Decision trees for error concealment in video decoding

Song Cen, Pamela Cosman, Faramarz Azadegan[†]

ECE Dept., Univ. of California at San Diego, La Jolla, CA, 92093-0407
[†]Rockwell Semiconductor Systems, A2-North, 9868 Scranton Road, San Diego, CA 92122
{scen,pcosman}@code.ucsd.edu faramarz.azadegan@rss.rockwell.com

Abstract: When macroblocks are lost in an MPEG decoder, the decoder can try to conceal the error by estimating or interpolating the missing area. Many different methods for this type of concealment have been proposed, operating in the spatial, frequency, or temporal domains, or some hybrid combination of them. In this paper, we show how the use of a decision tree that can adaptively choose among several different error concealment methods can outperform each single method. We also propose two promising new methods for temporal error concealment.

1 Introduction

When video signals are compressed and transmitted over unreliable channels, some strategy for error control or concealment must be employed. Possible strategies include forward error correction added at the encoder, early re-synchronization and post-processing methods employed by the decoder, and interactive requests for repeated data, involving both encoder and decoder. In this paper, we are concerned with the set of post-processing methods that can be employed by the decoder. We consider the single-layer case where coding modes, motion vectors, quantized DCT coefficients, and other information about macroblocks are all sent with the same priority. When errors strike the bitstream, we assume the decoder loses all information about that slice up to the next resynchronization point. In the absence of block interleaving, a horizontal swath of macroblocks is missing, and the decoder's postprocessing methods seek to conceal this from the viewer. A variety of alternatives exists: spatial domain interpolation, estimation in the frequency domain, temporal concealment involving locating appropriate blocks in a reference frame. Which method is best depends on the characteristics of the missing block, its neighbors, and the overall frame. In this work, we design a decision tree which can examine these characteristics and choose among several error concealment (EC) methods. The decision tree provides lower distortion in the concealed blocks than does the use of any single fixed concealment method among those tested. We also present two new methods for temporal EC, one which attempts to estimate the parameters of a global camera pan,

and one which splits the missing macroblock (MB) in half for separate use of motion vectors. The paper is organized as follows. In the remainder of the introduction, we review existing EC methods. In section 2 we discuss the CART^{TM} algorithm for classification tree design. We present in section 3 our new EC methods, as well as other ones available to our decision tree. Error simulations and results are in section 4.

Prior work on error concealment: A good recent review of EC methods for video compression is provided in [5]. The post-processing error concealment methods with which we are concerned can be divided into three main groups.

Frequency concealment (FC): In frequency concealment, some low-order DCT coefficients of the missing blocks are estimated using either the corresponding DCT coefficient of neighboring blocks, or using the neighbor's DC values. These methods cannot be used to estimate high-frequency coefficients.

Spatial concealment (SC): One can interpolate directly in the spatial domain. If one had neighboring blocks on all 4 sides, then each pixel in the missing MB could be reconstructed, for example, by using bilinear interpolation from the four nearest pixels. If MBs are available only above and below the missing MB, then one can do one-dimensional linear interpolation from the two nearest neighboring pixels. Other strategies exist, for example, directional interpolation that seeks to preserve edges [4]. In general, spatial concealment methods are the most complex, since a computation must be done for each pixel.

Temporal concealment (TC): One can attempt to reconstruct the motion vector of the lost MB, and use the referenced block for concealment. If the estimation of the motion vector (MV) is inaccurate, the block obtained will have distracting artifacts at the boundaries with its neighbors. The MV can be estimated using, for example, the average or median of the MVs from MBs above, below, and diagonal. Alternatively, each neighboring macroblock's MV can suggest a candidate reference block; the candidates are all checked to see which one produces the best match for the boundary pixels.

A variety of hybrid algorithms have been proposed. For example, in TC the referenced block can be further improved by spatial smoothing at its edges, to make it conform to the neighbors, at the expense of additional complexity. In [6], a MB is estimated by satisfying a weighted combination of spatial smoothness and temporal smoothness constraints.

Adaptive concealment methods: Often, error concealment involves using a single fixed method for reconstructing any MB which is lost, however, a few adaptive EC methods have been proposed. In [6], the coding mode and block loss patterns are clustered into four groups, and the weighting between the spatial and temporal smoothness constraints depends on the group. A further level of adaptivity appears in [3] and [2] where small fixed decision trees are used. In [3], temporal concealment is used for most blocks. However, a scene detector (which looks at the mean and variance of the MVs in a frame, as well as at the number of intra-coded blocks) attempts to detect scene changes and irregular motion, in which case TC is likely to do poorly. In that case, a decision is made next on complexity. If there are too many

lost blocks, then spatial concealment cannot be used in real time for this frame. If the complexity constraint is satisfied, then a last choice is made between FC and SC based on a 5-category classification of the overall activity and color requirements of the video. Similar criteria are used in [2]. A single-split decision tree (two terminal nodes) is proposed for P and B frames. If the variance of the MVs in the frame does not exceed a threshold, and also the number of intra-coded MBs does not exceed its threshold, then SC is used, otherwise TC is used. For I-frames, a tree with two splits (3 terminal nodes) is used to choose between SC and FC.

