A Fast Block Matching Technique Using a Gradual Voting Strategy

Jik-Han JUNG†, Hwal-Suk LEE†, Nonmembers, and Dong-Jo PARK†, Member

SUMMARY In this letter, a novel technique for fast block matching using a new matching criterion is proposed. The matching speed and image quality are controlled by the one control parameter called matching region ratio. An efficient matching scheme with a gradual voting strategy is also proposed. This scheme can greatly boost the matching speed. The proposed technique is fast and applicable even in the presence of speckle noise or partial occlusion.

key words: block matching, gradual voting strategy, direct-address table

1. Introduction

The block matching technique has been widely used for image coding, stereo vision, visual tracking and so on. However, the full-search algorithm (FSA) requires heavy computational burden so that faster algorithms have been proposed in the last two decades. They can mainly be divided into four categories: 1) fast search techniques that reduce the number of candidate locations, such as the three step search (TSS), the block-based gradient decent search (BBGDS), and the hexagon-based search (HEXBS) [1]–[3]; 2) techniques based on pixel patterns or motion field decimation like N-Queen decimation [4]; 3) algorithms using a mathematical inequality based on sum norms, such as the block sum pyramid (BSP) and the winner-update algorithm (WUA) [5], [6]; and 4) exploitation of different matching criteria [7]–[9].

Because the search pattern in the above group 1) and the template decimation pattern in group 2) have been determined on the basis of off-line statistical analyses, the performance of the block matching is degraded in real applications. In a dynamic environment, the best search pattern and the best template decimation pattern cannot be easily determined.

Based on the property that the best match also has a spatial intensity distribution similar to the template block, we propose a new matching method. It gradually increases the number of candidate locations and matching parts of the template block. As a result, the proposed method allows a dynamic decision for the search pattern and the template decimation pattern. In addition, an efficient matching scheme with a gradual voting strategy is proposed that boosts the processing speed. The proposed technique is fast and applicable even in the presence of speckle noise or partial occlusion.

2. Matching Criterion

2.1 Conventional Criterion

Common block matching algorithms utilize the sum of absolute differences for the cost function. Let us assume that there are two images, \( I_{r-1} \), reference image, and \( I_r \), current image. The matching error between the block at position \((x, y)\) in \( I_r \) and the candidate block at position \((x+u, y+v)\) in \( I_{r-1} \) is usually defined as following: 

\[
SAD_{(x,y)}(u, v) = \sum_{i=0}^{B-1} \sum_{j=0}^{B-1} |I_r(x+i, y+j) - I_{r-1}(x+u+i, y+v+j)| 
\]

where the block size is \( B \times B \). Most matching techniques search for the best estimate \((\hat{u}, \hat{v})\) that gives the global minimum of the matching error. That means, 

\[
(\hat{u}, \hat{v}) = \arg \min_{(u,v)\in A} SAD_{(x,y)}(u, v), \quad \text{where } A \text{ is the search area.}
\]

2.2 Proposed Matching Criterion

Let us assume that there is a candidate block at position \((\hat{u}, \hat{v})\) in the search area that has the same appearance as the template block. Then, the matching error of the candidate block is zero: 

\[
SAD_{(x,y)}(\hat{u}, \hat{v}) = 0.
\]

In this case, we can find the best matching position not by calculating all matching errors, but by searching for the block which has the same appearance. For example, the best match that is similar to the template (Fig. 1 (a)) can be found at position \((2, 1)\) in the search area (Fig. 1 (c)).

In order to measure similarity to the template block, the block matching score \( S \) and the pixel matching score \( P \) are defined as

\[
S_{(x,y)}(u, v) = \sum_{j=0}^{B-1} \sum_{i=0}^{B-1} P_{(x,y,u,v)}(i, j) 
\]

\[
P_{(x,y,u,v)}(i, j) = \begin{cases} 
1, & |I_r(x+i,y+j) - I_{r-1}(x+u+i,y+v+j)| \leq \delta_l \\
0, & \text{otherwise},
\end{cases}
\]

where \( \delta_l \) is the margin of the intensity difference, which increases gradually during the matching process. If we calculate the block matching score \( S_{(x,y)}(2, 1) \) in Fig. 1 (c), where \( B = 4 \) and \( \delta_l = 0 \), we have \( S_{(x,y)}(2, 1) = B^2 = 16 \). This score indicates that the candidate block at \((2, 1)\) is perfectly matched to the template.
2.3 Matching Procedure

