Low Complexity Video Compression Using Moving Edge Detection Based on DCT Coefficients

Chanyul Kim and Noel E. O'Connor

CLARITY: Centre for Sensor Web Technologies, Dublin City University, Glasnevin, Dublin, Ireland
Chanyul.kim,oconnorn@eeng.dcu.ie

Abstract. In this paper, we propose a new low complexity video compression method based on detecting blocks containing moving edges using only DCT coefficients. The detection, whilst being very efficient, also allows efficient motion estimation by constraining the search process to moving macro-blocks only. The encoders PSNR is degraded by 2dB compared to H.264/AVC inter for such scenarios, whilst requiring only 5% of the execution time. The computational complexity of our approach is comparable to that of the DISCOVER codec which is the state of the art low complexity distributed video coding. The proposed method finds blocks with moving edge blocks and processes only selected blocks. The approach is particularly suited to surveillance type scenarios with a static camera.

Low complexity video compression, Moving edge, DCT

1 Introduction

New digital video applications have recently emerged such as private internet broadcasting and wireless multimedia sensor networks. These kinds of applications fundamentally need a low power and low complexity encoder in order to operate on power limited devices such as wireless video phones, personal digital assistants (PDA) and sensor platforms. Of course, apart from coding computational complexity, the coding gain of a compression algorithm plays a vital role in determining its practical usefulness. Therefore, much research has targeted *reasonable* coding gains whilst keeping complexity low. However, the challenge comes in finding the optimum trade-off as the dual requirements of coding gains and low complexity are not comfortable bed-fellows.

H.264/AVC has been standardized with a target for coding gains regardless of complexity [1]. Although the power of digital devices has increased steadily, it is still hard to realize real-time operation on power limited devices. Recently, much research for achieving low complexity with reasonable coding gains has been performed using H.264/AVC [2–4]. In hybrid coding, such as the standardization efforts of ITU-T and MPEG, two main coding tools are used to obtain

reasonable coding gains: transform and prediction. The most time consuming functions of H.264/AVC are motion estimation, compensation and prediction. Therefore low complexity algorithms based on H.264/AVC typically attempt to reduce the complexity of these three functions. In [2, 3], the authors' approaches focus on reducing the complexity needed to find motion vectors and make mode decisions as required by the variable block size feature of H.264/AVC. Many fast algorithms are reviewed in [4]. As an example of other approaches investigated to obtain low complexity, Hiratsuka *et al.* proposed adaptive tree based video coding where their approach has similarity to 3D DCT based coding [5]. However, its drawback is low coding gains even though low complexity is achieved compared to standard codecs. Perhaps the most popular approach to obtain coding gains is region of interest (ROI) based coding. However this approach needs an additional step of object or region segmentation that requires significant computational overhead. In the case of low motion video such as surveillance, low complex algorithms are possible such as that proposed in [6]. Sriram S *et al.* proposed foveation based low complexity video coding [7]. Their approach uses human visual system (HVS) modeling to obtain coding gains whereby a DCT based foveation filter is used to reduce complexity for detecting foveation regions.

For implementing low complexity encoders, a coding paradigm named distributed video coded (DVC) based on Wyner-Ziv [8] and Slepian-Wolf [9] information theory has recently emerged. The basic concept is that the complexity shifts from the encoder to the decoder by using error correction codes (ECC). The distributed probability between the current frame and the previous frame is defined similar to a channel in communication theory and only parity bits of ECC are sent to the decoder. The decoder decodes the current frame with parity bits and a reference frame sent as side information. In this approach, the encoder only performs the matrix multiplications between the parity matrix and data bits, whilst the decoder requires more complexity to decode the ECC. Puri and Ramchandran proposed the PRISM codec for multimedia transmissions on wireless networks using syndromes [10]. Aaron and Girod proposed a video coding framework using intraframe encoding as side information with turbo codes named Stanford DVC and transform based coding [11, 12]. Recently, X.Artigas *et al.* proposed the DISCOVER codec including more advanced tools to obtain coding gains such as a rate-distortion module and virtual channel modeling. Despite their efforts, however, coding quality still suffers degradation compared to H.264/AVC [13].