The work discussed here is in a similar spirit, in that the EC method is chosen by a decision tree which looks at the context of the missing MB. We consider larger trees where the splits are chosen based on a training sequence of data, and thus where the trees can be tailored for particular sequences, or for an individual GOP (group of pictures). The decision trees would in this latter case need to be included as side information with each GOP. There are other approaches in which enhanced capability for error concealment comes at the expense of having to use side information. The MPEG-2 standard allows the use of concealment motion vectors (CMVs) to be transmitted for all intra-coded MBs; this leads to enhanced EC, but the bits employed for the CMVs detract from the source coding rate. Similarly, the use of more slices per horizontal strip costs more bits but allows faster re-synchronization. The transmission of EC decision trees provides yet an additional trade-off between the bit-rate for side information and the EC advantage under noisy conditions.

2 Classification tree design

The CART^{TM} algorithm for designing classification and regression trees has its origins in a 1984 monograph by Breiman, Friedman, Olshen, and Stone [1]. The general paradigm is as follows. Let x be a vector of measurements, and let C be a set of classes: $C = 1, 2, ...J$. In our case, the vector x is a set of measurements associated with a missing MB. For example, x will include information on the MVs for the MBs above and below, and on the number of non-zero AC coefficients for the MBs above and below. C is a set of possible EC methods. A classifier is a function $d(x)$ which assigns to every vector x a class from C; the classifier can be thought of as a partition of the measurement space into J disjoint subsets.

A learning sample or training sequence $\mathcal{L}$ consists of data (x_1, j_1), (x_2, j_2), ..., (x_N, j_N) on N cases where the class is known, that is, N macroblocks for which the best EC method is known. To form the training sequence, we can take each MB in the sequence, assume it is lost, reconstruct it using each of the EC methods, and see which one yields the smallest MSE. The measurement vector can include both ordinal and categorical variables. For example, the coding mode of the top MB is a categorical variable; its MV (if it has one) is an ordinal variable. The root node of the tree contains all the N training cases; a mix of best EC methods is present for the data in this root node. The goal of CART is to successively subdivide the training set using binary splits in such a way that the data associated with the terminal nodes of the tree do not have a mix of best EC methods; rather each node should be as "pure"

as possible. To accomplish this we must be able to measure the purity or impurity of a set of data; we used the Gini index of diversity [1].

During the design of the classification tree, we consider, for each terminal node of the tree, a standard set of possible splits of the data in that node. In the standard set, each split depends on the value of only a single variable. For each ordered variable x_m, we include all splitting questions of the form "Is $x_m \leq c$?" If x_m is categorical, taking values in $B = \{b_1, b_2, \ldots, b_L\}$, then we include all questions of the form: "Is $x_m \in S$?" as S ranges over all subsets of B.

There is a finite number of distinct splits, since the learning sample contains only N distinct points. For each single variable, we find the split which provides the greatest decrease in node impurity. We compare all of these, and find the best overall split of the data. A class assignment rule assigns a class $j \in \{1, 2, \ldots, J\}$ to every terminal node t. The class assigned to node t is denoted $j(t)$. A simple rule is to assign the most popular class for each terminal node; this is called the plurality rule.

Given a classifier d, we denote by $R^*(d)$ the "true misclassification rate" of d. There are three standard ways of estimating $R^*(d)$: cross-validation, test sample, and the resubstitution estimate (in which the classifier is designed using $\mathcal{L}$, and then the samples in $\mathcal{L}$ are run through the classifier to see how many of them get misclassified). One challenging question with the CART algorithm is how to grow the right-sized tree. On a learning sample, growing a larger and larger tree will continue to reduce the resubstitution estimate of misclassification error, until such time as each terminal node is completely pure and the resubstitution estimate is zero. However, such a tree is unlikely to perform well on other data. It is common to use either a test sample or cross-validation to handle this.

3 Error concealment methods

We considered the eight different EC methods listed in Table 1. The first column lists the name by which we reference the methods; the second column lists the types of frames for which it is used, and the last column summarizes how it works.

The spatial interpolation method works by linearly interpolating within a vertical column from the two nearest pixels in the adjacent top and bottom MBs. In frequency interpolation, the lowest 9 DCT coefficients (for each of the 6 blocks composing the missing MB as in MPEG-2 main profile) are estimated by a weighted average of the corresponding lowest 9 DCT coefficients of the blocks above and below. Both spatial and frequency interpolation can be used for any type of frame. However, this frequency interpolation requires the presence of the neighbor's DCT coefficients, thus both top and bottom MBs must be intra coded, which normally happens less than 5% of the time for P frames and 0.5% for B frames. So FC is not used as a concealment method for P and B frames.

There are five different methods which depend on the presence of other motion vectors in the frame, and so are not immediately applicable to I frames. Two of these are new, and are denoted "panning" and "top/botMV."

