SELECTIVE MOTION ESTIMATION STRATEGY BASED ON CONTENT CLASSIFICATION FOR HEVC SCREEN CONTENT CODING

Mengmeng Zhang [*], *Shuai Wang* [*] *and Bin Li* [§]

[*]North China University of Technology, Beijing, China
[§]Microsoft Research Asia, Beijing, China
Email: {muchmeng, dlz620301}@126.com; libin@microsoft.com

ABSTRACT

Motion estimation is one of the most time-consuming parts in video coding. This paper proposes a selective motion estimation strategy to reduce the complexity of the encoder for HEVC screen content coding. The proposed scheme is based on content classification. For each screen content inter frame, it is classified into screen content areas and camera-captured areas. Instead of using one single motion estimation search method for the whole frame, different methods are adaptively switched for different areas. Experimental results show that, compared with SCM-7.0 anchor, the proposed strategy achieves 15% and 13% encoding time saving on average merely with average 0.20% and 0.12% Y/G BD-rate loss for low delay and random access configurations, respectively.

Index Terms— Motion Estimation, Screen Content, HEVC, Classification

1. INTRODUCTION

The High Efficiency Video Coding (HEVC), developed by the Joint Collaborative Team on Video Coding (JCT-VC), is the newest video coding standard. It delivers the same subjective video quality at half the bit rate achieved by its previous video coding standard H.264/AVC [1].

Screen content videos is widely used in wireless displays, tablets, automotive displays, screen sharing etc. Unlike natural video captured by cameras, screen content usually contains cartoons, captures of typical computer screens, overlaid texts and news tickers, whose sharp edges and frequent transitions cannot be efficiently compressed by traditional video coding techniques. Thus, HEVC standard includes screen content as one of its extensions and investigates new techniques to improve the coding efficiency such as intra-block copy, palette mode and adaptive color transform [2].

Motion estimation is an essential process in video coding. It finds the best matched block position (Motion Vector) in the reference frames for every block in current frame. Motion estimation is the most computationally expensive process in the encoder because of variable sizes of block-matching and multiple reference frames [3]. Therefore, many fast motion estimation algorithms have been proposed in [4]-[7], including three step search, four step search, diamond search and hexagon search.

TZSearch method (TZS) is a fast motion estimation algorithm adopted in HM (HEVC Test Model) for its good performance [8]. While, it is poorly suited for predicting screen content for the complex characteristics. Hence, [9] proposes several modifications on TZS, which is adopted in SCM (HEVC Screen Content Test Model) and called TZSearchSelective method (TZSS). Although TZSS does perform well for screen content prediction, it brings a lot of computational complexity to the encoder, which is a problem need to be addressed. Thus, this paper proposes a selective motion estimation strategy based on content classification to reduce the complexity of the encoder for HEVC screen content coding. For a screen content inter frame, it is classified into screen content areas and camera-captured areas based on color information. Instead of using one single TZSS fast motion estimation method for the whole frame, TZS and TZSS are adaptively switched for different areas. Experimental results show that, compared with SCM-7.0 anchor, the proposed strategy achieves 15% and 13% encoding time saving on average merely with average 0.20% and 0.12% Y/G BD-rate loss for low delay and random access configurations, respectively.

The rest of this paper is organized as follows. A brief introduction to TZS and TZSS is given in Section 2. Then, the details of the proposed strategy are presented in Section 3. The experimental results are shown in Section 4. Finally, Section 5 concludes the whole paper.

2. TZSEARCH AND TZSEARCHSELECTIVE

TZS is a hybrid fast motion estimation algorithm combined with diamond or square search method and raster search method. It can be broadly concluded into 4 steps [10]: initial search center, diamond or square search, raster search and refinement. Firstly, the minimum of the median predictor, left predictor, up predictor, upper right predictor is selected as the start search center. Then, an 8-point diamond search or 8-point square search is used with different stride lengths. The motion vector with minimum SAD is taken as the

