MVA2000 IAPR Workshop on Machine Vision Applications, Nov. 28–30, 2000, The University of Tokyo, Japan

13—20

A block-based motion estimator capable of handling occlusions

Mark J.W. Mertens * Philips Research Labs Eindhoven

Gerard de Haan[†] Philips Research Labs Eindhoven

Abstract

We present a new block-based motion estimation strategy, which aims at correctly finding the velocity of picture blocks, even for background blocks in occlusion areas. This *tritemporal* ME calculates the motion between two pictures, switching the ME reference plane dependent on an *occlusion* detector. We also introduce a *retimer*, which can transform, locally in the picture, motion vectors valid for one time instant to another time instant. The retimer uses a *foreground/background detector*, of which we describe three varieties. Occlusion is a problem that plagues all block based motion estimation methods, and hence we see a utility of our method for applications like e.g. picture rate conversion, video compression, 3D matching or image sequence object extraction. An evaluation of the tritemporal estimator is included.

1 Introduction

For real-time, low-cost motion estimation we have developed the 3D Recursive Search block matcher [1], which, in contrast to the full search block matcher [2], does not have to evaluate all possible motion vector candidates within the search area, but only a limited number of smartly chosen candidates. If one has e.g. two rigidly moving objects one can understand that we only need to test the velocities of both objects, which were already found from a previous picture pair. Our primary application was picture rate conversion, in which case bad motion vector fields lead to an annoying *halo* in the interpolated pictures [3].

To estimate a vector, we divide a reference plane in 8x8 blocks and fetch with each tested motion vector candidate the pixel grey values form both adjoining pictures for comparison. Since for upconversion we need to interpolate a picture halfway between two originals, reason suggests that we put our reference plane at this halfway position ($\alpha_E = 0.5$). We determine the best matching motion vector among the candidates by minimising the sum of absolute differences or SAD (eq. 1):

$$\epsilon(\vec{C},\vec{X},n) = \sum_{\vec{x}\in B(\vec{X})} \left| I(\vec{x}-\alpha_E\vec{C},n) - I(\vec{x}+(1-\alpha_E)\vec{C},n+1)\right| \quad (1)$$

In this equation, $B(\vec{X})$ is the block of selected pixels at position $\vec{X}$ ($\delta x = 0\dots7$ by $\delta y = 0\dots7$), I the pixel grey value at position $\vec{x}$ and picture number or moment in time n, $\vec{C}$ the candidate vector under scrutiny, and α_E a constant ($0 \le \alpha_E \le 1$), determined by the temporal position between the two pictures of the reference plane to which the fetched grey value blocks are projected for the match. For most blocks a comparison can be performed in the described manner, but whatever reference plane (or α_E) we choose, there will always be a certain number of blocks in *ambiguity regions* (Fig. 1) that can not be matched correctly, because background (BG) data is not available in one of the two pictures, e.g. it is covered in the future picture $n+1$. The vector that results from the minimisation of the SAD for such a block is typically incorrect.

If we look at Fig. 1, we learn that if we put our reference plane anywhere between picture n and $n+1$, e.g. at $n+0.5$, we can have two cases for the blocks in the triangular ambiguity region, depending on the particular motion vector candidate. Either the vector fetches without crossing the foreground (FG) boundary f (e.g. with a candidate equal to the FG velocity v_F) two blocks from a *different* position in the background in n and $n+1$. Or, in case the candidate is e.g. the BG velocity v_B (which is the correct velocity for a block in the ambiguity triangle), a background block from picture $n+1$ is compared with a foreground block from n, in which case the SAD will be even higher than for an incorrect vector.

2 The tritemporal motion estimator

For *covering* (see Fig. 2), there is one position for the reference plane at which the ambiguity region disappears, namely at $n+1$ (or $\alpha_E = 1$). All pixel blocks in and around the occlusion area in the future

*Address: Prof. Holstlaan 4, 5656AA Eindhoven, The Netherlands. E-mail: mark.mertens@philips.com

†Address: idem E-mail: g.de.haan@philips.com

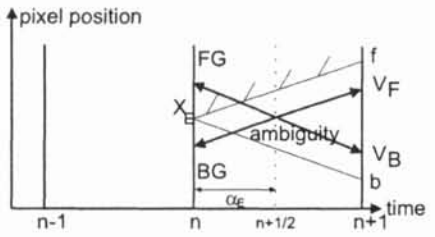

Figure 1: Ambiguity triangle for the uncovering case, and one-dimensional pictures for clarity. The foreground region boundary (the line f) moves away from background pixel trajectories (the line b).

