FACILITATING MOTION-BASED VISION APPLICATIONS BY COMBINED VIDEO ANALYSIS AND CODING

R.M.T.P. Rajakaruna, W.A.C. Fernando and J. Calic

I-Lab, Centre for Communication Systems Research,
University of Surrey, Guildford, United Kingdom
Email:{R.Rajakaruna, W.Fernando, J.Calic}@surrey.ac.uk

ABSTRACT

In order to jointly optimise the quality of video coding on one hand and video analysis on the other, this paper proposes a novel approach to enhance the reusable information content in compressed video domain. By introducing a hierarchical content driven motion estimation mechanism at the encoder, complemented by a statistical prediction of region-of-interest, this approach reduces the complexity and yet increases robustness of the compressed domain vision analysis applications. Taking the object tracking application as an example, we demonstrate that the motion vectors generated by the proposed method can be directly used to extract object information, achieving tracking performance comparable with a pixel domain approach. In addition, we show that the incurred rate distortion (RD) overheads and the effect on encoder complexity are minimal, especially when compared to the reduction of processing required for video analysis targeting a wide spectrum of computer vision applications.

Index Terms— Motion analysis, Feature Extraction, Compressed domain vision applications, video coding

1. INTRODUCTION

Motion based vision applications such as object segmentation and tracking provide prior information to variety of computer vision tasks ranging from activity recognition, automated surveillance to object monitoring and controlling systems. Nowadays, the majority of video files are in compressed form due to resource limitations in storage and transmission. As a result, video decoding tends to be the first step in implementation of vision applications. Therefore, processing the video in compressed domain [1]-[4], has the advantage of functioning at a fraction of computational cost, when compared to the pixel-domain techniques.

On the other hand, the processing carried out at the time of encoding focuses mainly on compression efficiency. For instance, selection criterion for block-based motion vectors is commonly formulated to achieve optimum rate distortion (RD) performance. As a result, motion vectors are not always related to actual motion. Therefore, the information available in the compressed data stream is ill-suited for vision applications. The present paradigm to address this is to introduce another dimension within the analysis as a confidence measure [2]-[4] to overcome this problem.

As an alternative, reusability of encoder processing can be increased by considering the requirements of vision applications at the time of video encoding. The idea of altering encoding parameters in order to support vision applications was used in [5] for scene change identification, via an additional frame with enforced H.264 block size selection. In [6] global motion estimation information was used to correct noisy vector fields to improve compressed domain video indexing. However, these approaches are only applicable to specific tasks, and in [5] the application should be aware of the changes made at the decoder.

In this paper, a motion estimation mechanism, based on frame content and motion statistics, is proposed. The Region-of-Interest (ROI), in terms of vision applications, for each frame, is predicted using the motion field statistics of the frame. The resulting motion field for the ROI are used in encoding the video, which enhance the reliability of motion information in the compressed data stream, while still adhering to the original encoding standard. As a result, any motion-based vision application, such as segmentation, tracking and indexing would benefit by the enhanced motion field.

The rest of the paper is organized as follows. Section 2 discusses the problem background, while the proposed method is outlined in section 3. Simulations results and the conclusion are given in sections 4 and 5 respectively.

2. BACKGROUND

Information directly available from the encoded video stream, used by vision applications can be twofold; the motion information and the transform coefficients of intra coded blocks or of prediction residuals. The motion estimation of all existing video coding standards is based on block matching approaches, where the motion vector is represented by a 2D translational model. Each frame is divided into blocks of size *n x m*, and matched with all candidate positions within the search region. The Sum of Absolute Difference (SAD) and Mean Square Error (MSE) are two commonly used criteria in selecting the best position.

For example, in the JVT implementation of H.264/AVC [7] the best predictor for the displacement of a block is found by minimizing the cost function [8]:

$$J(m,\lambda) = SAD(s,c) + \lambda \cdot R(m-p) \qquad (1)$$

In equation (1) m, p represent the motion vector and the prediction for the motion vector respectively, while λ is the Lagrange multiplier. The rate term *R (m-p)* represents the amount of information necessary to encode motion. In the *SAD* term, *s, c* represents the original video signal and the coded video signal respectively. Since the motion vectors are selected such that the rate distortion (RD) performance is optimized they do not necessarily represent actual motion of the objects in the scene, which leads to heavily noisy motion vectors.

Several algorithms are proposed in literature to process H.264/AVC motion vectors to enhance their reliability, prior to using them in compressed domain applications. In addition to motion analysis, a dissimilarity energy minimization approach was used in [2] based on texture and form, extracted by partly decoding the video. A model of global camera motion derived by predefined scene background detail, and block size variation in H.264/AVC was used as a confidence measure in [3] and [4] respectively. Instead, this proposal suggest to make use of encoder processing as a means to reduce the processing at receiver end applications by enforcing compressed features to carry useful information.

