AN EPIPOLAR RESTRICTED INTER-MODE SELECTION FOR STEREOSSCOPIC VIDEO ENCODING

Guolei Yang1, Luhong Liang2, Wen Gao1

1 Peking University, P.R.China
2 Key Lab of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, P.R.China

ABSTRACT

Fast stereoscopic video encoding becomes a highly desired technique because the stereoscopic video has been realizable for applications like TV broadcasting and consumer electronics. The stereoscopic video has high inter-view dependency subject to epipolar restriction, which can be used to reduce the encoding complexity. In this paper, we propose a fast inter-prediction mode selection algorithm for stereoscopic video encoding. Different from methods using disparity estimation, candidate modes are generated by sliding a window along the macro-block line restricted by the epipolar. Then the motion information is utilized to rectify the candidate modes. A selection failure handling algorithm is also proposed to preserve coding quality. The proposed algorithm is evaluated using independent H.264/AVC encoders for left and right views and can be extended to MVC. Experimental results show that encoding times of one view are reduced by 41.4% and 24.4% for HD and VGA videos respectively with little quality loss.

Index Terms—Stereoscopic video, epipolar restriction, inter-mode selection.

1. INTRODUCTION

Recently, stereoscopic video becomes realizable in real-world applications such as TV broadcasting, cinema and consumer electronics, which usually require real-time coding of high-definition (HD) stereoscopic videos. However, encoding of HD videos by latest standards such as H.264 [1] is rather time-consuming and it becomes more challenging to fast encode stereoscopic videos that basically consist of left and right views. It is well known that the two views of stereoscopic video are highly dependent on each other. Inspired by this, in this paper, we explore the fast coding methods according to the dependency between the views of stereoscopic video.

Fast coding methods have been intensively researched for monoscopic and stereoscopic videos for decades. As for monoscopic videos, low complexity algorithms such as fast inter-mode decision [2][3] have been widely used. These algorithms utilize the spacial and temporal dependencies of videos and can be independently applied to each view of stereoscopic video, but they ignore the dependency between different views, which is an important clue for further reducing coding complexity. As for stereoscopic videos, fast encoding approaches have been proposed to take advantages of characteristics such as epipolar restriction between views. For example, in [4], Xiaoming Li et al. proposed a fast disparity and motion estimation scheme for the H.264 MVC framework by predicting motion vectors with view disparity information. In [5], Ping He et al. also provided a fast mode selection and disparity estimation algorithm which employed disparity prediction adjustment. In order to predict the motion vectors or modes, these algorithms need to determine the correspondence between views using inter-view disparity estimation in pixel or sub-pixel level. However, disparity estimation itself is rather time-consuming and consequently slows down the encoding process.

In this paper, we proposed a fast inter-mode selection method for stereoscopic video coding using epipolar restriction between the left and right views. Instead of disparity estimation, the proposed method generates a collection of candidate modes by sliding a mode-searching window along the macro-block row restricted by the epipolar-line. Then, the generated modes are rectified according to the local motion information to improve prediction accuracy. The algorithm also utilizes the dependency on rate-distortion costs between the two views to predict mode selection failure and invoke a mechanism to preserve the coding quality. Currently, we implement the proposed method on separately coded left and right views, since this two-independent-video solution on stereoscopic video is low-cost and compatible with monoscopic video, which is a competitive feature for real-world industrial applications. Experimental results show that the proposed algorithm gains an average speed-up of 41.4% for HD and 24.4% for VGA videos in encoding the second view with a quality loss less than 0.1dB. Moreover, the proposed method can utilize most of the fast mode decision algorithms designed for monoscopic videos for further acceleration and can be extended to coding schemes that eliminate inter-view redundancy by disparity estimation and compensation like MVC.
The rest of this paper is organized as follows. Section 2 presents the proposed algorithm with analysis. Experimental results are given by section 3. Section 4 draws conclusion of the paper and describes our future works.

2. THE PROPOSED ALGORITHM

2.1 Overview of the proposed algorithm

Due to the dependency between views, it is possible to accelerate mode decision of one MB by selecting mode candidates from MBs on its epipolar line. Without loss of generality, we assume that the left view is already coded and information such as MB modes and motion vectors are stored to generate candidate modes for MBs in the right view. Fig. 1 illustrates the overview of the proposed algorithm, where \( MB_R(m,n) \) is the MB to be coded in row \( m \) and column \( n \) of the Right view. The algorithm consists of three steps in general to select the prediction mode of a MB.

