMSE between a reference frame $I_{ref}(x, y)$ and its preliminary local background sprite $I_{bs,t}(x, y)$ in step t is defined by

$$RMSE_t = \sqrt{\frac{1}{MN} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} (I_{ref}(i, j) - I_{bs,t}(i, j))^2} \quad (6)$$

where M and N are the dimensions of the reference frame and the preliminary local background sprite respectively. The preliminary local background sprite in step $t = 0$ is the reference frame itself so that $I_{bs,t=0} = I_{ref}$ and $RMSE_{t=0} = 0$. Since the foreground objects vanish step by step the RMSE value increases successively. Therefor, the difference of the RMSE values in two consecutive steps

$$\Delta RMSE_t = RMSE_t - RMSE_{t-1} \quad (7)$$



(a) Reference frame (b) Preliminary local background sprite, $t = 2$

Figure 8. Partitioning of images into blocks of size 25×25, sequence "Stefan", reference frame 230

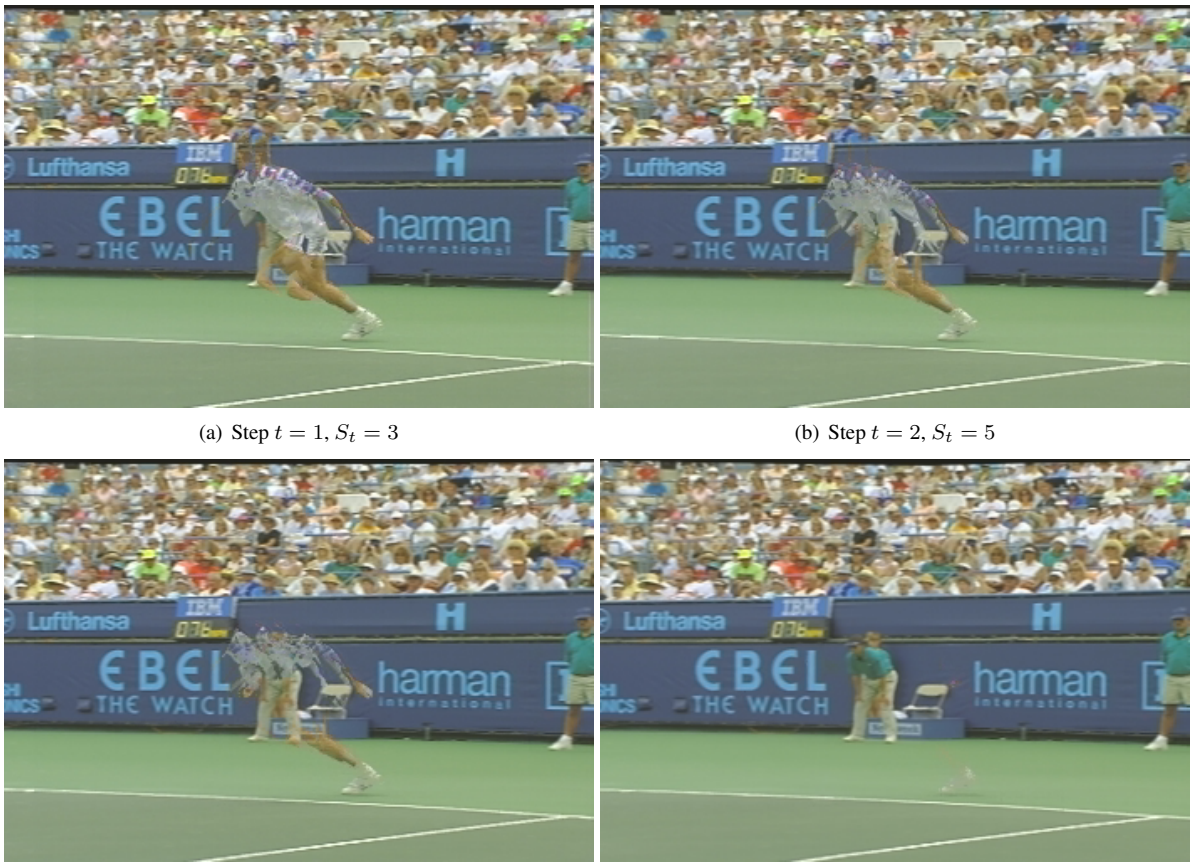
decreases. When the foreground objects are completely eliminated, the values $RMSE_t$ and $\Delta RMSE_t$ change only marginally.

