SPLINE-BASED BOUNDARY LOSS CONCEALMENT

Guido M. Schuster

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

Xiaohuan Li and Aggelos K. Katsaggelos

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

ABSTRACT

Object-based video coding requires the transmission of the object shape. This shape is sent as a binary α-plane. In lossy packet-based networks, such as the Internet, this information has a non-negligible probability of not arriving at the receiver, and hence its loss needs to be concealed. In this paper we propose a shape concealment technique utilizing Hermite splines. The algorithm has the following steps: (I) the received boundary is detected and the lost boundary parts are grouped using the packet loss pattern. (II) for each of these lost boundary parts, the received boundary points that border the area of the lost boundary parts are collected. These boundary points are then modelled by a second order Hermite spline. This model is subsequently used to match the velocity along the received boundary with the velocity of the concealing cubic Hermite spline. (III) since in most cases there are more than one concealing splines we draw every spline combination that does not result in an intersection and keep all possible results until the end. (IV) if there are more than one possible solutions we select the one that results in one overall closed non-intersecting boundary and fill the interior of the boundary to get the concealed α-plane. Experimental results which demonstrate and compare the performance of the proposed concealment method are given at the end of the paper.

1. INTRODUCTION

In this paper we concentrate on intra frame shape concealment and extend the work initially presented in [1]. For this problem, a maximum *a posteriori* probability (MAP) estimator was proposed in [2]. It uses the spatial redundancy in transmitted shape information on a statistical basis. As we will show in the experimental section, our boundary-based approach significantly outperforms this statistical approach. In [3, 4, 1] the binary α-plane is interpreted as a boundary that is missing some pieces. In [4] the concealment is achieved using an iterative line bending method, which does not offer the same degree of smoothness and control that we have in the proposed spline-based approach. In [3] the missing boundary pieces are estimated using a second order Bezier curve that matched the estimated derivatives $\frac{dy}{dx}$ at both ends. There are several problems with this approach. First, second order Bezier curves must be convex. In our experience this resulted often in incorrect approximations. Second, the method fails in the common case when more than one splines is needed to conceal the lost boundary area. Third, estimating derivatives $\frac{dy}{dx}$ is numerically not robust.

We assume that the α-plane describes a closed non-intersecting boundary and that only macro blocks (MB) can be lost. Depending on the packetization scheme one packet loss can result in several MBs losses. It is common to put a line of MBs (slice) into a packet and hence an entire row of α-plane data can be lost.

2. SINGLE LOST MACRO BLOCK CASE

First we simplify the problem by only looking at the case when only one MB is lost with only one boundary line. Consider Fig. 1a. First we detect the part of the boundary that is not missing (see Fig. 1b). Now we need to detect boundary points that touch the missing MB, the "connecting points". We collect these points in a clockwise fashion which will become important later. Then we collect for each connecting point a number of (20 in our experiments) "associated points". Associated points belong to the boundary and are 4 connected with a connecting point or another associated point (see Fig. 2a). The associated points set is an ordered set where the further away the point is from the missing MB, the farther back it is in the ordered set. Now we want to find for each set a mathematical model which will be used to estimate parameters useful for drawing the concealment spline between the two connecting points. We are planning to conceal the missing boundary using a cubic Hermite spline (see Fig. 2b), as defined below,

$$m(s) = \begin{bmatrix} m_x(s) \\ m_y(s) \end{bmatrix} = \begin{bmatrix} a_x s^3 + b_x s^2 + c_x s + d_x \\ a_y s^3 + b_y s^2 + c_y s + d_y \end{bmatrix}, \quad (1)$$

which has the interesting property that the parameters can be selected so that one can exactly match a start and a stop point ($m(start)$, $m(stop)$) and a start and a stop velocity ($mv(start)$, $mv(stop)$) as derived below:

$$mv(s) = \begin{bmatrix} \frac{dm_x(s)}{ds} \\ \frac{dm_y(s)}{ds} \end{bmatrix} = \begin{bmatrix} 3a_x s^2 + 2b_x s + c_x \\ 3a_y s^2 + 2b_y s + c_y \end{bmatrix}. \quad (2)$$

Clearly the goal of the mathematical model for the associated points set must be accurate velocity vectors that can be used for the cubic Hermite concealment spline. We decided to use a second order Hermite spline of the following form:

$$l(s) = \begin{bmatrix} l_x(s) \\ l_y(s) \end{bmatrix} = \begin{bmatrix} e_x s^2 + f_x s + g_x \\ e_y s^2 + f_y s + g_y \end{bmatrix}, \quad (3)$$

where again the velocity vector can be derived as follows:

$$lv(s) = \begin{bmatrix} \frac{dl_x(s)}{ds} \\ \frac{dl_y(s)}{ds} \end{bmatrix} = \begin{bmatrix} 2e_x s + f_x \\ 2e_y s + f_y \end{bmatrix}. \quad (4)$$

The naming of the splines is such that l stands for left, m stands for middle and r that stands for the right spline. The left spline (2^{nd}

order) is used to model the left associated points set, the right (2^{nd} order) to model the right associated points set and the middle (3^{rd} order) to connect the left and right splines smoothly by matching start and end point and velocities at those points.

