Adaptive Motion Compensation in MPEG

CHARLES CRUDEN
cruden@cs.ualberta.ca

XIAOBO LI
li@cs.ualberta.ca

Department of Computing Science,
University of Alberta,
Edmonton, Alberta, Canada, T6G 2H1

Abstract

Ever increasing bandwidth devoted to multimedia applications such as video demands better compression as a counterbalance. To this end, a method of increasing the compression of block based motion compensation schemes is presented. Each motion compensated frame is analyzed using a region growing method to determine a visually appealing foreground and background. Foreground areas then have their motion vectors refined, and both foreground and background have their quantization levels adjusted to produce a video with less accuracy in areas which are of less interest to the viewer. The resulting change in compression produces videos 5% to 45% smaller than the originals while maintaining a similar level of quality for the viewer.

1 Introduction

The demand for low bit rate video coding is growing steadily as more and more people connect to multimedia networks like the world wide web. While pure frequency domain techniques are growing more popular, techniques such as MPEG which combine motion compensation with spatial redundancy elimination remain the standard [5]. Increasing the efficiency of motion compensation techniques is therefore in our best interests.

Although some newer propositions for video compression suggest alternative approaches, the technique that has consistently remained among the leading performers, and indeed is still used in the four leading video compression standards (MPEG I, MPEG II, H.261, H.263), is block based motion compensation. Improvements to this general method would therefore find wide use. What is lacking in motion compensation which is present in more recent systems is an accounting for the human visual system (HVS) beyond just the filtering of the DCT. To resolve this, a system enhancing motion compensation with variable accuracy levels is proposed. The remainder of the paper will detail how the visual system is exploited to determine areas of more or less interest (Section 2), the algorithm used to exploit these areas (Section 3), the results of the compression algorithm on various image sequences (Section 4) and future work (Section 5).

2 Subjects in video

Various attempts have been made at exploiting HVS's deficiencies in eliminating information from video to improve compression, by introducing visually insignificant errors to produce more compressible data [2], by allowing for blurring in quickly moving objects, or by other methods. Our system takes advantage of the tendency of the HVS to track objects with disparate movement, relative to the rest of the movement in the video [3]. It also exploits the HVS's decreased acuity when dealing with objects away from the centre of the retina and with objects with a high retinal velocity.

When tracking objects in a scene, the HVS attempts to centre the object it is tracking on the retina. If the object being tracked is moving, or if the object is still and the background is moving, the motion vectors of the object will tend to be different from those surrounding it. This smooth motion tracking effectively segments the image into areas of high and low interest (foreground and background), with areas in the foreground being objects with differing motion vectors from the background of the image. As the eye's spatial acuity is greater in the centre of the retina [4], details in the object being tracked become more apparent, whereas details in the areas away from the object are less so. Centering the object also simultaneously decreases the retinal velocity of the object and increases that of the area surrounding it, further dropping acuity in

cases where motion is significantly different. For the compression routines, this locates an ideal place to trade a loss of detail for higher compression ratios.

This system has a drawback, in that it depends on the ability of the HVS to track objects in the video with reasonable accuracy. Tracking itself is affected mostly by image velocity, image acceleration and predictability of image motion [3]. For the small viewing areas common to current video compression, the first two aspects are minimized. By the time objects reach retinal velocities where tracking starts to be affected (around 30 deg/sec [1]), motion prediction has already failed due to limited search areas for motion vectors. The third aspect, being affected by the video's content, is out of the algorithm's control.

3 The algorithm

To exploit this tradeoff, motion estimation is followed by an analysis of the vectors generated to locate the area with highest motion relative to the background of the picture and a procedure to reduce or enhance detail as appropriate. The algorithm proposed is presented as an extension to MPEG I, but can readily be adapted to most block based motion compensation video compression schemes. It consists of three steps:

- an analysis of the motion vectors of the current frame
- quadtree coding of the foreground and background regions
- adaptive coding of motion vectors and macroblocks of the current frame

