AN ADAPTIVE INTERPOLATION SCHEME FOR 2-D MESH MOTION COMPENSATION

Pohsiang Hsu, K. J. Ray Liu

Dept. of Elect. Eng. and Inst. for Systems Research
University of Maryland, College Park, MD 20742
phsu,kjrliu@eng.umd.edu

Tsuhan Chen

AT&T Labs-Research
Red Bank, NJ 07701
tsuhan@research.att.com

ABSTRACT

Two dimensional mesh motion compensation produces blocking free prediction in contrast to block matching motion compensation by generating a smooth full motion field from the set of node motion vectors. This is clearly not appropriate across moving object boundaries where motion discontinuities exist. In this paper, we address this problem by proposing an adaptive interpolation scheme for patches that contain multiple motions (i.e., patches around motion boundaries). Simulation results show improvements over standard 2-D mesh motion compensation at a fraction of computational increase.

1. INTRODUCTION

For low bit rate applications, the block matching algorithm (BMA) is known to produce visually annoying blocking artifacts. One promising approach to remedy this problem is to use 2-D mesh based motion compensation [1, 2, 3, 4] which produces blocking-free prediction. Moreover, the 2-D mesh employs an affine model for the motion field making it capable of representing many common types of motion such as zooms and rotations that BMA can not handle. One inherent problem with 2-D mesh based motion estimation is that a globally smooth motion field model is assumed. Clearly, this is not true for most image sequences where the motion field contains discontinuities due to moving objects. Recognizing this fact, an approach was proposed in [5] using 2-D mesh motion compensation in a forward tracking framework where an irregular triangular mesh was designed to avoid placing grid points inside occlusion area and the mesh was tracked forward in time. This forward tracking approach involves node additions and deletions in newly discovered occlusion area where the maintenance of the mesh structure can become quite complex. In this paper, we investigate a simpler alternative approach in the framework of 2-D mesh based motion compensation with backward tracking and regular quadrilateral mesh using patchwise adaptive interpolation of the motion field.

2. PROBLEM FORMULATION

In 2-D mesh motion compensation with backward tracking, the current frame is covered with a nonoverlapping mesh and this mesh is deformed in the previous frame so that the contents of the two frames match with each other as closely as possible. The displacement of the nodes (i.e node motion vector) will be used to generate the full motion field through spatial transformation. To find the globally optimal set of node motion vectors requires exhaustive search due to the dependencies that exist among the node motion vectors which is too computationally intensive. Normally, a strategy that finds a local minimum is used [1] where the node motion vectors are first coarsely estimated by using BMA and then iteratively refined. In the case of quadrilateral mesh, the refinement process is an iterative minimization strategy where in each pass, every node motion vector is visited once in some scanning order and adjusted to find the position that results in the smallest prediction error while keeping the eight surrounding node motion vectors fixed. The process continues until no adjustments were made in a single pass or when a preset number of iterations has been reached.

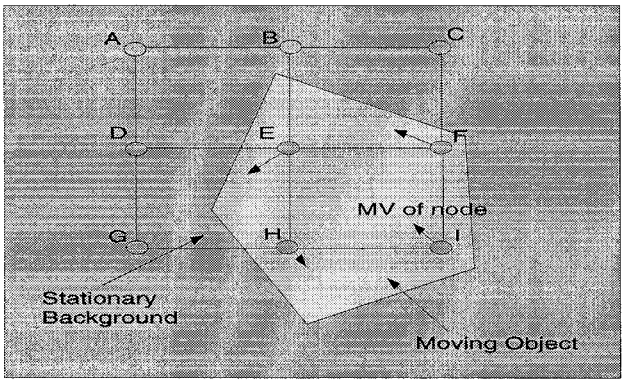

Figure 1: Illustration of incorrect interpolation of motion field in patches containing multiple motions

The problem with quadrilateral mesh motion estimation is the underlying assumption that only a continuous motion field exists within a patch. This implies that for patches containing multiple motions, the performance of quadrilateral mesh motion compensation will be unacceptable since we will be attempting to model the motions of different objects by a continuous motion field. Furthermore, the use of the iterative minimization give rise to the problem of error propagation in the refinement process where an unreliable

