Efficient and optimal block matching for motion estimation

Stefano Mattoccia Federico Tombari Luigi Di Stefano
Marco Pignoloni
Department of Electronics Computer Science and Systems (DEIS)
Viale Risorgimento 2, 40136 - Bologna, Italy
Advanced Research Center on Electronic Systems (ARCES)
Via Toffano 2/2, 40135 - Bologna, Italy
University of Bologna
{smattoccia, ftombari, ldistefano}@deis.unibo.it

Abstract

This paper proposes a novel technique for performing fast block matching for motion estimation which is optimal, meaning it yields the same results as a full-search investigation. The proposed technique derives from an approach previously proposed [10] for template matching and it is based on the deployment of a succession of lower bounding functions of the matching metric. Hence, an algorithm is outlined which efficiently exploits these bounding functions in order to rapidly determine non-matching block candidates, thus reducing the overall computational burden. Experimental results show that, compared to the brute-force approach, the proposed technique allows for notable reductions in terms of number of operations and computation times.

1. Introduction

Block matching is a common approach adopted in computer vision, in particular for the purpose of motion estimation for video compression, whose aim is to reduce temporal redundancy in video sequences. Let $\{I_r, I_t\}$ be two consecutive frames of a video sequence, and let I_r be subdivided into non-overlapping *blocks*, i.e. subwindows of size $N \times N$. Block matching aims at finding for each block of frame I_r the most similar block in frame I_t. Usually each block is not sought on the whole frame, but on a *search area* centred at the position of the block in frame I_r.

The *brute-force* approach (BF) for block matching relies on comparing each block of I_r with all possible *candidate blocks* belonging to the corresponding search area in I_t by computing a distance between blocks. The most commonly used distance for this scope is the *Sum of Absolute Differences* (SAD). Once the distances between the reference block and the candidate blocks have been computed, the best matching block is selected as the one corresponding to the minimum distance value found within the search area. Since this approach turns out to be computationally expensive, many techniques have been proposed in the last two decades with the aim of accelerating the BF (see [4] for a survey). These alternative approaches are usually denoted as either *optimal* or *non-optimal*, given they yield or not the same result as the BF. Non-optimal techniques [6], [9] usually reduce the search area in order to save computations, hence they don't guarantee the requirement of finding, for each block, the candidate block at the globally minimum distance within the search area. As a result, they tend to increase the distortion of the compressed video signal. Conversely, optimal techniques aim at accelerating the BA by selecting rapidly and safely many non-matching candidate blocks, so as to discard them without the need of computing the distance function. Non-matching candidate blocks are usually selected by means of lower bounds of the distance function [5], [3], [7].

This paper proposes a novel technique for optimal block matching. This technique derives from an approach previously proposed [10] for template matching which deploys a succession of lower bounds of the distance function characterized by increasing tightness and computational cost. These lower bounding functions are obtained by means of a partitioning of the blocks into non-overlapping subsets and by means of the concept of *partial distance*, which will be discussed in the following. The bounding functions are then deployed in an efficient matching scheme which turns out to speed-up significantly the BF.

This paper is structured as follows. Section 2 presents the notation and some previous work. Section 3 describes the proposed technique. Section 4 shows the experimen-

14th International Conference on Image Analysis and Processing (ICIAP 2007)
0-7695-2877-5/07 $25.00 © 2007 IEEE

IEEE COMPUTER SOCIETY

tal results obtained on typical benchmark video sequences. Finally, Section 5 draws the conclusions.

2. Notation and previous work

Let $I_r(x, y)$ be the reference frame, and $I_t(x, y)$ be the target frame where the block candidates are extracted. Let the block be of size $N \times N$ and the search area on the target frame be of size $M \times M$. Given the current block of the reference frame at coordinates $(\bar{x}, \bar{y})$, the BF usually computes for every candidate block belonging to the search area, i.e. $\forall u, v \in [1, \cdots, M]$, a *distance* between the current reference block and the candidate block:

$$D_p(\bar{x}, \bar{y}, u, v) = \sum_{i=1}^{N}\sum_{j=1}^{N} \big|I_r(\bar{x}+i, \bar{y}+j) - I_t(\bar{x}+u+i, \bar{y}+v+j)\big|^p \quad (1)$$

