STRATEGIES FOR FAST TRUE MOTION BLOCK MATCHING

Hendrik van der Heijden, Fabian Wenzel, Rolf-Rainer Grigat
Hamburg University of Technology
Vision Systems, 4-08/1
Harburger Schlossstr. 20
21079 Hamburg, Germany

Keywords: fast block matching, motion estimation, true motion.

Abstract: Block matching is a widely used method for fast motion estimation. Although using a very simple motion model, which does not fit most real world video material, many motion compensating video compression algorithms use block matching because of its speed. Applications based on true motion vector estimates often use an optical flow algorithm because of their higher need for accuracy at the expense of increased computing time. This paper presents a modified block matching algorithm suitable for true motion applications. A modified full search will be used on a cost function consisting of SAD and a vector field smoothing term. Several strategies as search center prediction, spiral search, early search termination and multilevel successive elimination are implemented to keep the computational demand low. This way, high-quality estimates can be computed in real-time.

1 INTRODUCTION

Motion Estimation (ME) is the basis for many digital image processing applications. In video coding, the objective of a ME algorithm is to minimize the residual intensity difference after motion compensation. Mostly *block matching* ME algorithms are used here because of speed. Typically, ME through minimizing intensity differences is not identical to determining true motion, which is defined as the physical 3D motion projected onto the image plane. If true motion estimates are required, ME is usually done by an optical flow (OF) algorithm, which tend to be time consuming. This paper adopts a motion smoothness constraint, as it is an essential aspect of optical flow algorithms, to be used in block matching. As search space subsampling algorithms like three step search are not suitable for true motion estimation, an adapted full search is used. Several techniques are implemented to keep computational demands low on today's desktop PCs. While the resulting motion field still has block resolution, correctness of estimates may improve from this added constraint, making it useful for some applications which otherwise would need a computational intensive optical flow approach.

2 BLOCK MATCHING

Block matching algorithms calculate a motion vector field, which consists of one vector for each block in the current frame pointing to its correspondence in a reference frame. The current frame I_t is divided into non-overlapping square blocks of size $B \times B$, typically 16×16 pixels. For each block $B_{i,j}$ at position (x, y) a corresponding block of the same size at position $(x + v_x, y + v_y)$ in the reference image I_{t+1} is determined by solving the following equation:

$$\mathbf{v} = (v_x, v_y) = \underset{(v_x,v_y)\in S}{\operatorname{argmin}} C_{(x,y)}(v_x, v_y) \qquad (1)$$

Hence, a matching error $C_{(x,y)}$ is minimized over a set S of search positions. The resulting displacement vector **v** is assumed to represent the motion of the block. Typically, a cost function like the sum of absolute differences (SAD) is chosen.

$$SAD_{(x,y)}(v_x, v_y) = \sum_{j=0}^{B-1}\sum_{i=0}^{B-1} |I_t(x+i, y+j) - I_{t+1}(x+v_x+i, y+v_y+j)| \qquad (2)$$

van der Heijden H., Wenzel F. and Grigat R. (2006).
STRATEGIES FOR FAST TRUE MOTION BLOCK MATCHING.
In *Proceedings of the First International Conference on Computer Vision Theory and Applications*, pages 359-363
DOI: 10.5220/0001368503590363

2.1 Construction of the Search Space

The search space S is a set of discrete search positions with integer coordinates, as the SAD function defines no finer resolution. A full search defines S as the set containing every position within a rectangle with size $2sw + 1$ (sw: search window size).

$$S = \{(v_x, v_y) | -sw \leq v_x, v_y \leq sw\} \quad (3)$$

Other algorithms like Three Step Search (Wang and Mersereau, 1999) or 2D Logarithmic Search (Jain and Jain, 1981) reduce the number of initial search positions to 9 resp. 5 and add more positions near the minimum position in several steps. As, due to their construction, a best match is not guaranteed to be found, only full search is used in the following.

Solving (1) is done by evaluating the SAD for every position in S. The SAD calculation can be aborted early, if the partial SAD already is above the lowest SAD found so far.

