A new algorithm of fast SSD calculation for motion estimation

E A Altman and E I Zakharenko
Omsk State Transport University, Omsk, 644050, Russia

Abstract. This paper considers motion estimation algorithms using a Sum of Squared Differences (SSD) metric and making an exhaustive search of motion vectors. A new fast algorithm, based on a new method of fast 2D convolution, is presented. It reduces a number of required arithmetic operations for finding all motion vectors in some area, compared to early published algorithms. The paper gives equations, underlines the presented algorithm, shows the tables comparing a new algorithm to another one in various cases. The new algorithm reduces the computational amount to 3.4%.

1 Introduction

A block matching algorithm is one of motion estimation methods. It is used for several problems of video processing, such as a video compression or an enhancement [1], [2].

The aim of this algorithm is looking to a fragment of a target frame which a best matching to a block of a source frame. The block of the source frame is a group of a neighbor pixels square form. A typical size of the block is 8×8 or 16×16 pixels. The fragment of the target frame is the frame block or an image, obtained from the frame by shifting him on half or quarter pixels. A searching area of the fragment is located around the block position in the source frame.

A block matching algorithms are classified by criteria of the matching and by methods of the searching.

As a criterion of the matching are used various metrics such as SAD (sum of absolute difference), SSD (Sum of Squared Differences) and others [3]. Commonly, SSD has the best quality of a matching evaluation, but the SAD requires less numbers of calculations [4].

There are two groups of methods for searching the best matching fragment. Methods of the first group find a fragment looking exhaustively over all possible positions of the block in the target frame [5].

Methods of the second group use a suboptimal algorithm, which presumes a smooth dependency of the used criterion on the fragment coordinate. They are based on an iterative searching procedure, have a small work time and a reasonable quality of result [6].

A choice of a group of methods depends on the problem that has to be solved. There are lots of problems, which demand the best quality of a block matching evaluation, and there are widely used problems, like motion estimation for video compression, which demand the fastest algorithms. Therefore, both groups of methods are being intensively researched.

In the paper we consider the first group. The main method of this group is brute force. There are two possibilities to perform an exhaustive searching with fewer operations, than brute force requires. The first possibility is an adaptation of fast algorithms of digital signal processing (like fast convolution) for calculation of a matching criterion (usually SSD) [7].
The second possibility is skipping surely bad positions of the fragment target frame. Under certain conditions applying this possibility reduces a number of arithmetic operations about 50 times (for example the New Fast Multilevel Successive Elimination Algorithm [8]).

These possibilities give the reduction of the number of operations of the same order [9]. Although elimination algorithms require less operations, fast algorithms have a regular computational scheme and often give faster implementation [10].

For the fast calculation of the matching criterion SSD, there are algorithms of fast calculation of a correlation, that based on a fast Fourier transform [4] or on number theoretic transforms [11]. This paper presents new algorithms of fast SSD calculation based on a new fast algorithm for computing the two-dimensional correlation [12].

2. Methods

The sum of squared difference (SSD) of two dimensional signals \( X \) and \( Y \) is equivalent to an Euclidean norm:

\[
SSD_{i,j} = \sum_{m=0}^{N_x-1} \sum_{n=0}^{N_y-1} (X_{m,n} - Y_{m+i,n+j})^2,
\]

where \( i, j \) – coordinates of the current candidate motion vector, \( Y \) is a candidate block in position \((i, j)\) relative to the current block \( X \), \( N_x \times N_y \) – size of the current block \( X \).

The double summation in the equation (1) requires a much calculations for each point of the result. To reduce the number of the operation, the expression (1) has been decomposed into three components:

\[
\sum_{m=0}^{N_x-1} \sum_{n=0}^{N_y-1} X_{m,n}^2 + \\
\sum_{m=0}^{N_x-1} \sum_{n=0}^{N_y-1} Y_{m+i,n+j}^2 - \\
\sum_{m=0}^{N_x-1} \sum_{n=0}^{N_y-1} X_{m,n} Y_{m+i,n+j},
\]

The first component (2) does not depend on the arguments of the (1), thus when searching for the minimum value of this metric its calculation can be omitted.

The evaluation of the second component (3) requires \( M^2N^2 \) multiplications and \( M^2(N^2-1) \) additions, where \( N \times N \) – size of the current block \( X \), \( M \times M \) – coordinate range \( i, j \). There are fast algorithms reducing number of calculations for its evaluating. They use an overlap of blocks, which appears when searching for motion vectors in adjacent points. These algorithms require \((M+N-1)(N+2(M-1)) \times (M+N-1)^2 \) multiplications and \((M+N-1)(N+2(M+N-1)(M-1)+M(N-1)+2M(M-1) \) additions [13].

The third component (4) is equivalent to a linear 2D correlation. It requires the largest number of operations among all component decomposition: \( M^2 \) multiplications and \( M^2(M-1) \) additions.

There are several ways for fast computation of convolution. The first group of methods uses fast algorithms for cyclic convolution based on fast Fourier transform or another number-theoretic transform [11]. In this paper, we consider methods of the second group based on "divide and conquer" principle – they divide convolution into several smaller convolutions.