At the beginning, foreground objects are still present in the preliminary local background sprite. Change in these areas leads to high values of $\Delta RMSE_t$. After several steps, most of the foreground is eliminated which leads to lower values of $\Delta RMSE_t$. It holds true that

$$\Delta RMSE_a >= \Delta RMSE_b$$

for $a <= b$. The value $\Delta RMSE_t$ can be interpreted as a measure for the information about the background that has been added to the local background sprite in step t.

Using the measure $\Delta RMSE_t$ is not without problems when only small foreground objects are present. Small objects only take a minor percentage of the whole frame.



(a) Step $t = 1$, $S_t = 3$

(b) Step $t = 2$, $S_t = 5$

(c) Step $t = 3$, $S_t = 7$

(d) Step $t = 8$, $S_t = 17$

Figure 7. Preliminary local background sprites, sequence "Stefan", reference frame 230

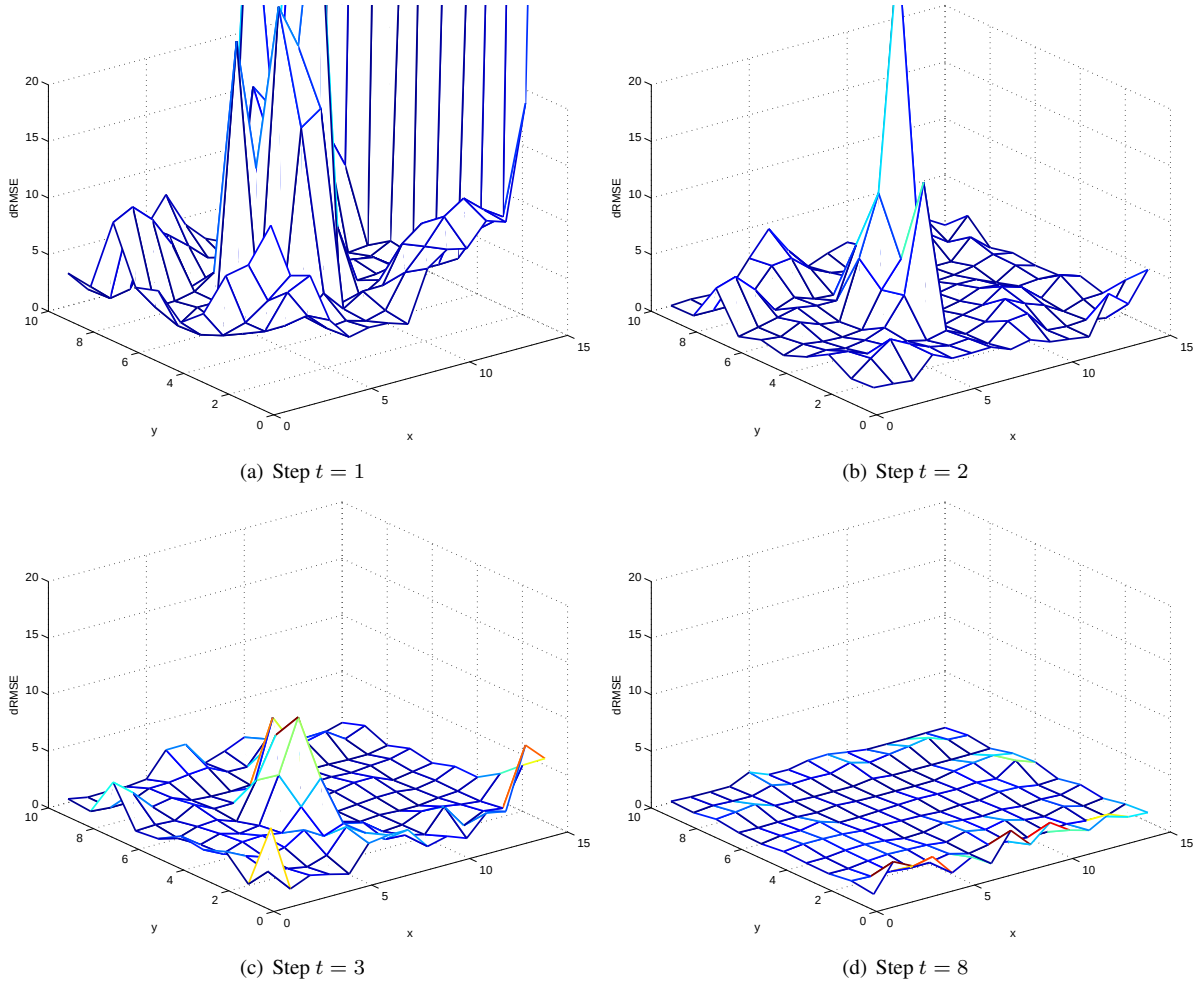
The influence of errors in these areas on the measure is small compared to the sum of errors in the background regions. In this case plotting $RMSE_t$ and $\Delta RMSE_t$ against time produces very flat curves which make a decision on the quality of the preliminary local background sprite very difficult. Therefore, we define matrices containing the blockwise calculated value $\Delta RMSE_t$, which we call dRMSE-matrices. Reference frame and preliminary local background sprite are divided into blocks of fixed size which can be seen in Figure 8. Within this work, various block sizes between 10×10 and 40×40 have been tested. The problem with small block sizes is that the profile of the dRMSE-matrices is of high frequency. With large block sizes the unwanted effect of averaging of large regions sets in again. In this work a fixed block size of 25×25 is used. The value $RMSE_t$ is then calculated for every block independently. Therefore, no averaging over the whole frame takes place. Distinct areas in the preliminary local background sprite can be evaluated independently. Furthermore, the difference to the block values in the step before is computed using Equation 7.

