

# Efficient, Robust and Fast Global Motion Estimation for Video Coding

*Frédéric Dufaux, Janusz Konrad*

1

Compaq Computer Corporation  
Cambridge Research Laboratory  
One Kendall Sq.,  
Cambridge, MA 02139, USA  
[dufaux@crl.dec.com](mailto:dufaux@crl.dec.com)

INRS-Télécommunications  
Institut National de la Recherche Scientifique  
16 Placedu Commerce,  
Verdun, QC, H3E 1M1, Canada  
[konrad@inrs-telecom.quebec.ca](mailto:konrad@inrs-telecom.quebec.ca)

EDICS number: IP1.11, IP1.1

## ABSTRACT

In this paper, we propose an efficient, robust, and fast method for the estimation of global motion from image sequences. The method is generic in that it can accommodate various global motion models, from a simple translation to an 8-parameter perspective model. The algorithm is hierarchical and consists of three stages. In the first stage, a low-pass image pyramid is built. Then, an initial translation is estimated with full-pixel precision at the top of the pyramid using a modified  $n$ -step search matching. In the third stage, a gradient descent is executed at each level of the pyramid starting from the initial translation at the coarsest level. Due to the coarse initial estimation and the hierarchical implementation, the method is very fast. To increase robustness to outliers, we replace the usual formulation based on a quadratic error criterion with a truncated quadratic function. We have applied the algorithm to various test sequences within an MPEG-4 coding system. From the experimental results we conclude that global motion estimation provides significant performance gains for video material with camera zoom and/or pan. The gains result from a reduced prediction error and a more compact representation of motion. We also conclude that the robust error criterion can introduce additional performance gains without increasing computational complexity.

## 1. Introduction

Video compression techniques rely on the reduction of statistical redundancies in the data and on the exploitation of the human visual system limitations. As the luminance and color in dynamic images are most correlated in the direction of motion, any redundancy removal is highly dependent on motion information. For this purpose, a nonlinear prediction technique, called *motion compensation* (MC), has been developed and used with remarkable success. It consists of two steps: estimating motion between successive video frames and then predicting the current frame from previously transmitted frames using the motion information. Motion compensation is a core technology of all video compression standards developed to date, such as MPEG-1 [1] and MPEG-2 [2].

A number of very different motion estimation algorithms have been proposed in the literature [3]. In general, motion in a sequence of images results from motion of a camera (e.g., pan, zoom) and from displacements of individual objects composing the scene. The former is often referred to as *global motion* and the latter as *local motion*. Most motion estimation techniques ignore this aspect and make no distinction between the global and local

<sup>1</sup>J. Konrad was with Compaq Computer Corporation, Cambridge Research Laboratory when this work was performed.

motion; global motion is taken into account only implicitly through local estimates. For instance, in terms of motion compensation, MPEG-1 [1] and MPEG-2 [2] rely on a local motion model of blocks under translation. However, it is usually advantageous to process global and local motion separately. Not only does it result in more precise estimates, but it also leads to a more compact representation of the motion information. Consequently, techniques for *global motion estimation* (GME) have been proposed [4][5][6][7][8][9]. Following these recent developments, two tools based on GME, *dynamicsprites* (DS) and *global motion compensation* (GMC), have been investigated in the framework of MPEG-4 [10].

GME techniques differ as far as the parametric model of camera motion is concerned and in the way that the model parameters are estimated. The following motion models have been used in the past: 3-parameter model corresponding to pan and zoom [5][6][8], 4-parameter model corresponding to pan, zoom and rotation [4], 6-parameter affine model [7][9], 8-parameter quadratic model [9], 8-parameter perspective model [9]. Parameter estimation methods can be classified into three categories: direct minimization of the prediction error by a differential technique [4], direct minimization of the prediction error by a matching technique [5][7], two-step method consisting of local motion estimation followed by fitting of global motion parameters [6][8][9].

