MOTION COMPENSATED SHAPE ERROR CONCEALMENT

Guido M. Schuster

Abteilung Elektrotechnik
Hochschule Rapperswil, Switzerland
guido.schuster@hsr.ch

Aggelos K. Katsaggelos

Department of Electrical and Computer Engineering
Northwestern University, Evanston, Illinois, USA
aggk@ece.nwu.edu

ABSTRACT

The introduction of Video Objects (VOs) is one of the innovations of MPEG-4. The α-plane of a VO defines its shape at a given instance in time and hence determines the boundary of its texture. In packet-based networks, shape, motion, and texture are subject to loss. In this paper we propose a post-processing shape error concealment technique that uses the motion compensated boundary information of the previously received α-plane. The proposed approach is based on matching received boundary segments in the current frame to the boundary in the previous frame. This matching is achieved by finding a maximally smooth motion vector field. After the current boundary segments are matched to the previous boundary, the missing boundary pieces are reconstructed by motion compensation. Experimental results demonstrating the performance of the proposed motion compensated shape error concealment method, and comparisons with our previously proposed spatial Hermite spline method are presented.

1. INTRODUCTION

One of MPEG-4's most prominent features, as compared to MPEG-1 and 2, is the presentation of video objects (VOs) [1]. Based on this, MPEG-4 video sequences are interpreted and manipulated in terms of visual objects. These objects are defined by their motion, texture and shape. Their particular shape at a given instance in time is described as a binary α-plane. The interpretation of the binary α-plane is that a 1 indicates a pixel that is part of the VO at that given time instance, while a 0 identifies a pixel belonging to the background. In packet-based networks, this shape information is subject to loss.

Various approaches to conceal lost shape information have been proposed. They can be classified into spatial methods, such as in [2, 3] (and references therein) and temporal methods, such as in [4, 5, 6] and the method proposed in this paper. The main advantage of temporal methods is that they can use the shape information of previous frames, while the spatial methods are only allowed to use the shape information contained in the current frame. Therefore one would expect that the temporal methods outperform the spatial ones, but clearly, spatial methods are the only choice when the previous frame does not contain a shape that is related to the current shape, as is, for example the case when a scene change occurs.

The rest of the paper is organized as follows: In section 2, we discuss how the individual boundary segments are defined and identified. In section 3, we propose a matching criterion which is based on a smoothness function of the motion vector set, and present an optimal solution to this matching problem by finding the shortest path through a trellis. In section 4, we discuss how the optimal motion vector set can be used to reconstruct the missing boundary pieces by motion compensating the corresponding parts of the previous boundary. In section 5, we present experimental results and compare the proposed temporal algorithm with the spatial Hermite spline based method [3]. In section 6, we summarize the paper and draw our conclusions.

2. IDENTIFYING THE SEGMENTS

We define the boundary of an α-plane as the collection of points that belong to the background which have at least one 4-connect neighbor that belongs to the object. For example, figure 1 displays the boundary of the 120th α-plane of the bream object. In case of a packet loss, the shape information is missing in the lost macro blocks (MB). This is shown in figure 2, which displays the partially received boundary of the 123rd α-plane (we assume a frame rate of 10 frames/second). As can bee seen in this figure, there are three different boundary segments. We identify these boundary segments by selecting their starting points as boundary points which are 8-connected to the missing MBs. To remove points that are paralleling the missing MBs we then select from these points the ones that have only one 8-connected boundary point neighbor. Hence these points must be starting points of boundary segments. We then follow in an 8-connect fashion these starting points in the direction of a line that could be drawn between the object and the background. This rule ensures that we are collecting the boundary points in a continuous fashion until we run out of boundary points. At that moment, the ordered set that defines this boundary segment is completely selected and we move to the next starting point. This ordering is important, since we will use the difference between consecutive motion vectors along these ordered boundary segments as the vector field smoothness measure later on.

3. MATCHING THE SEGMENTS

The basic concept is, that if we could find the corresponding boundary segments in the previous boundary (which might have been received without loss or might have been concealed), then we could use this information to recover the missing boundary parts in the partially received current boundary. This holds true since the corresponding boundary segments in the previous boundary are connected (i.e., the boundary in the previous frame is complete). If we could motion compensate those boundary segments connecting the corresponding boundary segments in the previous frame, we would recover the missing boundary segments in the current boundary.