3. PROPOSED SOLUTION

The motion in a video, resulting from both object movements and camera motion, relates directly to pixel movement across frames except for occlusions, re-appearances and illumination changes. It can be assumed that the estimation of pixel trajectories (i.e. optical flow) can be taken as an accurate estimation of the local motion. However, calculating optical flow for all pixels in a frame can be computationally expensive. Since the accuracy of motion vectors matters only in case of those corresponding to foreground objects, for vast majority of motion based applications it makes sense to limit the excessive calculations only to the ROI.

In order to extract the ROI, the proposed method analyse each frame to identify overall motion activity within the frame. This is done prior to encoding it as opposed to the common approach of processing one macroblock at a time. The flow vectors are calculated iteratively using an adaptation of Pyramidal Lucas Kande [9] optical flow algorithm. A magnitude based gradual rejection mechanism is used to extract the ROI using derived motion vectors at each pyramid level prior to moving to the next level. Finally, block motion vectors, needed for the video compression task, are calculated from the flow field. Resulting content based motion vectors are used to encode blocks representing the ROI, instead of conventional motion estimation. The blocks outside the ROI will remain untouched to minimize the impact on coding efficiency.

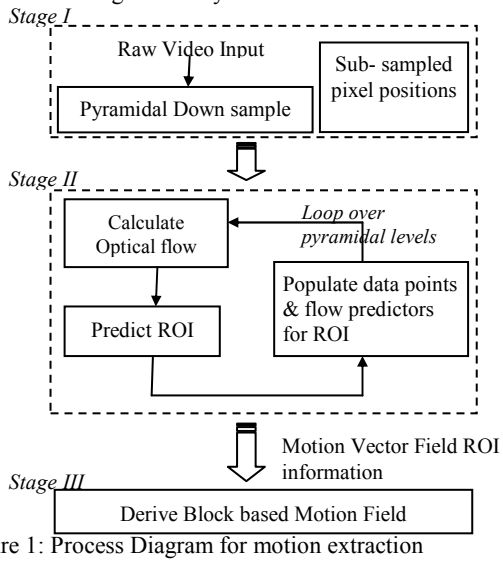

Figure 1: Process Diagram for motion extraction

The motion extraction algorithm consists of three stages as illustrated in Figure 1. Each stage is discussed in detail below.

Stage I: For two consecutive frame inputs, two pyramidal arrays of down-sampled frames are calculated, with number of pyramid levels L_M. If F^0 denote the original frame and the next pyramidal level F^1 is calculated by spatial filtering and sub sampling F^0, and F^2 from F^1 and so on. The input feature points to the optical flow algorithm are derived by taking all pixel positions of the frame sub sampled such that every N^{th} position is taken in *x,y* directions, where $N = 2^{L_M}$.

Stage II: The ROI and corresponding motion field are derived at this stage, initiating at the lowest level down-sampled frame, F^{L_M} . From the Lucas Kanade (LK) optical flow algorithm[10], if $u = (x,y)$ represent the position of a given pixel at time t and I_x, I_y is the intensity of the pixel at time t, the velocity V_x , V_y for the motion of the point across the plane of the frame compared against the next frame is given by,

$$\begin{pmatrix} V_x \\ V_y \end{pmatrix} = \begin{pmatrix} \sum I_{x_i}^2 & \sum I_{x_i} I_{y_i} \\ \sum I_{x_i} I_{y_i} & \sum I_{y_i}^2 \end{pmatrix}^{-1} \begin{pmatrix} \sum I_{x_i} I_{t_i} \\ \sum I_{y_i} I_{t_i} \end{pmatrix} \qquad (2)$$

Here, $\boldsymbol{I_t}$ denotes the temporal variation in pixel intensity compared against that of corresponding pixel in the next frame. In the pyramidal implementation of LK [9], pixel location $\boldsymbol{u}$ mapped onto to pyramid level L is defined by, $\boldsymbol{u/2^L}$, and the flow vectors calculated at one level are propagated into the next level as an initial guess of flow vector value. This redefines $\boldsymbol{I_t}$ as temporal variation in pixel intensity compared against $\boldsymbol{u}$ displaced by the flow vector calculated at the previous level scaled by a factor of 2, termed *flow predictor*.