In this paper, we propose a new GME technique [11] which is based on earlier work of Szeliski [12]. The camera motion is parameterized by a perspective model superior to the more restrictive 3-, 4- and 6-parameter models mentioned above. The prediction error is minimized by a gradient descent applied over a pyramid of input images. Since the method does not depend on a locally computed motion field, unlike [6][8][9], the estimation process is more robust. Furthermore, the resulting algorithm is cost-efficient in terms of the computational complexity. Compared to Szeliski's approach [12], the proposed technique includes a preliminary matching step to obtain a good initial guess of the solution and thus assuring convergence of the gradient descent in the presence of large displacements. Furthermore, the algorithm minimizes a truncated quadratic error function instead of the standard quadratic measure. The method is thus more robust to outliers, which would otherwise bias the estimation.

## 2. Global Motion Estimation

The proposed GME technique is designed to minimize the sum of squared differences (SSD) between the current frame  $I$  and the motion-compensated previous frame  $I'$ ,

**Equation 1:** 
$$E = \sum_{i=1}^N e_i^2, \text{ with } e_i = I'(x'_i, y'_i) - I(x_i, y_i),$$

where  $(x_i, y_i)$  denotes the spatial coordinates of the  $i^{\text{th}}$  pixel in the current frame,  $(x'_i, y'_i)$  denotes the coordinates of the corresponding pixel in the previous frame, and the summation is carried out over  $N$  pairs of pixels  $(x_i, y_i)$  and

$(x'_i, y'_i)$  within image boundaries. The method can easily be extended to arbitrarily shaped regions by restricting the sum to pairs of pixels within the region boundaries.

We propose to use a perspective (8-parameter) model for camera motion. To compute model parameters, implicit in  $(x'_i, y'_i)$ , we use a gradient descent method. The gradient descent minimizes either a quadratic or a truncated quadratic error function. In the latter case, the impact of outliers, that would otherwise bias the estimation, is reduced. In order to improve convergence and reduce computational complexity, a low-pass image pyramid is used; the gradient descent is applied at the top of the pyramid and then iterated at each level until convergence is achieved. To assure convergence in the presence of large displacements, the gradient descent should start within a convergence "basin" of the global minimum of  $E$ . To achieve this, an initial, coarse estimate of the translation component of the displacement is computed by applying an  $n$ -step search matching algorithm at the top level of the pyramid. A block diagram of the proposed approach is presented in Figure 1. Below, we explain the proposed approach in detail.

## 2.1 Motion model

Since the problem of motion computation is underconstrained, motion estimation techniques require an additional constraint either explicitly (e.g., via local smoothness or uniformity requirement) or implicitly (e.g., via parameterization of motion over a region of support). In this paper, we parameterize the camera motion over the whole image by a perspective motion model (8 parameters) defined as follows

$$\text{Equation 2: } \begin{aligned} x'_i &= (a_0 + a_2 x_i + a_3 y_i) / (a_6 x_i + a_7 y_i + 1) \\ y'_i &= (a_1 + a_4 x_i + a_5 y_i) / (a_6 x_i + a_7 y_i + 1) \end{aligned}$$

where  $(a_0, \dots, a_7)$  are the motion parameters. This model is suitable when the scene can be approximated by a planar surface, or when the scene is static and the camera motion is a pure rotation around its optical center. The simpler affine (6 parameters:  $a_6 = a_7 = 0$ ), translation-zoom-rotation (4 parameters:  $a_2 = a_5, a_3 = -a_4, a_6 = a_7 = 0$ ), translation-zoom (3 parameters:  $a_2 = a_5, a_3 = a_4 = a_6 = a_7 = 0$ ), and translation (2 parameters:  $a_2 = a_5 = 1, a_3 = a_4 = a_6 = a_7 = 0$ ) models are particular cases of the above model and can be easily derived from it.

## 2.2 Initial translation estimate

The purpose of the initial stage is to assure the convergence of the subsequent gradient descent algorithm. Note that this initial estimate does not need to be accurate. However, it must be robust enough to make the starting point of the gradient descent lie within a convergence "basin" of the global minimum of  $E$ .

For this purpose, an initial estimate of the translation components ( $a_0, a_1$ ) of the displacement are computed at the top level of the pyramid. This initial estimate is obtained by a matching technique, which minimizes  $E$  using a modified  $n$ -step search [13]. This method has the advantage of being simultaneously robust and fast.

