Efficient Motion Estimation Using a Sorting-based Early Termination Algorithm in H.264 Video Coding

Chou-Chen Wang*, Jung-Yang Kao** and Yu-Kai Lin*
* Department of Electronic Engineering, I-Shou University, Kaohsiung, Taiwan
** ICL, Industrial Technology Research Institute, Hsinchu, Taiwan

VRLAB 2008507042 윤창걸
Abstract
1. Introduction
2. Overviews of PDS and NPDS
3. The proposed fast ME module in H.264
4. Simulation results
5. Conclusions
The H.264/AVC video coding standard

- 7 variable block sizes in interframe coding (ranging from 16×16 to 4×4)
- The motion estimation with 7 modes needs very high computational complexity
- We propose a new and fast motion-estimation algorithm
  - Based on partial block distortion for sorted significant features
    - Bit-plane
    - Absolute difference of means (ADM)
Abstract

- Partial distortion searching (PDS) algorithm
  - Is a popular method for fast full search (FSS) in H.264
  - Proposed-PDS
    - Find the same motion vectors as FSS
  - Normalized PDS (proposed-NPDS)
    - Relatively accurate motion vectors with a large reduced computational load
  - Simulation results
    - Proposed-PDS
      - About 90% of the computation needed by the FSS (lossless)
    - Proposed-NPDS
      - About 68% of the computation needed by the FSS (very close)
Efficient motion vector estimation using 7 modes
Introduction

- **Most important issues (a real-time H.264 encoder)**
  - Increasing the speed of motion estimation (ME)
  - Fast ME using block matching algorithm (BMA)
    - Lossy motion estimation algorithm
      - NTSS [3], DS [4], HEXBS [5] and NPDS [6]
    - Lossless motion estimation algorithm
      - SEA [7], PDS [8] and ASSA [9]
Introduction

- Propose system
  - Using the characteristics of patterns similarity matching errors
  - The cost value “mcost” used to estimate MV of a mode
    - Proportional to the image complexity
      - Bit-plane
      - Absolute difference of means (ADM)
Sub-block PDS

The ME process is to obtain a MV for a target MB by using the block matching technique, which minimizes a measure of matching distortion between the target MB in the current frame and a candidate MB within a search window in a reference frame.

One of the most frequently used criteria to measure the matching distortion:

- The sum of absolute difference (SAD)

\[
D(x, y; u, v) = \sum_{i=0}^{15} \sum_{j=0}^{15} |I_t(x+i, y+j) - I_{t-1}(x+i+u, y+j+v)|
\]
Sub-block PDS

- The block distortion $D(x, y; u, v)$ is divided into 16 partial distortions ($d_p$), where each partial distortion consists of 16 points.

$$d_p(x, y; u, v) = \sum_{i=0}^{3} \sum_{j=0}^{3} \left| I_i(x + i + s_p, y + i + t_p) - I_{i-1}(x + j + s_p + u, y + j + t_p + v) \right|$$

- Where $(s_p, t_p)$ is the offset of the upper left corner point of the $p$th partial distortion from the upper left corner point of the candidate block.

- The $p$th accumulated partial distortion

$$D_p(x, y; u, v) = \sum_{i=1}^{p} d_i(x, y; u, v)$$
Overviews of PDS and NPDS

- Sub-block PDS
  - The calculation order $d_p$ is a sequential top-to-bottom matching order
  - During each block matching
    - PDS compares each accumulated partial distortion $D_p$ with the current minimum distortion ($D_{\text{min}}$) so that we can early detect the impossible candidate motion vectors (CMVs)
Normalized PDS (NPDS)

The pth partial distortion

\[ d_p(x, y; u, v) = \sum_{i=0}^{3} \sum_{j=0}^{3} \left| I_i(x + 4i + s_p, y + 4j + t_p) - I_{i-1}(x + 4i + s_p + u, y + 4j + t_p + v) \right| \]
Overviews of PDS and NPDS

