Real-time recursive motion segmentation of video data on a programmable device.

R.B. Wittebrood*and G. de Haan

Philips Research Laboratories, Video Processing and Visual Perception Group, Prof. Holstlaan 4, 5656 AA Eindhoven, The Netherlands

Abstract

Recently, we reported on a recursive algorithm enabling real-time object-based motion estimation (OME) of standard definition video on a digital signal processor (DSP). The algorithm approximates the motion of the objects in the image with parametric motion models and creates a segmentation mask by assigning to every block in the image the best matching model. It was found that the calculation of the segmentation mask was most critical for both the real-time behaviour and the quality of the motion vector field. This publication details the motion segmentation module and discusses a number of options which improve the quality/complexity ratio of the motion segmentation.

1 Introduction

Motion estimation (ME) is a key-technology for video processing applications. Image enhancement, standards conversion, source coding, structure from motion and image understanding techniques all benefit from ME. Since 1995, ME is commercially available for real-time consumer applications like 100Hz television [1]. The computational complexity and bandwidth requirements of these algorithms suggest implementation in application specific ICs (ASICs). For some applications, however, dedicated silicon is not a feasible solution, e.g. for low volume electronics, and ME on a programmable device might be a good alternative. Recently, OME algorithms were published, processing standard definition video in real-time on a DSP leaving enough resources for a complete motion compensated 3:2 pull-down elimination algorithm [2,3,4]. These algorithms are based on the observation that ME on sub-sampled images is a straightforward way to reduce the bandwidth and computational complexity with such an amount that real-time implementation is possible on a modern DSP. The sub-sampling reduces the block sizes significantly, e.g. to $2 \times 2$ pixels. Unfortunately, block based ME on such a small number of pixels gives noisy and inaccurate motion vector fields, as shown in Figure 1c. This improves when more pixels, and thus larger blocks, are used. However, the assumption of pure translational block motion fails on larger blocks and larger blocks have boundaries with even less correspondence to the actual object boundaries resulting in annoying artifacts, see Figure 1d. Since objects are usually larger than blocks, an object-based approach can estimate motion more accurately (on sub-sampled images) than traditional block-based approaches, without violating the object boundaries more than these traditional block-based approaches.

Figure 1: a) Image of the Renata sequence. Images b, c and d show the horizontal component of the motion in the Renata sequence estimated with a 3DRS estimator [5] on: b) normal images, c) sub-sampled images, d) sub-sampled images and large blocks. Compared to block-based ME on images with normal size, block-based ME on sub-sampled images yields noisy motion vectors. This improves when larger blocks are used, however, the increased mismatch between block boundaries and object boundaries results in heavy blocking artifacts.

This paper shows progress with the ME algorithm described in [3,4]. That algorithm approximates the motion
of the objects in the image with parametric motion models and creates a segmentation mask by assigning to every block in the image the best matching model. It was found that the calculation of the segmentation mask was most critical for both the real-time behaviour and the quality of the motion vector field.

In Section 2, a short introduction to OME in general and to the particular estimator described in [3, 4] is given. In Section 3, we focus on the construction of the segmentation mask and we describe a number of options to improve the quality/complexity ratio of this important part of the OME algorithm. In Section 4, results are discussed and conclusions are drawn in Section 5.

2 Prior work

The fundamental problem in OME is the well-known chicken and egg problem. On the one hand, the motion of an object can be determined if the exact location of that object in the image is known, i.e. if the segmentation of the object is known. On the other hand, the segmentation of an object in the image can be determined if the motion of that object is known.

![motion models](image1.png)

Figure 2: The chicken and egg problem of OME. The chicken: the motion of an object can only be estimated if its segmentation is known. The egg: the segmentation of an object can be determined if its motion is known. This figure also suggests a recursive solution.

The motion segmentation module determines the segmentation of the objects based on a description of their motion (motion models). The parameter estimation module determines the motion of the objects based on the location of the objects in the image (image regions). By looping through the two modules, the system could converge to a correct solution.