### 2.3 Gradient descent

The motion parameters  $\mathbf{a} = (a_0, \dots, a_7)$  are computed by minimizing  $E$ . Since the dependence of  $E$  on  $\mathbf{a}$  is nonlinear, we use the following iterative procedure (see [14])

$$\text{Equation 3: } \mathbf{a}^{(t+1)} = \mathbf{a}^{(t)} + \mathbf{H}^{-1} \mathbf{b}, \text{ or equivalently } \sum_{l=1}^n H_{kl} \delta a_l = b_k,$$

where  $\mathbf{a}^{(t)}$  and  $\mathbf{a}^{(t+1)}$  denote  $\mathbf{a}$  at iteration  $t$  and  $t+1$  respectively,  $\delta \mathbf{a} = \mathbf{a}^{(t+1)} - \mathbf{a}^{(t)}$  is the parameter update term,  $\mathbf{H}$  is an  $n \times n$  matrix equal to one-half times the Hessian matrix of  $E$  and is usually referred to as the curvature matrix [14],  $\mathbf{b}$  is an  $n$ -element vector equal to minus one-half times the gradient of  $E$ , and  $n$  is the number of parameters of the model. More specifically, the coefficients of the matrix  $\mathbf{H}$  and vector  $\mathbf{b}$  are given by

$$\text{Equation 4: } H_{kl} = \frac{1}{2} \sum_{i=1}^N \frac{\partial^2 e_i^2}{\partial a_k \partial a_l} \cong \sum_{i=1}^N \frac{\partial e_i}{\partial a_k} \frac{\partial e_i}{\partial a_l}, \text{ and } b_k = -\frac{1}{2} \sum_{i=1}^N \frac{\partial e_i^2}{\partial a_k} = -\sum_{i=1}^N e_i \frac{\partial e_i}{\partial a_k}.$$

See [14] for a justification of dropping these second-order derivatives in the computation of  $H_{kl}$  in Equation 4.

We are now in the position to describe the flow of the gradient descent algorithm as illustrated in Figure 1. The parameters  $a_0$  and  $a_1$  are first initialized with the translation vectors estimated in the initial matching stage. The gradient descent starts at the top level of the pyramid, then follows at the subsequent levels in a top-down approach. At each level, the gradient descent is iterated until a suitable convergence criterion is met. The projection of the motion parameters from one level onto the next one consists merely of multiplying  $a_0$  and  $a_1$  by 2, and dividing  $a_6$  and  $a_7$  by 2. Finally, the procedure stops at the base level of the pyramid where the final motion parameters are obtained.

Each iteration of the gradient descent consists of the following steps:

1. matrix  $\mathbf{H}$  and vector  $\mathbf{b}$  are computed (Equation 4),
2. the system defined in Equation 3 is solved using singular value decomposition (SVD) [14],
3. the resulting update term  $\delta \mathbf{a}$  is added to the current set of parameters:  $\mathbf{a}^{(t+1)} = \mathbf{a}^{(t)} + \delta \mathbf{a}$ .

The stopping criterion is defined as follows. At each level, at most  $N_{\max}$  iterations are carried out, but the process may stop earlier if the update term is smaller than a preset threshold. Two thresholds are used: a threshold  $\epsilon_1$  for the update of the translation parameters  $a_0$  and  $a_1$ , and a threshold  $\epsilon_2$  for the update of the remaining parameters.

## 2.4 Robust estimation

Since the motion model is global and applies to the complete image, it cannot account for local motion. In other words, from the point of view of the global motion model, local object motion may create outliers and therefore bias the estimate of global motion parameters. To remove the influence of such outliers, a robust estimator can be used [15].

More specifically, in this paper we consider M-estimators that minimize

$$\text{Equation 5: } \sum_{i=1}^N \rho(e_i),$$

where  $\rho$  is a symmetric positive-definite function with a unique minimum at  $e_i=0$ . In the original formulation (Equation 1), we have used a quadratic error function

$$\text{Equation 6: } \rho(e_i) = e_i^2.$$

A quadratic function gives a large weight to large errors, such as the ones introduced by outliers, therefore biasing the estimation process. In order to limit the sensitivity of the estimation process to outliers, we use the so-called *truncated quadratic* instead

$$\text{Equation 7: } \rho(e_i) = \begin{cases} e_i^2 & \text{if } |e_i| \leq t \\ 0 & \text{if } |e_i| > t \end{cases},$$

where  $t$  is a threshold. In other words, only those pixels for which the absolute value of the error term is below  $t$  are taken into account in the estimation process; other pixels are ignored.