Paper [14] describes a fast algorithm for SSD computation based on a decomposition of equation (4) into 9 convolution of quarter size. Two dimensional signals \( X \) and \( Y \) are divided into odd and even sub-signals on each dimension:

\[
X_{i,j} = [\hat{x}_{i,j}, \hat{x}_{i,j+2}, \ldots, \hat{x}_{i,j+N-2}]^T, \ Y_{i,j} = [\hat{y}_{i,j}, \hat{y}_{i,j+2}, \ldots, \hat{y}_{i,j+L-2}].
\]
The equation (4) may be rewritten as:

\[ C_{i,j} = \begin{bmatrix} c_{i,j} \\ c_{i+1,j} \\ c_{i,j+1} \\ c_{i+1,j+1} \end{bmatrix} = \begin{bmatrix} Y_{i,j} & Y_{i+1,j} & Y_{i+1,j+1} \\ Y_{i+1,j} & Y_{i+2,j} & Y_{i+2,j+1} \\ Y_{i,j+1} & Y_{i+1,j+1} & Y_{i+1,j+2} \\ Y_{i+1,j+1} & Y_{i+1,j+2} & Y_{i+2,j+2} \end{bmatrix} \begin{bmatrix} X_{0,0} \\ X_{1,0} \\ X_{0,1} \\ X_{1,1} \end{bmatrix}, \quad (6) \]

Introducing the notation:

\[ A_0 = X_{0,0} - X_{1,0}, \quad A_1 = X_{0,1} - X_{1,1}, \quad B_{i,j} = Y_{i,j} + Y_{i+1,j}, \quad (7) \]

the equation (6) can be transformed into:

\[ C_{i,j} = \begin{bmatrix} (B_{i,j} + B_{i,j+1})X_{0,0} \\ (B_{i,j} + B_{i+1,j+1})X_{1,0} \\ (B_{i,j+1} + B_{i+1,j+2})X_{0,1} \\ (B_{i+1,j+1} + B_{i+1,j+2})X_{1,1} \end{bmatrix} + \begin{bmatrix} -1 & 0 \\ 0 & -1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} B_{i,j}(X_{0,0} - X_{0,1}) \\ B_{i,1}(X_{0,0} - X_{1,1}) \end{bmatrix} + \begin{bmatrix} -1 \\ -1 \\ -1 \\ -1 \end{bmatrix}(A_0 - A_1). \quad (8) \]

The calculation of a convolution of quarter size requires $9/16(N^2 \times M^2)$ cycles. In addition, we have to calculate input data for these convolutions (preprocessing) and to unite their output data (post processing) for obtaining the result of source convolution. The preprocessing includes $5N^2/4+2(N+M−2)×(N+M−1)+(N+M−2)^2$ cycles. The post processing takes $10M^2/4$ cycles.

Decomposition (8) may perform recursively until a trivial convolution of size $2 \times 2$. Without pre and post processing operations, cost of convolution is divided on $16/9$ times for each decomposition.

The cost of pre and post processing reduces a win obtained by the decomposition. For small size convolutions this cost is too big.

In this paper, we propose another decomposition method for 2D convolution for the motion estimation problem. This method has small expenses for pre and post processing and suits well to our task.

3. Results
Let's introduce the designations:

\[ A_0 = X_{0,0} - X_{1,0}, \quad A_1 = X_{0,1} - X_{1,1}, \]

\[ B_{0,i,j} = Y_{i,j} - Y_{i+1,j}, \quad B_{1,i,j} = Y_{i,j+1} - Y_{i+1,j+1}, \quad (9) \]

\[ c_{i+1,j} = X_{0,0}Y_{i+1,j} + X_{0,1}Y_{i+1,j+1} + X_{1,0}Y_{i+2,j} + X_{1,1}Y_{i+2,j+1}, \]

\[ c_{i+1,j+1} = X_{0,0}Y_{i+1,j+1} + X_{0,1}Y_{i+1,j+2} + X_{1,0}Y_{i+2,j+1} + X_{1,1}Y_{i+2,j+2}. \]

Using these designations we can transform the equation (6) to the next:

\[ C_{i,j} = \begin{bmatrix} A_0B_{0,i,j} + A_1B_{1,i,j} + c_{i+1,j} \\ A_0B_{1,i,j} + A_1B_{0,i,j} + c_{i+1,j+1} \\ c_{i+1,j} \\ c_{i+1,j+1} \end{bmatrix}, \quad (10) \]
There are 12 convolution of the quarter size in the equation (4). They require \(3/4(N^2\times M^2)\) cycles. The preprocessing and post processing take jointly includes \(N^2/2+3/2M^2\) cycles, where \(N\times N\) – size of the current block \(X\), \(M\times M\) – coordinate range \(i, j\).