Figure 9 shows the corresponding matrices for the example in Figure 7. At the beginning the plot of the matrix is very wavy. With increasing steps t the matrices flatten successively. This means, the more temporally neighboring frames are transformed into the local background sprite's coordinate system the less information about the background of the reference frame is gained. The peak in the middle of the matrices in Figure 9 results from the moving tennis player and the area behind him that is disclosed. However, the RMSE in the background regions doesn't change significantly. After 8 blending steps (Figure 9(d)) – 16 frames and the reference frame have been blended together – the matrix is nearly flat in all regions. This corresponds with the results in Figure 7.

The quality of the preliminary local background sprites now is assessable in a very differentiated way. Assuming the generation of the local background sprite is to be aborted when there is no more information added in any region, meaning the matrix presented is flat in every region. We present three possible evaluation criteria for dRMSE-matrices.

The easiest way to evaluate the matrices is their maximum value. The maximum provides information about the biggest change of a block in the preliminary local background sprite. The curve of the maximum plotted against step t can be seen in Figure 10(a).

Another way to evaluate the matrices is the variance of their values. The variance provides information about the distribution of values in a dRMSE-matrix. The curve of the variance plotted against step t can be seen in Figure 10(b).



(a) Step $t = 1$ (b) Step $t = 2$

(c) Step $t = 3$ (d) Step $t = 8$

Figure 9. dRMSE-matrices using blocks of size 25×25, sequence "Stefan", reference frame 230

The last way to evaluate the matrices is inspired by the window-based feature tracker of Kanade et al. [10]. In their work, they define that a good feature is one that can be tracked well. This is the case when its texturizing is high. Therefore, the gradient $\nabla g = (g_x \; g_y)^{\mathrm{T}}$ is generated for each possible window. The gradients then are used to create a symmetric 2×2 matrix

$$\mathbf{Z} = \nabla g \nabla g^{\mathrm{T}} = \begin{bmatrix} g_x^2 & g_x g_y \\ g_x g_y & g_y^2 \end{bmatrix} \tag{8}$$

The eigenvalues of matrix $\mathbf{Z}$ are related to the texture. Two small eigenvalues mean a nearly homogeneous intensity profile. A large and a small eigenvalue correspond to a unidirectional texture pattern. Two large eigenvalues represent any highly textured pattern that can be tracked well. The dRMSE-matrices can be compared to the intensity profile of a feature window. On the contrary, we are looking for a nearly constant profile which means two small eigenvalues. Therefore, in every step t matrix $\mathbf{Z}$ is computed from the gradientes of the dRMSE-matrices. The curves of the eigenvalues plotted against step t can be seen in Figure 10(c).