We propose very low complexity video compression in a similar vein to ROI based coding. The proposed method does not perform object or region segmentation which can introduce computational complexity but rather detects moving edges using only DCT coefficients. The approach can be adapted from low motion to high motion sequences. The motivation for the research is presented in the next section. The proposed approach is explained in Section 3. The performance of the proposed method is compared to H.264/AVC and the DISCOVER codec in Section 4. Finally, concluding remarks are made in Section 5.

2 Motivation

The computational complexity of video coding comes from the motion estimation, compensation and prediction block. As mentioned in Section 1, many researchers focus on reducing motion estimation time to obtain low complexity in hybrid coding. A motion search algorithm has the role of finding matched blocks without any prior knowledge of the image. If prior knowledge of image contents was available, this would enable us to perform better motion estimation, but focusing on the regions that really need it, thereby satisfying the requirements of high coding gains and low complexity at the same time. A ROI-based encoder effectively attempts to predict the content of the scene to achieve coding gains. However, it usually requires an additional segmentation block and pre-/post-processing of the video which means significant computational cost [14]. We propose a ROI-based scheme that detects only moving edges and not complete moving objects. This facilitates low complexity encoding, particularly as this detection can be done using the DCT directly. Static edges give a hint for the boundary of object and moving edges give information of on which area has motion. Using this knowledge of moving regions then allows us to perform low complexity motion estimation. Moving edges are detected through classification of edges using DCT coefficients in a 4×4 sub block. An algorithm for reducing falsely detected moving edges is also suggested in this paper. Block based moving edge classification is more adaptable for video compression than a frame based approach since a frame based approach needs to find the block position and edge types which requires additional computational complexity.

3 Proposed Approach

The proposed video compression approach can use a modified standard H.264/AVC encoder to generate the necessary information corresponding to the edge direction number (ED) and the standard deviation of the AC DCT coefficients (SD) as well as to generate the required intra coded frame. However, in this paper, our previously proposed intra coding method is used as the intra encoder [15]. The encoder proposed in this paper consists of the moving edge detection and video compression functional blocks as shown in Fig. 1. The moving edge detection function block detects not only moving edges but also removes false moving edges via a reduction function block (RFME). Both processes are performed using 4×4 DCT coefficients.

After moving edge detection, video compression in our method is performed with a similar approach to a standard H.264/AVC codec. However, in order to obtain low complexity, our video compression method does not use variable block size (only 4×4 blocks are used in this paper), $\frac{1}{4}$ pel motion estimation/compensation or rate distortion optimization as used in H.264/AVC. Of course, this causes degradation of video quality, however, since our ROIs correspond to only moving edge blocks the slight degradation of video quality is offset by the bit savings obtained. Therefore, the overall rate-distortion performance