We want to calculate the left and right spline parameters such that the splines go through the connecting points and estimate the x and y coordinates of the associated points in the MSE sense. To achieve this, we use the fact that the associated point set is ordered and that the distance between two consecutive points is equal to 1 or 1.41 pixels. This can be written in matrix form for the x-dimension as follows (an equivalent expression holds for the y-dimension):

$$\begin{bmatrix} s_1^2 & s_1 \\ s_2^2 & s_2 \\ s_3^2 & s_3 \\ \vdots & \vdots \\ s_N^2 & s_N \end{bmatrix} \cdot \begin{bmatrix} e_x \\ f_x \end{bmatrix} = \begin{bmatrix} x_1 - x_0 \\ x_2 - x_0 \\ x_3 - x_0 \\ \vdots \\ x_N - x_0 \end{bmatrix}, \quad (5)$$

where (x_0,y_0) is the connecting point, (x_N,y_N) is the last point in the associated points set, s_1 is the value of s along the associated points set at (x_1,y_1), s_2 is the value of s along the associated points set at (x_2,y_2), and so on. The MSE estimates ($[\hat{e}_x, \hat{f}_x]^T$ and $[\hat{e}_y, \hat{f}_y]^T$) for the above parameters can now be found using the pseudo inverse. Having found these estimates we can express an estimate of the velocity at the point $s = 0$, which is at the connection point, as follows: $\hat{lv}(s = 0) = [\hat{f}_x, \hat{f}_y]^T$. Figure 2a shows the two second-order Hermite splines (the left and the right) and also the estimated velocities at the connecting points.

As discussed above we want to use a cubic Hermite spline for concealing the missing boundary. The main reason for this is that matched velocities along the concatenated splines (left, middle, right) results in visually pleasing curves. Furthermore by matching the connecting points and the velocities at the connecting points, the three concatenated splines result in a continuous and differentiable curve. Therefore we want to calculate the parameters of the cubic Hermite spline such that the following equation holds in the x-dimension (and equivalently in the y-dimension):

$$\begin{bmatrix} 0 & 0 & 0 & 1 \\ s_m^3 & s_m^2 & s_m & 1 \\ 0 & 0 & 1 & 0 \\ 3s_m^2 & 2s_m & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} a_x \\ b_x \\ c_x \\ d_x \end{bmatrix} = \begin{bmatrix} l_x(0) \\ r_x(0) \\ -\frac{dl_x}{ds}|_{s=0} \\ \frac{dr_x}{ds}|_{s=0} \end{bmatrix}, \quad (6)$$

where s_m is the distance along the cubic Hermite spline in pixels. The above equation needs to be solved for the unknown parameters, which can be done in a straightforward way. Now the cubic Hermite spline can be drawn (Fig. 2b). What is left to do is connect the detected boundary with this concealment spline and fill in the interior (see Fig. 1c and Fig. 1d).

3. MULTIPLE LOST MACRO BLOCKS CASE

As can be seen in Fig. 3a, for this case, we need to group lost MBs together since 4 connected MBs create a lost group of MBs that have to be concealed together. The proposed algorithm detects these groups of 4 connected lost MBs and treats them independently one group at the time. Clearly if the group would only have 2 connecting points we could simply use the method discussed so far. The problem arises when there are more than 2 connecting points. Since every line that goes into the group must come out of it there is always an even number N of connecting points. Since a spline connecting point A with point B is the same as a spline connecting point B with point A, the total number K of possible splines is $K = \frac{N!}{2^{N/2}(N/2)!}$. Based on our initial assumption that the boundary is not self-intersecting, we can exclude all combinations that intersect. We use the fact that if straight-line connections intersect, then spline connections must also intersect.