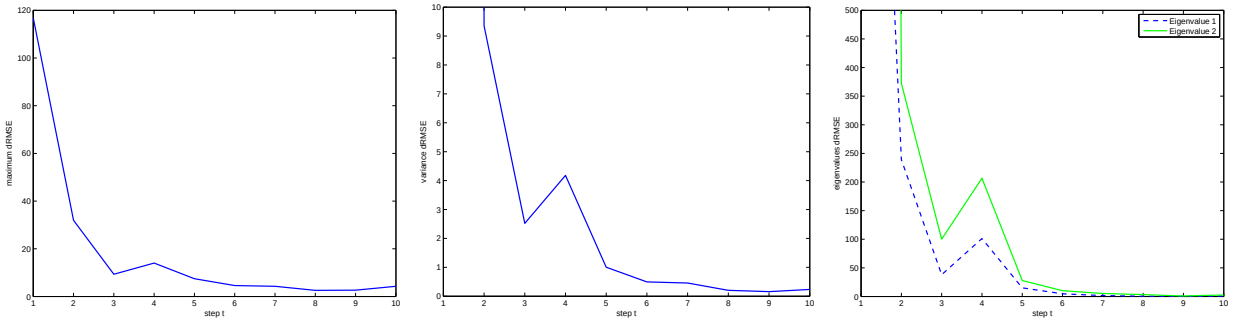
For every evaluation criterion used we assume local background sprite quality sufficient when maximum, variance and eigenvalues respectively lie below their predefined threshold values. The used thresholds can be see in Table 1. Outliers – step $t = 4$ in the examples in Figure 10 – come from sudden movements of the foreground objects or changes in camera speed.

Evaluation criterion	Threshold
Maximum	5.0
Variance	0.2
Eigenvalues	5.0

Table 1. Thresholds for evaluation criteria

4. EXPERIMENTAL EVALUATION

We have evaluated our approach using four test sequences. The first sequence is called "Allstars (Shot 1)" (352×288, 250 frames) and is recorded from a soccer broadcast of german television. It mainly consists of translational motion but is difficult to handle for global motion estimation because of its low-frequency content (green pitch). The second sequence is called "Mountain" (352×192, 100 frames) and is part of a BBC documentation. It shows a leopard chasing an animal while moving down a hill.



(a) Maximum (b) Variance (c) Eigenvalues

Figure 10. Various evaluation critera for dRMSE-matrices, sequence "Stefan", reference frame 230

This sequence consists also mainly of translational motion but has a high-frequency background (mountains). The third sequence is called "Race1 (View 0)" (544×336, 100 frames) and is part of an MPEG testset for multiview sequences. It shows a kart race. The forth sequence is the well-known "Stefan" sequence (352×240, 300 frames). Both the content and the motion of the sequence are very complex. It consists of low-frequency (tennis court) and high-frequency parts (audience). The camera motion is composed of translation, scaling and perspective transformation.

For evaluation of background quality we compute the background PSNR of the model using equation

$$PSNR = 10 \cdot \log_{10}\left(\frac{255^2}{MSE}\right) \tag{9}$$

for the sequence's luminance part. We have generated ground truth masks of the foreground objects for all sequences to compute the PSNR only between the background pixels of the original reference frame and the background model.

First, we compare all possible evaluation criteria, i.e. maximum, variance and eigenvalues, for the generation of local background sprites. Figure 11 shows the background PSNR for the background models generated. Table 2 shows the mean PSNR values. One can clearly see that for every sequence the maximum criterion produces the best background quality (bold values in Table 2). This is due to the fact that local background sprite generation stops using the least temporally neighboring frames when the maximum criterion is chosen.

The high values in the last quarter of sequence "Allstars (Shot 1)" (Figure 11(a)) come from the camera that is static in that part of the sequence. The translational initialization of the global motion estimation algorithm is very exact and the gradient descent algorithm produces very small errors.

The PSNR profile of sequence "Mountain" in Figure 11(b) is almost constant. Except for low values at the beginning and at the end of the sequence. These are explainable with errors in registration.

The low values around frame 30 of sequence "Race1 (View 0)" (Figure 11(c)) come from a fast camera pan that produces big translational motion. Contrary to the last part of sequence "Allstars (Shot 1)" the translational initialization in that part of the sequence produces higher errors for the global motion estimation.

The peaks around frames 25, 50 and 120 and the drop in the last 30 frames of sequence "Stefan" (Figure 11(d)) have similar causes. The camera motion is static or nearly static in parts with peaks and moves very fast at the end of the sequence.