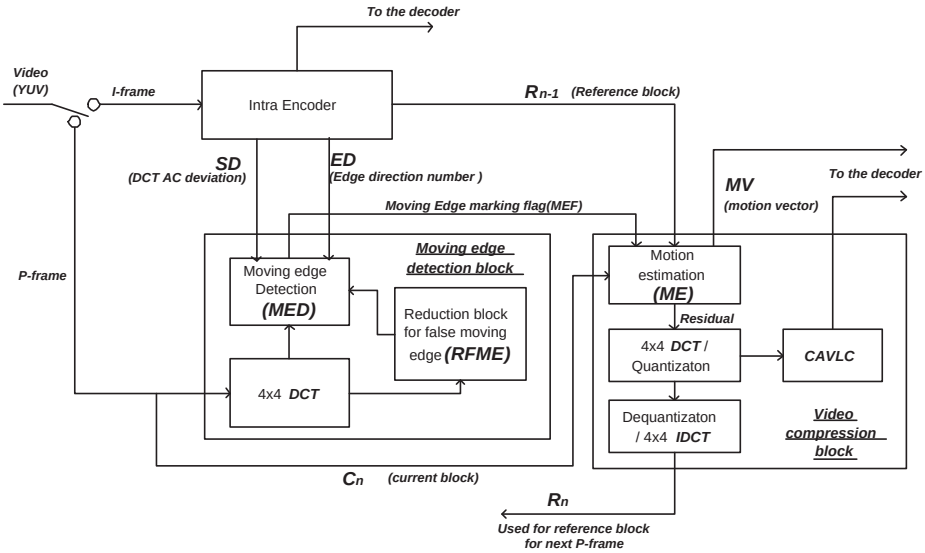

Fig. 1. The proposed video compression functional blocks, bold italics indicate abbreviations used in the paper.

is not severely degraded compared to H.264/AVC (see Fig. 6 and the discussion in the results section). The detailed functionalities of the proposed method are explained in the following.

3.1 Moving Edge Detection

Moving edge detection If the current 4×4 block has a specific edge feature (such as horizontal, vertical, diagonal or texture as shown in Fig. 2(a)), the standard deviation (SD) of AC coefficients as calculated in equation (1) indicates whether edge or non-edge information is present [16]. As an object moves in the scene it covers and uncovers background around its borders. Also, the object may deform, changing its shape. Both of these phenomena result in a change of the edge characteristics within blocks on the object's boundary. This can be used to detect moving edge blocks from frame to frame. There are 3 possibilities:

- An edge block changes to a non-edge block;
- A non-edge block changes to an edge block;
- The edge direction within the block changes to another one of directions depicted in Fig. 2(b).

$$\sigma_{ac} = \frac{\sum_{i=1}^{15} C_i^2}{15} - (\frac{\sum_{i=1}^{15} C_i}{15})^2 \tag{1}$$

Where σ_{ac} is a standard deviation of all AC coefficients in a 4x4 block, C_i is the i^{th} DCT coefficient as depicted in Fig. 2(a). If the current block is a non-edge

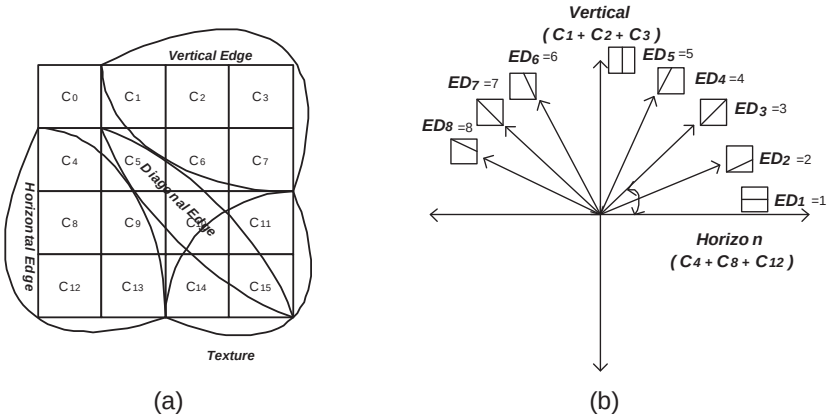

Fig. 2. (a) Edges related to DCT coefficients; (b) Edge direction classification as numbers from 1 to 8.

block, we set the edge strength measure to a pre-defined threshold value(TH). Then the edge strength value of a non-edge block at the same spatial position in the previous frame is examined. If the difference of edge strength value is zero between two blocks, this means that there is no change between blocks, so this block is not a candidate for a moving edge. If the difference of edge strength value is more than zero, there is a transition from an edge to a non-edge or a non-edge to an edge. We consider both cases as moving edges. Algorithm 1 shows a simple procedure for obtaining a moving edge. Whilst straightforward to implement, this algorithm means that moving edges are easily affected by the threshold value so that many false edges are introduced as shown in Fig. 3(b). Thus, we need to reduce these false moving edges. This algorithm, based on edge direction modeling, is explained in the next section.