Usually $p = 1$ and $D_p(\bar{x}, \bar{y}, u, v)$ coincides with the SAD function. Then the best matching offset position (u_w, v_w) is selected as the one corresponding to the global distance minimum, i.e.:

$$(u_w, v_w) = \arg \min_{u,v \in [1,\cdots,M]} \{D_p(\bar{x}, \bar{y}, u, v)\} \quad (2)$$

The *triangular inequality* applied to the distance function (1) leads to:

$$D_p(\bar{x}, \bar{y}, u, v) \geq \Big|\big[\sum_{i=1}^{N}\sum_{j=1}^{N} |I_r(\bar{x}+i, \bar{y}+j)|^p\big]^{\frac{1}{p}} - \big[\sum_{i=1}^{N}\sum_{j=1}^{N} |I_t(\bar{x}+u+i, \bar{y}+v+j)|^p\big]^{\frac{1}{p}}\Big|^p \quad (3)$$

Hence, the right-hand term of inequality (3) is a lower bounding function for the distance term. We will refer to it as β_p. In the case $p = 1$, β_1 equals:

$$\beta_1(\bar{x}, \bar{y}, u, v) = \Big|\sum_{i=1}^{N}\sum_{j=1}^{N} |I_r(\bar{x}+i, \bar{y}+j)| - \sum_{i=1}^{N}\sum_{j=1}^{N} |I_t(\bar{x}+u+i, \bar{y}+v+j)|\Big| \quad (4)$$

which can be obtained by summing all elements of the reference block and candidate block. For this reason, this is a term which can be rapidly computed by means of incremental techniques such as [8], [2]. Term β_p can be used to select non-matching candidate blocks by testing, for each candidate block, the following sufficient condition before the computation of the distance term:

$$\beta_p(\bar{x}, \bar{y}, u, v) > D_m \quad (5)$$

with D_m representing the minimum distance value *found so far*. If condition (5) is verified, then β_p can not represent the global best match due to (3): hence, it can be discarded and term D_p needs not to be evaluated. Conversely, if condition (5) doesn't hold, term D_p is computed from scratch. The use of β_1 as a bounding function of the L_1 distance to speed-up an optimal block matching process has already been proposed in [7].

3. Proposed algorithm

In the following a set of additional lower bounding functions for the distance term is derived. As it will be shown, these lower bounding functions are tighter to the distance term compared to β_p, but they are also computationally more demanding. Hence, for all candidate blocks for whom condition (5) doesn't hold, instead of computing D_p from scratch we propose to deploy such additional conditions in order to increase the number of discarded non-matching blocks.

First of all, we propose to partition each block into r non-overlapping subsets. Even though there are no particular constraints about how to partition the blocks, according to the implementation that will be used to obtain the experimental results we propose to partition the blocks by rows. Furthermore, as the block side is typically a power-of-two value, we propose to choose r between any power-of-two submultiples of the block side which allows for subsets having all the same size $N \times \frac{N}{r}$. This allows increasing the overall computational efficiency of the technique, as it will be shown further on. In the following we will refer to the generic subset height as n, i.e. $n = \frac{N}{r}$.

Hence, given a block of I_r at position $(\bar{x}, \bar{y})$, and a candidate block of I_t at offset $(\bar{u}, \bar{v})$, once r is chosen, a generic *partial bound* term computed on subset $t, t \in [1, \cdots, r]$ can be defined as:

$$\alpha_{p,t}(\bar{x}, \bar{y}, \bar{u}, \bar{v}) = \Big|\big[\sum_{i=1}^{N}\sum_{j=1+n\cdot(t-1)}^{n\cdot t} \big|I_r(\bar{x}+i, \bar{y}+j)\big|^p\big]^{\frac{1}{p}} - \big[\sum_{i=1}^{N}\sum_{j=1+n\cdot(t-1)}^{n\cdot t} \big|I_t(\bar{x}+\bar{u}+i, \bar{y}+\bar{v}+j)\big|^p\big]^{\frac{1}{p}}\Big|^p \quad (6)$$

Similarly, a generic *partial distance* term computed on subset $t, t \in [1, \cdots, r]$ can be defined as:

$$D_{p,t}(\bar{x}, \bar{y}, \bar{u}, \bar{v}) = \sum_{i=1}^{N}\sum_{j=1+n\cdot(t-1)}^{n\cdot t} \big|I_r(\bar{x}+i, \bar{y}+j) -$$