Next, the local background sprites are compared to common background models. We use on one hand the local background sprites provided by the worst criterion – which in all cases is variance, except for sequence "Race1 (View 0)" which is eigenvalues. On the other hand we use reconstructed background sequences from single, multiple and super-resolution sprites repspectively. The plots for the comparison can be seen in Figure 12. Table 3 shows the mean PSNR values. One can clearly see that the new approach outperforms common background models even when the worst evaluation criterion is used at about $2 - 6$ dB (bold values in Table 3).

For sequence "Stefan" (Figure 12(d)) only the first 250 frames are taken into account for the mean PSNR value as the super-resolution sprite existed only for that range. The three peaks in the curve for multiple sprites result from the generation process that adds the original pixels at the position of the reference frame in the background sprite. Therefore, the quality at the three reference frames is nearly optimal.

5. CONCLUSION

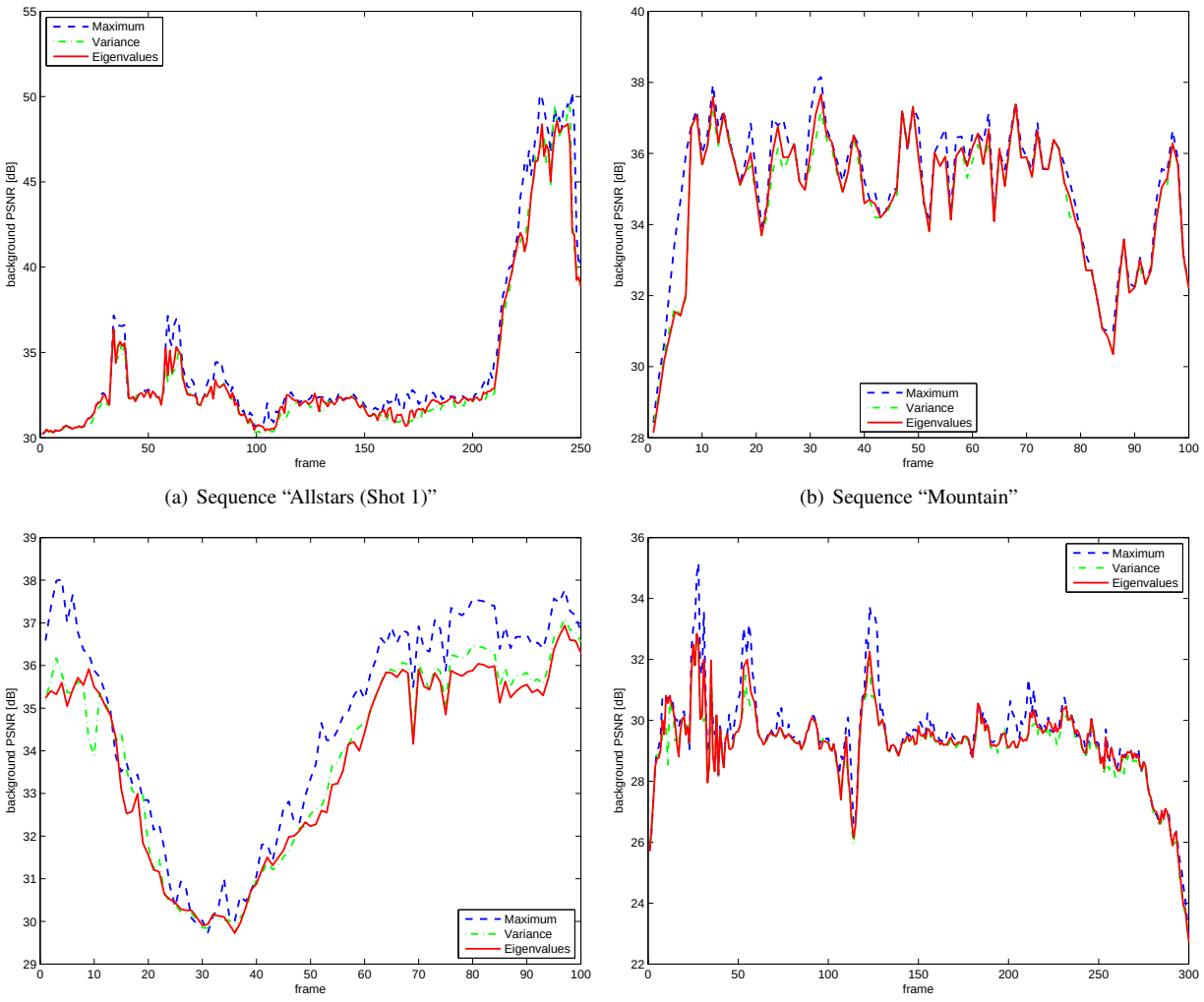
We have presented a novel approach for background modeling of video sequences. Conventional background sprites model the background of the sequence in one image that usually is of large dimensions. Other than that local background sprites are generated for each reference frame individually and are of the same size as the reference. When using conventional background sprites for applications like image segmentation the frames containing the background information have to be reconstructed from the sprite image. This step is not necessary when using local background sprites. Therefore, distortion is reduced. It has been shown that the quality of the background model us-