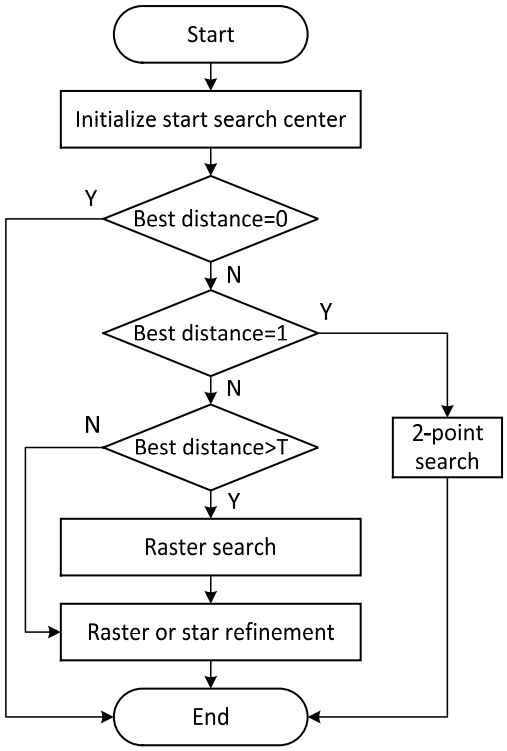


Fig. 1. Flowchart of TZS.

center search point for further steps. 2-point search, raster search and raster or star refinement will be finally used based on the stride of the motion vector with minimum SAD. The flowchart of TZS is shown in Figure 1.

TZS performs well for camera-captured sequences. However, screen content always contains sharp edges, high spatial frequencies and non-monotonic error surface, which cannot be efficiently predicted by the traditional TZS.

Therefore, in the 17th JCT-VC meeting, [9] proposes three modifications on TZS to sufficiently captured the spatial and temporal characteristics exhibited by screen content, which becomes TZSS. The modifications are mainly summarized as follows [9].

- Multi stage approximated error cost computation with early exit at each stage.
- Modified initial search method with a fixed diamond search traversing uniformly across the initial search space
- Modified early skip detection based on residual signals merge MV only.

Table 1 shows the performance of TZS versus TZSS for several sequences. It is observed that for sequences with a large proportion of moving texts or graphics, such as *FlyingGraphics*, *Desktop*, *Console*, *MissionControlChlip3*, TZS brings a lot of loss in terms of Y/G BD-rate (Bjøntegaard Delta Bit Rate of Component Y/G). On the contrary, for camera-captured content sequences like *EBURainFruits* and *Kimono1*, TZS hardly brings any loss on Y/G BD-rate. Meanwhile, TZS saves 20%~39% encoding time, which means the computational complexity of TZS is less than TZSS. Therefore, we can conclude that though TZS is not so efficient for screen content, it performs well enough in terms of both quality and complexity for camera-captured content.

Table 1. TZS vs. TZSS (32 frames, low delay, various QPs)

Sequences	Y/G BD-rate (%)		Encode time saving (%)	
	YUV444	RGB	YUV444	RGB
sc_flyingGraphics	2.92	2.84	34	30
sc_desktop	2.45	2.65	22	20
sc_console	6.44	4.39	30	24
MissionControlClip3	4.75	4.82	22	20
EBURainFruits	0.04	-0.01	25	21
Kimono1	-0.22	0.06	39	30

3. THE PROPOSED STRATEGY

Screen content is usually a mixture of screen content and camera-captured content. Hence, an adaptively selective motion estimation strategy is proposed based on content classification. For each screen content inter frame, it is firstly classified into screen content areas and camera-captured areas based on 8×8 (denoted as a unit) level. Then, CU (Coding Unit) whose units are all camera-captured will be regarded as a camera-captured CU and subsequently adopted TZS motion estimation because, as previously stated, it performs well enough for camera-captured content. Otherwise, CU will be regarded as screen content and remains TZSS motion estimation. The flowchart of the proposed strategy demonstrates in Figure 2.