0-7695-2877-5/07 $25.00 © 2007 IEEE
IEEE COMPUTER SOCIETY

$$I_t(\bar{x}+\bar{u}+i,\bar{y}+\bar{v}+j)\big|^p \quad (7)$$

It is important to note that from the property of the triangular inequality the following relation holds for each subset t:

$$D_{p,t}(\bar{x},\bar{y},\bar{u},\bar{v}) \geq \alpha_{p,t}(\bar{x},\bar{y},\bar{u},\bar{v}) \quad (8)$$

Hence, a novel lower bounding function of the distance term can be obtained by summing up all the r partial bound terms computed on each subset of the block:

$$\beta_{p,r}(\bar{x},\bar{y},\bar{u},\bar{v}) = \sum_{t=1}^{r} \alpha_{p,t}(\bar{x},\bar{y},\bar{u},\bar{v}) \quad (9)$$

The fact that (9) represents a lower bound for the distance term D_p is guaranteed by the triangular inequality and because:

$$D_p(\bar{x},\bar{y},\bar{u},\bar{v}) = \sum_{t=1}^{r} D_{p,t}(\bar{x},\bar{y},\bar{u},\bar{v}) \quad (10)$$

It is worth pointing out that also term $\beta_{p,r}$, similarly to term β_p, can be computed rapidly by means of incremental techniques. This is because each partial bound term represents a summation of the elements of the image on a subset of a block. In particular, as the size of all subsets is the same, only one incremental scheme is needed. The computation of $\beta_{p,r}$ needs more calculations compared to β_p due to the computation and sum of the single partial bounds embodied in (9). Nevertheless, with $p = 1, 2$, $\beta_{p,r}$ represents a closer approximation of the distance term D_p compared to β_p [10]. Hence it has to be applied, testing the following sufficient condition (11), on those candidate blocks which during the block matching process were not previously rejected by the sufficient condition derived from β_p.

$$\beta_{p,r}(\bar{x},\bar{y},\bar{u},\bar{v}) > D_m \quad (11)$$

Now, for all candidate blocks which are not rejected by (11), tighter lower bounding functions can be derived by computing one or more partial distance terms on the various subsets. In particular, by replacing the partial bound term $\alpha_{p,1}$ with the corresponding partial distance term $D_{p,1}$ we determine a further lower bounding function as:

$$\beta_{p,r-1}(\bar{x},\bar{y},\bar{u},\bar{v}) = D_{p,1}(\bar{x},\bar{y},\bar{u},\bar{v}) + \sum_{t=2}^{r} \alpha_{p,t}(\bar{x},\bar{y},\bar{u},\bar{v}) \quad (12)$$

Term $\beta_{p,r-1}$ represents a closer approximation of the distance term D_p compared to term $\beta_{p,r}$. In fact, since

$$\alpha_{p,1}(\bar{x},\bar{y},\bar{u},\bar{v}) \leq D_{p,1}(\bar{x},\bar{y},\bar{u},\bar{v}) \quad (13)$$

the following inequalities hold:

$$\beta_{p,r}(\bar{x},\bar{y},\bar{u},\bar{v}) \leq \beta_{p,r-1}(\bar{x},\bar{y},\bar{u},\bar{v}) \leq D_p(\bar{x},\bar{y},\bar{u},\bar{v}) \quad (14)$$

Unfortunately, term $\beta_{p,r-1}$ is also more demanding in terms of computations compared to term $\beta_{p,r}$ as it incorporates a partial distance term which can not be computed by means of incremental schemes. However, it has to be computed only for those candidate blocks which were not discarded by the application of bound $\beta_{p,r}$. As a result, the sufficient condition derived from bound $\beta_{p,r-1}$ is:

$$\beta_{p,r-1}(\bar{x},\bar{y},\bar{u},\bar{v}) > D_m \quad (15)$$

By increasingly replacing each partial bound term with the corresponding partial distance term computed on the same subset other lower bounding functions can be determined, which are characterised by increasing tightness to the distance term and increasing computational weight. However, as previously, each bounding function has to be applied only to those candidate blocks which were not discarded by the sufficient condition tested at the previous step. Following this approach, up to r sufficient conditions can be tested, associated with terms $\beta_{p,r}, \cdots, \beta_{p,1}$. The last lower bounding function contains only one residual partial bound term:

$$\beta_{p,1}(\bar{x},\bar{y},\bar{u},\bar{v}) = \sum_{t=1}^{r-1} D_{p,t}(\bar{x},\bar{y},\bar{u},\bar{v}) + \alpha_{p,r}(\bar{x},\bar{y},\bar{u},\bar{v}) \quad (16)$$