Various methods have been proposed to solve this problem. Reference [6] classifies the OME algorithms into two categories: direct and indirect methods. The indirect methods use motion vector fields (or optical flow fields) estimated by a non-object-based estimator and partition these fields into segments which can be described by a single parametric model, or in which the motion is continuous [7, 8]. The direct methods tackle the chicken and egg problem in different ways. Some authors try to solve the problem simultaneously, usually in a Bayesian framework [9, 6], others use a hierarchical approach [10, 11, 2], or use image segmentation methods to obtain initial segmentation fields [12, 13]. We use a recursive approach, as proposed in [3, 4]. Figure 2 explains the chicken and egg problem and suggests a recursive solution.

In our earlier contribution to OME [3, 4], we identified an object by a unique label $O$. With every object $O$, a motion model in the form of a parameter vector $\mathbf{P}_o = (t_x, t_y, z_x, z_y)$ was associated. This parameter vector described the translation and zooming of the object. The relation between the parameter vector of an object $O$ and the motion (displacement) vector, $\mathbf{D}_o$, of the object at position $\mathbf{f} = (x, y)$ in the image is:

$$\mathbf{D}_o(\mathbf{P}_o, \mathbf{f}) = \begin{bmatrix} t_x + z_x (x - x_{zc}) \\ t_y + z_y (y - y_{zc}) \end{bmatrix}$$

(1)

with $(x_{zc}, y_{zc})$ the centre of zoom.

The segmentation of an object was described in a segmentation mask $\mathbf{M}(\mathbf{X})$ indicating for every block, $B(\mathbf{X})$, to which object it belongs. Note that an upper case $\mathbf{X}$ indicates locations on the block grid and a lower case $\mathbf{f}$ indicates locations on the pixel grid. The segmentation mask is constructed by minimising (neglecting any interpolations necessary for sub-pixel accurate motion models):

$$\varepsilon_b(\mathbf{P}_o, \mathbf{X}, n) = \sum_{\mathbf{f} \in B(\mathbf{X})} |F_b(\mathbf{f}, n) - F_b(\mathbf{f} - \mathbf{D}_o(\mathbf{P}_o, \mathbf{f}_{zc}), n - 1)|$$

(2)

for every block, $B(\mathbf{X})$, over all available motion models, $\mathbf{P}_o$. Here $n$ is the image number of the sub-sampled luminance image $F_b$, $\mathbf{P}_o$ is the parameter vector associated with object $O$ and $\mathbf{f}$ is the location of block $B(\mathbf{X})$. For computational efficiency the displacement vector, $\mathbf{D}_o$, is not calculated for every pixel, $\mathbf{f}$, in block $B(\mathbf{X})$ but only at a central pixel $\mathbf{f}^c$.

The parameter estimation module determines the parameters of the motion models of the objects by minimising another error measure over a small set of feature points, $F_S$:

$$\varepsilon_p(\mathbf{P}_o, \mathbf{X}, n) = \sum_{\mathbf{X} \in F_S} W_o(\mathbf{X}) \varepsilon_b(\mathbf{P}_o, \mathbf{X}, n)$$

(3)

where:

- $W_o(\mathbf{X}) > 1$, e.g. 16, in case, in the previous iteration, motion model $O$ was assigned to block $B(\mathbf{X})$ with a low segmentation error, $\varepsilon_b(\mathbf{P}_o, \mathbf{X}, n) < T_e$. 

2
\( W_0(\mathbf{X}) < 1 \), e.g. 1/16, in case, in the previous iteration, another motion model was assigned to the block with a low segmentation error.

\( W_0(\mathbf{X}) = 1 \), otherwise.

The feature points are chosen as:

\[
FS(n) = \left\{ \mathbf{X} \mid \sum_{\mathbf{X} \in H(\mathbf{X})} |F(\mathbf{X}, n) - F(\mathbf{X}, n-1)| \geq T_f \right\}
\]

as described in [4]. An overview of the complete OME algorithm from which we are started is given in [3, 4]. For the remainder of this paper, we shall focus on the motion segmentation module.