Based on this, we want to exclude all possible combinations of connecting points that result in straight-line intersection. We derive this recursively, and we start with 2 connecting points that are shown as the first circle with two connecting points and one line in Fig. 5. Note that we use as a representative group of lost MBs a circle and use the symbol GX, where X is the number of connecting points. Since we collect the connecting points in an ordered (in our case clockwise) direction, the effective shape of the group does not matter hence we use a circle for this derivation. For the 2 connecting points, there is clearly only one-way to connect them, which does not intersect. For 4 connecting points this is more interesting, since there are now 3 ways to connect them but, by inspection of the second row in Fig. 5, only 2 of them do not intersect. For 6 connecting points there are 15 ways to connect them but by inspection of the third row in Fig. 5, only 5 do not intersect. For 8 connecting points there are already 105 ways to connect them but only 14 of them do not intersect. This is not shown in Fig. 5, instead we use the trees drawn in Fig. 6 to explain how we got to the number 14. The G4 tree is made up of 2 branches of G2s. This is based on the idea that the dashed line connecting the uppermost point in the circle with another point can either be connected to the point to the left of the uppermost point or to the point to the right of the uppermost point. If it would be connected to the lowermost point, then the remaining connection line would have to intersect this line. Hence we have two non-intersecting branches of the G4 tree. Now if it is connected to its neighbor in the counter clockwise direction, then we are left with 2 points on the right, which form a G2 group. If it is connected to its neighbor in the clockwise direction, then we have left 2 points on the left, which again form a G2 group. Hence a G4 group can through this tree be represented recursively in terms of G2 groups. Since there is only one valid combination per G2 group and the G4 group is made up of 2 G2 groups, there are then 2 valid G4 combinations for lines that do not intersect. Clearly this is the same result we got in the second row of Fig. 5. This idea can be extended to the G6 group below the G4 group in Fig. 6. The dashed line creates now 3 branches, where the leftmost branch is a G4 group, the middle branch is a product of 2 G2 groups and the rightmost branch is another G4 group. Hence the number of combinations for a G6 group is 2+1*1+2 = 5, which again we can also see in row 3 of Fig. 5. Clearly this recursive argument can be also used for a G8 and a G10 group, which is done in Fig. 6. The G8 group results in 14 combinations and the G10 group in 42. In practice such many connecting points per group is rather unlikely but the presented recursive procedure can not only be used to count the number of combinations, but also to draw them. We need to draw all these combinations, since the fact that a straight-line does not result in an intersection on a circle does not guarantee that a spline in an arbitrarily shaped group of 4 connect lost MBs does not intersect. We test this by drawing the splines of all these combinations and only keeping the ones which do not intersect each other. This is shown in Fig. 3b and Fig. 3c . Note in Fig. 3b and Fig. 3c that the concealment spline for the upper group is unique, since it is a G2 group, while for the lower group, 2 possible solutions exist since

it is a G4 group and the splines did not intersect. As mentioned before, we collect each non-intersecting solution per group and the total combinations of all these solutions are then considered at the end to select a final reconstructed boundary. In our example we have two groups, of which the first one created one solution and the second one two. Therefore the total number of solutions is 1*2=2 as seen in Fig. 3b and Fig. 3c.

After concealment solutions for all groups have been generated, a final solution needs to be selected. We use the assumption that the boundary is a closed curve and check all of the solutions which in our example excludes Fig. 3c. Since it is theoretically possible that there is more than one closed boundary among the feasible solutions, we then select the shortest one. What is left to do is fill the interior of the final reconstructed boundary with ones and set everything else to zero which results in the reconstructed α-plane, shown in Fig. 3d.

4. EXPERIMENTAL RESULTS

We measure the performance of the proposed algorithm by counting the number of incorrect pixels in the concealed α-plane. We use an i.i.d. drop rate as the model for the packet channel and for the first set of experiments, we assume that one MB is put into one IP packet. Column "MB" in Table 1 shows the performance for this packetization scheme of the proposed concealment system "Hermite Spline", as well as that of two other schemes proposed in [2], "Median" and "MAP". The Median method replaces a missing pixel by the median of a neighborhood. We use the same 18-pixel neighborhood as proposed in [2]. This Median filtering is repeated iteratively until the concealed α-plane does not change anymore. The MAP method is motivated as a maximum *a posteriori* estimator for the missing pixels of a binary image that uses a Markov random field as image model. In [2] the model is selected such that the resulting estimator is simply a weighted version of a median filter. The weighting is such that boundary lines in the four neighbors that would extend into the missing MB are more likely to result. In other words, the pixels in the missing MB are selected such that dominant edges are conserved. The results reported in column "MB" in Table 1 are the averages of 1000 realizations for each drop rate for the first frame of the video object "bream". From this it becomes clear that the proposed Hermite spline method outperforms the Median and the MAP method significantly. In the second part of this experimental section, we repeat the experiment but we are using a different packetization scheme. We now put a row of MBs (a so called "slice") of the video object in one IP packet. The main difference is that when a packet is lost now, we loose an entire row of MBs. The results of this experiment (using again 1000 realization for each drop rate) are reported in column "Slice" in Table 1 and the proposed method is again the most successful. To show the subjective effects of the three methods, we also show in Fig. 4a a sample received α-plane and in Fig. 4b, Fig. 4c and Fig. 4d the resulting three reconstructed α-planes. Figure 4 clearly shows that the Hermite spline method performs also subjectively the best.