![Fig.1 Epipolar restricted mode selection](image)

(1) **Reusable mode generation.**

According to epipolar restriction, the most possible modes that can be reused by \( MB_R(m,n) \) should locate along the MB line with the same row as \( MB_L(m,n) \). To generate reusable modes, we employ a mode searching window that slides along this MB row. The sliding step of the window can be one or more sub-blocks to balance accuracy and efficiency. When the window moves to the position of one MB, the prediction mode used by this MB is added into the set of reusable modes.

(2) **Mode rectification.**

When the searching window located across two adjacent MBs, both of their prediction modes must be considered. Here, we employ the motion vectors of these two MBs to generate modes that may not appear in this MB line but is still likely to be the prediction mode of \( MB_R(m,n) \). Generated modes and reusable modes selected in step (1) are added into the set of rectified modes. Details of the rectification algorithm are described in Section 2.2.

(3) **Mode selection.**

Candidate modes for inter-prediction are selected from the set of rectified modes using Rate-Distortion Optimization (RDO). Since the similarities between the left and right views are rather complex due to various factors such as different motion and depth of objects in the scene, there must be some situations that the optimal prediction mode for \( MB_R(m,n) \) does not appear in the rectified modes set. Although the possibility of such situations is relatively low according to our observation, it still reduces the coding quality. To solve this problem, we employ rate-distortion cost (RDcost) thresholds determined according to the dependency of the two views to predict mode selection failure and invoke a mechanism to preserve the coding quality. Details are described in Section 2.3.

2.2 Rectification of reusable modes

Objects and scenes are similar in the two views for most parts. However, factors like view-disparity and various depths of objects cause slightly difference between views. For example, content of one MB could be separated into two MBs in the other view and the two parts may belong to different objects or two area of the same object with great difference. Observation on prediction modes of MBs with similar contents in different views reveals possible effect of such difference. Though not common, some MBs with close content hardly share the same or even close prediction mode, especially those content natural borders of objects. For these MBs, simply select reusable modes would cause serious error. An effective solution to this problem is to identify that two neighbor MBs or sub-blocks content two part of one object or two different objects, and then generate potential prediction modes on different conditions to reduce the possibility of mode prediction failure caused by the inter-view difference. Therefore, we employ a mode rectification scheme.

When the searching window moves across two MBs, namely, Left-MB and Right-MB, mode rectifications is performed to generate such potential modes. We use the Motion Vector (MV) of Left-MB and Right-MB or their sub-MBs i.e. \( MV_{left} \) and \( MV_{right} \) to identify contents of the two MBs, since different parts of one object are likely to have close MVs. Admittedly, for static object with low of zero MV, such a MV based scheme cannot work well, but the mode rectification process can still deal with most common moving objects and reduce prediction failure as much as possible. There are 4 conditions when potential modes need to be analyzed in mode rectification.

(1) **Both Left-MB and Right-MB use 16x16 mode.** We calculate \( Dist = |MV_{left} - MV_{right}| \) as the distance of \( MV_{left} \) and \( MV_{right} \). If \( Dist > Th_{diff} \), where \( Th_{diff} \) is the threshold used to judge whether the two vectors are generally different from each other or not, then we add 8x16 as a rectified mode.
(2) Both Left-MB and Right-MB use 8x16 mode. Calculate $\text{Dist} = |MV_{\text{left}} - MV_{\text{right}}|$. Here, $MV_{\text{left}}$ is MV of the right 8x16 sub-block in Left-MB and $MV_{\text{right}}$ is MV of the left 8x16 sub-block in Right-MB. If $\text{Dist} < T_{\text{sim}}$, where $T_{\text{sim}}$ is the threshold used to judge whether the two vectors are close enough or not, then we add 16x16 as a rectified mode.

(3) Both Left-MB and Right-MB use 16x8 mode. Calculate the MV distance between the two upper sub-blocks and that between the two bottom sub-blocks. If any of the two distances is larger than $T_{\text{diff}}$, then we add 8x8 as a rectified mode.

(4) Only Left-MB or Right-MB used mode 8x8. If no smaller modes (8x4, 4x8, 4x4) is used in the two 8x8 sub-blocks in left side of Right-MB or right side of Left-MB, we calculate the MV distance between the upper and bottom 8x8 sub-blocks near the MB border. If the distance is smaller than $T_{\text{sim}}$, then we add 8x16 as a rectified mode.