Further speedup is possible by reducing the number of search positions (early abort of search), the complexity of SAD calculation (using estimates for the SAD) and optimizing the visiting order of the search positions. All of these possibilities are used in this work and described in the following sections.

2.2 Multilevel Successive Elimination Algorithm

The Multilevel Successive Elimination Algorithm (MSEA) (Gao, 2003) speeds up SAD calculation. It defines several lower bounds SAD^l on the SAD with decreasing quality and computational demands to speed up SAD calculation.

Consider P_0 as the set of all pixel positions in a block. $P_{l,i}$ is the level-l quad-tree partition i of P_0, it contains all pixels of sub-block i with size $\frac{B}{2^l} \times \frac{B}{2^l}$. Vectors **a** and **b** contain all pixels of the two blocks to be compared. The evaluation of a new search position consists of subsequently computing SAD^L, SAD^{L-1} ... SAD^0, and is aborted as soon as $SAD^l > SAD^*$, SAD^* being the lowest SAD found so far.

$$SAD \leq SAD^l = \sum_{0 \leq j \leq l} \left| \sum_{i \in P_{l,j}} a_i - \sum_{i \in P_{l,j}} b_i \right| \quad (4)$$

The sums over block partitions are precalculated for every position in the reference image I_{t+1} and every level, taking additional memory of $2(L + 1)$ times the size of a single input image. Precalculation time for B=16 and 2 levels equals that of 4.5 SAD calculations per block. In this case, most positions with sub-optimal SAD values can be eliminated with $\frac{1}{16}$ of the operations a regular SAD would take.

2.3 Prediction and Search Position Order

It can be seen in section 2.2 that MSEA performes well if a good search position is known as early as possible. This fact can be utilized as neighboring blocks tend to have highly correlated motion vectors (MVs). In our approach MVs of three adjacent blocks (left, top and top-right, which have already been computed), three temporally adjacent block MV (same block, bottom-left and bottom-right in the previous motion field) and the zero motion MV are evaluated first. This choice of neighbors is taken from (Tourapis et al., 2001) and extended by the two diagonal temporal candidates from (de Haan et al., 1993) to allow better prediction of upward motion.

The MV having the lowest SAD defines the search center. As this prediction most of the time leads to the global minimum SAD being near the search center, the remaining positions of S are visited in expanding spiral order starting from the search center. The effective size of the search window will be larger than that of a static search center, because the search window size now limits the maximum detectable acceleration instead of the maximum speed. The search is aborted, if n rings have been completed without finding a lower cost position. n can be adjusted to favor speed or quality of the algorithm. In the following experiments $n = 3$ is used.

2.4 Modification Towards True Motion

Optical flow algorithms often incorporate a motion field smoothness constraint so that local variations increase cost (Brox et al., 2004). Equations determining MVs usually include image intensities of just a few pixels directly, which barely is a sound measure of image part similarity, even in strongly textured regions. Constraining adjacent MVs to be similar adds indirect influence of more pixels, expanding the effective image area used for similarity measurement, which lowers noise sensitivity. As a second effect, uniform image regions adopt MVs of surrounding textured regions.

While block matching directly includes enough pixels for textured regions, the SAD measure is not meaningful in poorly textured regions, regions with non-zero gradient in only one direction or when calculated on periodic image content. In this cases, taking adjacent MVs into account may improve correctness of determined motion vectors. In our work, we combine the SAD with an additive term increasing with MV distance from its surrounding MVs, leading to a modified cost function (5).

$$C(u,v) = SAD(u,v) + dB^2 f(\min_i \|(u,v) - N_i\|) \quad (5)$$

$$f(x) = \begin{cases} x^2 & \text{if } x \leq 1 \\ x & \text{else} \end{cases} \quad (6)$$

N_i are the left, top and top-right neighbor block MVs, the distance to the nearest MV is embedded in a function f which reduces damping for small (subpixel) deviations. As only the distance to the nearest MV is considered, the damping term penalizes positions which do not have at least one similar neighbor MV. A neighbour MV reliability based weighting of influence would be preferable, but was not considered here. Damping factor d can be adjusted to the properties of the input images. In our experiments $d = 0.5$ is a good value for most cases with $B = 16$.