5. SUMMARY AND CONCLUSIONS

In this paper we propose a novel shape error concealment method based on Hermite splines. We use second order Hermite splines to estimate the velocities at the connecting points and a cubic Hermite spline matching these two velocities to conceal the lost boundary segment. We propose a solution when there are more than two connecting points per group using a recursive generation of all possible non-intersecting solutions. Finally we collect all possible solutions that result from the local decisions made during the concealment of the groups and make a final decision on the reconstructed boundary based on the assumption that the boundary is a closed non-intersecting curve.

The proposed shape concealment method interprets shape information in terms of a closed non-intersecting boundary. It explores the known shape information on a relatively high level (velocity along a curve), and thus making better use of the available information than the statistical methods. The Hermite spline method significantly outperforms the Median and the MAP method in objective tests. What is even more important is the improvement in subjective quality which results from the built in enforcement of the boundary smoothness.

6. REFERENCES

[1] X. Li, G. M. Schuster, and A. K. Katsaggelos, "Curve-fitting algorithms for shape error concealment," in *Proceedings of the 5th Nordic Signal Processing Symposium*, October 2002, pp. 4–7.

[2] S.Shirani, B. Erol, and F. Kossentini, "A concealment method for shape information in MPEG-4 coded video sequences," *IEEE Transactions on Multimedia*, vol. 2, no. 3, pp. 185–190, September 2000.

[3] 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.

[4] X. Li, G. M. Schuster, and A. K. Katsaggelos, "A recursive shape error concealment algorithm," in *Proceedings of the International Conference on Image Processing*, Rochester, New York, USA, September 2002.

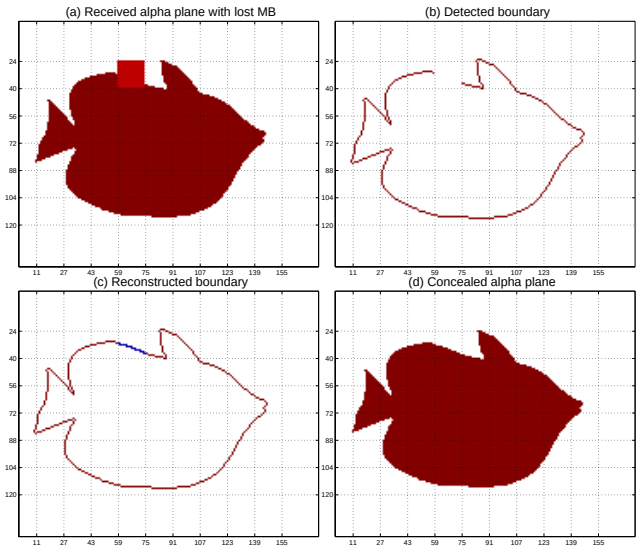

Fig. 1. α-plane concealment for one missing MB

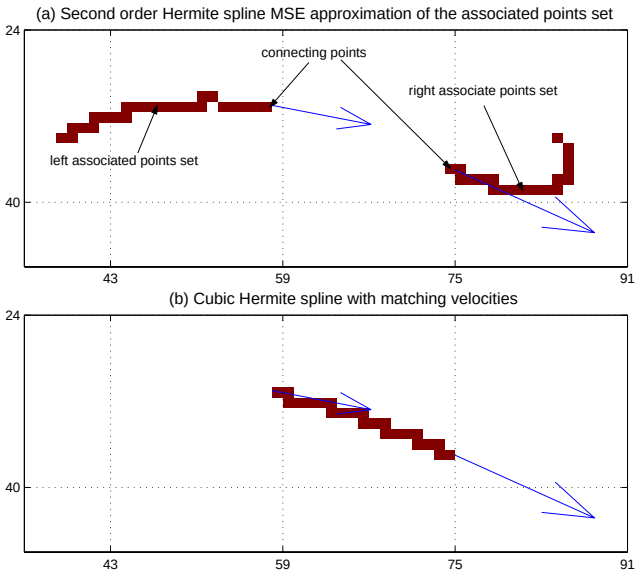

Fig. 2. Matching connecting points and velocities of the splines

Drop rate	Hermite Spline		MAP		Median	
	MB	Slice	MB	Slice	MB	Slice
2%	4	3	6	5	7	4
8%	32	33	40	63	49	60
16%	77	106	102	221	122	195
24%	126	262	199	464	212	411
Average	60	101	87	188	97	168

