AFFINE PREDICTION AS A POST PROCESSING STAGE

Roman C. Kordasiewicz†, Michael D. Gallant‡, and Shahram Shirani†

†McMaster University, ‡LSI Logic Corporation

ABSTRACT

Translational motion vectors (MV)s and macro block (MB) frame partitioning are the predominant means of motion estimation (ME) and motion compensation (MC). However, the translational motion model does not describe sufficiently complex motion such as rotation, zoom or shearing. To remedy this one can start computing more advanced motion parameters and/or partition the frame differently. However these approaches are either very computationally expensive and/or have limited search ranges. Thus, in this paper we propose a novel post processing stage which can be easily incorporated into most of the current coders. This stage generates the predictor for each inter MB, based on an affine motion model using translational motion vectors. Our approach has very low computational complexity, however average PSNR gains of up to 0.6 dB were realized for video sequences with complex motion.

Index Terms— motion vectors, prediction, motion estimation, motion compensation, affine motion,

1. INTRODUCTION

Block matching (BM) and translational motion vectors (MV)s form the basis of the traditional motion estimation (ME) and motion compensated prediction (MCP). The basic idea behind traditional ME is to take a rectangular region from the current frame and to match it as closely as possible to a region in a reference frame. The match criterion is most often based on sum of absolute differences (SAD) or mean square error (MSE) measure. There are two key aspects of block matching and translational MVs important to our discussion. First, a single motion vector is used to represent the motion of the entire MB region, which can lead to motion field discontinuities and therefore blocking artefacts. Second and most importantly, translational motion vectors can not effectively model more complicated motion such as camera rotation, zoom, or object shearing. However, traditional BM is relatively fast, simple, performs well, and easily lends itself to hardware implementations.

Polynomial [1] and affine [2] motion models were envisioned to replace the translational motion vectors, since they much more effectively model complex motion. However, determining advanced motion coefficients for rectangular regions is still computationally challenging in real time applications. Thus a simplified affine motion estimation using triangular mesh regions was proposed [3, 4, 5]. Here the affine MVs for each of the pixels in the mesh area is computed using a simple scan line algorithm [6]. One of the biggest detractors of the mesh based approaches is the fact that the mesh lines should not cross over [3]. Because of this, the search areas in the initial search and in the refinement are very small and are directly tied to the mesh size. In addition, an extra row and column of MVs has to be sent and/or stored, and the mesh partitioning is independent of video which can further limit the performance of the mesh-based approach. However, mesh-based approaches can still yield competitive results because they model more complex motion very effectively [3].

To address the limitations in both the traditional BM and in the mesh based approaches, we propose a novel affine prediction stage. This stage can be incorporated into any of the existing traditional BM coders without affecting their syntax.

2. PREDICTION USING AFFINE MOTION

H.264 is the one of the latest video compression standards, significantly outperforming its predecessors. To gain such compression, a large degree of flexibility is built into the specification and this is likely to continue in future standards. Therefore, in this work we try to address the various MB sizes, MB modes (intra, inter, copy), and the complexity involved in converting current software and/or hardware to leverage on affine motion compensated prediction.

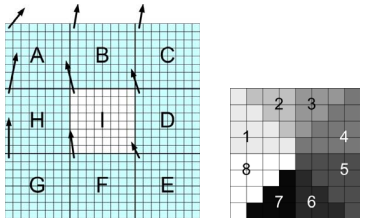
2.1. Basic description

We propose that using our approach does not have to impact existing motion estimation (ME) and mode decisions in most of the current encoders. As long as a codec partitions frames into MBs and it finds translational MVs for each of the inter coded regions, it can benefit from our approach. In practice the prediction is often performed on per MB basis as the

†Roman C. Kordasiewicz and Dr. Shahram Shirani are with Department of Electrical and Computer Engineering, ITB A320, McMaster University, 1280 Main Street West, Hamilton, Ontario L8S-4K1, their respective emails are kordasi@grads.ece.mcmaster.ca , shirani@mail.ece.mcmaster.ca

‡Dr. Michael D. Gallant is with LSI Logic Corporation, 97 Randall Drive, Waterloo, Ontario N2V-1C5, his email is mgallant@lsi.com

frame is scanned in raster order, however with our approach a small buffer containing neighbour MB mode and MV information must be maintained. In this paper we refer to generating the final predicted frame as post-processing, however one should be aware that this is still performed within the coding loop, and thus the difference frame is affected. As a result the affine post-processing motion compensation has to be performed in both the encoder and the decoder. However the complexity of our method is very low, and it is comparable to loop filtering, which is an integral part of many existing codecs.