The truncated quadratic is used only within the gradient descent part of the algorithm; it is not applied in the initial matching stages since the estimation of the initial translation parameters is coarse anyway. Using a truncated quadratic function, the same formulation as described in Section 2.3 can still be applied. Furthermore, the robust estimator does not increase the computational complexity. In fact, it marginally reduces the complexity as fewer pixels are accumulated in  $E$ .

The robust estimation consists of the following steps. During the first iteration at the top level of the pyramid, the quadratic error function is used (Equation 6) and a histogram of  $|e_i|$  is computed. Then, an absolute threshold  $t$  is derived from the histogram so as to exclude the samples resulting in the top  $T$  percent of the distribution of  $|e_i|$ , as illustrated in Figure 2. In subsequent iterations, the truncated quadratic error function is used (Equation 7) with the previously computed threshold  $t$ . The threshold  $t$  is updated at subsequent levels of the pyramid; a new histogram is computed after the first iteration at each level.

### 3. Simulation Results

This section reports simulation results using the proposed GME technique. Results are expressed in terms of coding efficiency for the dynamics sprites (DS) tool considered in the MPEG-4 Version 2 Visual Working Draft [10]. The simulations are performed using the MPEG-4 Verification Model. By using DS, every macroblock in the current frame can be predicted using either the previous frame and local motion vector (standard inter-frame coding), or a long-term memory (sprite) and global motion parameters.

Simulations have been carried out using the following sequences: *Foreman*, *CoastGuard*, *TableTennis*, *Stefan*, and *MIT*. Furthermore, two sequences with arbitrarily shaped objects have been also used: *Coast\_shore* and *Stefan\_bckgd*. They correspond to the shore in *CoastGuard* and to the background in *Stefan*, respectively. The test conditions, including the spatial resolution, frame rate, number of frames, quantization parameter, local motion vector range, as well as the motion model used for GME, are summarized in Table 1. All these sequences in Table 1 have a significant camera motion, resulting in inter-frame displacements as large as 30 pixels. The first four sequences are composed of a background and one or several foreground objects each characterized by a distinct local motion. Thereby, robust estimation should improve the performance of GME by removing the influence of the outlier pixels corresponding to local motion. Conversely, in the last three sequences, the whole image is moving with a coherent global motion and robust estimation is not expected to result in any gain. Note that for the sequences *CoastGuard*, *MIT*, and *Coast\_shore*, either a translational or a affine motion model was selected for GME due to the characteristics of the motion present. Note, that using a perspective model in these cases would essentially lead to similar results, but at a higher computational cost.

The number of levels of the low-pass image pyramid influences the convergence performance of the algorithm. In our experiments, the number of levels has been heuristically set to 3, based on the spatial resolution of the test sequences and the amount of inter-frame motion present. The pyramid has been constructed using a 3-tap filter with coefficients  $[1/41/21/4]$ . A modified 3-step search [13] has been used in the initial matching with a search range of  $\pm 4$ ,  $\pm 2$ , and  $\pm 1$  pixels in the first, second and third step respectively. This results in a total of 25 matching positions and a maximum displacement of  $\pm 7$  pixels at the top pyramid level corresponding to  $\pm 28$  pixels at full-resolution. We have further chosen  $N_{\max}=32$ ,  $\epsilon_1=0.1$ , and  $\epsilon_2=0.001$  in the stopping criterion of the iterative gradient descent, as a good compromise between performance and complexity.

Table 2 compares the performance of the MPEG-4 encoder without DS versus the cases when DS is used along with the proposed GME. In the latter case, results are shown for four combinations:

- no initialization and non-robust quadratic estimator (i.e., threshold of  $T=0\%$ ) that corresponds to the method proposed in [12],

- initialization by 3-step matching and non-robust quadratic estimator ( $T=0\%$ ),
- no initialization and robust truncated quadratic estimator ( $T=10\%$ ),
- initialization by 3-step matching and robust truncated quadratic estimator ( $T=10\%$ ).