Let us assume that there is no exact copy in the search area. Then the normal matching procedure is processed as follows:

1. Initialize $\delta_I$ to 0.
2. Calculate matching score $S(x, y)(u, v)$ for each block at every $(u, v)$ position.
3. Search the matching position $(\tilde{u}, \tilde{v})$ with a block matching score of $S(x, y)(\tilde{u}, \tilde{v})$ that is equal to or greater than $R \cdot B^2$, where $R$ is the matching region ratio given primarily: $S(x, y)(\tilde{u}, \tilde{v}) \geq R \cdot B^2$.
4. If no candidate $(\tilde{u}, \tilde{v})$ is found, increase the margin of the intensity difference $\delta_I$ by 1 and return to Step 2.
5. If several candidates are found in Step 3, choose the best candidate that has the smallest mean of absolute difference (MAD). For fairness, MAD’s should be calculated over only the matched pixels, where the pixel matching score $P(x, y, u, v)(i, j)$ is 1.

Figure 1 (b) shows another template block that has 6 different intensities in comparison with the first template Fig. 1 (a). And Fig. 2 shows the matching process of a candidate block at position (2, 1) in the search area. Let us assume that $R$ is set to 0.75. If the margin $\delta_I$ is zero, only 10 pixels are matched, which have gray color in Fig. 2 (a). However, as the margin of the intensity difference $\delta_I$ increases, the number of matched pixels also increases like Figs. 2 (b) and 2 (c). In the case $\delta_I = 2$, we can obtain the matching position with $R \cdot B^2 = 0.75 \times 4^2 = 12$ matched pixels. The final block matching scores of all candidates are shown in Fig. 2 (d) and the best candidate is located at position (2, 1).

3. Fast Matching Scheme

It is time consuming to repeatedly calculate all scores when the margin of the intensity difference $\delta_I$ varies, so we adopt a direct-address table and a gradual voting strategy to avoid excessive computation.

3.1 Direct-Address Table

First, we generate a direct-address table for the search area as depicted in Fig. 3. By using this table, all pixels with a specific intensity can be found directly. For example, if we need pixel positions that have an intensity of 254 in the search area, we can find the pixel positions at a glance, which are (5, 4) and (5, 6), as noted in Fig. 3.

3.2 Gradual Voting Strategy (GVS)

With the generated direct-address table, we can find the pixel position $(u, v)$’s that have the same intensity as a pixel
Algorithm 1  Fast Matching Scheme

I. VARIABLES
Num(k) : number of pixels in search area which have intensity k
Direct_address_u,v(k, l) : location (u, v) of the l-th pixel of intensity k
Template(i, j) : intensities of the template block
Vote_table(u,v) : block matching score table, S(u, v)
Max_vote : maximum score of S(u, v)
Max_vote_u, Max_vote_v: (u, v) value of the Max_vote
δI : margin of intensity difference
B : block size
R : matching region ratio, control parameter

II. GRADUAL VOTING STRATEGY (GVS)
Vote_table(u,v) = 0
Max_vote = 0
δI = 0
While( Max_vote < R·B² )
  For( j = 0 ; j < B ; j ++ )
    For( i = 0 ; i < B ; i ++ )
      k = Template(i, j) ± δI
      u = Direct_address_u(k, l) - i
      v = Direct_address_v(k, l) - j
      Vote_table(u, v) ++
      If ( Max_vote ≤ Vote_table(u, v) )
        Max_vote = Vote_table(u, v)
        Max_vote_u = u
        Max_vote_v = v
      End If
    End For i, j
  End For j
  δI = δI + 1
End While
Return (Max_vote_u, Max_vote_v)

at position (i, j) in the template block. Then, the candidate block at position (u − i, v − j) is allotted one point. After accumulating scores, we determine the best match that has the maximum score. Until the best match reaches a threshold score R·B², the margin of the intensity difference increases. The proposed algorithm is stated in Algorithm 1.

3.3 Reduction of Computational Cost

According to (1), evaluating one S(u, v) requires B² pixel decisions for each δI. Without fast matching scheme, it requires 256 × B² operations to exploit full δI’s, from 0 to 255. However, in the proposed fast matching scheme, each pixel-to-pixel matching is treated only once so that only B² operations are required totally. It can greatly boost the matching speed.