### 3 Motion segmentation

The most expensive part of the OME algorithm is the motion segmentation module. According to [3] it accounts for 70% of the DSP cycles consumed by the ME algorithm. In the following subsections, the motion segmentation module is described and a number of interesting optimisation possibilities is analysed. In Subsection 3.1, we discuss options to improve the consistency of the motion segmentation and in Subsection 3.2, two options for reducing the computational complexity of the algorithm are discussed.

#### 3.1 Consistency improvements

**3.1.1 Independent vs. predictive motion segmentation**

The motion segmentation module determines for every block in the image which motion model fits best. How well a motion model fits for a given block, \( B(\mathbf{X}) \), is expressed by eq. 2. A segmentation is obtained by minimising this error over all motion models:

\[
M(\mathbf{X}) = \arg\min_{O} \left\{ \varepsilon_b(\mathbf{P}_o, \mathbf{X}, n) \right\}
\]

Note that an object consist of a segmentation and a motion model, hence the minimisation over \( O \). This independent segmentation resembles the vector assignment of a full search (FS) block matcher, and indeed similar problems occur. The main problems of a FS block matcher are the inconsistency of the resulting vector field and the high computational complexity and required bandwidth. In the past, a lot of research was aimed at reducing the disadvantages of a FS block matcher, surveys are given in [6, 5]. A successful block matcher (for standards conversion purposes) both in accuracy and smoothness is the 3D recursive search (3DRS) block matcher. The approach used in this block matcher is also applicable here. Instead of minimising eq. 2 over all motion models, prediction is used to limit the number of motion models evaluated in the minimisation. In the predictive segmentation method, eq. 2 is minimised over a small set, \( CS \), of candidate motion models.

\[
M(\mathbf{X}) = \arg\min_{O \in CS} \left\{ \varepsilon_b(\mathbf{P}_o, \mathbf{X}, n) \right\}
\]

The candidate set, \( CS \), contains predictions selected from the segmentation mask in a spatio-temporal neighbourhood around the current block. Since objects are usually larger than blocks, the neighbours of the current block will often belong to the same object as the current block, and can therefore be used as spatial predictions. Moreover, the object to which the current block belongs, will be present in both the current frame and the previous frame (or else ME is not possible). Thus, blocks from the previous segmentation mask can also be used as temporal predictions. The predictions propagate the motion models from frame to frame (temporal predictions) and from block to block (spatial predictions). Figure 3 shows two candidate sets located on the blocked grid of the segmentation mask. The blocks are scanned from left to right and from top to bottom as shown by the dotted arrows. The current block, \( 'C' \), indicates the block for which eq. 2 is currently minimised. The segmentation mask above the thick solid black line originates from the current recursion of the segmentation module. The blocks above this line can be used as spatial predictions, \( 'S' \). The blocks below this line can be used as temporal predictions, \( 'T' \), since their segmentation originates from the previous recursion. The spatial predictions, \( 'S' \), are located as close as possible to the current block, \( 'C' \), for maximum correlation and propagate motion models in the direction indicated by the solid arrows. The temporal predictions are located at some distance of the current block to account for object motion, as explained below.

**Figure 3:** Possible configurations for prediction candidates located on a blocked segmentation mask. For a full explanation see text. a) illustrates a configuration with two spatial predictions and two temporal predictions opposite to the spatial predictions. b) shows a minimum prediction configuration with two spatial and one temporal prediction.
Figures 4a and 4b show two subsequent frames of an edge (i.e. the transition from white blocks to gray blocks) moving in the direction of the upper left corner. Based on the predictions, the task is to select the correct motion model (the white blocks) for block 'C'. Figures 4c and 4d show two prediction configurations placed on the segmentations at the moment 'C' is processed. The temporal prediction, 'T', in Figure 4c probes the wrong motion model because the edge didn’t pass that position at the time instance of the previous recursion. The temporal prediction, 'T', can probe the correct motion model if it is placed at a larger distance from 'C', as can be seen in 4d.

The actual choice of the distance is a trade-off between two issues. On the one hand, a large distance enables accurate detection of the object edges of rapidly moving objects, at the cost of a reduced ability to segment small objects. On the other hand, a very small distance enables the segmentation of small objects, but limits the accuracy with which object edges can be estimated. Experiments show that the location of the temporal prediction as indicated in 4d is a good trade-off.