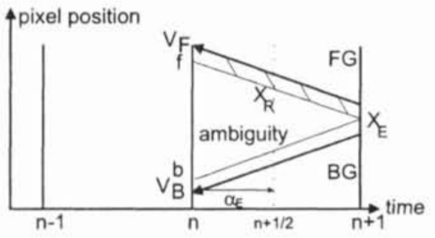

Figure 2: Ambiguity triangle for the covering case. The uncertainty becomes zero for the reference plane $n+1$

picture $n+1$, can also be found in the past picture n, hence their motion can be determined (vectors v_F and v_B in Fig. 2). Similarly, in the past picture there are pixel blocks, which are not visible anymore in the future picture (because they became covered), and these contain all pixels needed for the interpolation of $n+0.5$. Analogously in the case of *uncovering*, we should put the reference plane at picture n ($\alpha_E = 0$) to obtain the correct vectors for all blocks. The largest region of incorrect vectors for the uncovering case occurs if we put the reference plane at $\alpha_E = 1$.

Whether we have a region in the picture where the background is being covered or uncovered is determined by an *occlusion or covering/uncovering detector* that we described in patent [4]. The *tritemporal estimator* now switches the reference plane time moment α_E dependent on the outcome of the covering/uncovering detector. In non-occlusion areas, we take $\alpha_E = 0.5$ and project the blocks bidirectionally from both the past and future picture to the reference block position, i.e. compare them at the temporal position of the interpolated picture. For covering blocks, we take $\alpha_E = 1$ and match a block in the future picture in its own position with a block fetched with the candidate motion vector from the past picture. In uncovering areas of the picture we take $\alpha_E = 0$, in other words we match the blocks in the past picture with the blocks fetched from the future picture. (Eq. 2) summarises the tritemporal motion estimator in formulas.

$$\epsilon_c(\vec{C},\vec{X},n) = \begin{cases} \sum_{\vec{x}\in B(\vec{X})} \left| I(\vec{x}-\vec{C},n) - I(\vec{x},n+1) \right| & (covering) \\ \sum_{\vec{x}\in B(\vec{X})} \left| I(\vec{x},n) - I(\vec{x}+\vec{C},n+1) \right| & (uncovering) \\ \sum_{\vec{x}\in B(\vec{X})} \left| I(\vec{x}-\frac{\vec{C}}{2},n) - I(\vec{x}+\frac{\vec{C}}{2},n+1) \right| & (otherwise) \end{cases} \tag{2}$$

The benefit gained from switching α_E is that no incorrect vectors result from the minimisation, but the price to be paid is that the vectors obtained are not always in the correct block positions. E.g. the foreground edge (the block position where the vector changes from the BG to the FG velocity) is found at the position X_E, in stead of at its real position X_R (see Fig. 2). This is because the time moment of the matching was incorrect ($\alpha_E = 1$ in stead of $\alpha_E = 0.5$).

In fact the obtained vector field is not valid at any single time moment, but simultaneously at three different time moments, depending on the position in the image, hence the name tritemporal ME. In general for picture rate conversion, the vector field has to be *retimed* to the desired *upconversion position* (typically $\alpha_U = 0.5$). In the specific case of a sequence in which the FG object is stationary and the BG moves behind it, no retiming of the tritemporal vector field is necessary. This happens often in film material where the cameraman tracks the main subject (see Fig. 3).

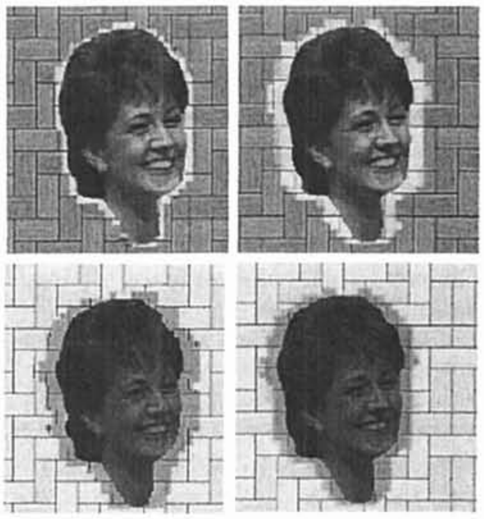

Figure 3: Top row: Stationary FG and moving BG for left Tritemporal and right reference ME (with $\alpha_E = 0.5$ always). Bottom row: both MEs for moving FG and stationary BG. Note that for a stationary FG object the tritemporal ME gives a negligible vector field mismatch, but for a moving FG it performs worse.