Table 1. Average number of error pixel. MB: i.i.d. MB loss; Slice: i.i.d. Slice loss

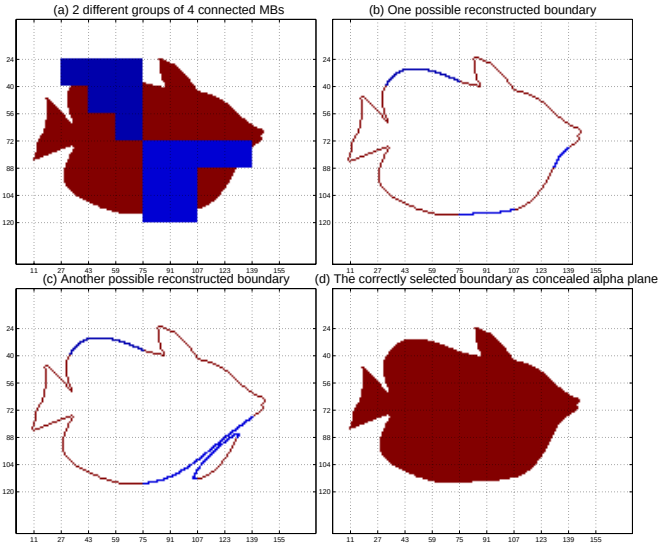

Fig. 3. α-plane concealment for 2 groups of missing MBs

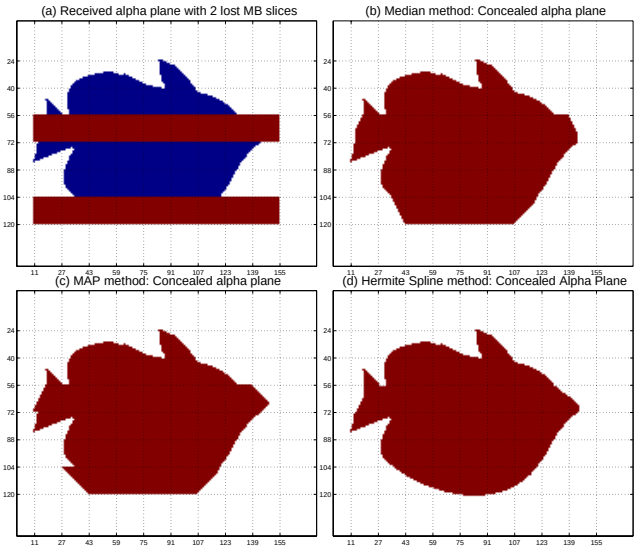

Fig. 4. Visual comparison of the three concealment methods

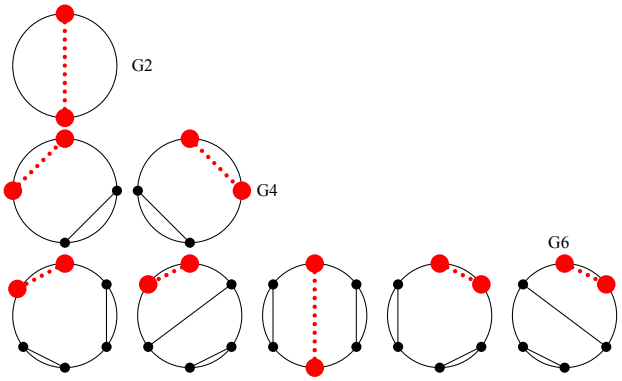

Fig. 5. Different groups GX with different numbers of connecting points (X=2, 4 or 6) and all possible non-intersecting combinations

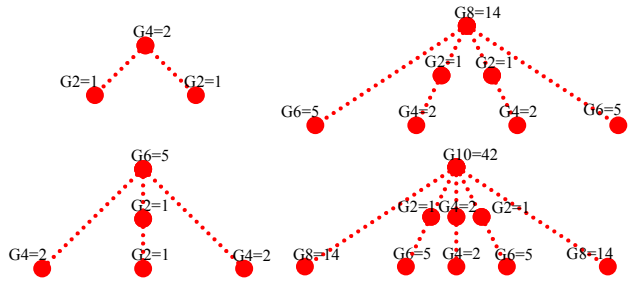

Fig. 6. Recursive calculation of the number of non-intersecting combinations for a given group