0-8186-8183-7/97 $10.00 © 1997 IEEE

node motion vector will affect the refinement of other node motion vectors around it. To illustrate the points made so far, consider the scenario shown in Fig. 1 where we have a rotating object on a still, inhomogeneous background. Take patch D-E-G-H for instance. In this patch, there are two distinct motions (i.e. rotating object and still background). With 2-D mesh based motion compensation, the motion field inside this patch is generated through bilinear interpolation using the node motion vectors D,E,G, and H. This means that incorrect motion vectors will be generated inside this patch creating a warped background which is visually annoying and generates large distortion. Ideally, the best thing to do for patch D-E-G-H is to use a combination of node motion vectors E and H for the object portion and a combination of motion vectors D and G for the background portion. This process can be seen as changing the interpolation of the patch from fixed bilinear interpolation of four nodes to an adaptively defined interpolation pattern that adjusts to the scene content. This is also similar to a motion field segmentation [6] of the patch. However, this process requires that the exact contour of the object be known at the decoder incurring a rate penalty. To balance the tradeoff between accurate boundary description (lower distortion) versus the rate needed to describe the contour, we make the assumptions that each patch contains at most two moving objects and the division between the moving objects can be approximated by using a straight line. Moreover, the possible line divisions are quantized to a small set to further decrease the amount of rate used to represent them. To summarize, we propose to construct a set of interpolation patterns to allow adaptivity in motion field interpolation for patches that contain multiple motions due to occlusion.

3. ADAPTIVE INTERPOLATION

Currently, the set of interpolation patterns used in our system contains 32 shapes plus the default shape (i.e. Bilinear interpolation of four nodes) as shown in Fig. 2. Each pattern consists of 16 by 16 pixels corresponding to a 16 by 16 patch in the mesh and it specifies how the patch should be interpolated given the four vertices A,B,C,D that surrounds it. Given the fact that we are dividing the region of interest with a straight line, two types of configuration are possible namely, a division of the patch into two regions each contains two nodes or a division of the patch into two regions where one region contains one node and the other contains three nodes. For the first configuration, we need to have a method of interpolating each region from its two node motion vectors. Here we propose to use linear interpolation to generate the line between the two nodes and then repetitively fill the rest of the region using the interpolated line. As an example, the regions labeled A,B are interpolated as follows:

$$mv(x,y) = mv_B * \frac{x}{16} + mv_A * (1 - \frac{x}{16}).$$

In the second configuration, we have a region containing three nodes and another containing one node. Here we simply use constant interpolation to fill both regions where for the region with three nodes, we choose the node that is located diagonally from the node in the other region. As an example, the region labeled A are interpolated as follows:

$$mv(x,y) = mv_A.$$

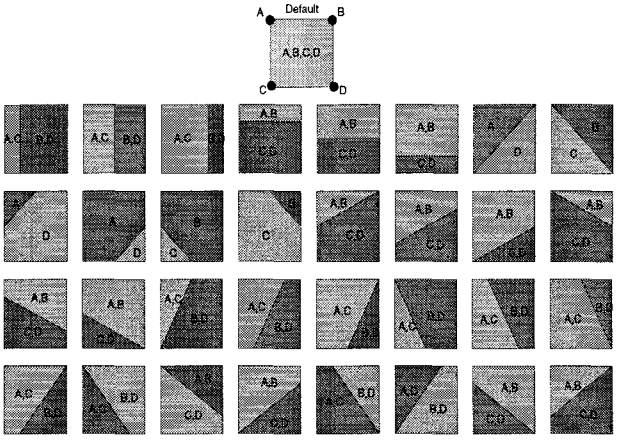

Figure 2: Set of interpolation patterns

4. INTERPOLATION PATTERN SELECTION

In this section, we address the problem of how to choose the interpolation pattern adaptively for each patch. First, we perform the standard 2-D mesh motion compensation and we compute the prediction error for each patch. A patch is considered as a candidate for adaptive interpolation if its prediction error is greater than a given threshold. For these candidate patches, the actual decision is done in a rate-distortion framework[7] as described below. Let $D(i)$ denotes the resulting prediction error in using interpolation pattern $i, i = 0, ..., 32$ for the candidate patch where pattern 0 denotes the default bilinear interpolation. Let

$$i^* = arg \min_{i=1,...,32} D(i).$$

be the index to the interpolation pattern excluding the default pattern that resulted in the minimum residual.