Algorithm 1 Moving edge detection

if $\sigma_{sd} < TH$ **then**
  set non-edge, SD = TH, save SD value to SD_n
else
  edge, save SD value to SD_n
end if
if $current\ frame \neq intra\ frame$ **then**
  if $\mid SD_n - SD_{n-1} \mid > 0$ **then**
    moving edge
  else
    static edge
  end if
end if

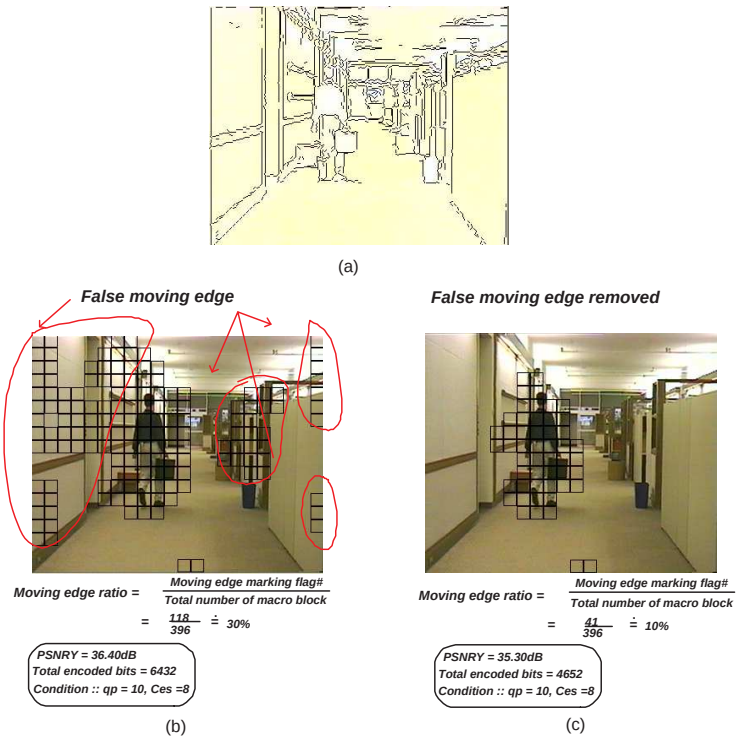

Fig. 3. (a) An edge image for 50^{th} frame of Hall Monitor, qp =10, $C_{es} = 8$: (a) an edge frame considering edge directions; (b) After moving edge detection, with false moving edges indicated; (c) After false moving edge reduction – moving macro blocks decrease by 20%, PSNR = -1.1dB, required bits = -28% compared to (a).

Reducing false moving edges A false moving edge arises from image noise and an incorrect threshold value for deciding moving edges with the SD of DCT coefficients. The image noise could be eliminated by a pre-processing filter; however, this inevitably generates more computational complexity. We cannot simply increase the threshold used, since real moving edges are also removed when a strong threshold value is applied. Therefore we should select real moving edges out of the total suggested moving edges whilst keeping a reasonable threshold value by observing critical variation of the edge direction (more than 45°) between the current block and the reference block. Our reduction block for false moving edges (RFME) can select real moving edges by defining edge directions and measuring the edge direction difference. Edge directions can be obtained as following:

$$\theta = \arctan\langle \frac{\sum_{i=1}^{3} C_i}{\sum_{i=1}^{3} C_{4i}} \rangle \tag{2}$$

Where C_i is the i^{th} DCT coefficient of a 4×4 sub-block. We consider edge directions , $0°, \pm 45°, 90°, \pm 26.5°, \pm 63.4°$ shown in Fig. 2(b), which are used in H.264/AVC intra prediction. The obtained edge map by classifying edge directions is depicted in Fig. 3(a). We allocated numbers from one to eight each of these directions by equation (2). The difference of an edge direction number (ED) between the current block and the reference block is defined as a distance (D) as in equation (3).