Hence, if the associated sufficient condition

$$\beta_{p,1}(\bar{x},\bar{y},\bar{u},\bar{v}) > D_m \quad (17)$$

doesn't hold, that is, the current candidate block can not be safely discarded even by means of this last test, the computation of the distance term D_p is completed by calculating the partial distance term corresponding to the $r-th$ (last) subset and D_p is then compared with D_m.

It is worth noting that the lower bounding functions can be computed iteratively. That is, each lower bound needs the computation of a single partial distance term, plus two addition operations in order to update the bound value adopted at the previous step. It is also interesting to note that the adoption of any different partitioning scheme which maintains the subdivision of the blocks into equally sized subsets would perform similarly and would require the same memory footprint. The choice of r turns out to be a parameter of the proposed technique. Increasing r would mean having lower bounding functions which better approximate the distance term, but would also need more computations for their calculations. As for block matching

IEEE COMPUTER SOCIETY

the size of the blocks is typically $N = \{8, 16\}$, we experimentally found out that in most cases the best results are obtained by choosing r equal to $\{4, 8\}$.

Furthermore it is important to point out that the performances of our method are affected by the initial value of D_m. In fact, if D_m is initialized with a value close to the global minimum of the distance function to be found in the search area, the sufficient conditions embodied in the proposed approach have good chances to discard a high number of non-optimal candidate blocks. Conversely, if D_m is initialized e.g. to the maximum value which the distance can assume, no blocks can be discarded within the initial positions until a local minimum is found. For this reason, a simple but effective improvement to the outlined method is to initialize D_m to $D_{m,0}$, that is the distance value corresponding to the candidate block at offset $(u, v) = (0, 0)$. We also propose an alternative approach, that is to initialize D_m to the global minimum corresponding to the best matching offset position found at the previous frame for the same block, $D_{m,mp}$. In order to deploy this, the technique has to keep trace of the motion flow of the previous frame. This approach can be seen as a very basic motion predictor, as it assumes as most probable motion offset that found in the previous frame.

Although other optimal techniques deploying lower bounding functions obtained by summations over partitioned blocks have been proposed [5], [3], [1], only our technique exploits the concept of partial distance in order to further refine the bounding functions.

4. Experimental results

In this section some experimental results are presented which compare the block matching technique described in Sec. 3 with the BF. The distance used is the SAD (i.e. $p = 1$), the block size is 16×16 (i.e. $N = 16$) and the search is performed on both directions on an offset equal to $[-16, +16]$, hence $M = 32$. For what means the proposed technique, we show the results obtained by choosing $r = \{4, 8\}$, which, as said before, turns out to be the best choice in most cases. The testing sequences used for the comparison are typical video sequences used for benchmarking motion estimation algorithms and are shown in Figures $1 \div 10$[1]. All algorithms have been implemented in C on a Linux workstation with a 1.5 GHz Pentium M CPU.

Tables 1 and 2 show the speed-ups in terms of ratios of measured execution time of the proposed algorithm versus the BF, the former referring to $r = 4$, the latter to $r = 8$. In both tables, the second column ($D_{m,0}$) refers to the initialization of D_m as the distance corresponding to candidate block at offset $(0, 0)$, while the third column ($D_{m,mp}$) refers to the initialization of D_m by means of the motion predictor as explained in Section 3. As it can be inferred from the tables, the proposed technique can speed-up notably the BF along the whole dataset. In the $D_m = D_{m,0}, r = 4$ case the speed-ups range from 2.1 to 12.9, in the $D_m = D_{m,mp}, r = 4$ case they range from 2.0 to 13.4. Similarly, in the $D_m = D_{m,0}, r = 8$ case the speed-ups range from 2.0 to 12.3, in the $D_m = D_{m,mp}, r = 8$ case they range from 2.0 to 12.5. For what means the method for initializing D_m, the motion predictor approach seems to bring more benefits, as it yields to almost the same results as the other approach in all cases except for the *Flower garden* sequence, where the speed-up obtained is more than doubled.