Because of the low cost of pre and post processing, the algorithm based on the equation (10) is more effective than the algorithm based on the equation (8) in case of small sized convolution. Table 1 shows the numbers of operations of both algorithms for several sizes of the convolution. The size of \(X\) is \(N\times N\) and \((2N-1)\times(2N-1)\) – size of \(Y\).

**Table 1.** The number of operations of the convolution for two methods of the decomposition.

| N   | The number of operations |
|-----|-------------------------|
|     | 9 convolution | 12 convolution |
| 2   | 28           | 28           |
| 4   | 501          | 485          |
| 8   | 6 106        | 6 709        |
| 16  | 65 631       | 86 189       |
| 32  | 666 556      | 1 073 533    |
| 64  | 6 565 473    | 13 170 845   |

We can reduce the number of operations by using the various methods on different stages of the recursive decomposition. The idea of a new fast algorithm of the convolution consists in the recursive decomposition using the best method on each stage. Samples of a benefit of the new algorithm are shown in table 2.

**Table 2.** The number of the operations of the various algorithms of the calculation of the convolution.

| N   | The number of operations |
|-----|-------------------------|
|     | The fastest algorithm from the previously known | New algorithm |
| 2   | 28          | 28          |
| 4   | 485         | 485         |
| 8   | 6 106       | 5 962       |
| 16  | 65 631      | 64 335      |
| 32  | 666 556     | 654 892     |
A new algorithm of a SSD calculation uses the described above algorithm of the convolution to find the third component of the equation (4). Table 3 shows the number of operations is required for the SSD calculation by various algorithms.

**Table 3.** The number of operations required for the SSD calculation by various algorithms.

| N    | The fastest algorithm from the previously known | New algorithm |
|------|-----------------------------------------------|---------------|
| 2    | 207                                           | 207           |
| 4    | 2 178                                         | 2 106         |
| 8    | 21 003                                        | 20 355        |
| 16   | 201 528                                       | 195 696       |
| 32   | 1 926 597                                     | 1 874 109     |
| 64   | 18 309 546                                    | 17 837 154    |

4. Discussion

The presented algorithm of the motion estimation reduces a quantity of calculation in most cases. It doesn’t take an additional memory or other resources. A disadvantage of this algorithm is a complexity and a bigger size of its implementation.

For a practical application of the discussed methods, the next useful step of research is developing a code generation software. In order to generate a program for a particular case it must analyze the available methods of decomposition and pick the best one. Such approach also enables taking into consideration a hardware specific.

In the sense of a theoretical research, the represented algorithm decreases the number of the operations for calculation of 2D convolution, but the question of the minimum quantity of operations is still open.

References

[1] Madisetti V 2009 *Wireless, networking, radar, sensor array processing, and nonlinear signal processing* (CRC press)

[2] Anand Deshpande P P and Rao D 2014 Motion Estimation Algorithms in Video Super-Resolution *Fifth International Conference on Recent Trends in Information, Telecommunication and Computing*

[3] Paramkusam A and Reddy V 2011 An efficient fast full search block matching algorithm with SSD criterion *India Conference (INDICON) Annual IEEE* pp 1–6
[4] Setkov A 2011 Matching quality investigation of Motion Estimation Algorithms 9th conference of open innovations community fract pp 133–140
[5] Mattoccia S, Tombari F, Di Stefano L and Pignoloni M 2007 Efficient and optimal block matching for motion estimation Image Analysis and Processing. 14th International Conference pp 705–710
[6] Lam C-W, Po L-M and Cheung C H 2003 A new cross-diamond search algorithm for fast block matching motion estimation Neural Networks and Signal Processing, 2003 (Proceedings of the 2003 International Conference vol 2), pp 1262–1265
[7] Lee C-H and Chen L-H 1997 A fast motion estimation algorithm based on the block sum pyramid IEEE Transactions on Image Processing 6(11) pp 1587–1591
[8] Rao K S, Rao K N, Paramkusam A and Bhavana G 2017 An efficient multilevel motion estimation in video coding International Journal of Mechanical Engineering and Technology 8 pp 724–730
[9] Zakharenko E I and Altman E A 2015 Fast Full-Search Motion Estimation Method Based On Fast Fourier Transform Algorithm AIST (Supplement) pp 183–186
[10] Kilthau S L, Drew M S and Moller T 2002 Full search content independent block matching based on the fast fourier transform Image Processing. International Conference vol 1 pp I–I
[11] Toivonen T Heikkilä J and Silvén O 2002 A new algorithm for fast full search block motion estimation based on number theoretic transforms Proceedings of the 9th International Workshop on Systems, Signals, and Image Processing pp 90–94
[12] Altman E A and Zakharenko E I 2015 Fast algorithm for computing the two-dimensional correlation for video processing Proceedings of TUSUR 2(36) pp 119–124
[13] Gao X, Duanmu C and Zou C 2000 A multilevel successive elimination algorithm for block matching motion estimation IEEE Transactions on Image Processing 9(3) pp 501–504
[14] Naito Y, Miyazaki T and Kuroda I 1996 A fast full-search motion estimation method for programmable processors with a multiply-accumulator Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. IEEE International Conference 6 pp 3221–3224