In the proposed method, flow calculations at the lowest level were initiated using only a sub-sampled set of pixel positions, instead of using the entire array of positions. At each level, the magnitude of the calculated flow vector, $V^L_{x,y}$ is compared against a threshold, θ_L to predict the ROI as follows:

$$\left|V^L_{x,y}\right| \begin{cases} < \theta_L & \text{Excluded from ROI} \\ \geq \theta_L & \text{Included in ROI} \end{cases}$$

In the experiments, $\theta_L = \mu_L/2$ (except for $L=0$), and $\theta_0=1$, were chosen empirically, where μ_L is the mean value of magnitude of the flow vector field generated at level L.

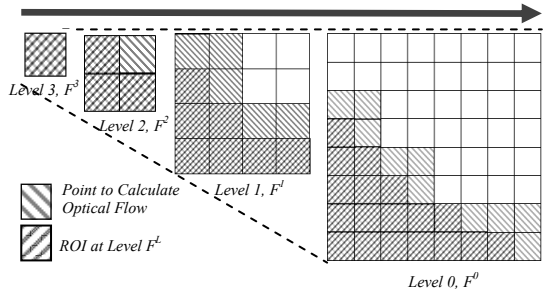

Figure 2: ROI prediction at pyramidal levels for an 8x8 block

If a data point *u'* falls within the ROI prediction, it is propagated to the succeeding level with all four corresponding pixel positions populated by data points as illustrated in Figure 2, to compensate for the previous sub sampling. The flow prediction vectors for the additional points are assumed to be equal to that of *u'*. The points that fall outside are simply discarded.

V^0, the flow field for the original frame, i.e. for level F^0 is passed to stage III. The flow outside the ROI taken as $(0,0)^T$.

Stage III: The block motion vectors are calculated here, using the flow field V^0, and the ROI definition. For a given block at *(a,b)*, of size *p x q,* corresponding motion vector $MV_{a,b}$ is given by,

$$MV_{a,b} = \frac{\sum_{\forall n \in W} [n]}{|W|} \quad (3)$$

Here, W is the set of pixels defined by,

$W = \left\{ n / n \in \left\| V^0_{x,y} \right\| > \theta_0; a \le x < (a+p); b \le y < (b+q) \right\}$, and $|W|$ is the cardinality of the set.

4. SIMULATION RESULTS

Simulations of the proposed system were conducted using four test sequences; '*Football*', '*Soccer*', '*Foreman*' and '*Silent*' such that a range of object and camera motion is considered. All four sequences were in CIF (352 x 288) resolution, 4:2:0 sub-sampling format and only the first 100 frames were used. Results for the proposed system were compared against the output of H.264/AVC using JVT reference software, JM ver.10.1 [11]. All sequences were encoded with 8x8 block partition, and an initial I frame and a sequence of P frames were used as the GOP structure. Frame referencing in JM was limited to one previous frame. Search range in JM and the output of motion analysis module were restricted to 16 for comparability. A capture of the motion analysis process is given in Figure 3.

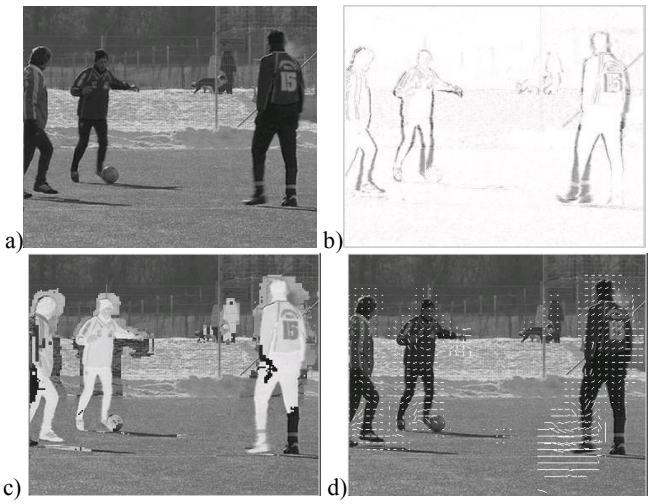

Figure 3: a)'*Soccer*' sequence original frame #4, b)frame difference between frame #4-5, c)identified ROI denoted in false colour and d)the block motion field derived for ROI overlaid.

The results are outlined based on four categories. The accuracy of the motion vectors derived by equation (3), in describing actual motion is demonstrated using the PSNR of motion compensated frames. Each predicted frame has been motion compensated from original preceding frame, assuming lossless frame compression to eliminate effect of compression loss within this measure. The RD performance and complexity of the system, when these motion vectors are used within H.264 are compared with the output of JM 10.1. Finally, the derived motion vectors are used in an application scenario to demonstrate their effectiveness.