2.5 Subpixel Resolution

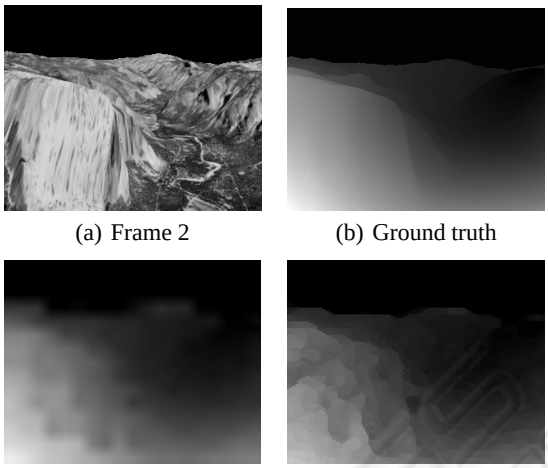
After determining the best matching pixel position for a block, a second search is done around this location refining the MV to subpixel resolution. A 2D logarithmic search with an initial search distance of 1/2 pixel evaluates the cost function on the current block and a linearly interpolated block from the reference image. MSEA is only applicable at full-pixel positions, so subpixel refinement can take more time than all other ME components together.

2.6 Motion Field Upsampling

In order to provide motion vectors for every pixel in an image, the block-based vector field is upsampled with one of three algorithms: constant and linear interpolation as well as hierarchical block erosion(de Haan et al., 1993). The latter divides each block in four sub-blocks and assigns the vector median of its parents vector and the vectors of its two directly adjacent blocks of parent size to each subblock. This filter is applied iteratively down to subblocks of size 2×2 pixels, then constant interpolation is applied to save time.

3 EVALUATION

We evaluated our proposed algorithm with respect to computing time (on an Athlon64 3000+) and accuracy. In order to evaluate the true error, the synthetic yosemite sequence (Quam, 1984) with known correct motion field (ground truth) has been used. A simple 3×3 smoothing filter kernel is applied to decrease aliasing effects. Figure 1 shows a sample frame, ground truth and motion estimates.



(a) Frame 2 (b) Ground truth

(c) estimate, linear interpolation, B=16 (d) estimate, block erosion, B=8

Figure 1: Yosemite sequence, velocity fields of correct motion (ground truth) and two estimates.

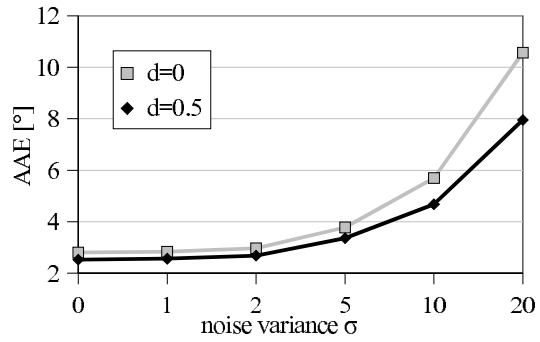
Table 1: AAE improvement when using our smoothness constraint. For $d = 0$, our cost function degrades to standard SAD. Other parameters are $B = 16$, $sw = 16$, $1/32$ pixel MV resolution, linear interpolation. Computation times are given for the whole sequence.

Sequence	d	AAE	time
Yosemite with	0	$3.84° \pm 6.91°$	399ms
black sky	0.71	$3.43° \pm 5.71°$	389ms
Yosemite with	0	$2.80° \pm 3.72°$	328ms
sky cropped	0.5	$2.52° \pm 2.19°$	351ms

Figure 3 displays the average angular error (Horn and Schunck, 1981) and its standard deviation at different damping factors. As the boundary between sky and land cannot be modeled adequately with blocks, a second measurement is taken with the top 72 pixel lines cropped. Results of these tests in table 3.1 show slight improvements of angular error and standard deviation when using our motion field smoothness constraint. While computation gets a bit faster due to the large black area in the first test, our extension increases computation time by 7% in the second test.