0-7803-8554-3/04/$20.00

For the motion compensation, we require a motion vector field as shown in figure 3. We define this field as the set of displacement vectors that indicate for each current boundary segment point, the previous boundary point it came from. For finding a corresponding boundary segment in the previous boundary, we assume that the motion vector field that achieved this matching is smooth. We measure the smoothness as the sum of the squared differences between consecutive motion vectors and look for the optimal motion vector field $v_0^*, \ldots, v_N^*$ that minimizes this smoothness measure. This can be expressed as follows,

$$v_1^*, \ldots, v_N^* = \arg \min_{v_1,\ldots,v_N} \Big(\sum_{j=2}^{N} \parallel v_j - v_{j-1} \parallel^2 + \alpha \sum_{j=1}^{N} \parallel v_j \parallel^2 \Big), \quad (1)$$

where N is the number of points in the current boundary segment and v_j the motion vector displacing a previous boundary point to the point j in the current boundary segment. The motion vector v_j belongs to a set of admissible motion vectors V_j for point j, which consists of all motion vectors in a search window of a given size (± 16 pixels in the reported experiments) around point j, which actually displace a boundary point from the previous boundary to point j of the current boundary. The second term which is weighted by α (which is set equal to 10^{-6} in the reported experiments) is added such that if there are equally smooth motion vector fields, the one which results in the smallest total displacement is selected.

For finding the solution to the above optimization problem, we use a shortest path algorithm through a trellis, similar to the approach presented in [7, 8]. Figure 4 shows a partial trellis with one node and one edge cost. The horizonal axis of the trellis consists of the points on the boundary segment and at each point, all of the admissible motion vectors represent the nodes at this boundary point (points on the vertical axis). Between two consecutive boundary points there exists a fully connected mesh of edges, each with an associated edge cost. The edge cost corresponds to the squared difference between the motion vectors used for point j and point $j-1$, the first term in the cost function in equation 1. Each node also has an associated cost, which is the squared length of the motion vectors weighted by α. This corresponds to the second term in the cost function in equation 1. Clearly, the shortest path through the entire trellis results in the motion vector field that minimizes the cost function defined in equation 1. The obvious choice for such a shortest path algorithm would be the well known Viterbi algorithm. In the next section, we use this motion vector field for motion compensating the missing boundary pieces.

4. MOTION COMPENSATING THE MISSING BOUNDARY PIECES

Figure 3 shows the situation after all segments have been matched. The idea again is to conceal the missing current boundary segments with the motion compensated connecting boundary segments in the previous boundary. For this purpose these connecting boundary segments must be identified. This is achieved by going through the current boundary segments and erasing all boundary points in the previous boundary that the motion vectors are originating from. We label the start and stop points of all these remaining connecting boundary pieces by simply following the motion vectors of the start and stop points of the current boundary segments as shown in figure 5. ("cstart" and "pstart" denote the start points from the current and the previous frame boundaries, respectively; similarly for the "cstop" and "pstop"). We then motion compensate these remaining connecting boundary pieces using the two motion vectors which connect the start and the stop points in the previous and in the current boundaries. Since these motion vectors are usually not the same, we use a linear interpolation of these motion vectors for the motion compensation for the points on the connecting boundary segments in the previous boundary. Hence we move along the connecting boundary segment and create the concealment pixels in the current boundary by displacing the connecting pixels with a motion vector that is calculated in this fashion, $(M-m)/(M-1) * v_{start} + (m-1)/(M-1) * v_{stop}$, where M is the number of pixels in the connecting boundary segment, m the index that counts the pixels in the connecting boundary segment (m starts at 1 and ends at M) and v_{start} and v_{stop} the motion vectors of the start and/or the stop points respectively. The result of this motion compensation is shown in figure 5, where the missing boundary segments are now recovered by the motion compensated connecting boundary segments of the previous boundary. All that has to be done now is to flood fill the interior of the object and this flooded area represents the concealed alpha plane.