Table 3 and 4 show the speed-ups in terms of ratios of number of elementary operations of the proposed algorithm versus the BF and, as previously, the former refers to $r = 4$, the latter to $r = 8$. The elementary operations considered refer to the high-level code, and are subdivided into three groups: *"If"* for branch instructions, *"+/-"* for additions and subtractions, *"Abs"* for absolute values. Similarly to Table 1, columns $2, 3, 4$ ($D_{m,0}$) refers to the initialization of D_m as the distance corresponding to candidate block at offset $(0, 0)$, while columns $5, 6, 7$ ($D_{m,mp}$) refers to the initialization of D_m by means of the motion predictor. As it can be seen, the proposed technique allows for a significant reduction in terms of operations for what regards additions, subtractions and absolute values. Obviously, the number of branch operations is always increased compared to the BF, due to the high number of tests performed when applying the sufficient conditions for discarding candidate blocks. Nevertheless, it is worth noting that for all the tested video sequences, the percentage of branch instructions never accounts for more than 0.13% of the total number of operations performed by the BF.

Table 1. Speed-ups of the proposed algorithm vs. BF, $r = 4$

Sequence	$D_{m,0}$	$D_{m,mp}$
Claire	12.9	13.0
Miss America	2.1	2.0
Salesman	12.6	13.4
Flower garden	2.9	7.3
Table tennis	2.4	3.0
Grandmother	5.3	4.9
Mr. Chest	12.4	11.9
Trevor	5.9	5.8
Surfside	3.9	3.9
Football	5.2	5.3
Average	6.6	7.1

[1] Notation $\sharp i(k)$ means $i - th$ frame out of a sequence of k frames

0-7695-2877-5/07 $25.00 © 2007 IEEE

IEEE COMPUTER SOCIETY

Table 2. Speed-ups (ratios of operations) of the proposed algorithm vs. BF, $r = 8$

Sequence	$D_{m,0}$	$D_{m,mp}$
Claire	12.3	12.4
Miss America	2.0	2.0
Salesman	11.5	12.5
Flower garden	3.9	7.4
Table tennis	2.4	3.2
Grandmother	4.8	4.6
Mr. Chest	11.9	11.7
Trevor	5.5	5.6
Surfside	3.6	3.7
Football	5.0	5.1
Average	6.3	6.8

Table 3. Speed-ups (ratios of operations) of the proposed algorithm vs. BF, $r = 4$

		$D_{m,0}$			$D_{m,mp}$	
Sequence	If	$+/-$	Abs	If	$+/-$	Abs
Claire	0.8	21.5	23.9	0.8	21.4	23.8
Miss America	0.4	2.6	2.7	0.4	2.7	2.8
Salesman	0.8	23.2	26.4	0.8	27.0	31.2
Flower garden	0.5	3.7	3.9	0.6	11.6	12.7
Table tennis	0.4	3.0	3.1	0.4	4.2	4.4
Grandmother	0.6	6.8	7.2	0.5	6.7	7.0
Mr. Chest	0.8	21.7	25.0	0.7	21.7	25.0
Trevor	0.7	8.3	8.8	0.6	8.5	9.1
Surfside	0.5	4.7	4.9	0.5	4.7	5.0
Football	0.6	7.8	8.4	0.5	7.8	8.3

5 Conclusions

We have described a novel technique for performing fast and optimal block matching. The proposed technique relies on the concept of partial distance and on a proper partitioning of the blocks in order to deploy a succession of lower bounding functions for the distance term which allow for rapidly and safely discarding non-matching candidates, hence decreasing the computational burden of the block matching process. The computation of the bounding functions can be rapidly executed by means of standard incremental techniques. Experimental results show the better efficiency of the proposed technique with regards of the BA, in terms of reduction of number of operations as well as in terms of reduction of execution times. Future works will be aimed at comparing our technique with other state of the art fast optimal block matching algorithms.