4. Simulation Results

When the size of the search area is M × M and the block size is B × B, the number of operations of the full-search algorithm is (M − B + 1)²B². Our proposed algorithm requires only M² operations to generate the direct-address table and B² for matching in the best case. For performance evaluation, five template blocks of size 8 × 8 in Figs. 4(b)-(f) were used, and the search area of size 256 × 256 is given in Fig. 4(a).

The operation count and execution time per block matching are summarized in the Table 1. The four well-known block matching schemes, FSA, TSS, BSP, and WUA, were contrasted with the proposed scheme (GVS), where the matching region ratio R is 0.1. The five algorithms were implemented using Visual C++ 6.0 on a machine with the Intel Core 2 Quad Q8200 2.33 GHz processor. Five sub-tables were extracted with five different template blocks, which are noted below each sub-table. Notice that TSS is fast but its PSNR is low.

Tests I-IV show that the proposed technique outperforms the conventional algorithms. The proposed algorithm is fast and applicable even in the presence of speckle noise or partial occlusion, as emphasized in Tests II & III. A counterexample is given in Test V, where additive white gaussian noise is severe. In such case, the effect of matching region ratio become clear. As the matching region ratio increases, matching error become reduced with degradation of speed, as shown in the Table 2. The matching region ratio R is an important parameter controlling the matching speed and the
Table 1  Performance comparison.

| Algorithm | Operations Number | Execution Time μs | Speed % | PSNR (dB) |
|-----------|-------------------|-------------------|---------|-----------|
| FSA       | 3968064           | 100               | 60855   | 100       | 1         |
| TSS       | 3648              | 0.09              | 64      | 0.11      | 951       |
| BSP       | 62064             | 1.56              | 3819    | 6.28      | 16        |
| WUA       | 622069            | 1.37              | 3425    | 5.63      | 18        |
| GVS       | 24117             | 0.61              | 2355    | 3.87      | 26        |

* Test I : Template Block of Clear Image

| Algorithm | Operations Number | Execution Time μs | Speed % | PSNR (dB) |
|-----------|-------------------|-------------------|---------|-----------|
| FSA       | 3968064           | 100               | 64000   | 100       | 1 17.86   |
| TSS       | 3648              | 0.09              | 67      | 0.1       | 955      |
| BSP       | 363273            | 15.93             | 13180   | 20.59     | 5 17.86   |
| WUA       | 95385             | 2.4               | 4578    | 7.15      | 14 17.86  |
| GVS       | 22175             | 0.56              | 2352    | 3.54      | 14 17.86  |

* Test II : Template Block of Noisy Image (White & Black 10%)  

| Algorithm | Operations Number | Execution Time μs | Speed % | PSNR (dB) |
|-----------|-------------------|-------------------|---------|-----------|
| FSA       | 3968064           | 100               | 66426   | 100       | 1 11.92   |
| TSS       | 3648              | 0.09              | 71      | 0.11      | 949      |
| BSP       | 1773523           | 44.69             | 30688   | 46.2      | 2 11.92   |
| WUA       | 791765            | 19.95             | 20221   | 30.44     | 3 11.92   |
| GVS       | 18214             | 0.46              | 2352    | 3.54      | 11 11.92  |

* Test III : Template Block of Noisy Image (White & Black 30%)  

| Algorithm | Operations Number | Execution Time μs | Speed % | PSNR (dB) |
|-----------|-------------------|-------------------|---------|-----------|
| FSA       | 3968064           | 100               | 62206   | 100       | 1 29.2    |
| TSS       | 3648              | 0.09              | 65      | 0.1       | 957      |
| BSP       | 365313            | 9.21              | 9049    | 14.55     | 7 29.2    |
| WUA       | 82561             | 2.08              | 4470    | 7.19      | 14 29.2   |
| GVS       | 70079             | 1.77              | 4447    | 7.15      | 14 29.2   |