3.1 Noise Sensitivity

To evaluate noise sensitivity in textured regions, we added gaussian noise with variance $\sigma \in [0, 1, 2, 5, 10, 20]$ to the Yosemite sequence with cropped sky and measured the AAE with and without our damping extension. Results in figure 2 shows some improvement in noisy cases.



(a) Frame 2

Figure 2: Noise sensitivity, average angular error of motion estimates on Yosemite sequence (sky cropped) with added gaussian noise.

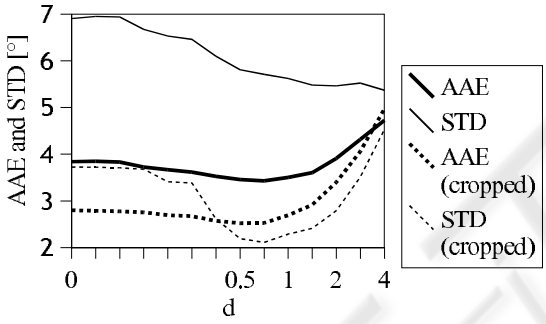


Figure 3: Yosemite sequence, average angular error (AAE) and standard deviation (STD) of full image with black sky and with sky cropped at different damping factors, B=16, upsampling via linear interpolation. Minimum values are 3.43 ± 5.71 (full) resp. 2.53 ± 2.11 (cropped).

Table 2: Yosemite sequence, linear interpolation, $B = 16$, $sw = 16$, $d = 0.71$ (optimal). For a block size of $B = 8$ two additional results are given. For comparison, a good 100% density 2D OF result (Farnebäck, 2000) is listed, its computing time is 16s per frame on a 360MHz Sun Ultra 60.

resolution	AAE	time/frame
1 pixel	$8.94° \pm 8.94°$	11.9ms
1/2	$5.02° \pm 6.08°$	14.7ms
1/8	$3.55° \pm 5.73°$	23.0ms
1/32	$3.43° \pm 5.71°$	32.6ms
1 (BS=8)	$9.10° \pm 9.35°$	24.9ms
1/32 (BS=8)	$3.54° \pm 5.31°$	53.7ms
OF	$1.40° \pm 2.57°$	see caption

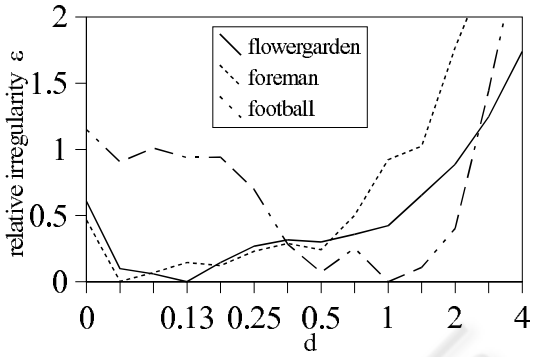


Figure 4: Relative irregularity at different damping factors, minimum set to zero, block erosion, B=16.

3.2 Real World Sequences

In a second experiment, real world sequences with unknown ground truth, a motion field irregularity measure ε (7) is used, which is defined as the average angular error (AAE) between a forward ME to the next image and a backward ME to the previous image.

$$\varepsilon(I_t) = AAE(-BM(I_t, I_{t+1}), BM(I_t, I_{t-1})) \quad (7)$$

This measure yields low values when computed correspondences are stable. False correspondences result in motion vectors jumping from one frame to another, i.e. in large values of ϵ. Experiments with various parameters on several synthetic image sequences indicate an approximately linear connection between ε and ground truth AAE.

Figure 4 shows the irregularity of three standard sequences, flowergarden (uniform foregound/background motion), foreman (first 70 frames only, medium motion) and football (fast, turbulent motion).