On all the other conditions, no extra mode needs to be generated. For clarity, Fig.2 illustrates mode rectification process on condition (3) as an example.

2.3 Implementation

The algorithm implementation shows in Fig.3. The process of reusable mode generation and mode rectification are performed for three times, in step 1, 3, and 5, on different areas of epipolar MB line, which are confined by $N_1$, $N_2$, and $N_3$. For each time, a set of candidate modes ($M$, $M'$, and $M''$) is selected for RDO. In step 2.2, 4.2, and 6.2, threshold of RDcost ($T_{\text{high}}$) is used to predict mode selection failure. In step 2.1, 4.1, 6.1, and 8.1, if the RDcost of $\text{mode}_i$ is smaller than $T_{\text{low}}$, we select $\text{mode}_i$, as prediction mode of $\text{MB}_i(m,n)$, then end the algorithm for early termination. Here, we employ the RDcost of MBs on the epipolar line to generate appropriate thresholds. Firstly, we define $RD\text{cost}_{\text{max}}$ and $RD\text{cost}_{\text{min}}$ as the maximum and minimum value of RDcost among the 10 MBs on the given MB’s epipolar line. Then, the two thresholds are define by $RD\text{cost}_{\text{max}}$ and $RD\text{cost}_{\text{min}}$ by

$$T_{\text{low}} = (1 + \beta)RD\text{cost}_{\text{max}}$$

for early termination of RDO and mistake prevention.

Algorithm: Fast Mode Selection of $\text{MB}_i(m,n)$

1. Move sliding window from $\text{MB}_i(m,n)$ to $\text{MB}_i(m,n-N_3)$ with a step-length of 8 pixel to gain candidate modes collection $M$.
2. for each mode $\text{mode}_i \in M$, do 2.1.
   2.1 calculate RDcost($\text{mode}_i$)
3. 2.2 if RDcost of all modes is larger than $RDTh_{\text{high}}$, go to Step 3. else goto Step 4.
4. for each mode $\text{mode}_i \in M'$ and $\text{NOT} \in M$, do 4.1.
   4.1 calculate RDcost($\text{mode}_i$)
5. 4.2 if RDcost of all modes is larger than $RDTh_{\text{low}}$, go to Step 5. else, go to Step 9.
6. for each mode $\text{mode}_i \in M'$ and $\text{NOT} \in M$ or $M$, do 6.1.
   6.1 calculate RDcost($\text{mode}_i$)
7. 6.2 if RDcost of all modes is larger than $RDTh_{\text{high}}$, go to Step 7. else Go to Step 9.
8. If all possible modes are already tested, go to Step 9. else go to Step 8.
9. Calculate RDcost of all untested modes.
10. Select mode with smallest RDcost as the prediction mode. end.

3. EXPERIMENTS

The algorithm was developed on an JM (Joint Mode) 16.0 H.264 encoder software. Configuration of the encoder and parameters of algorithm is given by Table 1.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_{\text{diff}}$</td>
<td>1.414</td>
</tr>
<tr>
<td>$T_{\text{sim}}$</td>
<td>0.525</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>0.01</td>
</tr>
<tr>
<td>$\beta$</td>
<td>-0.05</td>
</tr>
<tr>
<td>$N_1$</td>
<td>3</td>
</tr>
<tr>
<td>$N_2$</td>
<td>6</td>
</tr>
<tr>
<td>$N_3$</td>
<td>9</td>
</tr>
</tbody>
</table>

Testing sequences are categorized into two groups. The VGA group contains three VGA (640x480) sequences used for MVC. They are Ballroom, Exit, and Vassar. Only the first two views of these sequences are used to represent the left and right views. On the other hand, the HD group contains 3 HD (1920x1080) sequences. River is capture by two cameras set on a boat traveling down a river, containing different views of riverside and boats, the whole picture is moving. Space is a computer-graphic animation, containing high-detailed satellites, the Earth et al. Soccer is a Soccer game record with high movements taken by a 3D-camera. For each sequence, 250 frames are tested. The average performance of the algorithm of Qp ranged from 26 to 32 is given by Table 2. The speed-up is calculated by

$$\text{SpeedUp} = \frac{T - T_{\text{proposed}}}{T}, \quad (3)$$

Due to copyright restriction, we are not authorized to publish any part of these HD videos in any form.