Sequence	Algorithm	PSNR [dB]
Allstars (Shot 1)	Local background sprite (maximum)	**34.4345**
	Local background sprite (variance)	33.7043
	Local background sprite (eigenvalues)	33.8240
Mountain	Local background sprite (maximum)	**35.1592**
	Local background sprite (variance)	34.7299
	Local background sprite (eigenvalues)	34.8040
Race1 (View 0)	Local background sprite (maximum)	**34.6666**
	Local background sprite (variance)	33.9092
	Local background sprite (eigenvalues)	33.7659
Stefan	Local background sprite (maximum)	**29.5146**
	Local background sprite (variance)	29.0908
	Local background sprite (eigenvalues)	29.2079

Table 2. Mean background PSNR values for local background sprites generated using all evaluation criteria

Sequence	Algorithm	PSNR [dB]
Allstars (Shot 1)	Single sprite	29.0938
	Local background sprite (variance)	**33.7043**
Mountain	Single sprite	28.8877
	Local background sprite (variance)	**34.7299**
Race1 (View 0)	Single sprite	30.1350
	Local background sprite (eigenvalues)	**33.7659**
Stefan (1-250)	Multiple sprites	24.7312
	Super-resolution sprite	27.4404
	Local background sprite (variance)	**29.4479**

Table 3. Mean background PSNR comparing reconstructed single/multiple/super-resolution sprites with local background sprites (worst evaluation criterion)



(a) Sequence "Allstars (Shot 1)"

(b) Sequence "Mountain"

(c) Sequence "Race1 (View 0)"

(d) Sequence "Stefan"

Figure 11. Background PSNR of local background sprites using various evaluation criteria for dRMSE-matrices

ing this new technique clearly outperforms conventional background models.

Further work is required concerning all fixed thresholds used in this work. The block sizes of the dRMSE-matrices can be adaptively adjusted using a preliminary segmentation that provides information about the mean object size in the reference frame. The threshold values for the evaluation criteria can also be adaptively adjusted. The curves are similar to an exponential path. Curve fitting techniques can estimate the path using past values and adjust the threshold depending on a possible convergence.