5. EXPERIMENTAL RESULTS

Clearly, there are many different ways to quantify the performance of the proposed concealment algorithm. As an absolute measure, we use the total number of incorrectly concealed pixels in the concealed α-plane. If many MBs are lost though, this number tends to be higher than when few MBs are lost. Therefore we use a relative measure, the ratio of the incorrectly concealed pixels divided by the total number of lost pixels. Before we can run tests on the proposed concealment algorithm, we need a model for the packet network, particularly for the packet loss. We assume that the Internet would be used to transport the data packets as RTP/UDP/IP packets. Therefore we use a model studied in [9]. This model is a two state Markov chain, with a loss state and a not-loss state. In the loss state, the packet is lost and in the not-loss state, the packet arrives at its destination. This model allows for correlated losses and one of the parameters of the model is the so called conditional loss probability, *clp*, which is the probability that a packet is lost, given that the previous packet was lost. This is equivalent to the probability that the current state is the loss state, given that the previous state was the loss state. The higher this probability, the more correlated the losses are. The other parameter of the model is the unconditional loss probability, *ulp*, which is the overall probability that a packet is lost, or equivalently, that the Markov chain is in the loss state. For the presented simulation results, we use the parameters given in table 3 of [9]. We selected the third column, which represents a *ulp* of 12% and a *clp* of 27%. This corresponds to a relatively heavily loaded Internet. We also use the parameters *ulp* of 4% with a *clp* of 27% which represents a relatively lightly loaded Internet. Such lighter congestions were not studied in [9], but the trend in table 3 of [9] is such that the *clp* is getting smaller as the *ulp* gets smaller. Not reducing the *clp* but keeping it equal to 27% corresponds therefore to a worst case scenario *clp* for the 4% *ulp* Internet. Next, we need a model of how the MBs are put into RTP/UDP/IP packets. MPEG-4 uses a concept called application layer framing, by defining video packets (VP). In application layer framing, the application layer manages how the data is put into transport packets and the application is also responsible for synchronization. A video packet can contain one or more MBs and it

is the smallest unit of synchronization, since the MBs do not carry an MB address. In the experiments, we assume two packetization modes. In the MB mode, one MB is put into one VP and that VP is put into one RTP/UDP/IP packet. This mode is inefficient, since the overhead of the VP/RTP/UDP/IP packet is now added to each MB. It would be more efficient to put several MBs into one VP/RTP/UDP/IP packet. On the other hand, this mode is good for concealment, since most of the time only one MB is lost and the neighboring MBs are available for the concealment scheme. We use this mode as a baseline to show what the algorithm could achieve in the best packetization case. In the slice mode, we put all MBs of a row (a so called slice) into one VP which in turn is put into one RTP/UDP/IP packet. This mode is clearly more efficient, since the overhead is amortized over several MBs. Unfortunately this efficiency makes the concealment of a lost packet more difficult, since now MBs are lost in slices, which means there is a larger connected area of lost data that needs to be concealed. This model though is a more realistic model than the MB mode for the packetization done in MPEG-4 systems. In an error prone packet-based environment, it is important, that the information in a given video packet is independent of the information in any other video packet. If this is not the case, error propagation occurs as pointed out in [10]. As also suggested in [10], in MPEG-4 shape coding this independence is achieved if the intermode is not used. Hence in our experiments, we only use intramode shape coding. In table 1 we compare the performance of the proposed temporal scheme with a state of the art spatial scheme such as the one we proposed in [3]. We use three sequences (Akiyo, Weather and Bream), two unconditional loss probabilities (4% and 12%) and two different packetization schemes (MB mode and Slice mode). For concealing the current boundary, we use the previous boundary, even when that boundary was also concealed. We assume that the first frame of the sequence arrived without information loss. Under these circumstances table 1 clearly shows the performance gain the proposed scheme achieves over the Hermite spline method. This gain is especially large when slices are lost, since then the spatial method has very little information surrounding the lost boundary and hence the performance deteriorates.

Experiment	A.	W.	B.	Mean	Δ Hermite
MB4%MC	0.4%	1.3%	2.3%	1.4%	1.4%
MB12%MC	0.5%	1.5%	2.4%	1.5%	1.7%
Slice4%MC	0.6%	1.7%	3.3%	1.9%	7.0%
Slice12%MC	0.7%	1.9%	3.3%	2.0%	7.2%

Table 1. Average number of incorrectly concealed pixels divided by the average total number of lost pixels. The averages are taken over 100 realizations of every third frame (10 frames per second) of the 300 frames QCIF sequences Akiyo (A.), Weather (W.), and Bream (B.) for each packetization scheme and unconditional loss probability. The column Δ Hermite shows the average improvement of the proposed temporal method to the spatial Hermite spline method proposed in [3].