3 The retimer

As we noted in the previous section, the motion vector field originating from the tritemporal estimator, yields incorrect motion vectors for the interpolated picture between X_E and the true edge position of the foreground X_R. In typical video sequences, we will *locally* just have an occlusion of one BG object by one FG object. Hence, if we know that the velocity was incorrectly determined to be one of the two locally occurring velocities, we just have to fill in the other velocity instead.

The retimer performs the following actions. First it determines where the retiming should occur. It looks for a velocity edge X_E and marks a sufficiently broad *occlusion region* around this edge. Second it calculates how much blocks exactly should be corrected, by rounding the FG velocity to block precision. Third it determines which retimer correction action should be applied (e.g. replace the estimated FG velocity by the BG velocity). It turns out that there are 8 different retimer cases, depending on:
1. Covering versus uncovering
2. The sign of the foreground velocity
3. On which side of the velocity edge X_E the foreground is

To illustrate point 3, Fig. 4 shows two complementary cases of covering, where the obtained 2 velocities are exactly the same, but where the FG is on a different side of the edge. In the next section we describe 3 foreground/background detection strategies. Fig. 5 illustrates the correction of the retimer.

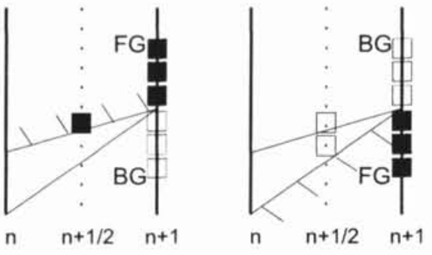

Figure 4: Left: In case the FG object is above the BG, we have to replace 1 incorrect BG velocity block (see $t+1$) by a FG velocity block (black). Right: In case the FG object is below the BG, we have to replace 2 incorrect FG velocity blocks by BG velocity blocks (white).

4 Foreground/background (FB) detection

With the aid of a previous motion vector field we can determine which object regions or velocities

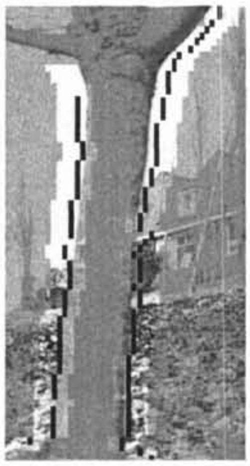
Figure 5: Processing of the retimer. The velocity of the tree was such that one row of blocks had to be adapted. The lighter band shows the total *occlusion region* that is analysed. The black lines show the blocks that where correctly changed from FG velocity to BG velocity, so that the motion vector field snugly fits the tree object.

belong to the FG and which to the BG. A background block is a block for which both the grey value pixels and the associated velocity vector disappear under covering. We use the correspondence of the vectors in our FB detectors, since we judge them to yield simpler, more reliable measures than pixel based measures. A disadvantage is that the quality of the FB detector depends on the obtained motion vector fields.

In a first strategy, the *average vector FB detection*, we make use of the fact that any vector $\vec{v}_{\text{fetch}} = k\vec{v}_{\text{F}} + (1-k)\vec{v}_{\text{B}}$, where k is smaller than 1 and $\vec{v}_{\text{F}}$ and $\vec{v}_{\text{B}}$ are the velocities of the FG and BG objects around X_E, fetches a BG velocity from the previous vector field in the case of covering and a FG velocity in the case of uncovering (see Fig. 6). The safest vector to use is the average vector $\vec{v}_{\text{av}} = 0.5\vec{v}_{\text{F}} + 0.5\vec{v}_{\text{B}}$. A variant of this first strategy fetches the background vector from the future for uncovering.

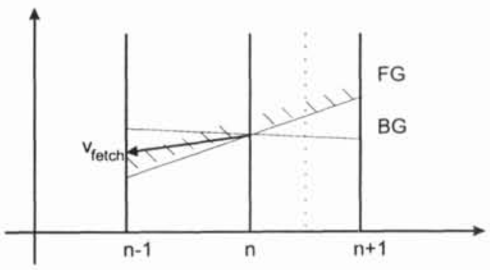

Figure 6: Average vector FB detection: if we look at the vector present in the position determined by the average vector between the FG and BG velocities, we always find the FG velocity for uncovering and the BG velocity in case of covering.

A second strategy, the *edge crossing FB detec-*

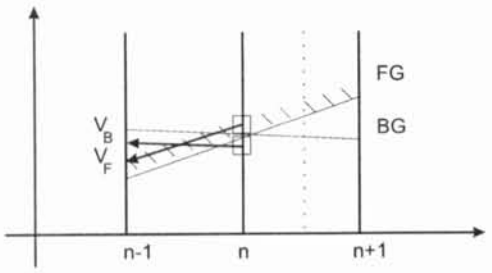

Figure 7: Edge crossing FB detection (see text).