![Figure 4: Influence of temporal prediction.](image)

Figures 3 shows that the motion models are propagated in the direction of the lower-right corner. The two spatial predictions enable propagation in two dimensions, this increases the convergence speed of the segmentation dramatically. Optimally, the propagation should be in all directions. This can be accomplished by rotating the scanning direction and candidate configurations from recursion to recursion. Figures 5 shows four different scanning directions and the corresponding propagation cones in light grey.

![Figure 5: Influence of scanning directions on propagation of motion models.](image)

Additional to the spatial and temporal predictions, other candidates randomly chosen from the available motion models are needed. These candidates are called the random predictions and are necessary in the situations where the spatial and temporal predictions are wrong. Once a random prediction is assigned to a block, it will be propagated in the form of spatial and temporal predictions.

In order to keep the operation count low, the number of predictions must be kept as small as possible. A minimum prediction configuration consists of at least two spatial predictions, one temporal prediction and one random prediction. Two spatial predictions are necessary to enable 2-D spatial propagation, the temporal prediction enables temporal propagation and the random prediction enables estimation of the correct motion models when the spatial and temporal predictions fail, e.g. at scene changes or when new objects appear.

Because of the high correlation of the spatial and temporal predictions within objects, they will often be the same. Only one match error, \(\varepsilon_b(\bar{P}, \bar{X}, n)\), has to be calculated in this case, which reduces the computational complexity.

In Figure 6a, an image from an noiseless artificial sequence is given. Figures 6b and 6c show the segmentation results of the independent and predictive segmentation methods respectively. It can be seen that only errors in occlusion areas are made, no measures to prevent this can be taken. Figure 6d shows the independent segmentation
result in case noise is added to the artificial sequence, Figure 8a shows the predictive result in case of noise.

\[
\begin{align*}
\epsilon^p_k(F_0, \tilde{X}, n) &= \epsilon_0(F_0, \tilde{X}, n) + p
\end{align*}
\]  

(7)

The penalty, \( p \), is used to suppress erroneous assignments. This means that predictions which have a high probability to be wrong should be penalised more. The random prediction has the lowest probability to give the correct motion model and will account for a large amount of errors in case of noise. The temporal prediction has a good correlation to the current block, but since it originates from the previous recursion, it will result in more errors than the spatial prediction. It follows that the penalty, \( p_t \), is assigned one of three values, based on the type of prediction used in eq. 7, \( p \in \{ p_s, p_t, p_r \} \). Since the spatial prediction is most reliable the spatial penalty, \( p_s \), is set to zero. The temporal penalty, \( p_t \), of the temporal predictions is chosen small and the random penalty of the random prediction is chosen high. This leads to the following: \( p_r > p_t > p_s = 0 \).

On edges, assumptions as stated above are not true anymore and penalising might, therefore, result in less accurate edges and merging of objects. The resources of modern DSPs prevent that measures can be taken to reduce this risk. However, the increased consistency and accuracy within objects outweighs the decreased accuracy on edges. Penalising candidates can be seen as an additional constraint to solve the ill-posedness of the ME problem. Smoothness constraints are frequently used in Bayesian ME [14, 9] and optical flow equation methods [15].

In Figure 8, a comparison is given between segmentation result of the predictive method, with and without penalties. The improvement in consistency is clear.

\[
\begin{align*}
\epsilon^p_k(F_0, \tilde{X}, n) &= \epsilon_0(F_0, \tilde{X}, n) + p
\end{align*}
\]  

(7)

3.1.2 Penalties

Although the predictive segmentation is effective, aperture problems, noise and repeating image structures may result in erroneous motion model assignments. A computationally efficient means to reduce this risk, especially for the aperture problem and noise, is to penalise temporal and random predictions:

Although the predictive segmentation is effective, aperture problems, noise and repeating image structures may result in erroneous motion model assignments. A computationally efficient means to reduce this risk, especially for the aperture problem and noise, is to penalise temporal and random predictions:

\[
\epsilon^p_k(F_0, \tilde{X}, n) = \epsilon_0(F_0, \tilde{X}, n) + p
\]  