3.1 Motion vector analysis

The first stage of coding requires the generation of motion vectors as specified by the MPEG standard. This done, analysis of the vectors to determine a foreground and a background can proceed. Based on the visual system observations above, the background of a video sequence will be defined as the largest area of relatively homogeneous movement: the foreground will be the remainder of the video sequence. While this description of the foreground tends to encompass more than just the object being focused on, it is obviously visually preferable that more of the image be coded with greater detail rather than less. So long as the object the eye is focusing on what is in the foreground, the difference should not be noticeable. Locating the foreground and background proceeds as the result of an image segmentation process performed on the motion vectors generated. The assumption is made that part of the background, whatever it may be, will encompass one of the four corners of the video sequence. While this is not always the case, for the great majority of sequences it will hold. Proceeding from this assumption, a region growing algorithm is started at each of the four corners of the frame. The current implementation provided two algorithms for growing regions: a single-link region growing algorithm, and a centroid-link region growing algorithm. The single-link region growing algorithm examined only 4-connected neighbours of the starting block. A new block was added to the region if the motion vectors of the two blocks differed by at most one in both the X and Y directions. i.e. for vectors M = $\{M_x, M_y\}$ and N = $\{N_x, N_y\}$

$$|M_x - N_x| + |M_y - N_y| \leq 1 \qquad (1)$$

Considering 8-connected neighbours, or allowing a less restrictive joining equation usually created too large a background region.

The centroid-link region growing algorithm considered 8-connected neighbours of the starting square. A new block was added to the region if its motion vector satisfied the following equation

$$t \geq \left[\frac{N^2}{N+1}(M_i - \overline{X_i})^2/S_i^2\right]^{\frac{1}{2}} \quad i = 1,2,3 \qquad (2)$$

where M_1 is the mean of the previous motion vectors' X components, M_2 is the mean of their Y components and M_3 is the mean of their lengths, S_1, S_2, and S_3 are the standard deviation of their X components, Y components and lengths and X_1, X_2 and X_3 are the current vector's X component, Y component and length.

Although only i = 1 and 2 would appear to be sufficient, providing for both the X and Y components of the vector to be statistically similar to the rest of the region, they turn out not to be. They account truly only for the direction of the vectors over the region, generating a mean vector by averaging the X and Y components of the vectors in the region. Examination of figure one shows averaging the X and Y components of vectors A and B would produce the shorter vector in C, whereas what is really needed is the longer C. This can be obtained by averaging the lengths of A and B and applying it to the averaged direction.

Once region growing at all four corners is completed, the average vectors of the regions are compared, and sufficiently similar regions (under the

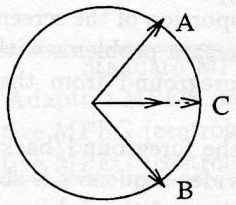

Figure 1: two ways of finding the average of A and B

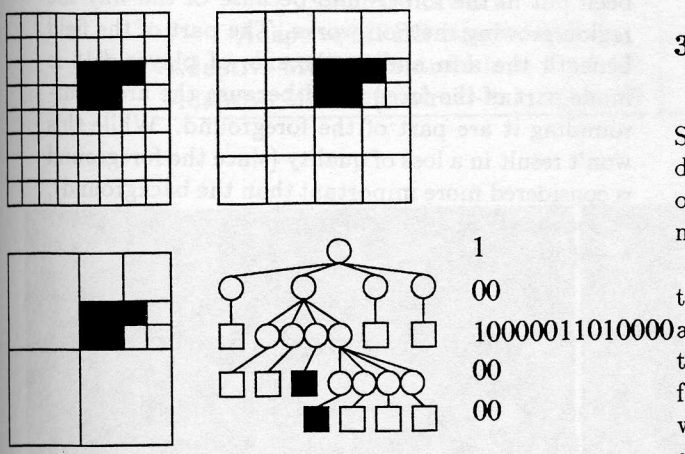

Figure 2: steps in producing the quadtree

criteria of equation 2) are considered merged, even if they aren't connected. The largest of these new regions is taken as the background of the image: the remainder is the foreground.

There are cases where region growing fails to provide regions of any useful size. In cases such as these (where the region grown was 3 blocks or less in size), the region is ignored and detection of a foreground/background for that area is considered to have failed. In cases where detection fails in all four corners, the frame is coded as a normal predicted frame.

3.2 Quadtree coding

To communicate the decision of which blocks are in the foreground and which in the background, a quadtree is sent at the start of each motion compensated frame.

The quadtree itself is built by the encoder by making an array matching the blocks of the video image, with each element of the array either zero or one, zero for the background, one for the foreground. Arrays one half the size of the previous are progressively built by looking at 2x2 blocks of the previous array and assigning the value 1 if they are dissimilar, or 0 if they all have the same value. If the previous array size is odd, only the elements of the 2x2 block which are still in the previous array are examined. The array building process continues until one of the array's dimensions falls below 1. The contents of the arrays are encoded by doing a depth first traversal and sending the resulting series of ones and zeroes. If a zero occurs at an element which isn't a leaf of the array tree, the value of a leaf element in the tree rooted at that element is sent after the zero.