Table 4. Speed-ups in terms of reduction of N. elementar Ops, proposed algorithm vs. BF, $r = 8$

		$D_{m,0}$			$D_{m,mp}$	
Sequence	If	$+/-$	Abs	If	$+/-$	Abs
Claire	0.8	25.3	30.1	0.7	25.2	29.9
Miss America	0.3	2.8	3.0	0.3	2.9	3.1
Salesman	0.8	28.8	36.0	0.8	32.1	41.2
Flower garden	0.4	4.3	4.6	0.6	15.2	18.1
Table tennis	0.3	3.6	3.9	0.4	5.6	6.2
Grandmother	0.5	7.2	7.9	0.5	7.1	7.7
Mr. Chest	0.8	29.9	39.2	0.7	29.9	39.2
Trevor	0.6	9.7	10.9	0.5	10.0	11.3
Surfside	0.4	5.1	5.6	0.4	5.2	5.6
Football	0.5	9.5	10.8	0.5	9.6	10.9

References

[1] M. Brunig and W. Niehsen. Fast full-search block matching. *IEEE Transactions on Circuits and Systems for Video Technology*, 11(2):241–247, 2001.

[2] F. Crow. Summed-area tables for texture mapping. *Computer Graphics*, 18(3):207–212, 1984.

[3] X. Gao, C. Duanmu, and C. Zou. A multilevel successive elimination algorithm for block matching motion estimation. *IEEE Trans. Image Processing*, 9(3):501–504, 2000.

[4] Y. Huang, C. Chen, C. Tsai, C. Shen, and L. Chen. Survey on block matching motion estimation algorithms and architectures with new results. *The Journal of VLSI Signal Processing*, 42(3):297–320, 2006.

[5] C. Lee and L. Chen. A fast motion estimation algorithm based on the block sum pyramid. *IEEE Trans. Image Processing*, 6(11):1587–1591, 1997.

[6] R. Li, B. Zeng, and M. L. Liou. A new three-step search algorithm for block motion estimation. *IEEE Transactions on Circuits and Systems for Video Technology*, 4(4):438–442, 1994.

[7] W. Li and E. Salari. Successive elimination algorithm for motion estimation. *IEEE Trans. on Image Processing*, 4(1):105–107, 1995.

[8] M. Mc Donnel. Box-filtering techniques. *Computer Graphics and Image Processing*, 17:65–70, 1981.

[9] A. Moradi, R. Dianat, S. Kasaei, and M. Shalmani. Enhanced cross-diamond-hexagonal search algorithms for fast block motion estimations. In *IEEE Conference on Advanced Video and Signal Based Surveillance*, pages 558–563, Como, Italy, September 2005.

[10] F. Tombari, S. Mattoccia, and L. Di Stefano. Template matching based on the l_p norm using sufficient conditions with incremental approximations. In *Proc IEEE Int. Conf. on Advanced Video Surveillance Systems (AVSS 2006)*, Sydney, Australia, November 2006.

0-7695-2877-5/07 $25.00 © 2007 IEEE

IEEE COMPUTER SOCIETY

Figure 1. *Claire* **sequence:** $\sharp 50(60)$

Figure 2. *Miss America* **sequence:** $\sharp 50(149)$

Figure 3. *Salesman* **sequence:** $\sharp 50(448)$

Figure 4. *Flower garden* **sequence:** $\sharp 50(60)$

Figure 5. *Table tennis* **sequence:** $\sharp 50(60)$

Figure 6. *Grandmother* **sequence:** $\sharp 50(60)$

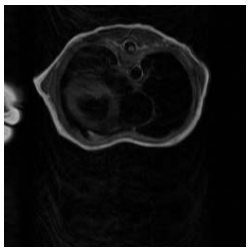

Figure 7. *Mr. Chest* **sequence:** $\sharp 50(77)$

Figure 8. *Trevor* **sequence:** $\sharp 50(99)$

Figure 9. *Surfside* **sequence:** $\sharp 10(19)$

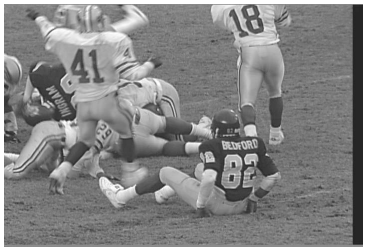

Figure 10. *Football* **sequence:** $\sharp 50(59)$

IEEE COMPUTER SOCIETY