(7)

The penalty, \( p \), is used to suppress erroneous assignments. This means that predictions which have a high probability to be wrong should be penalised more. The random prediction has the lowest probability to give the correct motion model and will account for a large amount of errors in case of noise. The temporal prediction has a good correlation to the current block, but since it originates from the previous recursion, it will result in more errors than the spatial prediction. It follows that the penalty, \( p_t \), is assigned one of three values, based on the type of prediction used in eq. 7, \( p \in \{ p_s, p_t, p_r \} \). Since the spatial prediction is most reliable the spatial penalty, \( p_s \), is set to zero. The temporal penalty, \( p_t \), of the temporal predictions is chosen small and the random penalty of the random prediction is chosen high. This leads to the following: \( p_r > p_t > p_s = 0 \).

On edges, assumptions as stated above are not true anymore and penalising might, therefore, result in less accurate edges and merging of objects. The resources of modern DSPs prevent that measures can be taken to reduce this risk. However, the increased consistency and accuracy within objects outweighs the decreased accuracy on edges. Penalising candidates can be seen as an additional constraint to solve the ill-posedness of the ME problem. Smoothness constraints are frequently used in Bayesian ME [14, 9] and optical flow equation methods [15].

In Figure 8, a comparison is given between segmentation result of the predictive method, with and without penalties. The improvement in consistency is clear.
of sub-sampling. However, for $2 \times 2$ blocks, the number of segmentation errors suddenly rises at $p = 512$ (for this artificial sequence). The reason for this is that the penalty is so high that the background is propagated to the head of Renata. The smaller the block size, i.e. the higher the sub-sampling, the sooner this breakthrough occurs.

![Figure 9: Number of segmentation errors as function of the random penalty for the artificial sequence, see Figure 6a. The horizontal axis indicates the random penalty, $p$, the vertical axis the number of segmentation errors made by the predictive segmentation method. Noise is fixed on $\sigma = 5$. Results for different sub-sampling levels are given indicated by the corresponding block sizes.](image)

### 3.2 Complexity reduction methods

#### 3.2.1 Block hopping

Since the correlation of the spatial prediction with the current block is higher than the correlation of the temporal and random predictions, the segmentation error of the spatial prediction will, in general, be lower than the segmentation error of the temporal and random predictions. From graph $\triangle$ in Figure 10, it can be seen that there is only a small probability that the segmentation errors of the temporal and random predictions are smaller than the segmentation error, $E_s$, of the spatial prediction. This probability is even smaller with a decreasing segmentation error $E$.

A considerable reduction in calculations is obtained when the spatial prediction is assigned to the current block if the segmentation error, $e_b(P_o, \hat{X}, n)$, of this prediction is below a certain threshold $T$. The segmentation errors of the other predictions are not calculated unless $e_b(P_o, \hat{X}, n)$ exceeds threshold $T$. The reduction in calculations can be seen from graph $\Box$ in Figure 10. The reduction in calculations can be exchanged for more segmentation accuracy by varying the threshold $T$.

#### 3.2.2 Block sub-sampling

Another scheme for the reduction of computational complexity is block sub-sampling. When using block sub-sampling, the motion segmentation module only minimises the match error (eq. 2) for a subset of the blocks in the image. An interpolation step assigns motion models to the skipped blocks. The block sub-sampling should not break the convergence of the predictive segmentation, therefore the prediction configuration should be adjusted to the new sub-sampled block grid. The quincunx pattern given in Figure 11 reduces the number of $e_b(P_o, \hat{X}, n)$ with a factor of two. In subsequent recursions, the quincunx phase can be inverted, i.e. looking at Figure 11 in one recursion the gray blocks are used and in the next recursion the white blocks. To keep maximum correlation between the spatial predictions and the current block, the spatial predictions are placed as close as possible to the current block on the quincunx grid. The temporal predictions are most valid at block locations which are processed in the previous recursion. If the quincunx pattern alternates with subsequent recursions, the structure given in Figure 11b is preferred over the configuration of Figure 11a.