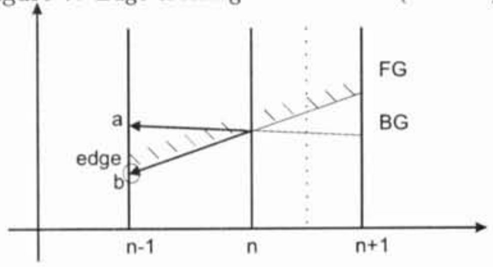

Figure 8: Edge projection FB detection (see text).

tion, is a prediction correction strategy in case the motion vector fields are not perfect. It uses the fact that for uncovering, BG positions projected to the past with the BG velocity have a higher probability (theoretically 1) of crossing towards the foreground region than when they are projected with $\vec{v}_{av}$. Because we do not know a priori which velocity is the background velocity, we project the two positions on either side of the edge with its own velocity $\vec{v}_{self}$ (see Figure 7). If the estimator works correctly, both velocities should project to the same FG velocity, and the velocity that is most different from the fetched velocity is the BG velocity. If another case occurs (e.g. the BG velocity projects to a BG region) than the FB output is labelled as indecisive.

A third strategy, the *edge projection FB detection*, checks for e.g. covering whether the edge between $\vec{v}_F$ and $\vec{v}_B$ in the previous image $n-1$ is present at position a or b (see Figure 8). If the edge is detected at position b, then the upper velocity, which projected to this position, is the FG velocity, and vice versa. Care should be taken that the velocities around the edge in $n-1$ are the same velocities as in n.

5 Analysis results

Table 1 presents the outcome of three error measures for a tritemporal and reference 3D Recursive Search motion estimator. The first is the mean square error MSE. We project the previous $(n-1)$ and next $(n+1)$ original picture with the obtained motion vector field $\vec{v}(\vec{x})$ to the current position (n). We average the pixel values of both projections, and calculate the mean square error with the current original pixels (Eq. 3). If pixels are fetched from outside the image, they are discarded in the summation. The average MSE was taken over 10 images from the sequence.

$$MSE(n) = \sum_{\vec{x}} \left(\frac{[I(\vec{x}-\vec{v}(\vec{x}), n-1) + I(\vec{x}+\vec{v}(\vec{x}), n+1)]}{2} - I(\vec{x}, n) \right)^2 \quad (3)$$

The subjective MSE (SMSE) is modeled on the human visual system [5]. It averages the pixel values over a 2x2 block before taking a third power of the subsampled differences. NIP is the number of pixels with an incorrect velocity. We tested the tritemporal estimator on two artificial sequences, for which we know the velocities exactly (Pk is the top sequence of Fig. 3, and Fl is the head superimposed on the flower texture of Fig. 5).

Measure:	Pk Trit.	Pk Ref.	Fl Trit.	Fl Ref.
MSE	107	285	87	128
SMSE	5165	10825	4955	8186
NIP	6967	56642	5107	12398

Table 1: Error measures for two sequences (Pk, Fl) for the tritemporal and reference ME.

6 Conclusions and further work

We have presented a simple motion estimator, that uses only two frames, and a stored previous motion vector field, that can find the correct velocities with high precision for almost all video sequences. In upconversion, a correct vector field leads to reduced halo, and in other applications a better vector field is also necessary or desirable.

In the future we will continue to make the method even more robust for complex video sequences and more accurate.

References

[1] G. de Haan, P.W.A.C. Biezen, H. Huijgen and O.A. Ojo, "True Motion Estimation with 3-D Recursive Search Block-Matching", IEEE Tr. on Circuits and Systems for Video Technology, Vol.3, October 1993, pp. 368-379.

[2] A.M. Tekalp, "Digital Video Processing", Prentice Hall PTR, 1995, ISBN 0-13-190075-7.

[3] O.A. Ojo and G. de Haan, "Robust motion-compensated video upconversion", IEEE Tr. on Consumer Electronics, Vol. 43, Nov. 1997, pp. 1045-1056.

[4] G. de Haan and A. Pelagotti, 'Problem area location in an image signal', Patent no. WO0011863, 21-08-98.

[5] H. Marmolin, "Subjective MSE measures", IEEE Tr. on Systems, Man, and Cybernetics, Vol. SMC-16, May 1986, pp. 486-489.