Results are expressed in terms of average total bitrate and average Peak-Signal-to-Noise-Ratio (PSNR) of the luminance component. From Table 2, the benefit of using GME appears very clearly. The use of DS leads to very significant bits savings, ranging from 4% to 52%. The gains due to the use of GME can be attributed to a more precise estimation of motion and its more compact representation. DS has also the advantage of storing past information along  $n$ -term memory further improving motion-compensated prediction. It can also be observed that for the first three sequences the use of the robust estimator in GME ( $T=10\%$  versus  $T=0\%$ ) results in savings ranging from 6 to 15% in terms of bitrate for similar or better PSNR. For the other sequences, the robust estimation still outperforms the non-robust one, although the savings are much smaller. Without initialization, the GME technique does not converge for some frames of certain sequences resulting in erroneous motion parameters. Although the initial matching achieves only a small gain in terms of coding efficiency, it is essential to produce reliable motion parameters that can be used to build a sprite.

Figure 3 shows the gain of the robust GME over the non-robust GME (i.e.,  $T=0\%$ ) as a function of the threshold  $T$ . For all the test sequences, a threshold between 5% and 15% is a good operating point; a threshold too large not only removes outliers but also legitimate pixels. Therefore, the performance declines with increasing  $T$ . Figure 4 shows the layout of pixels declared as outliers for *Foreman* and *CoastGuard* ( $T=10\%$ ). Clearly, the robust estimator removes pixels corresponding to local motion: the face in *Foreman*, and the boats and the water in *CoastGuard*.

In terms of the computational complexity, the proposed GME technique is running at a speed ranging from 2 to 8 frames/second on a 533 MHz Alpha 21164 PC, depending on the motion model and video sequence used. The complexity of the initial matching step is negligible when compared to that of the whole algorithm.

## 4. Conclusions

In this paper, we have presented an efficient, robust, and fast GME method. The hierarchical algorithm first computes a coarse estimate of the translation component by an  $n$ -step matching, and then iterates a gradient descent to obtain and refine the 8 parameters of a perspective model. The algorithm minimizes a truncated quadratic error function, thereby reducing the impact of outliers that would otherwise bias the estimation.

Applying the proposed GME within an MPEG-4 coding system, we have shown that it provides significant performance gains. Indeed, bits savings ranging from 4% up to 52% have been achieved. Furthermore, we have also shown that the robust truncated quadratic estimator outperforms the non-robust quadratic one on sequences

containing object motion, with savings up to 15% in terms of bitrate for similar or better PSNR. Moreover, we have observed that the initial matching is essential for the algorithm to converge in the presence of large inter-frame displacements. Finally, the resulting algorithm is computationally very cost-effective.