$$D = |[(M + shift)\%8)] - 4| \quad \text{if}[(N + shift)\%8] == 4 \tag{3}$$

Where $[n]$ is a near integer number n, % is a remainder, M and N are EDs of the reference block and the current block respectively. When D is greater than a pre-defined threshold value, usually set to two, we consider this block as a moving block since the edge direction has changed more than 45°. The maximum difference value of ED occurs between ED_1 and ED_8 as shown in Fig. 2(b). However, the difference edge direction between ED_1 and ED_8 is only 26.5° even though the difference between them is generated as the maximum value (7). Therefore a compensation routine should be applied. If the ED of the current block and the reference block are N, M respectively, the edge difference is not calculated as $|N-M|$ but as in equation (3). First, the shift value is found as a pre-condition of equation (3). The ED of the current block shifts to the centre number (which is 4 since eight numbers are allocated for all directions), therefore the total difference between them is calculated as a shifted value of ED of the current block. For instance, let the current ED is one (horizontal direction) and a reference ED is eight($-26.5°$), the shift value is three and the distance (D) is not seven but one. This case is then not a candidate for a moving edge by setting the threshold to two(45°). Fig. 3(b)(c) shows the result of a moving edge detection and false moving edge reduction blocks. Although PSNR is ultimately degraded 1.1dB compared to considering all moving edges, the required encoder bits are also reduced by 28%. Also the number of macro blocks required to process is decreased dramatically, significantly reducing the processing time. In the example shown in Fig. 3(c), Only 41 macro blocks are considered for further processing out of all 118 macro blocks if false moving edges are not removed.

3.2 Video Compression Block

The video compression block consists of four blocks: motion estimation and compensation (ME), DCT/Quantization, IDCT/Dequantization and content adaptive variable length coding (CAVLC). The ME block is performed only for macro blocks which have the moving edge flag (MEF) from the MED block as shown in Fig. 1. The flag is set for each macroblock by considering neighboring blocks near moving edges as shown in Fig. 4. For example, if a 4×4 moving block is located in the 6^{th} sub-block of a macro block in the raster scan order, we select the search area so that the moving edge sub-block is located in the centre. The resulting overlapping areas with neighboring macro blocks are depicted in grey in Fig. 4. The moving edge flag (MEF) is also set for these macro blocks.

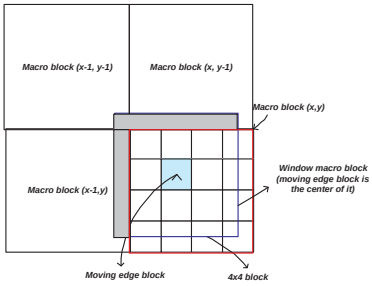

Fig. 4. Additional needed macro blocks to maximize the motion search range of a moving edge 4×4 sub block

After deciding the macro blocks that require processing, ME is then performed with full search with a 16×16 search range, no variable size blocks (i.e. one motion vector per a 4×4 sub block), no sub-pel compensation and no motion prediction to guarantee low complexity. The IDCT/Dequantization is performed for reconstructing the current image. After ME, content adaptive variable length coding (CAVLC) is performed in the same way as in H.264/AVC. The in-loop-filter (de-blocking filter) is not used in the proposed method.

4 Implementation and Experimental Results

This section verifies the performance of the proposed low complex video compression framework. The performance is compared to H.264/AVC reference software KTA 1.6 based on JM 11.0 with baseline profile released in Jan. 2008[1], H.263+ based on TMN code[2] and the DISCOVER codec [13]. The KTA reference has more advanced features than H.264/AVC such as an adaptive quantization matrix selection, a $\frac{1}{8}$ pel motion compensated prediction, an adaptive prediction error etc. H.263+ is designed as a low bitrate compression by adding several annexes which can substantially improve coding efficiency. The DISCOVER codec is the state of the art in distributed video coding, focusing on low complexity encoder[3]. Only luminance coefficients are considered in this paper, so U and V coefficients and the de-blocking filter are disabled in the reference software.