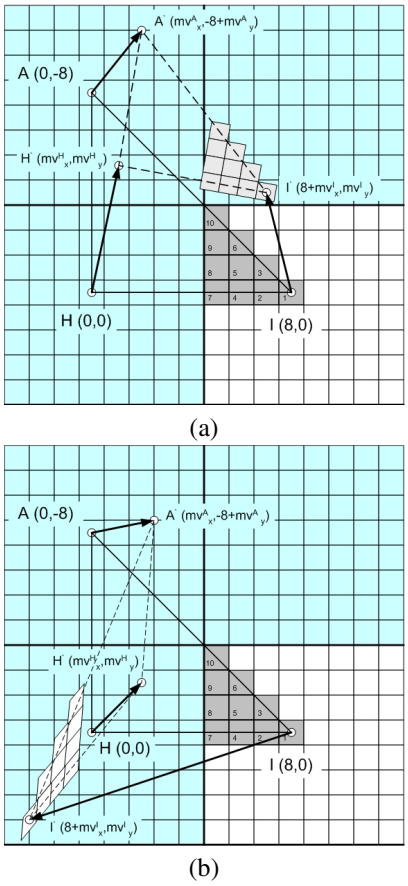
(a) (b)

Fig. 1. (a) Small frame partitioned into 9 8×8 blocks, and translational MVs for each of the blocks. (b) Eight partitions of the 8×8 block

After traditional motion compensated prediction, the frame partitioning is known along with MVs and other MB mode information. For now we assume that only "inter" 8×8 blocks are used, as shown in Fig. 1 (a), where a 24×24 pixel region with 9 blocks and their MVs is shown. Now consider finding a predictor for the I^{th} (center) block. Traditionally to obtain this predictor, every pixel is predicted from the reference frame using a single MV which in this figure is $(d_x, d_y) = (-1, -4)$. However, we can partition the I^{th} block into 8 symmetric triangular regions as in Fig. 1 (b). We then consider predicting region 1 of the I^{th} block using the affine motion model shown in Fig. 2 (a). This is done by taking into account the motion of the neighbouring blocks and forming a right angled triangle (A, H, I). The affine motion vector (AMV) $\vec{v} = [v_1, v_2, v_3, v_4, v_5, v_6]$, defines the pixel location mapping between the current and reference frames using the following formula [7]:

$$\begin{aligned} x'(x,y) &= v_1 x + v_2 y + v_3 \\ y'(x,y) &= v_4 x + v_5 y + v_6 \end{aligned} \tag{1}$$

with, $\vec{x} = [x, y]$, being pixel position in the current frame with respect to the triangle (A, H, I) centered at H, and $\vec{x}' = [x', y']$ being a pixel position in the reference frame. Now, we assume that corner pixels of the triangle can be predicted using the translational MV for each of their respective blocks. This is done because the translational MV is most likely accurate near the center of the block [8]. Since the positions of $\vec{x}$ and $\vec{x}'$ are known at the corners of the triangle, it is possible to solve for the affine parameters, resulting in:



(a)

(b)

Fig. 2. Examples of using affine motion to predict the 1'st region in an 8×8 MB. (a) Positive example. (b) Example when a motion field discontinuity occurs.

$$\begin{aligned} v_1 &= (mv_x^I - mv_x^H) >> 3 \\ v_2 &= (mv_x^H - mv_x^A) >> 3 \\ v_3 &= mv_x^H \\ v_4 &= (mv_y^I - mv_y^H) >> 3 \\ v_5 &= (mv_y^H - mv_y^A) >> 3 \\ v_6 &= mv_y^H \end{aligned} \tag{2}$$

with $(mv_x^A, mv_y^A), (mv_x^H, mv_y^H), (mv_x^I, mv_y^I)$ being the MVs for regions A,H, and I respectively, and $>>$ denotes a shift right operator. We have purposefully chosen the size of the triangle to be 9 such that there are no division operations [5]. Once the affine parameters are computed, then region 1 pixel positions in the previous frame are determined by doing a simple addition or subtraction as we move from pixel to pixel based on the scan line algorithm [6]. This is thanks to the following recursive formulas:

$$\begin{aligned} x'(x \pm 1, y) &= x'(x,y) \pm v_1 \\ x'(x, y \pm 1) &= x'(x,y) \pm v_2 \\ y'(x \pm 1, y) &= y'(x,y) \pm v_4 \\ y'(x, y \pm 1) &= y'(x,y) \pm v_5 \end{aligned} \tag{3}$$

which can be easily derived from Equation (1). In the end only 10 pixels are predicted for region 1 of the I^{th} block. The other 7 regions of this block are predicted in a very similar way. Note that the traditional frame partitioning into MBs is maintained, and new prediction regions were defined (shown in Fig. 1 (b)), in contrast to the mesh based approaches.