## References

- [1] ISO/IEC JTC1 IS 11172-2 (MPEG-1), "Information Technology - Coding of Moving Pictures and Associated Audio for Digital Storage Media Up to About 1.5 Mbit/s", 1993.
- [2] ISO/IEC JTC1 IS 13818-2 (MPEG-2), "Information Technology - Generic Coding of Moving Pictures and Associated Audio", 1994.
- [3] F. Dufaux and F. Moscheni, "Motion Estimation Techniques for Digital TV: A Review and a New Contribution", *Proc. IEEE*, vol. 83, pp. 858-879, June 1995.
- [4] S. F. Wu and J. Kittler, "A Differential Method for Simultaneous Estimation of Rotation, Change of Scale and Translation", *Signal Processing: Image Communication*, vol. 2, no. 1, pp. 69-80, May 1990.
- [5] D. Adolph and R. Buschmann, "1.15 Mbit/s Coding of Video Signals Including Global Motion Compensation", *Signal Processing: Image Communication*, vol. 3, no. 2-3, pp. 259-274, June 1990.
- [6] Y. T. Tse and R. L. Baker, "Global Zoom/Pan Estimation and Compensation for Video Compression", In *IEEE Proc. ICASSP'91*, vol. IV, pp. 2725-2728, Toronto, Canada, May 1991.
- [7] F. Moscheni, F. Dufaux, and M. Kunt, "A New Two-Stage Global/Local Motion Estimation Based on a Background/Foreground Segmentation", In *IEEE Proc. ICASSP'95*, pp. 2261-2264, Detroit, MI, May 1995.
- [8] H. Jozawa, K. Kamikura, A. Sagata, H. Kotera, and H. Watanabe, "Two-Stage Motion Compensation Using Adaptive Global MC and Local Affine MC", *IEEE Trans. On Circ. and Syst. for Video Tech.*, vol. 7, no. 1, pp. 75-85, Feb. 1997.
- [9] M. Etoh and T. Ankei, "Parametrized Block Correlation - 2D Parametric Motion Estimation for Global Motion Compensation and Video Mosaicing", *IEICE TR PRMU97*, July 1997.
- [10] ISO/IEC JTC1/SC29/WG11 N1993 "MPEG-4 Version 2 Visual Working Draft Rev 2.0", Feb. 1998.
- [11] J. Konrad and F. Dufaux, "Improved Global Motion Estimation for N3", ISO/IEC JTC1/SC29/WG11 M3096, Feb. 1998.
- [12] R. Szeliski, "Image Mosaicing for Tele-Reality", TR94/2, Digital Equipment Corp., Cambridge Research Lab., May 1994.
- [13] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion Compensated Interframe Coding of Video Conferencing", In *Proc. Nat. Telecommun. Conf.*, pp. G5.3.1-G.5.3.5, New Orleans, LA., Dec. 1981.
- [14] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes in C: The Art of Scientific Computing*, Cambridge Univ. Press, Cambridge, 1988.
- [15] P. Meer, D. Mintz, A. Rosenfeld, and D. Y. Kim, "Robust Regression Methods for Computer Vision: A Review", *Int. Journal of Computer Vision*, vol. 6, no. 1, pp. 59-70, 1991.

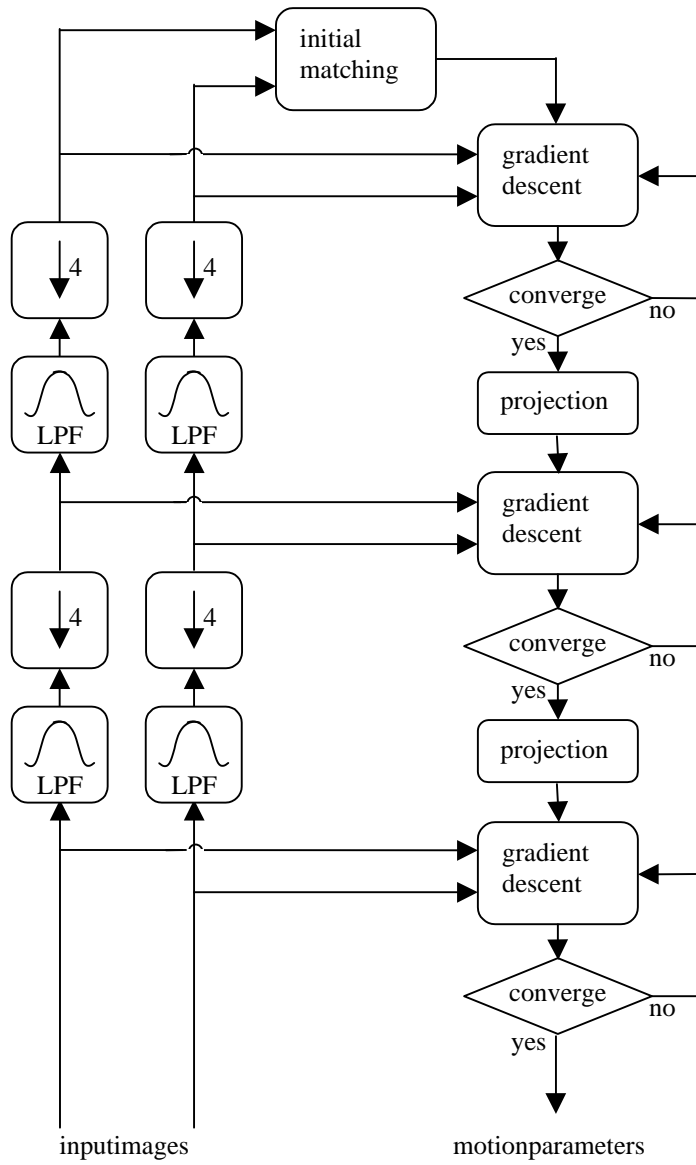


Figure 1: Block diagram of the proposed global motion estimation shown for the case of a 3-level hierarchical implementation.