4.1 Motion vector accuracy

The luminance PSNR of motion compensated frames is given for '*Foreman*' sequence in Figure 4 at three quantization parameters (QP), and the summary of results for all four sequences is given in Table 1.

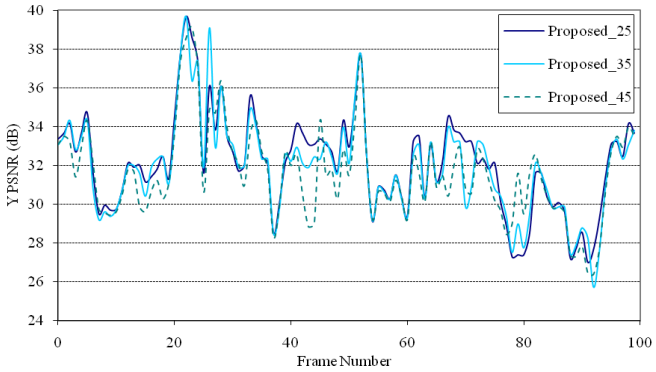

Figure 4: PSNR with different QP values, (*method_QP*)

	Avg. Area within ROI	Avg PSNR, dB	ME time (s)	
			JM	Proposed
Football	60.52%	24.34	152.17	147.06
Soccer	76.36%	34.30	185.14	119.89
Foreman	54.50%	31.89	139.51	96.53
Silent	17.17%	36.72	147.06	70.81

Table 1: PSNR performance and motion estimation time

4.2 Rate Distortion performance

RD performance of the proposed solution is compared with that of JM for each of the sequence considered, as given in Figure 5. Results illustrates that the proposed system performs quite similar to JM, although some macro blocks contain non-optimized motion information compared with those of JM.

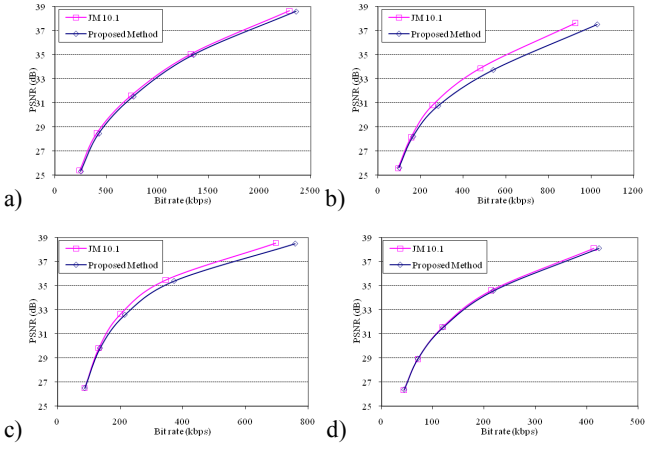

Figure 5: RD performance comparison a) '*Football*', b) '*Soccer*', c) '*Foreman*' and d) '*Silent*

4.3 Complexity

In the proposed method, complexity mainly resides in the implementation of equation (2). However, both I_x and I_y terms need only be calculated once for all pixels at a given pyramidal level, irrespective of the summation window size. This compares

favourably with the *SAD* calculations required at each pixel, for each candidate location in block based motion estimation. It is also demonstrated based on the time taken in motion estimation (ME) by the proposed method and JM compared in Table 1. JM ME time (for QP 25) given here is scaled down to the percentage area of ROI. The time taken by the proposed method is observed to be less than that of JM, for high and medium motion sequences.

4.4 Application Scenario: Object tracking

In order to demonstrate the effectiveness of achieved results, a simple tracking application was developed, where feature points are propagated along the sequence of frames by displacing them according to motion vectors. Object bounding box was then derived based on the distribution of feature points.

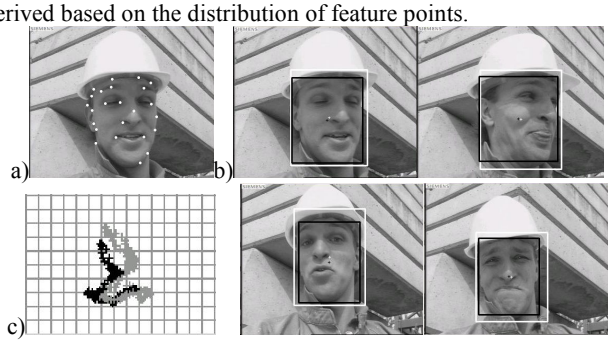

Figure 5: a) Frame #1 with selected feature points, b)Tracked boundary for frame #2,8,40 and 86, c)Trajectory of boundary centre points over 100 frames ([120,120] to [240,200] is shown).