We note that the exhaustive search to find the best interpolation patterns out of the 32 modes can be performed efficiently by taking advantages of the fact that there are many overlapped regions in the set of patterns where the computation of the displaced frame difference and the generation of the motion field need only to be done once. For example, we only need to compute the motion vectors for the four line segments AB, BD, DC, and CA once and these interpolated motion vectors can then be re-used to generate the motion field in the patterns that contain regions of two nodes each. Moreover, some patterns share part of the same pixel residuals so that these parts need to be computed only once for example, left quarter part of Mode 1 and Mode 3.

Now if $D(i^*) < D(0)$ then we encode the resulting residuals from using pattern i^* and pattern 0 to get R_0 and R_{i^*} otherwise the default interpolation pattern is chosen.

Once we've computed R_0 and R_{i^*}, the decision rule for deciding whether to used adaptive interpolation becomes:

If $R_0 > R_{i^*} + R_{shape}$,
Choose adaptive interpolation using pattern i^*
else
Choose bilinear interpolation.

where R_{shape} denotes the number of bits required to encode the interpolation pattern information. Due to the fact that the interpolation information is compressed, we do not know R_{shape} a priori so we will need to make an estimate of it. We made a conservative estimate and chose a fixed constant for $R_{shape} = 10$ bits. This issue will be discussed further in the section describing the coding of the patterns.

After the set of interpolation patterns is chosen, we re-optimize the set of node motion vectors by applying the refinement algorithm[1] to refine the motion vectors with respect to the set of new interpolation patterns. We note here that only those node motion vectors surrounding the patch whose interpolation pattern changed needs to be refined.

5. CODING OF THE INTERPOLATION PATTERNS

The interpolation pattern information is coded as follows. First, we create a interpolation classification matrix C where if $C_{i,j} = 1$ then patch (i, j) is adaptively interpolated otherwise $C_{i,j} = 0$ and the actual interpolation patterns associated with the nonzero components of C are organized in a raster scan order and coded using a five bits fixed length codeword. This process is demonstrated in fig. 3 where on the left we show the interpolation pattern chosen for each block in a frame and on the right we show how the information is decomposed into C and the actual interpolation patterns.

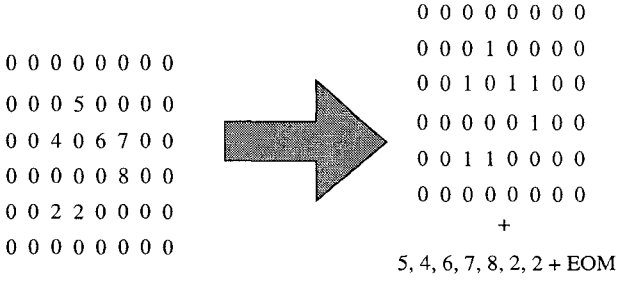

Figure 3: Set of interpolation patterns

In conclusion, we see that adding an interpolation pattern requires 5 bits to signal the actual pattern plus the bits needed to signal that the current patch will be adaptively interpolated. Since the process of run length encoding is highly nonlinear and dependent both on the locations of other adaptively interpolated patches and the number of adaptively interpolated patches, we found through experiments that 5 bits is a good approximation to the average number of bits needed for a wide range of interpolation maps which is how we decided on $R_{shape} = 10$.