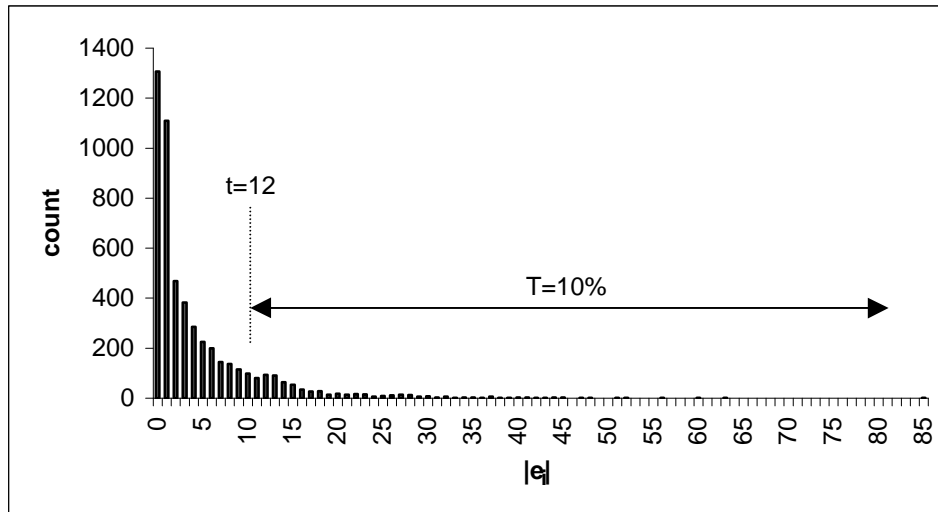


Figure 2: Histogram of  $|e_i|$  with the absolute threshold  $t$  that excludes the top  $T\%$  of the histogram.

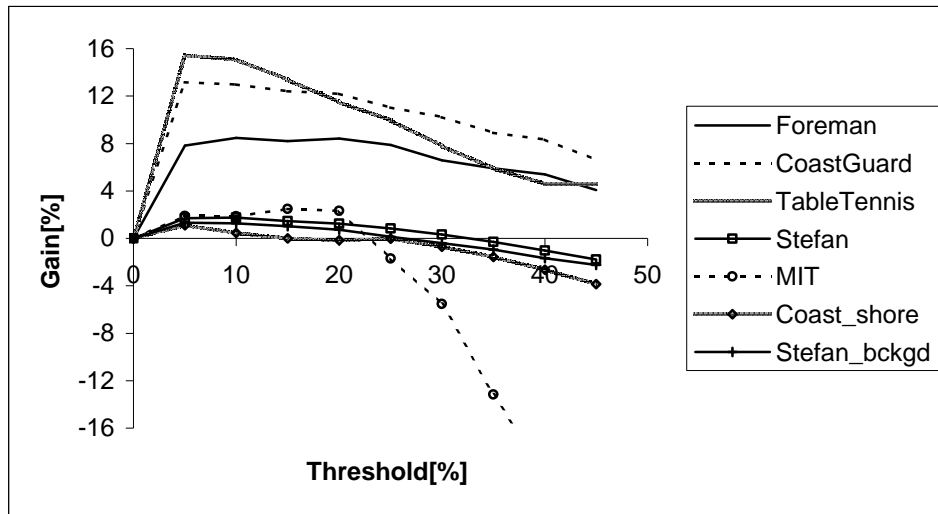


Figure 3: Bitrate gain of robust Global Motion Estimation over non-robust as a function of threshold.



Figure 4: Original images and layouts of pixels declared as outliers (black) for *Foreman* and *Coast Guard*.