References

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

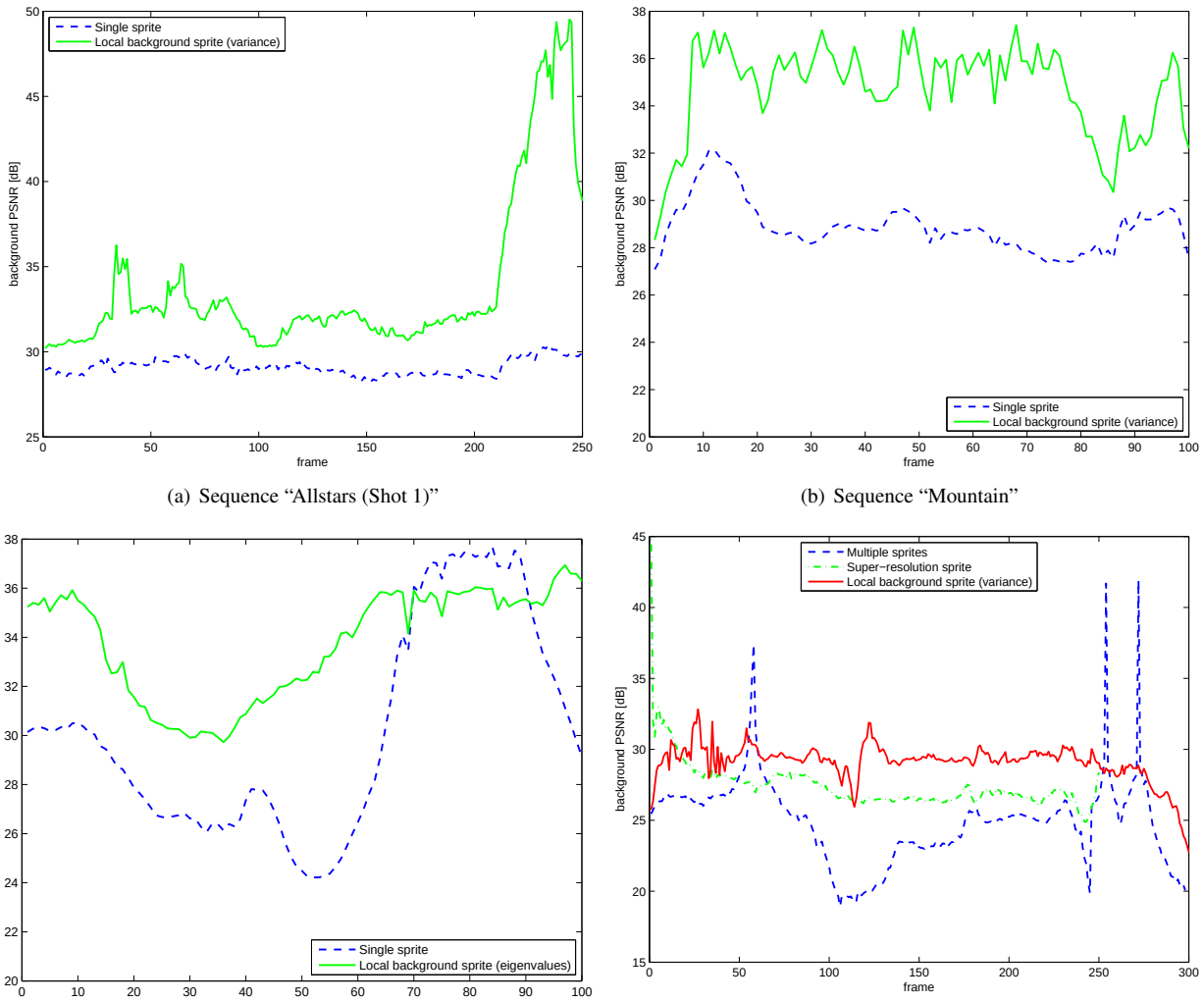
[2] D. Farin, P. H. N. de With, and W. Effelsberg, "Video object segmentation using multi-sprite background subtraction," in *Int. Conf. on Multimedia and Expo (ICME)*, Taipei, Taiwan, June 2004.

[3] A. Krutz, M. Kunter, M. Mandal, M. Frater, and T. Sikora, "Motion-based object segmentation using sprites and anisotropic diffusion," in *8th International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS)*, Santorini, Greece, June 2007.

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

[5] 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 International Conference on Image Processing (ICIP'07)*, San Antonio, USA, Sept. 2007.

[6] G. Ye, M. Pickering, M. Frater, and J. Arnold, "A robust approach to super-resolution sprite generation," in *Int. Conf. on Image Processing (ICIP'05)*, Genova, Italy, Sept. 2000.



(a) Sequence "Allstars (Shot 1)"

(b) Sequence "Mountain"

(c) Sequence "Race1 (View 0)"

(d) Sequence "Stefan"

Figure 12. Comparing background PSNR of reconstructed single/multiple/super-resolution sprite with local background sprite (worst evaluation criterion)

[7] 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.

[8] A. Krutz, M. Frater, and T. Sikora, "Improved image registration using the up-sampled domain," in *Int. Conf. on Multimedia Signal Processing (MMSP'06)*, Victoria, Canada, Oct. 2006.

[9] Frederic Dufaux and Janusz Konrad, "Efficient, robust, and fast global motion estimation for video coding," *IEEE Transactions on Image Processing*, vol. 9, pp. 497–501, 2000.

[10] S. Baker and I. Matthews, "Lucas-kanade 20 years on: A unifying framework," *International Journal of Computer Vision*, vol. 56, pp. 221–255, Feb. 2004.