![Figure 10: $\triangle$) The probability that the segmentation errors of the temporal and/or random predictions are smaller than the segmentation error, $E_s$, of the spatial candidate. $\Box$) The average number of $e_b(P_o, \hat{X}, n)$ calculations per block when block hopping with threshold $T$ is used.](image)

![Figure 11: Quincunx segmentation grid and prediction configurations. In subsequent recursions of the motion segmentation module the segmentation errors are calculated for the gray and white blocks respectively. With alternating quincunx phases, the configuration in b) is preferred.](image)
without and with interpolation of skipped blocks.

Figure 12: Predictive motion segmentation of a frame from the Renata sequence. a) motion segmentation on a quincunx grid, b) interpolated segmentation.

4 Results

The accuracy and consistency of the motion segmentation methods can be expressed in the Modified Mean Squared Error (M2SE) and the Spatial Inconsistency (SI) measures used in [5]:

\[
M2SE = \sum \left[ F_s(\mathbf{x}, n) - F_{mc}(\mathbf{x}, n) \right]
\]

with:

\[
F_{mc}(\mathbf{x}, n) = \frac{1}{2} F_s(\mathbf{x} - \mathbf{D}(\mathbf{x}), n - 1) + \frac{1}{2} F_s(\mathbf{x} + \mathbf{D}(\mathbf{x}), n + 1)
\]

\[
SI = \sum_{\mathbf{x}} \sum_{k=-1}^{1} \sum_{k=-1}^{1} \left\| \mathbf{D}(\mathbf{x}_c) - \mathbf{D}(\mathbf{x}_c) - \left[ \frac{kW_b}{|W_b|} \right] \right\|
\]

with \( W_b \) and \( W_h \) the block width and height and \( \mathbf{D}(\mathbf{x}) \) the motion vector at pixel \( \mathbf{x} \). \( \| \cdot \| \) is the \( L_1 \) vector norm. \( \mathbf{x}_c \) runs through all locations corresponding to block centres.

In general one can say that when segmentation method A results in both a lower M2SE and a lower SI than segmentation method B, that segmentation method A is more capable to estimate the correct motion-segmentation. The sequences in Figure 13 are used to compare the motion-segmentation methods. Four sequences are used: 'BBC drum text', 'Renata', 'car gate' and 'roller coaster'. In 'BBC drum text' the image is translating to the left, the artificially inserted text is stationary. In the 'Renata' sequence, Renata is moving to the right and the background is slightly zooming-in. In the 'car gate' sequence the camera is zooming in on the scene, the car moves towards the camera and the gate is moving to the right. The background of the 'roller coaster' sequence is stationary and the roller coaster is following the track towards the right of the scene, effectively resulting in a zooming rotation. The motion models, used to test the motion segmentation module, are least squares approximations of the motion of a set feature points manually selected from the objects in the sequences. A 3DRS ME algorithm [5] is used to determine the motion of the feature points.

Figure 13: Video sequences used in experiments. The arrows indicate the motion. a) 'BBC drum text': content is translating to the left, artificially inserted text is stationary. b) 'Renata': Renata is moving to the right, camera is slightly zooming in. c) 'car gate': camera is slightly zooming in, car is moving towards camera, gate is moving to the right. d) 'roller coaster': scene is stationary, roller coaster is following rails to the right.

Figure 14 shows how the M2SE and SI of the segmentation depend on the temporal and random penalties with a zero spatial penalty. The best M2SE performance is achieved in the area where the random penalty is higher than the temporal penalty, and where the temporal penalty is higher than zero, i.e. \( p_r > p_t > 0 \). This is in line with our expectations.

Figure 14 shows distinct discontinuities where the temporal penalty equals the spatial penalty, i.e. \( p_t = p_s = 0 \), and where the random penalty equals the temporal penalty, i.e. \( p_r = p_t \). This supports the hypothesis that the spatial predictions have a higher correlation to the current block than the temporal predictions and that the temporal predictions have a higher correlation to this block than the random predictions. Obviously, if predictions with less correlation to the current block are preferred above better predictions, more errors will be made resulting in a higher M2SE. That the transitions at \( p_t = p_s \) and \( p_r = p_t \) are sharp is due to the spatial and temporal predictions which cause the propagation of the erroneous model.