6. SIMULATION RESULTS

The simulations were performed using ninety frames of the H.263 color test sequences Carphone and Salesman in QCIF format. The frame rate was set to be 8.33 Hz with the original frame rate at 25 Hz. We compared the coded results of our scheme with 2-D quadrilateral mesh and H.263(TMN5)[8] with advance prediction mode on. Same DCT error residual coding was used for our scheme and 2-D quadrilateral mesh while the motion vectors were differential coded using prediction from it's neighboring motion vectors as specified in H.263. For both our scheme and 2-D mesh, we used a grid partition of 16 by 16 pixels and an initial search area of ± 7 pixels with the refinement search area of ± 2 pixels. The node motion vectors were quantized to half-pixel accuracy. For both schemes, the intraframe was coded using a fixed QP of 15 and we vary the QP used for the Interframe to generate the rate-distortion points.

In Fig. 4, we showed the rate-distortion curve and it indicates that the performance of our scheme is better than both H.263 and 2-D mesh for these two sequences.

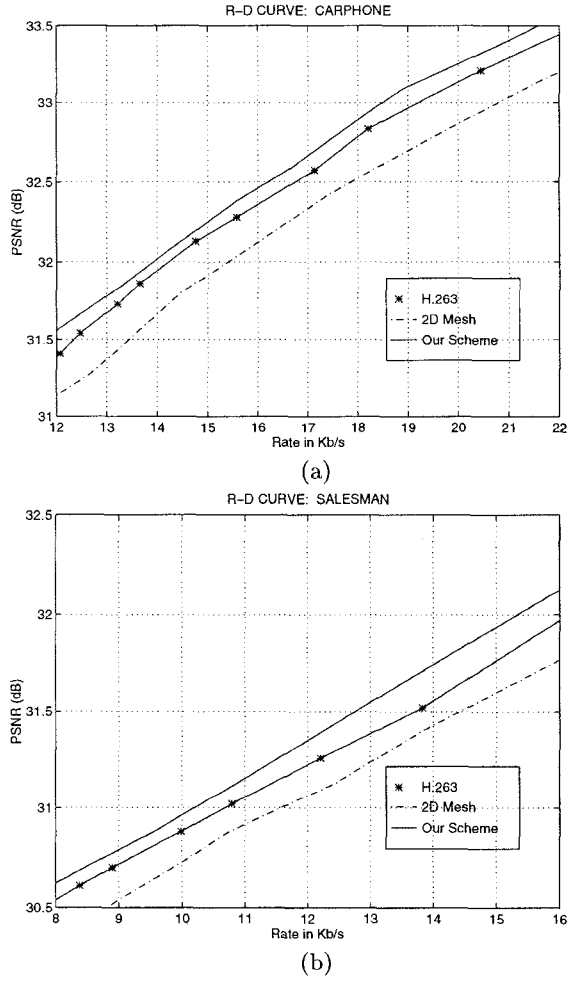

Figure 4: Rate-Distortion Curves

Lastly, we show in Fig. 5 some motion compensated results using the original frames only. In Fig. 5(a), (b), we show the two original frames of the sequence Carphone used in this experiment. As we can see from Fig. 5(c), the BMA compensated frame looks blocky and part of the left side of the face of the person was chopped off. On the other hand, Fig. 5(d) shows that 2-D mesh produced a smooth result but part of the background on the left side of the face was warped. Lastly, we show the result produced by our scheme in Fig. 5(e) along with the chosen interpolation patterns in Fig. 5(f). As we can see, the prediction is still smooth but the background portion is no longer warped.