Hall monitor with QCIF and CIF resolution, Camera and Foreman sequences at QCIF resolution, down-sampled to 15Hz are selected as test sequences. The Hall Monitor and Camera sequence have no global motion and they are representative of surveillance applications. Foreman has significant global motion, leading PSNR saturation explained in Chapter 4.1. All tests are performed with GOP size 8 and one I-frame per every seven P-frames (IPPPPPPPI). Our low complex video compression framework is written in ANSI C++ and a Intel Integrated Performance Primitive 5.3 library. All tests are performed on an Intel

[1] The source code is available at http://iphome.hhi.de/suehring/tml/download/KTA
[2] The source code is available at http://whkong.myrice.com/download/src/vcomp
[3] The executable DISCOVER codec is available at http://www.discoverdvc.org/

Core(TM)2 Duo 3.6GHz with 2GB RAM using Window XP version 2002 with service pack 2.

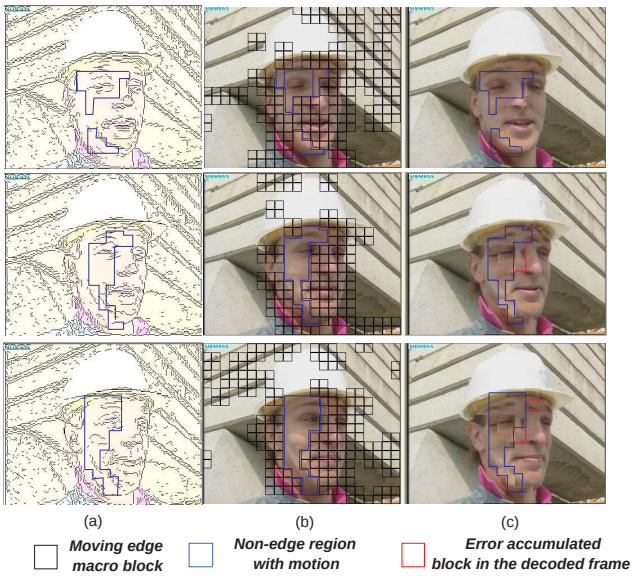

Fig. 5. (a) Edge images (b) Moving edge macro blocks: regions classified as non-edges but that have motion are not detected properly as shown by the area within the blue line (c) Decoded frames: accumulated errors are displayed within the red line

4.1 Rate-Distortion performance

Fig. 6 depicts the compression results compared to the H.264/AVC, H.263+ and the DISCOVER codecs. The proposed method shows good compression, albeit degraded by 2dB compared to H.264/AVC. However, this constitutes an increase of almost 4dB and 1.5dB compared to the DISCOVER codec and H.263+ for the Hall Monitor and Camera sequence as shown in Fig. 6(a),(b)&(c). However, PSNR saturation occurred for the Foreman sequence as shown in Fig. 6(d). This problem arises from detecting non-edge blocks that actually exhibit motion. For example, when the camera is constantly moving, this generates moving edges near the object boundary. If there are non-edge regions with motion inside the object, this is not detected by our approach. Fig. 5 shows the block artifacts in the decoded frame. Regions classified as non-edges but that have motion are not detected properly as shown by the area within the blue line. This generates block artifacts due to the difference in DC values of the current and reference sub blocks. This error is accumulated (red lines) until an intra frame is encountered. Therefore, our application is particularly suited to surveillance-type applications without global motion or non-edge regions with motion.