*Foreman* and *Coast*

Sequence	Spat.Resol.	Fps	frames	Q	MVrange	Model
<i>Foreman</i>	<i>352x288</i>	<i>15</i>	<i>150</i>	<i>31</i>	<i>±32</i>	<i>Perspective</i>
<i>CoastGuard</i>	<i>352x288</i>	<i>15</i>	<i>150</i>	<i>31</i>	<i>±16</i>	<i>Translation</i>
<i>TableTennis</i>	<i>352x240</i>	<i>30</i>	<i>300</i>	<i>20</i>	<i>±32</i>	<i>Perspective</i>
<i>Stefan</i>	<i>352x240</i>	<i>30</i>	<i>300</i>	<i>20</i>	<i>±32</i>	<i>Perspective</i>
<i>MIT</i>	<i>352x288</i>	<i>15</i>	<i>150</i>	<i>31</i>	<i>±32</i>	<i>Affine</i>
<i>Coast_shore</i>	<i>352x288</i>	<i>15</i>	<i>150</i>	<i>31</i>	<i>±16</i>	<i>Translation</i>
<i>Stefan_bckgd</i>	<i>352x240</i>	<i>30</i>	<i>300</i>	<i>20</i>	<i>±32</i>	<i>Perspective</i>

Table 1: Image sequences and coding parameters used.

Coding Initialization Estimator	noDS	DS none $T=0\%$	DS 3-stepmatching $T=0\%$	DS none $T=10\%$	DS 3-stepmatching $T=10\%$
Sequence	Rate[kb/s] PSNR[dB]	Rate[kb/s] PSNR[dB]	Rate[kb/s] PSNR[dB]	Rate[kb/s] PSNR[dB]	Rate[kb/s] PSNR[dB]
<i>Foreman</i>	<b>125.39</b> <b>29.02</b>	<b>111.66</b> <b>28.74</b>	<b>110.36</b> <b>28.73</b>	<b>105.32</b> <b>28.69</b>	<b>101.00</b> <b>28.65</b>
<i>CoastGuard</i>	<b>130.98</b> <b>26.40</b>	<b>111.46</b> <b>26.28</b>	<b>109.16</b> <b>26.26</b>	<b>97.34</b> <b>26.23</b>	<b>94.99</b> <b>26.23</b>
<i>TableTennis</i>	<b>136.42</b> <b>28.93</b>	<b>149.64</b> <b>28.58</b>	<b>149.63</b> <b>28.59</b>	<b>127.25</b> <b>28.87</b>	<b>127.05</b> <b>28.88</b>
<i>Stefan</i>	<b>463.80</b> <b>26.52</b>	<b>457.29</b> <b>26.46</b>	<b>453.26</b> <b>26.46</b>	<b>452.04</b> <b>26.48</b>	<b>445.36</b> <b>26.46</b>
<i>MIT</i>	<b>105.92</b> <b>27.32</b>	<b>51.54</b> <b>27.04</b>	<b>51.54</b> <b>27.04</b>	<b>50.41</b> <b>27.04</b>	<b>50.58</b> <b>27.04</b>
<i>Coast_shore</i>	<b>54.34</b> <b>26.68</b>	<b>36.04</b> <b>26.53</b>	<b>35.18</b> <b>26.53</b>	<b>35.77</b> <b>26.54</b>	<b>35.02</b> <b>26.55</b>
<i>Stefan_bckgd</i>	<b>464.91</b> <b>26.61</b>	<b>454.64</b> <b>26.57</b>	<b>445.54</b> <b>26.56</b>	<b>450.67</b> <b>26.57</b>	<b>439.93</b> <b>26.56</b>

**Table 2: Comparison of MPEG-4 coding with and without Dynamic Sprites; in the latter case, results for four combinations of initialization and estimator type are shown (see text for details).**

List of figure/table captions:

**Figure 1:** Block diagram of the proposed global motion estimation shown for the case of a 3-level hierarchical implementation.

**Figure 2:** Histogram of  $|e_i|$  with the absolute threshold  $t$  that excludes the top  $T\%$  of the histogram.

**Figure 3:** Bit rate gain of robust Global Motion Estimation over non-robust as a function of threshold.

**Figure 4:** Original images and layout of pixels declared as outliers (black) for *Foreman* and *Coast Guard*.

**Table 1:** Image sequences and coding parameters used.

**Table 2:** Comparison of MPEG-4 coding without and with Dynamic Sprites; in the latter case, results for four combinations of initialization and estimator type are shown (see text for details).