3.3 Computation Time

Table 3 gives computation times for three standard sequences for pixel resolution and the additional time per subpixel bit, which has shown to be nearly constant independent of subpixel resolution, for block sizes $B = 8$ and $B = 16$. The impact of MSEA on computation time was also measured in a short experiment. While a spiral full search speeds up by a factor of two, our described search algorithm gains only about 10% speed for $d = 0$ and loses up to 4% with $d = 0.5$ in the flowergarden sequence. In the latter case, the overhead introduced by MSEA is larger than the time it saves. MSEA gives best speedup on sequences with poor motion predictabiliy, which usually are one taking the most time.

Table 3: Computing times per frame for three sequences (block erosion, sw=16, d=0.5).

block size	sequence	time for pixel res.	additional time per subpixel bit
8	flowergarden	29.1ms	7.6ms
	foreman	31.3ms	8.3ms
	football	33.1ms	8.3ms
16	flowergarden	14.4ms	5.5ms
	foreman	16.6ms	6.6ms
	football	19.8ms	6.7ms

As can be calculated from table table 3, CIF (352x288) sized image sequences can be processed in real time (25Hz, below 40ms per frame) with 1/8 pixel resolution, even when containing turbulent motion like in the football sequence. As processing time is data dependent, subpixel resolution or the early termination criterion can be adaptively adjusted to meet real time requirements for every frame. Only one vector per block is computed, so our results have lower resolution than optical flow estimates. On comparable hardware, a recent OF implementation (Bruhn and Weickert, 2005) takes about 150ms per 160x120 pixel frame.

4 CONCLUSION

We presented strategies for fast true motion block matching. Our damping extension improves block matching results towards optical flow estimation and reduces noise sensibility. Block erosion upscaling of motion estimates to pixel resolution removes visible block structure. Prediction and MSEA have been adopted to keep computational demands low enough for real time processing. As motion discontinuities are rather badly modeled by the block structure used, there is a desire for better edge treatment. This can be done in several ways, for example weighting the SAD cost with a windowing function and finding correspondences for overlapping blocks as in the experimental BBC Dirac video codec. Another approach would be recognizing blocks containing more than one motion domain and determining a decomposition of block content and motion vectors with minimal SAD for each part (Orchard, 1993).

REFERENCES

Brox, T., Bruhn, A., Papenberg, N., and Weickert, J. (2004). High accuracy optical flow estimation based on a theory for warping. In *Proceedings of the 8th European Conference on Computer Vision*.

Bruhn, A. and Weickert, J. (2005). Towards ultimate motion estimation: Combining highest accuracy with real-time performance. To appear in Proc. 10th International Conference on Computer Vision (ICCV 2005).

de Haan, G., Biezen, P., Huijgen, H., and Ojo, O. (1993). True-motion estimation with 3d recursive search block matching. *IEEE Trans. on Circuits and Systems for Video Tech.*, 3(5).

Farnebäck, G. (2000). Fast and accurate motion estimation using orientation tensors and parametric motion models. In *Proceedings of 15th IAPR International Conference on Pattern Recognition, September 2000.*

Gao, Duanmu, Z. (2003). A multilevel successive elimination algorithm for block matching motion estimation. *IEEE Trans. on Image Processing*, 9(3).

Horn, B. and Schunck, B. (1981). Determining optical flow. *Artificial Intelligence*, 17(1-3):185–203.

Jain, J. and Jain, A. (1981). Displacement measurement and its application in interframe image coding. *IEEE Trans. Commun.*, COM-29:1799–1808.

Orchard, M. (1993). Predictive motion-field segmentation for image sequence coding. *IEEE Trans. on Circuits and Systems for Video Tech.*, 3:54–70.

Quam, L. (1984). Yosemite image sequence.

Tourapis, A. M., Au, O. C., and Liou, M. L. (2001). Predictive motion vector field adaptive search technique (PMVFAST) - enhancing block based motion estimation. In *Proceedings of Visual Communications and Image Processing 2001 (VCIP'01)*.

Wang, H. and Mersereau, R. (1999). Fast algorithms for the estimation of motion vectors. *IEEE Transactions on Image Processing*, 8(3).