2.2. Meeting real system requirements

So far we have demonstrated how a frame partitioned into 8×8 blocks can be predicted using an affine motion field. However in real coders there are numerous special cases and modes, which are addressed in the following paragraphs.

- Picture Boundary: At the edges of the frame some MBs don't have neighbours, and thus no affine prediction can be performed. Thus if a region in Fig. 1 (b) does not have neighbours with valid MVs, then that region is predicted using the translational MV for that MB.

- MB size: In most of the current video coders, a frame can be partitioned into rectangular regions of size $i \times j$, where i and $j \in \{32, 16, 8, 4\}$. Any block region, of which one or both dimensions are 4 pixels, would marginally benefit from using the affine motion field. Such regions are thus always predicted using translational MVs. However, any region larger than 8×8 would be considered for the sake of prediction as several 8×8 blocks. For example, a 16×16 MB would be considered as four 8×8 blocks each inheriting the MV of the larger block. The 8×8 blocks are the basic building block of our prediction engine. Since they are small enough to represent relatively fine motion field density, and are large enough to make use of affine motion prediction.

Note that when a 16×16 MB is considered as four 8×8 block, then the center 8×8 pixels are predicted using purely translational MV. Predicting the center of a larger block with translational MV is a very good trade off between complexity and performance. From the complexity point of view very little extra has to be done to find a translationally predicted region, and there is no extra work in the prediction. From the performance point of view the largest prediction error occurs on the periphery of the MB [8], and our approach targets that region.

- MB mode: There are many reasons why it is beneficial to encode regions with different modes (intra, inter, copy), and it is important to accommodate them all. Intra MBs can be treated the same way as the edge of picture. For all inter coded regions we would try to perform affine motion field prediction when the neighbour information allows for that. The copy MB regions can be treated for the purposes of prediction as inter coded regions with 0 MVs.

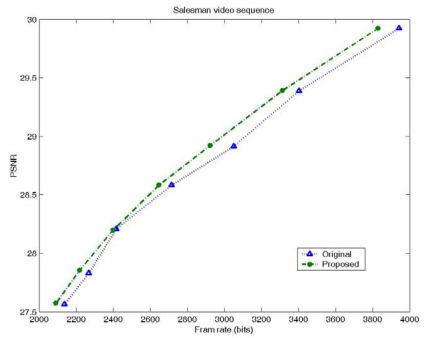
- Motion field discontinuities: In order to increase compression large search regions must be sometimes considered. With large search areas there is often a possibility of finding a predictor which is not necessarily in the same general direction as of its neighbours, as shown in Fig. 2 (b). One can consider this a discontinuity in the motion field. In such situations, predicting using the affine motion often yields poor results, and thus such situations have to be excluded. Since limiting the search range often deteriorates the performance of ME and motion compensation (MC) it is more beneficial to develop a simple test for such cases. In the example Fig. 2 (b) it is sufficient to check that $8 + mv_x^I > 0$. Similar tests can be devised for other cases. In addition, some discontinuities may lead to predicting from very narrow regions, which often results in poor results. To remedy this problem, another basic test can be applied.

3. RESULTS

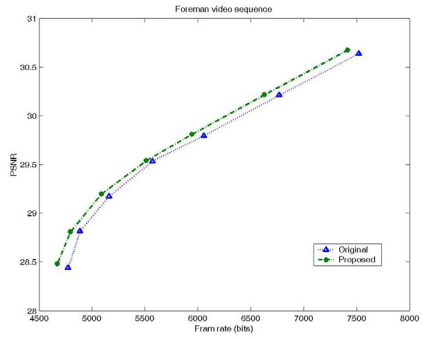
In our experiments a typical video coder was used, with ME, motion compensation, difference MB generation, entropy coding of the residual MBs, and entropy coding of the MVs and MB mode information. The translational motion search area was ± 15 pixels, first performed on sub-sampled images and then on the original video. The translational MV refinement performed, was ± 2 *step*s, where *step* was a function of the quantization parameter (ie. the smallest MV delta that did not get quantized to zero). In addition, four CIF video sequences were tested `salesman`, `foreman`, `mobile` and `modified mobile`. Since no suitable video sequence was found with significant amounts of affine motion, `modified mobile` was generated form the `mobile` video sequence by adding visible and natural amounts of camera zoom and rotation. These video sequences were first encoded using the unmodified coder, and by a coder with an affine predictor generation stage for the final predictor MBs. We call these two methods "original" and "proposed". The advantage of using our method is clearly visible in Fig. 3 where PSNR vs. bit rate plots are shown for these video sequences. In Fig. 3 (a) and (b) video sequences with less complex motion are shown, yielding average PSNR gains of 0.1 and 0.12 dB respectively. This is quite expected since in absence of any significant affine motion our method converges to standard translational prediction. However, because of the low computational complexity of our method, we can still justify using our method even for these video sequences. The real motivation for using the affine prediction stage can be seen in Fig. 3 (c) and (d), where average PSNR gains of 0.18 and 0.6 dB were respectively realized. This is due to the more complex types of motion visible in these video sequences. The PSNR gains can be interpreted as average bit rate reductions of 4.3% and 9.5% respectively. This is quite impressive considering that these gains are a result of computationally simple post processing stage.