6. SUMMARY AND CONCLUSIONS

In this paper we proposed a novel temporal shape error concealment technique which is based on matching the received boundary segments with the previously received (or concealed) boundary. The matching is performed by finding a maximally smooth motion vector field using a shortest path through a trellis. The missing boundary pieces are then recovered by motion compensating the connection boundary segments in the previous boundary using the maximally smooth motion vector field. In addition to being able to conceal a missing boundary piece well, when there is a corresponding connecting boundary segment in the past, this method also results in a matching value, which is the cost along the optimal path. One can now build a system that first tries to use the proposed temporal shape concealment method, but which switches to the spatial method we proposed in [3] when the matching value becomes too large for a particular missing boundary piece. That would indicate, that a smooth motion vector field cannot be found and hence the underlying assumption that the previous boundary is similar to the current boundary might be violated. In these cases, the spatial method might perform better and hence a hybrid system might result in the best performance.

7. REFERENCES

[1] ISO/IEC JTC1/ SC29/ WG11 N2502, "Text for ISO/IEC FDIS 14496-2 Visual," November 1998.

[2] G. M. Schuster, X. Li, and A. K. Katsaggelos, "Spline-based boundary loss concealment," in *Proceedings of the International Conference on Image Processing*, Barcelona, Spain, September 2003.

[3] G. M. Schuster, X. Li, and A. K. Katsaggelos, "Shape error concealment using Hermite splines," *accepted for publication in the IEEE Transactions on Image Processing*.

[4] M-J. Chen, C-C Cho, and M-C. Chi, "Spatial and temporal error concealment algorithms of shape information for MPEG-4 video," in *IEEE International Conference on Consumer Electronics*, June 2002.

[5] M. R. Frater, W.S. Lee, M. Pickering, and J.F. Arnold, "Error concealment of arbitrarily shaped video objects," in *Proceedings of the International Conference on Image Processing*, Chicago, Illinois, USA, October 1998, vol. 3, pp. 507–511.

[6] P. Salama and C. Huang, "Error concealemt for shape coding," in *Proceedings of the International Conference on Image Processing*, Rochester, New York, USA, September 2002, vol. 2, pp. 701–704.

[7] D. Geiger, A. Gupta, L. A. Costa, and J. Vlontzos, "Dynamic programming for detecting, tracking, and matching deformable contours," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 3, pp. 294–302, March 1995.

[8] B. Tom, S.N. Efstratiadis and A.K. Katsaggelos, "Motion estimation of skeletonized angiographic images using elastic registration," *IEEE Transactions on Medical Imaging*, vol. 13, no. 3, pp. 450–460, September 1994.

[9] J.C. Bolot, "Characterizing end-to-end packet delay and loss in the Internet," *Journal of High-Speed Networks*, vol. 2, no. 3, pp. 305–323, December 1993.

[10] N. Brady, "MPEG-4 standardized methods for the compression of arbitrarily shaped video objects," *IEEE Transactions of Circuit and Systems for Video Technology*, vol. 9, no. 8, pp. 1170–1189, December 1999.