* Test IV : Template Block of Noisy Image (Additive Gaussian, μ = 0, σ = 30)  

| Algorithm | Operations Number | Execution Time μs | Speed % | PSNR (dB) |
|-----------|-------------------|-------------------|---------|-----------|
| FSA       | 3968064           | 100               | 65388   | 100       | 1 17.15   |
| TSS       | 3648              | 0.09              | 65      | 0.1       | 957      |
| BSP       | 1152905           | 29.05             | 22105   | 33.81     | 3 17.15   |
| WUA       | 816485            | 20.58             | 20501   | 31.35     | 3 17.15   |
| GVS       | 62862             | 1.58              | 4098    | 6.27      | 16 13.04  |

* Test V : Template Block of Noisy Image (Additive Gaussian, μ = 0, σ = 30)  

Table 2  Effect of matching region ratio.

| Ratio (R) | Operations Number | Execution Time μs | Speed % | PSNR (dB) |
|-----------|-------------------|-------------------|---------|-----------|
| FSA       | 3968064           | 100               | 65388   | 100       | 1 17.15   |
| 0.1       | 62862             | 1.58              | 4098    | 6.27      | 16 13.04  |
| 0.2       | 227484            | 5.73              | 11435   | 17.49     | 6 14.49   |
| 0.3       | 308968            | 7.9               | 14632   | 22.38     | 4 14.01   |
| 0.4       | 603883            | 15.22             | 28516   | 43.61     | 2 14.25   |
| 0.5       | 857826            | 21.62             | 37634   | 57.55     | 2 14.91   |
| 0.6       | 1154511           | 29.1              | 58050   | 77.77     | 1 17.15   |

Table 2  Effect of matching region ratio.

that the best match also has a spatial intensity distribution similar to the template block. The proposed scheme gradually enlarges the candidate locations and the matching parts of the template block. As a result, the proposed method allows a dynamic decision regarding the search pattern and the template decimation pattern. In addition, an efficient matching scheme with a gradual voting strategy is proposed that boosts the processing speed. We described the pseudo code for the proposed algorithm and compared the performance of the proposed algorithm with those of two representative algorithms: the full-search algorithm and the winner-update algorithm. Simulation results show that the proposed matching algorithm is fast and applicable even in the presence of speckle noise or partial occlusion.

5. Conclusion

We proposed a new matching method based on the property of image quality, and can be chosen by observing the statistics of speckle noise or partial occlusion.

References

[1] T. Koga, K. Inuma, A. Hirano, Y. Iijima, and T. Ishiguro, “Motion-compensated interframe coding for video conferencing,” Proc. National Telecommunications Conference, pp.G5. 3. 1–G5. 3. 5, 1981.
[2] L.K. Liu and E. Feig, “A block-based gradient descent search algorithm for block motion estimation in video coding,” IEEE Trans. Circuits Syst. Video Technol., vol.6, no.4, pp.419–422, 1996.
[3] C. Zhu, X. Lin, L.P. Chau, K.P. Lim, H.A. Ang, and C.Y. Ong, “A novel hexagon-based search algorithm for fast block motion estimation,” Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP ‘01), pp.1593–1596, 2001.
[4] C. Wang, S. Yang, C. Liu, and T. Chiang, “A hierarchical n-queen decimation lattice and hardware architecture for motion estimation,” IEEE Trans. Circuits Syst. Video Technol., vol.14, no.4, pp.429–440, 2004.
[5] C. Lee and L. Chen, “A fast motion estimation algorithm based on the block sum pyramid,” IEEE Trans. Image Process., vol.6, no.11, pp.1587–1591, 1997.
[6] Y. Chen, Y. Hung, and C. Fuh, “Fast block matching algorithm based on the winner-update strategy,” IEEE Trans. Image Process., vol.10, no.8, pp.1212–1222, 2001.
[7] C. Zhu, W. Qi, and W. Ser, “Predictive fine granularity successive elimination for fast optimal block-matching motion estimation,” IEEE Trans. Image Process., vol.14, no.2, pp.213–221, 2005.
[8] O. Urban and S. Ertürk, “Constrained one-bit transform for low complexity block motion estimation,” IEEE Trans. Circuits Syst. Video Technol., vol.17, no.4, pp.478–482, 2007.
[9] Y.W. Huang, C.Y. Chen, C.H. Tsai, C.F. Shen, and L.G. Chen, “Survey on block matching motion estimation algorithms and architectures with new results,” J. VLSI Signal Processing, vol.42, no.3, pp.297–320, 2006.