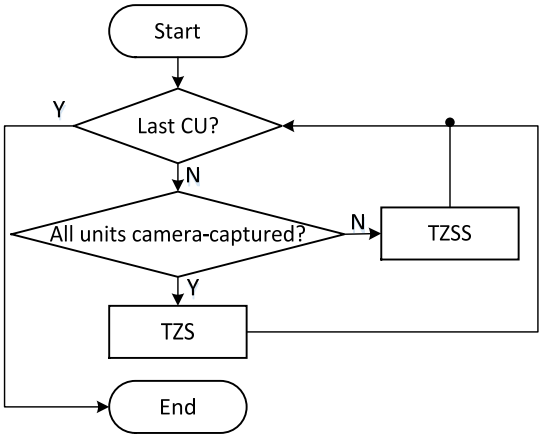


Fig. 2. Flowchart of the proposed strategy.

Figure 3 illustrates luma component of a screen content block and a camera-captured block extracted from a typical screen content frame. It is obvious that the characteristics of them are entirely different. The colors in a camera-captured content block are typically abundant and close to each other. While, the colors in a screen content block is usually dominated by several major colors which have significantly different values with one another. Based on above analyses, a classification scheme is designed based on color information. The details are as follows.

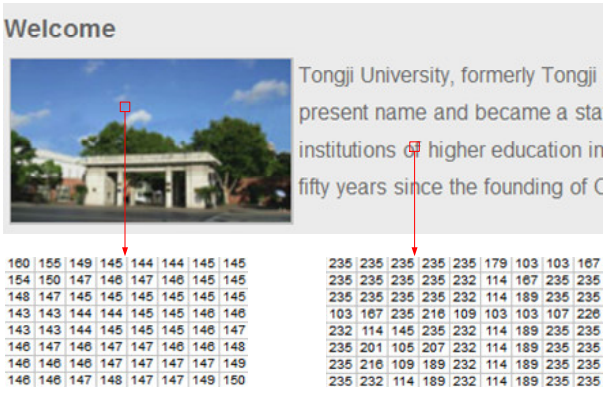


Fig. 3. Luma component of a screen content block and a camera-captured block.

Firstly, each screen content inter frame is divided into non-overlapped 8×8 units. Secondly, the total color number and the color range of luma component for each unit are calculated, denoted as N and R respectively. R is the range between the maximum color value and the minimum color value, defined as

$$R = V_{\max} - V_{\min}, \tag{1}$$

where $V_{\max}$ and $V_{\min}$ are the maximum color value and the minimum color value. Finally, parameter D will be derived as

$$D = \begin{cases} 0, & N = 1 \\ \left\lfloor \dfrac{R}{N-1} \right\rfloor, & Otherwise \end{cases}, \tag{2}$$

where $\lfloor \bullet \rfloor$ denotes round down operator. Since D is in terms of the ratio of R and N, camera-captured content unit tends to obtain a smaller D but screen content unit tends to be much bigger. Therefore, the whole frame will be recognized by D as follows.

- If $0 < D \le Threshold$ (set at 4), the unit will be regarded as camera-captured content.
- Otherwise, the unit will be regarded as screen content.

The classification scheme is so simple that it may lead to a misjudgment in some circumstances. For example, the edges in a camera-captured content may be misjudged as screen content due to the sharp contrast, as shown in Figure 4 (a), and screen content whose foreground and background are extremely close may be misjudged as camera-captured content, as presented in Figure 4 (b). Thus, a conditional merge process after classification is designed to refine the accuracy of classification. The merge performs from 16×16 level to 32×32 level till 64×64 level by the rules below.

- In current level, if the percentage of camera-captured content units or screen content units reaches to 75%, all the other units will be forced to be the category of majority;

After the merge, the classification based on color information is completed. The adaptively selective motion estimation strategy will be performed based on the classification subsequently. Figure 5 gives a flowchart of the classification scheme and Figure 6 illustrates a classification result.