- Normalized PDS (NPDS)
  - The NPDS matches all the search points inside the search window as that in the FSA
  - The search begins at the origin search point and moves outward with a spiral scanning path
Normalized PDS (NPDS)

- The NPDS compares each accumulated partial distortion $D_p$ with the normalized minimum distortion (NMD: $pD_{\text{min}} / 16$)
  - $D_{\text{min}}$ is the current minimum distortion
- The comparison starts from $p = 1$ and proceeds toward $p = 16$
- The comparison stop
  - The normalized partial distortion of the CMVs is greater than the NMD
- The end of comparison (i.e. $p = 16$)
  - If $D_{16}$ is smaller than $D_{\text{min}}$, then this CMV becomes the new current minimum point
- Computational complexity is reduced by high rejection of impossible CMVs at early stage
The proposed fast ME module in H.264

Proposed system

- The motivations of the proposed algorithm
  - Two significant features
    - Bit-plane
    - Absolute difference of means (ADM)

- Binary bit-plane
  - Bit-plane

\[ B_p^l(x+4i+s_p, y+4j+t_p) = \begin{cases} 
1 & \text{if } I_l(x+4i+s_p, y+4j+t_p) \geq m_p^l \\
0 & \text{otherwise} \end{cases} \]
The proposed fast ME module in H.264

**Proposed system**

- If a sub-block has a bit-plane that is very different from that of another sub-block, the two blocks are unlikely to have similar image characteristics and result in a high matching error.

- The Hamming distance based on two bit-planes is used as the measurement of similarity.
  
  - The Hamming distance $H_p$
    
    $$H_p = \sum_{i=0}^{3} \sum_{j=0}^{3} B_p^{I_i}(x+4i+s_p, y+4j+t_p) \oplus B_p^{I_{i-1}}(x+4i+s_p, y+4j+t_p)$$

  - The Hamming distance for each sub-block in a MB is carried out and the magnitudes are sorted in the descending order.
The proposed fast ME module in H.264

- Proposed system
  - Absolute difference of means (ADM)

\[ ADM_p = \left| m_p^I - m_p^{I-1} \right| \]

- The ADM magnitudes for the same \( H_p \) are resorted in the descending order
The proposed fast ME module in H.264

- **Proposed system**
  - **The First stage**
    - Compute directional differences using binary pattern matching for all pixels in the center block of the search window
    - The magnitudes of Hamming distances of sub-blocks in a MB are sorted in descending order
  - **The Second stage**
    - If the sub-blocks have the same Hamming distances, then the ADM magnitudes of these sub-blocks are resorted in descending order
  - The positions used to determine the order of matching priority for all the candidate blocks in the search window
    - According to the arranged sub-block, we find the best MV using the same search procedure as the NPDS
The proposed fast ME module in H.264
Simulation results

(a) Foreman sequence.
(b) Mother and daughter sequence.

Fig. 6 Compare the PSNR-Y performance of different fast ME modules in H.264.

Table I. Average checking sub-block per mode required by different fast ME modules in H.264 with search window $31 \times 31$ in 30 frames/s.

<table>
<thead>
<tr>
<th>Sequences</th>
<th>Methods</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>16×16</td>
<td>16×8</td>
</tr>
<tr>
<td>Forman</td>
<td>JM-FS</td>
<td>3.01</td>
</tr>
<tr>
<td></td>
<td>Proposed-PDS</td>
<td>2.85</td>
</tr>
<tr>
<td></td>
<td>Proposed-NPDS</td>
<td>1.01</td>
</tr>
<tr>
<td>Mother daughter</td>
<td>JM-FS</td>
<td>2.73</td>
</tr>
<tr>
<td></td>
<td>Proposed-PDS</td>
<td>2.60</td>
</tr>
<tr>
<td></td>
<td>Proposed-NPDS</td>
<td>1.01</td>
</tr>
</tbody>
</table>
Conclusions

- By using the calculation order according to pattern similarity, we can obtain faster elimination of impossible candidate vectors than fast PDS algorithm in the H.264 encoder.