4. CONCLUSION

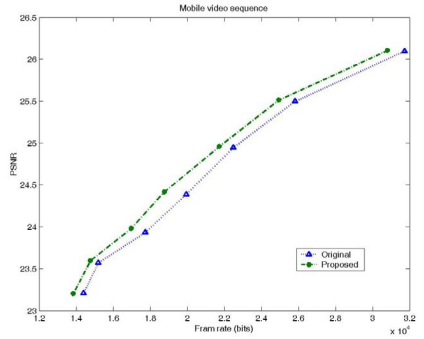
Complex motion is a challenge for video coders using translational MVs, resulting in poorly predicted frames. Correct-



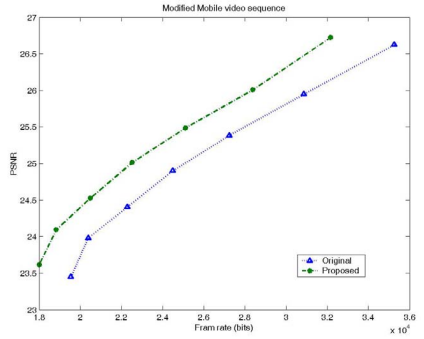
(a)



(b)



(c)



(d)

Fig. 3. PSNR vs. frame bit rate plots, depicting the "original" and "proposed" results for (a) `salesman`, (b) `foreman`,(c) `mobile` and (d) `modified mobile` video sequences.

ing this problem by using more advanced motion models in motion estimation and motion compensation during mode decisions is still highly prohibitive in real time applications. Therefore, in this paper we propose a novel post processing prediction stage, which converts the translational MVs into affine MVs. This affine prediction stage can be easily defined and integrated into any of the existing coders which use translational MVs and rectangular MB regions. The additional computational overhead is very low, and it is in orders of few additions, shift operations per pixel. For video sequences with less complex motion, our method yields comparable and often better video quality to translational prediction. However, the average PSNR improvement as a result of using our method in a video sequences which contain complex motion were observed to reach up to 0.6dB.

5. REFERENCES

[1] Marta Karczewicz, Jacek Nieweglowski, and Petri Haavisto, "Video coding using motion compensation with polynomial motion vector fields," *Signal Processing: Image Communiation*, vol. 10, pp. 63 – 91, 1997.

[2] Nokia Inc. - Nokia Research Center, "MVC coder/decoder submitted to ITU-T," 2000.

[3] M. Sayed and W. Badawy, "An affine-based algorithm and SIMD architecture for video compression with low bit-rate applications," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, April 2006.

[4] W. Badawy and M. Bayoumi, "A scalable affine core for mesh-based video object motion compensation," *The 7th IEEE International Conference on Electronics, Circuits and Systems*, vol. 2, Dec. 2000.

[5] W. Badawy and M. Bayoumi, "A multiplication-free parallel architecture for affine transformation," *IEEE International Conference on Application-Specific Systems, Architectures, and Processors*, July 2000.

[6] K. Kannappan, "An interleaved scanline algorithm for 2-d affine transformations of images," *Proceedings of the 35th Midwest Symposium on Circuits and Systems*, vol. 1, Aug. 1992.

[7] Roman C. Kordasiewicz, Michael D. Gallant, and Shahram Shirani, "Modelling the effect of quantizing affine motion vectors on rate and energy of difference macroblocks," *ICIP International Conference on Image Processing*, September 2005.

[8] Michael T. Orchard and Garry J. Sullivan, "Overlapped block motion compensation: An estimation-theoretic approach," *IEEE Transactions on Image Processing*, vol. 3, Sept. 1994.