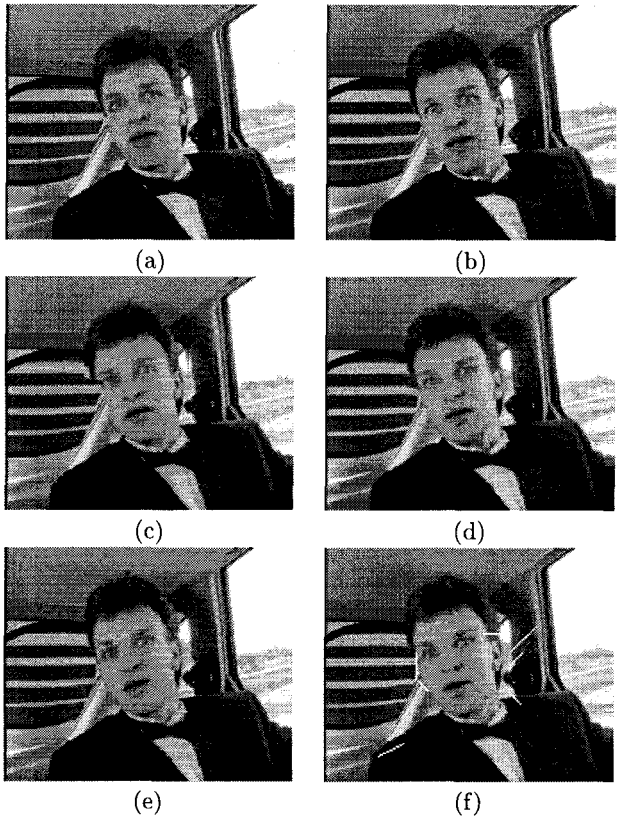

Figure 5: Motion Compensated Results: (a) Original frame 30 (b) Original frame 33 (c) BMA compensate frame 33 (d) 2-D mesh compensated frame 33 (e) Compensated frame 33 using Our Scheme (f) Interpolation Map used for Our Scheme

7. CONCLUSION

In this paper, we presented a video codec that combined 2-D backward regular mesh with adaptive interpolation aimed at improving prediction in patches that contains multiple motion. The simulation results indicate that while our codec is better than using 2-D mesh alone, it only gives slight PSNR improvements over block-based motion compensation approach (i.e. H.263 with Advanced Prediction) suggesting that the use of globally smooth motion field is inappropriate and there are limitations placed on the use of uniform backward tracking mesh due to it's inability to track scene contents. More promising approach is to work in an object-based framework by using forward tracking with separate mesh defined for each moving objects[5] and this will be left for future research.

8. REFERENCES

[1] Y. Nakaya and H. Harashima, "Motion Compensation based on Spatial Transformations," *IEEE Trans. on Circuits and Syst. for Video Tech.*, vol. 4, no. 3, pp. 339–356, Jun. 1994.

[2] Y. Wang and O. Lee, "Active Mesh - A Feature Seeking and Tracking Image Sequence Representation Scheme," *IEEE Trans. o Image Processing*, vol. 3, no. 5, pp. 610–624, Sept. 1994.

[3] Y. Wang and O. Lee, "Use of Two-Dimensional Deformable Mesh Structures for Video Coding, Part I - The Synthesis Problem: Mesh-Based Function Approximation and Mapping *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 6, no. 6, pp. 636–646, Dec. 1996.

[4] Y. Wang, O. Lee, and A. Vetro, "Use of Two-Dimensional Deformable Mesh Structures for Video Coding, Part II - The Analysis Problem and a Region-Based Coder Employing an Active Mesh Representation" *IEEE Trans. on Circuits and Systems for Video Technology* , vol. 6, no. 6, pp. 647–659, Dec. 1996.

[5] Y. Altunbasak and A. Tekalp, "Occlusion-Adaptive 2-D Mesh Tracking," in *Proc. ICASSP*, Atlanta, Georgia, May 1996, pp. 2108–2111.

[6] M. Orchard, "Predictive Motion-Field Segmentation For Image Sequence Coding," *IEEE Trans. on Circuits and Syst. for Video Tech.*, vol. 3, no. 1, pp. 54–70, Feb. 1994.

[7] G. M. Schuster and A. K. Katsaggelos, "A Video Compression Scheme with Optimal Bit Allocation between Displacement Vector Field and Displaced Frame Difference," in *Proc. ICASSP*, Atlanta, GA, May 1996.

[8] Telenor Research. H.263 encoder/decoder tmn1.5 ver. 1.7. *ftp://bonde.nta.no/pub/tmn*,1996.

[9] K. McConnell, D. Bodson, and R. Schaphorst, *FAX: Digital Facsimile Technology and Applications*, Artech House, 1992.