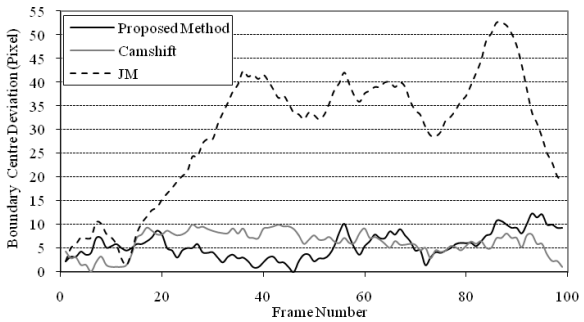

Figure 6: Comparison of tracking accuracy

Method	Avg. deviation compared with ground truth(Pixel)
JM (MV based)	29.74
Proposed Method (MV based)	**5.50**
Camshift (Pixel based)	6.15

Table 2: Tracking performance over 100 frames

Subjective results are outlined in Figure 6, for 'Foreman' sequence, where motion driven tracking with motion vector output from the proposed system(denoted in black), was compared with the OpenCV implementation of Camshift face detection algorithm [12] (denoted in white/gray), initialized with identical bounding area. Accuracy of tracking was measured by the deviation of the resulting bounding rectangle from that of manually marked ground truth. The output for the proposed method, JM and Camshaft are compared in Figure 7 and Table 2. The motion vectors from the proposed method perform consistently, and the result is comparable with that of pixel domain tracking; Camshift, while the trajectory for JM motion vectors strongly diverge from ground truth. Moreover, the foremost processing required at the application level is in extracting motion vectors, as opposed to the pixel level processing required by Camshift.

5. CONCLUSIONS

This paper proposes a method for enhanced reuse of information in compressed video at the time of encoding in order to reduce the processing required for compressed domain vision applications. This has been achieved by a heirarchical content driven motion estimation mechanism, complemented by a statistical region-of-interest prediction criterion. The evaluation of proposed method is conducted on four test sequences, that cover a range of motion types. In an object tracking application it is demonstrated that the resulting motion vectors can be directly used to extract object information, with the tracking performance comparable to the pixel domain approach. Additionally the RD performance and the complexity of encoder are affected only by a small margin. That can be weighed against the reduced processing required in analysing the video, for multiple applications. In the future, this work will be extended by incorporating the effect of global motion on the overall frame statistics.

6. REFERENCES

[1] R.V. Babu, K.R. Ramakrishnan, S.H. Srinivasan, "Video object segmentation: a compressed domain approach", *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 462- 474, April 2004

[2] W. You, M.S.H. Sabirin, and M. Kim, "Moving Object Tracking in H.264/AVC bit stream", *International Workshop on Multimedia Content Analysis and Mining (MCAM'07),* 2007.

[3] G. Takacs, V. Chandrasekhar, B. Girod and R. Grzeszczuk, "Feature Tracking for Mobile Augmented Reality Using Video Coder Motion Vectors," *6th IEEE and ACM Int. Symposium on Mixed and Augmented Reality, 2007*, pp. 141 – 144, 2007.

[4] S. De Bruyne, W. De Neve, K. De Wolf, D. De Schrijver, P. Verhoeve and R. V. de Walle, "Temporal Video Segmentation on H.264/AVC Compressed Bitstreams", *The International MultiMedia Modeling Conference (MMM),* 2007

[5] S. K. Kapotas, A. N. Skodras, "A new data hiding scheme for scene change detection in H.264 encoded video sequences", *Proceedings of the IEEE International Conference on Multimedia and Expo* ,pp 277-280, 2008

[6] C. Kas, H. Nicolas, "Joint global motion estimation and coding for scalable H.264/SVC high-definition video streams", *International Workshop on Content-Based Multimedia Indexing,* pp. 433-438, 2008

[7] D. Marpe, , T. Wiegand, G. J. Sullivan, "The H.264/MPEG4 Advanced Video Coding Standard and its Applications", *IEEE Communications Magazine*, August 2006

[8] A.M. Tourapis, "Fast ME in the JM reference software", ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6, Document JVT-P026, Poznań, PL, 24-29 July, 2005

[9] J.Y. Bouguet, "Pyramidal Implementation of the Lucas Kanade Feature Tracker", Intel Corporation, Microprocessor Research Labs, 2000

[10] B.D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision" *Proceedings of Imaging understanding workshop*, pp 121—130, 1981

[11] JVT H.264/AVC reference software version JM 10.1, http://iphome.hhi.de/suehring/tml/download/

[12] Intel Corporation. Open computer vision library, 2006. http://www.intel.com/technology/computing/opencv/index.htm.