Figure 15 gives the results of the different segmentation methods discussed above, in terms of M2SE, SI and
\( \varepsilon_k(P_0, X, n) \)-calculations. The results are normalised with respect to the independent method. It can be seen that the M2SE of the predictive and predictive+block hopping methods are better than the M2SE of the independent method. The predictive+block hopping method limits the increase in M2SE, with respect to the predictive method, by verifying a spatial prediction for every block, this verification is not available in the predictive+block sub-sampling method, and for this reason this method has a higher M2SE than the predictive+block hopping (and even the independent) method. The reductions of the spatial inconsistency and of the complexity are the main advantages of the predictive method. Although the SIs of the optimisations (predictive+block hopping and predictive+block sub-sampling) are lower than the SIs of the predictive method, the optimisations have a lower quality than the predictive method because the M2SEs of these optimisations are significantly higher. The verification of the spatial candidate in the predictive+block hopping method limits the M2SE but increases the calculations with respect to the predictive+block sub-sampling method.

Figure 14: a) The M2SE and b) the SI as a function of the penalties for temporal and random candidates. The spatial penalty is zero. The M2SE and the SI are averaged over the four sequences shown in Figure 13.

Figure 15: Results for different methods and sequences in terms of M2SE, SI and complexity X. The bars, from black to light gray, indicate the results for the independent, predictive, predictive+block hopping and predictive+block sub-sampling. a) M2SE, b) SI and c) calculations. The results are given in a ratio to the results of the independent method.

5 Conclusions

A real-time recursive motion segmentation method for a programmable device and a number of options to improve the quality/complexity ratio have been proposed. A predictive method has been described which improves the consistency of the motion segmentation with respect to the independent method. A further significant improvement is obtained when penalising predictions which are less likely to be correct. Options to reduce the computational complexity are also given. A predictive+block hopping method is proposed, which hops to the next block if the segmentation error of the spatial prediction is below a certain threshold. Another method, predictive+block sub-sampling, is proposed which reduces the complexity even further simply by minimising the segmentation error for a subset of the blocks, while interpolating the skipped blocks. The predictive+block hopping and predictive+block sub-sampling exchange quality for a reduction in complexity, while still achieving reasonable results.

6 Acknowledgements

The authors would like to thank Dr. Ir. M. Mertens from Philips Research Laboratories for stimulating discussions during the course of this work.
References


Biography

*Rimmert Bart Wittebrood* was born in Haarlem, The Netherlands, on February 1, 1972. He received his M.Sc. degree in electrical engineering from the University of Twente in 1998 and subsequently, he joined Philips Research Laboratories in Eindhoven, where he became a member of the Video Processing and Visual Perception group. He is currently involved in the research of motion estimation algorithms for video applications, including scan rate conversion systems, motion artefact reduction algorithms and coding applications. His main work is focused on the development of real-time motion estimation algorithms for programmable devices.

*Gerard de Haan* received the B.Sc., the M.Sc., and the Ph.D. degree from Delft University of Technology in 1977, 1979 and 1992 respectively. He joined Philips Research in 1979. Currently, he is a Research Fellow in the group Video Processing & Visual Perception of Philips Research Eindhoven and a Professor at the Eindhoven University of Technology. He has a particular interest in algorithms for motion estimation, scan rate conversion, and image enhancement. His work in these areas has resulted in several books, about 70 papers (www.ics.ele.tue.nl/~dehaan/publications.html), some 50 patents and patent applications, and several commercially available ICs. He was the first place winner in the 1995 ICCE Outstanding Paper Awards program, the second place winner in 1997 and in 1998, and the 1998 recipient of the Gilles Holst Award. The Philips 'Natural Motion Television' concept, based on his PhD-study received the European Innovation Award of the Year 95/96 from the European Imaging and Sound Association. Gerard de Haan is a Senior Member of the IEEE.