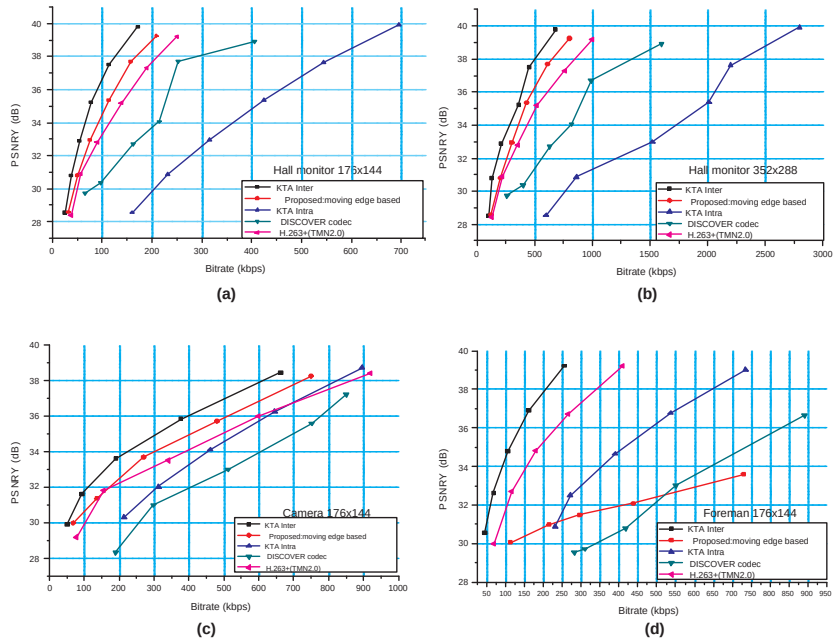

Fig. 6. Rate-Distortion – (a)(c) QCIF (b) CIF resolution: These sequences have no-edge regions without motion; (d) This sequence has no-edge regions with motion. For (a)(b)(c) the PSNR of the proposed method is degraded by maximum 2dB compared to H.264, for (d) PSNR saturation has occurred

4.2 Execution Time

Fig. 7(a) shows execution time for 30 frames at hall monitor and foreman sequence according to coding methods. Our approach is almost the same or better than DISCOVER codec well known as a very low complexity encoder. Additional DCT is introduced to obtain moving edges, but it does not generate severe computational complexity due to its integer transform that occupies less than 3% of whole complexity in H.264 [17]. Moving edges give which macro blocks should be treated as motion estimation block. Typically, the moving edges are less than one of tenth of the whole macro blocks, sometimes the number of moving edges goes to zero (no needed any processing) as shown in Fig. 7(b). The execution time is less than 5% compared to H.264/AVC intra coding since our approach does not use advanced coding tools introduced in H.264 and whole macro blocks.

5 Conclusion and Future Considerations

In this paper, a low complexity video compression algorithm based on detecting moving edges in the compressed domain is suggested. In terms of coding gains, PSNR is degraded by 2dB and enhanced by 1.5dB and 4dB compared to

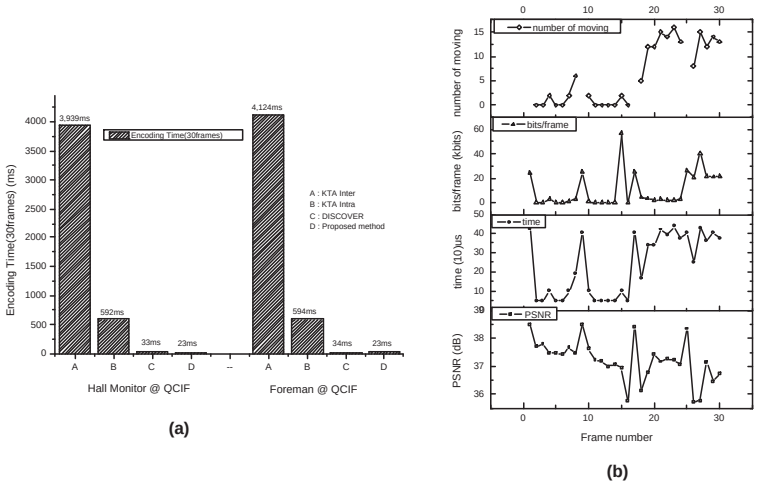

Fig. 7. (a) Average complexity comparison between encoder (30frames) (b) PSNR, encoding time, the number of encoded bits, the number of moving edge macro block verse frame number (hall monitor with QCIF)

H.264/AVC, H263+ and the DISCOVER codec respectively. In terms of computational complexity, it shows almost the same complexity as the DISCOVER codec which is the state of the art in low complexity distributed video coding. However, error accumulation occurs in non-edge areas with motion. Clearly, in the future we need to consider not only moving edges but also the entire moving object in order to overcome this drawback. We would also like to investigate integrating our approach into a DVC framework.