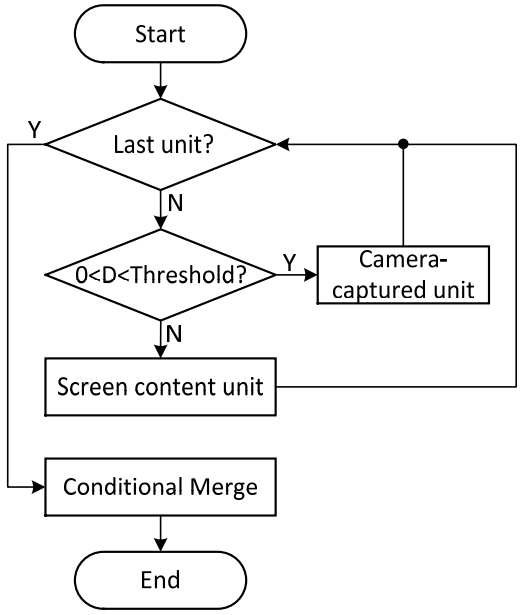
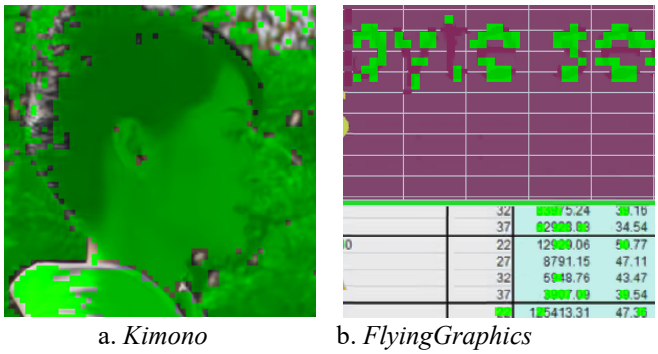


Fig. 5. Flowchart of the classification scheme.



a. *Kimono* b. *FlyingGraphics*

Fig. 4. Misjudgments (Green: camera-captured units).

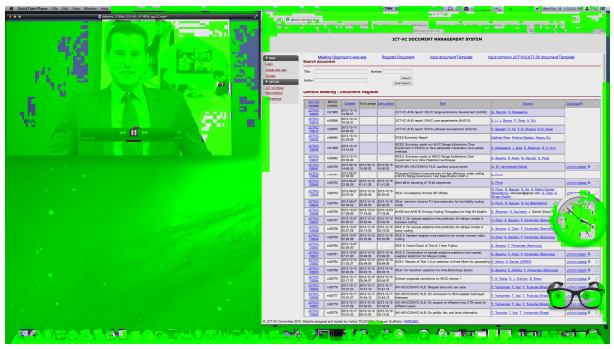


Fig. 6. A classification result.

4. EXPERIMENTAL RESULTS

The proposed strategy is evaluated on reference software SCM-7.0 under lossy screen content coding conditions recommended by JCT-VC. The configurations include low delay and random access. The hardware platform is Intel Core i7-4790 CPU @ 3.60GHz and 3.60GHz, 16.00GB RAM with Microsoft Windows 10 64-bit operating system. A total of 100 frames of each sequence in YCbCr444 (denoted as YUV444) or RGB format are tested with different QPs (22, 27, 32, 37). The experimental results are provided in terms of Y/G BD-rate and encoding time saving (TS) shown in Table 2 and Table 3. TS are defined as

$$TS = \frac{T_{anchor} - T_{proposed}}{T_{anchor}} \times 100\%. \quad (3)$$

Table 2. Experimental results of low delay

Sequences	Y/G BD-rate (%)		TS (%)	
	YUV444	RGB	YUV444	RGB
sc_flyingGraphics	0.01	0.21	0	-1
sc_desktop	0.01	0.01	-1	-1
sc_console	0.00	0.00	-1	0
sc_web_browsing	0.07	0.24	4	4
sc_map	0.51	0.34	31	26
sc_programming	0.28	0.06	15	12
sc_SlideShow	0.61	0.47	21	17
Basketball_Screen	0.23	0.42	28	22
MissionControlClip2	0.21	0.14	17	16
MissionControlClip3	0.27	0.28	13	13
sc_robot	0.43	0.41	32	29
EBURainFruits	0.05	0.03	16	13
Kimono1	0.00	-0.05	39	25
Average	0.20		15	