3.3 Adaptive coding of motion vectors and macroblocks

So far, only an increase in the size of the compressed data has been achieved. To balance this two methods are used to reduce the size of the residual which needs to be coded after motion compensation.

The first is an adaptive coding of the motion vectors. For the blocks in the foreground of the image, a refinement is performed on the motion vector for the block by searching the area around it for a more accurate vector. Each possible vector within $\pm\frac{1}{2}$ pixel is searched as an alternate vector and the one with the least error is selected. This provides greater accuracy for the foreground against the standard accuracy for the background, at a cost of some additional work to find the refined motion vector and to generate the half pixel block values.

Additionally, different quantization levels can be set for the foreground and background macroblocks. Normally, the coefficients of the residual left over after motion compensation are quantized with the same step size across the entire frame. Instead, we specify two steps sizes, one for the foreground and one for the background, so that more precision can be focused in areas of interest.

3.4 Coding efficiency

For most block based motion compensation compression schemes, the greatest percentage of the time spent encoding the sequence is spent searching for motion vectors, performing repeated block comparisons. The extensions proposed do not alter this: less than half of one percent of processing time is spent locating the foreground and generating the quadtree to represent it. Where additional significant work is performed is in refining the motion vectors.

The amount of additional work required to refine the foreground of a given frame is dependent greatly on the size of the foreground that ends up being selected. Each macroblock selected for re-

quires an additional 8 block compares for the refinement step in the above algorithm. (Compare this with anywhere from 13 to 400 block compares required for the original motion vector estimation over a ± 10 pixel search area, depending on the search algorithm used.) Assuming a reasonable proportion of the macroblocks are selected as foreground, this could mean up to roughly 30% more block comparisons in a worst case scenario. In practice, an average of 20% more processing time is required in comparison to full pixel encodings. Half pixel encodings are executed in roughly the same amount of time.

4 Results and analysis

Four standard video sequences (caltrain, football, tennis and trevor [7]) were compressed using standard MPEG I and MPEG I with the extensions proposed. The encoder used in both cases was a version of the U.C.Berkeley mpeg_encode program [8], modified to produce both standard MPEGs and MPEGs which used adaptive motion vector coding and quantization. The results of compression of these sequences are shown in tables one and two.

Although many values of t were tried for the centroid link compression runs, the resulting videos fell squarely into two groups: one of high quality and low compression (for $t \leq 0.15$) and one of lower quality but high compression (for $t \geq 0.16$). The reason for this appears to be a sensitivity to the discrete values of the motion vectors.

As can be seen, the method of extracting a foreground and background provided sufficient room for compression to drop file size from between 5% to 45%, while maintaining a fairly similar signal to noise ratio. More importantly, visual inspection of the frames of the sequences shows little noticeable difference between the foreground of the MPEGs compressed with the adaptive method, and those without. Figure 3 shows frame 14 of the tennis sequence, on the left compressed using the standard method, and on the right compressed using the adaptive method (centroid link, t = 0.16). Unfortunately, with the higher compression ratios obtained by centroid-link, $t \geq 0.16$, artifacts begin to appear in some of the sequences, as the region growing method has chosen some of the active foreground as part of the background.

As would be expected, compression improves in cases where the ratio of background to foreground is higher: the foreground of tennis being isolated down to the moving ball and the player's arm, and the foreground of trevor being the person talking. Football has a large number of moving objects which take up a large proportion of the screen, as does caltrain. There are certain problems with the method of isolating the foreground from the background though.

An example of the foreground/background isolation in one of the video sequences is shown in figure 4. While the outline of the subjects is generally good, there are portions of the image which should be in the background (the field) but which have been put in the foreground because of the way the region growing method works. The part of the field beneath the arm and to the side of player #18 is made part of the foreground because the areas surrounding it are part of the foreground. While this won't result in a loss of quality (since the foreground is considered more important than the background), it does result in the compression being less than it could be.

5 Conclusion and future work

A method for selecting a visually appealing foreground and background for an arbitrary video sequence has been presented. Region growing at the corners of the video selects areas of homogeneous movement which forms the background: the foreground is the remainder of the video. Using this segmentation, the motion vectors for the foreground are refined, and the macroblocks coded with a denser quantization, while the visually less important background is coded with a sparser quantization. Applying this method to various standard image sequences has shown a 5% to 45% increase in compression while maintaining a similar SNR and visual image quality, with minimal computational overhead.