Acknowledgements

Authors would like to acknowledge the support of Samsung Electronics and Science Foundation Ireland under grant 07/CE/I1147.

References

1. 14496-10, H.: Advanced video coding. Technical report, ITU-T (2003)
2. Tourapis, H.Y.C., Tourapis, A.M.: Fast motion estimation within the h.264 codec. In: Multimedia and Expo, 2003. ICME '03. Proceedings. 2003 International Conference on. Volume 3. (July 2003) 517–20
3. gon Kim, D., jung Yoo, C., bae Chang, O., mi Kim, E., Choi, J.R.: Improved fast mode decision algorithm for variable macro block motion compensation in h.264. In: Information Technology Convergence, 2007. ISITC 2007. International Symposium on, Joenju (November 2007) 184–187

4. Wong, H.M., Au, O.C., Chang, A., Yip, S.K., Ho, C.W.: Fast mode decision and motion estimation for h.264 (FMDME). In: Circuits and Systems, 2006. ISCAS 2006. Proceedings. 2006 IEEE International Symposium on. (May 2006)
5. Hiratsuka, S., Goto, S., Baba, T., Ikenaga, T.: Video coding algorithm based on adaptive tree for low electricity consumption. In: Circuits and Systems, 2004. Proceedings. The 2004 IEEE Asia-Pacific Conference on. Volume 1. (December 2004) 5–8
6. Magli, E., Mancin, M., Merello, L.: Low-complexity video compression for wireless sensor networks. In: Multimedia and Expo, 2003. ICME '03. Proceedings. 2003 International Conference on. Volume 3. (July 2003) 585–8
7. Sriram Sankaran, R.A., Khokhar, A.A.: Adaptive multifoveation for low-complexity video compression with a stationary camera perspective. In: Proc. SPIE, Vol. 5685, 1007. (2005)
8. Wyner, A., Ziv, J.: The rate-distortion function for source coding with side information at the decoder. In: IEEE Transactions on Information Theory. Volume 22. (Jan 1976) 1–11
9. Slepian, D., Wolf, J.: Noiseless coding of correlated information sources. In: IEEE Transactions on Information Theory. Volume 19. (July 1973) 471–480
10. Puri, R., Ramchandran, K.: Prism: A new robust video coding architecture based on distributed compression principles. In: Proc. Allerton Conf. (Oct 2002)
11. Aaron, A., Rane, S., Zhang, R., Girod, B.: Wyner-ziv coding for video: applications to compression and error resilience. In: Data Compression Conference, 2003. Proceedings. DCC 2003. (March 2003) 93–102
12. Aaron, A., Rane, S., Setton, E., Girod, B.: Transform-domain wyner-ziv codec for video. In: Proceedings of SPIE Visual Communications and Image Processing Conference, San Jose, USA (Jan 2004)
13. Artigas, X., Ascenso, J., Dalai, M., Klomp, S., Kubasov, D., Ouaret, M.: The discover codec: Architecture, techniques and evaluation. In: Picture Coding Symposium, Lisbon, Portugal (2007)
14. Chi, M.C., Chen, M.J., Yeh, C.H., Jhu, J.A.: Region-of-interest video coding based on rate and distortion variations for h.263+. Image Commun. **23**(2) (2008) 127–142
15. Kim, C., Noel.E.O'Connor: Low complexity intra video coding using transform domain prediction. In: International Conference on Visualization, Imaging, and Image Processing, Palma de Mallorca, Spain (2008)
16. Labit, C., Marescq, J.: Temporal adaptive vector quantization for image sequence coding. In: SPIE, Advances in Image Processing. Volume 804., Hague, Netherlands (Apr 1989)
17. Kim, C., Noel.E.O'Connor: Reducing complexity and memory accesses in motion compensation interpolation in video codecs. In: China-Ireland International Conference on Information and Communications. (2007)