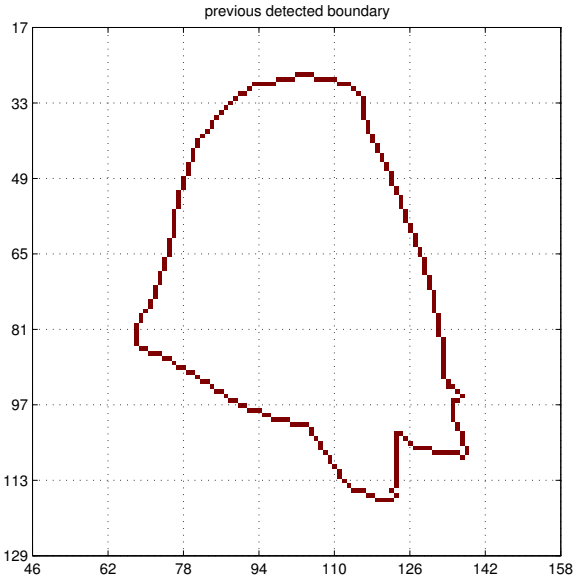

Fig. 1. Boundary of α-plane of frame 120 of the bream object

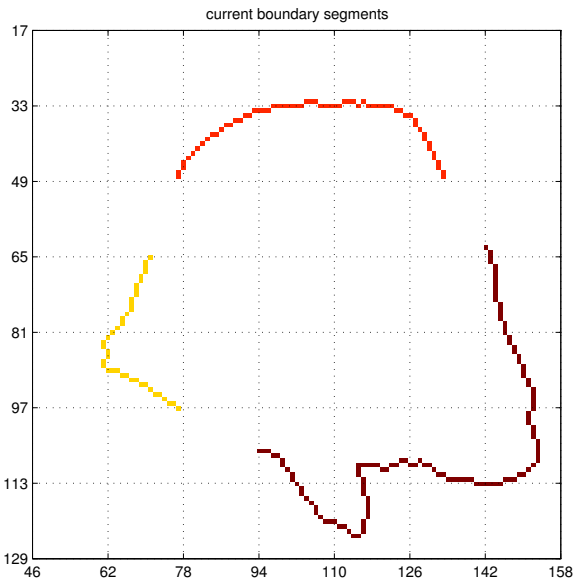

Fig. 2. Partially received boundary of α-plane of frame 123

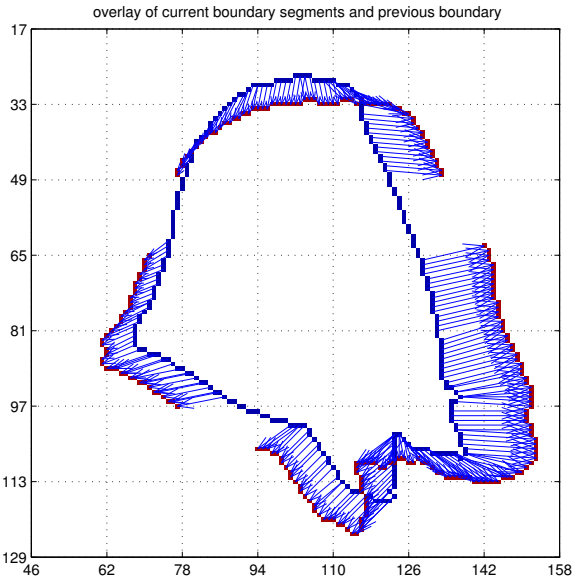

Fig. 3. Displayed motion vector field for the boundary segments

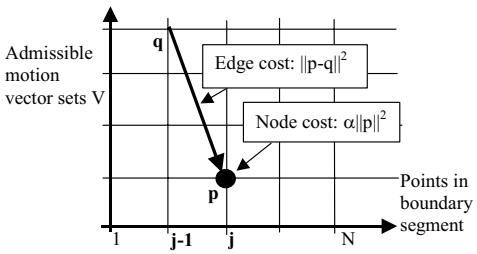

Fig. 4. Partial trellis for finding the optimal motion field

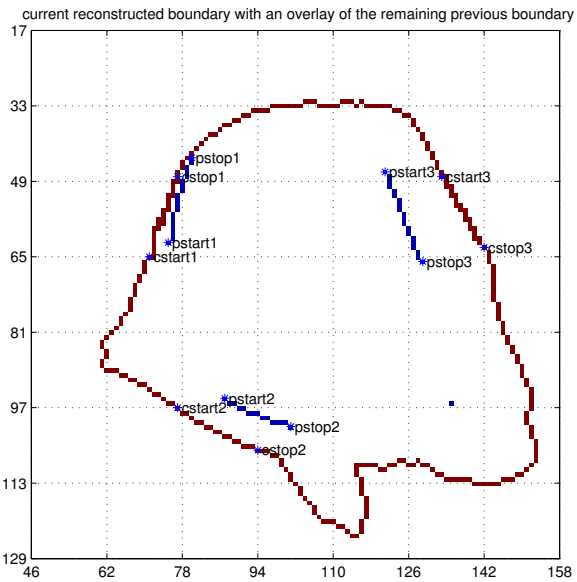

Fig. 5. Concealed boundary of α-plane 123 using the correct boundary segments from α-plane 120