The method presented shows definite advantages for video compression, producing video of quality almost as good as standard methods, while providing appreciable gains in compression ratios. The method for extracting foreground and background has a few drawbacks which could be corrected for. Improvements to this by searching for a reasonable approximation of the camera movement of the video and subtracting that from the motion vectors before using region growing (similar to [6]) are being investigated and show promise for achieving even better segmentation of the video.

References

[1] van den Ber, A.V. and Collewin, H.: *Hu-*

Table 1: Compressed file sizes in bytes

	caltrain	football	tennis	trevor
Standard MPEG	261,357	1,302,865	1,358,250	129,654
Adaptive MPEG (single-link)	200,315	1,235,113	1,058,459	86,606
Adaptive MPEG (centroid-link, $t \leq 0.15$)	319,900	1,334,907	1,113,177	124,100
Adaptive MPEG (centroid-link, $t \geq 0.16$)	154,214	866,169	726,028	73,443

Table 2: PSNR of predicted frames in dB

	caltrain	football	tennis	trevor
Standard MPEG	33.2	33.3	30.6	35.1
Adaptive MPEG (single-link)	30.0	31.5	26.7	31.8
Adaptive MPEG (centroid-link, $t \leq 0.15$)	32.2	33.0	27.3	34.4
Adaptive MPEG (centroid-link, $t \geq 0.16$)	28.5	28.2	25.2	31.1

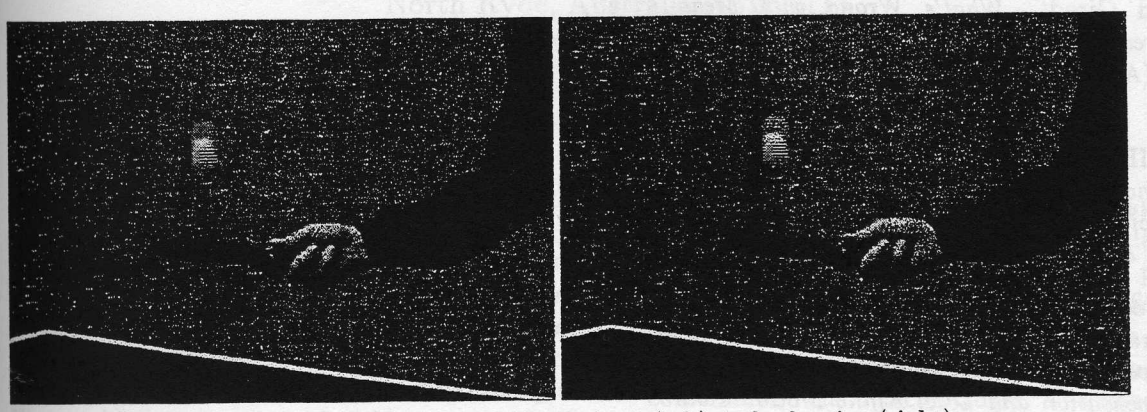

Figure 3: frame 14 of tennis from standard (left) and adaptive (right) sequences

Figure 4: frame 32 of football with foreground highlighted

man Smooth Pursuit: Effects of Stimulus Extent and of Spatial and Temporal Constraints of the Pursuit Trajectory, Vision Research, 26(8), pp.1209-1222

[2] Chou, Chun-Hsien and Chen, Chi-Wei: *A Perceptually Optimized 3-D Subband Codec for Video Coding over Wireless Channels*, IEEE Transactions on Circuits and Systems for Video Technology, April 1996, pp.143-156

[3] Michael P. Eckert and Gershon Buchsbaum: *The Significance of Eye Movement and Image Acceleration for Coding Television Image Sequences*, Digital Images and Human Vision (Andrew B. Watson ed.), Bradford Book, 1993

[4] Bernd Girod: *What's Wrong with Mean-squared Error?*, Digital Images and Human Vision (Andrew B. Watson ed.), Bradford Book, 1993

[5] ISO/IEC 11172-2, *MPEG-1 - Information Technology - Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to about 1.5 Mbits/s, Part 2: Video*

[6] H. Jozawa, K. Kamikura, A. Sagata, H. Kotera, H. Watanabe: *Two-Stage Motion Compensation using Adaptive Global Motion Compensation and Local Affine Motion Compensation*, IEEE Transactions on Circuits and Systems for Video Technology, February 1997, pp.75-85

[7] ftp://ipl.rpi.edu/pub/image/sequence

[8] ftp://mm-ftp.cs.berkeley.edu/pub/multimedia