- 340 -
where \( T \) is the coding time of JM16.0 and \( T_{\text{proposed}} \) is that of the proposed algorithm. As shown in Table 2, the algorithm reached 41.36\% speed-up on average for HD videos and 24.39\% speed-up for VGA videos. The algorithm performs better for HD than VGA videos because HD videos tend to use more large prediction modes compared to VGA videos since objects in HD video are naturally consists of more MBs. Therefore, MBs that contains object border, which causes increase of candidate modes, is less than VGA video.

<table>
<thead>
<tr>
<th>Sequence(HD)</th>
<th>River</th>
<th>Space</th>
<th>Soccer</th>
<th>Ave.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speed-up(%)</td>
<td>40.30</td>
<td>42.96</td>
<td>40.82</td>
<td>41.36</td>
</tr>
<tr>
<td>AP(\text{PSNR}(\text{dB}))</td>
<td>-0.04</td>
<td>-0.05</td>
<td>-0.04</td>
<td>-0.04</td>
</tr>
<tr>
<td>(\text{ABit-Rate}(%))</td>
<td>+0.87</td>
<td>+1.00</td>
<td>+0.89</td>
<td>0.92</td>
</tr>
<tr>
<td>Sequence(VGA)</td>
<td>Ballroom</td>
<td>Exit</td>
<td>Vassar</td>
<td>Ave.</td>
</tr>
<tr>
<td>Speed-up(%)</td>
<td>25.99</td>
<td>24.70</td>
<td>22.48</td>
<td>24.39</td>
</tr>
<tr>
<td>AP(\text{PSNR}(\text{dB}))</td>
<td>-0.05</td>
<td>-0.05</td>
<td>-0.04</td>
<td>-0.05</td>
</tr>
<tr>
<td>(\text{ABit-Rate}(%))</td>
<td>+1.01</td>
<td>+0.39</td>
<td>+0.58</td>
<td>0.66</td>
</tr>
</tbody>
</table>

In evaluating the reduction on computational complexity, we also calculated the number of candidate modes tested before the prediction mode is selected for each MB. Compared with the proposed algorithm, original H.264 carries an exhaustive mode selection, i.e. it tests all the 8 modes during mode decision.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Average Candidates Number</th>
<th>Save rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>HD</td>
<td>River</td>
<td>3.48</td>
</tr>
<tr>
<td></td>
<td>Space</td>
<td>3.27</td>
</tr>
<tr>
<td></td>
<td>Soccer</td>
<td>3.70</td>
</tr>
<tr>
<td>VG</td>
<td>Ballroom</td>
<td>4.88</td>
</tr>
<tr>
<td></td>
<td>Exit</td>
<td>4.94</td>
</tr>
<tr>
<td></td>
<td>Vassar</td>
<td>4.83</td>
</tr>
</tbody>
</table>

We also combine the proposed algorithm with a classical fast mode decision algorithm designed for monoscopic video in [6]. Experiments of the hybrid algorithm reveals that combined with the proposed algorithm, the monoscopic fast mode decision algorithm gains a higher speed-up of 70.3\%, compared with 32.1\% gained by [6] itself for HD videos, while the quality drops slightly by 0.009dB with bit-rate increased by 0.046\%. Due to the binocular suppression theory[7], quality loss of one view in stereo video should be discounted compared with that of monoscopic videos.

Finally, although the proposed algorithm is realized based on an JM16.0 software, it can work on MVC framework (or JMVM software) since their mode decision process is similar. But inter-view prediction modes of MVC cannot directly benefit from the algorithm without modification on schemes.

### 4. CONCLUSION AND FUTURE WORKS

This paper proposed an epipolar restriction based mode selection algorithm for stereoscopic video coding. Different from methods using disparity estimation, candidate modes are generated by sliding a window along MB row restricted by epipolar-line. Mode rectification and RDcost threshold were also used to improve efficiency and accuracy of mode selection. Experiments show that the proposed algorithm significantly reduces coding complex especially for HD videos. The algorithm can also be combined with monoscopic fast mode selection algorithms. Since the similarities between views are rather complex, in next step, we will focus on improving mode rectification by superior means other than motion vector to further reduce candidate modes.

### 5. ACKNOWLEDGEMENTS

This paper is partially supported by National Basic Research Program of China (973 Program) under contract 2009CB320905 and National Natural Science Foundation of China under contracts 60832004.

### 6. REFERENCES