Table 3. Experimental results of random access

Sequences	Y/G BD-rate (%)		TS (%)	
	YUV444	RGB	YUV444	RGB
sc_flyingGraphics	0.24	0.06	-1	0
sc_desktop	0.00	-0.01	14	11
sc_console	0.00	0.00	3	4
sc_web_browsing	0.00	-0.03	0	-1
sc_map	0.22	0.20	14	10
sc_programming	0.12	0.09	18	17
sc_SlideShow	0.23	0.46	25	23
Basketball_Screen	0.14	0.15	16	17
MissionControlClip2	0.04	0.01	10	10
MissionControlClip3	0.07	0.07	9	9
sc_robot	0.34	0.38	26	21
EBURainFruits	0.02	0.07	14	12
Kimono1	0.08	0.20	32	21
Average	0.12		13	

Compared with SCM-7.0 anchor, the proposed scheme achieves -1%~39% and -1%~32% encoding time saving for low delay and random access configurations, respectively. For sequences with screen content throughout the screen, such as *Flying Graphics*, *Desktop*, *Console* and *Web Browsing*, the encoder speedup performance is poor, which because almost all units in frame are recognized as screen content and remain TZSS motion estimation strategy. The complexity brought by content classification scheme is exposed in this situation. However, for sequences mixed with camera-captured content and screen content, the encoder acceleration is significant. Because a number of units in frame are classified into camera-captured content and subsequently accelerate motion estimation process by applying TZS method.

Though the experimental results are various for different screen content sequences, on average, the proposed scheme achieves 15% and 13% encoding time saving with merely 0.20% and 0.12% Y/G BD-rate loss, which is negligible.

5. CONCLUSION

This paper proposes a selective motion estimation strategy to reduce the complexity of the encoder for HEVC screen content coding. The proposed strategy classifies each inter frame into screen content areas and camera-captured areas. Instead of using one single motion estimation search method for the whole frame, different methods are adaptively switched for different areas. As the experimental results shown, the proposed strategy is efficient for HEVC screen content coding time reduction with a negligible quality loss.

ACKNOWLEDGEMENTS

This work is supported by the Natural National Science Foundation of China (No.61370111, No.61103113), Beijing Nova Programme (Z14111000180000), and Beijing Youth Talent Project (CIT&TCD 201504001).

6. REFERENCES

[1] G. J. Sullivan, J. R. Ohm, W. J. Han and T. Wiegand, "Overview of the High Efficiency Video Coding (HEVC) Standard," in *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649-1668, Dec. 2012.

[2] J. Xu, R. Joshi and R. A. Cohen, "Overview of the Emerging HEVC Screen Content Coding Extension," in *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 1, pp. 50-62, Jan. 2016.

[3] N. Purnachand, L. N. Alves and A. Navarro, "Fast Motion Estimation Algorithm for HEVC," *2012 IEEE Second International Conference on Consumer Electronics - Berlin (ICCE-Berlin)*, Berlin, 2012, pp. 34-37.

[4] B. T. Koga, K. Linuma, A. Hirano , Y. Iijima, and T. Ishiguro, "Motion-compensated Interframe Coding for Video Conferencing,". *Proc. NTC81, G5.3.1--G5.3.5*.

[5] L. M. Po and W. C. Ma, "A novel four-step search algorithm for fast block motion estimation," in *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 3, pp. 313-317, Jun 1996.

[6] S. Zhu and K. K. Ma, "A new diamond search algorithm for fast block matching motion estimation," *1997 IEEE International Conference on Information, Communications and Signal Processing,* vol.1, pp.525-525.

[7] C. Zhu, X. Lin and L. P. Chau, "Hexagon-based search pattern for fast block motion estimation," in *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, no. 5, pp. 349-355, May 2002.

[8] HM Reference Software 3.4 [Online]. Available: https://hev c.hhi.fraun hofer.de/svn/svn_HEVCSoftware

[9] R. Krishna, J. Sole, L. Zhang and M. Karczewicz. "AhG8: On fast inter search method for screen content coding, " *JCTVC-Q0147,* Valencia, Apr. 2014.

[10] N. Purnachand, L. N. Alves and A. Navarro, "Improvements to TZ search motion estimation algorithm for multiview video coding," *2012 19th International Conference on Systems, Signals and Image Processing (IWSSIP)*, Vienna, 2012, pp. 388-391.