Panning: When a camera is panning, many MBs in a scene should have similar MVs

Method Name	Ftype	How it works
spatial	I,P,B	interpolate linearly from boundary pixels in top/bot MBs
frequeny	I	weighted average of first 9 DCT coefficients of top/bot MBs
panning	I,P,B	use the camera panning motion vector
top/botMV	P,B	use top MV for top 8×16 sub-MB, use bottom MV for bottom 1/2
averageMV	P,B	use the average motion vectors of top and bottom MBs
useonlyMV	P,B	top or bot MB is I-coded ⇒ use the only MV available
spat+onlyMV	P,B	use only available MV for nearest half, spatial interp. for rest
copyPmb	I	copy co-sited MB from previous P frame if it's I-coded or has MV=0

Table 1: Our set of available methods for error concealment

that correspond to the true panning motion. Individual MBs might have different MVs for a variety of reasons such as noise and object motion. Also, MBs in a background area of low variation might find any number of excellent matching blocks in the reference frame; the MV corresponding to the actual pan might not be chosen as the best if the area is homogeneous. If the global panning MV can be estimated, it might constitute a better estimate for EC purposes than the MVs of the neighboring blocks. We estimate the panning MV by putting all non-zero MVs for the current frame into a histogram with 47×47 bins. The histogram bin with the largest count is assumed to represent the global pan. This was found to give better results compared to just averaging together all non-zero MVs for the frame, since averaging includes objects which may be moving contrary to the global panning direction. This method can be applied to I frames by using the panning parameters estimated from the previous P frame.

Top/botMV: In a P or B frame, if both the top and bottom MBs have MVs associated with them, we can estimate the MV for the missing MB by taking the average of the ones above and below (averageMV method). If the MVs above and below are very different in magnitude or direction from each other, it might not make sense to average them. Instead, we might wish to use the MV for the block above for the top half of the missing MB (8×16 sub-macroblock for luminance and 4×8 sub-block for chrominance), and use the MV for the block below for the bottom half (top/botMV method). This method performs very well, but it has the disadvantage that since we are not providing one single MV for the missing MB, we cannot consider the error concealer as a front-end to a standard MPEG-2 decoder. If exactly one of the top or bottom MBs is intra-coded, then we have only one motion vector to go by. We might want to use this one as the MV for the entire missing MB (useonlyMV) or we might want to use it only for the half MB to which it is closer, using spatial interpolation for the other half (spat+onlyMV).

The last method in Table 1 was employed only for I frames. If the co-sited MB in the previous P frame was intra-coded, or had zero motion vector, then that MB might be useful directly as a replacement for the missing MB. If, however, the co-sited MB had a non-zero motion vector, then likely it is not an accurate reconstruction of the current missing I frame MB. This method is referred to as copyPmb.

4 Experiments and results

The CART algorithm was provided initially with a large set of input variables that attempted to describe the spatial and temporal context of a missing MB. For missing I, P, and B frame MBs, there were 29, 62, and 101 input parameters, respectively. Only a small subset of these were actually used by the CART algorithm to divide the data sets; most of the input parameters were found to be similar to (but not as good as) other parameters in the set in their ability to purify the tree nodes. Some of these input parameters are listed in Table 2. The table includes the name by which we reference the parameter, the types of frame for which it is used, and whether it is an ordinal (O) or categorical (C) variable. For categorical variables, the number of categories is listed. The last column describes what the parameter represents.

Parameters related to macroblock position			
MBROW	I,P,B	O	vertical position of lost MB
MBCOL	I,P,B	O	horizontal position of lost MB
GOPINDEX	I,P,B	O	GOP number of the lost MB
PICINDEX	P,B	O	frame number (among the same type) of the lost MB
Parameters related to object motion:			
MBTYPETB	P	C(9)	describes top/bot MBs intra/zeroMV/nonzeroMV mode
MBTYPETB	B	C(16)	describes top/bot MBs intra/forw/back/interp mode
MBSKIPTB	P,B	C(4)	describes whether top/bot MB skipped or not
MVTOPH	P,B	O	horizontal forward MV of top MB
MVTOPAMP	P,B	O	amplitude of forward MV of top MB
MVDIFAMP	P,B	O	amplitude of difference vector: (top MB forward_MV - bottom MB forward_MV)
MVANGPDT	P,B	O	angle difference between panning MV and top MV
MVPDTSUM	P,B	O	sum of horiz. and vert. diff. between pan and top MV
Parameters related to panning motion:			
PNPZERO	I,P	O	percentage of MBs with MV equal 0 in (prev) P frame
PNPFOR	I,P	O	percentage of forward MV coded MBs in (prev) P frame
PNPMVV	I,P	O	vertical Panning MV in (previous) P frame
PNPMVANG	I,P	O	angle of Panning MV in (previous) P frame
Parameters related to texture:			
TXNNZTOP	I,P,B	O	# of non-zero DCT coefficients in top 6 blocks
TXPNZTOP	I,P,B	O	position of last non-zero DCT coef (sum over top 6 blks)
TXSUM4TP	I,P,B	O	sum of selected DCT coefficients in top 6 blocks
TXDEVTOP	I,P,B	O	range of grey values in top 4 luminance blocks

Table 2: Examples of parameters provided as inputs to the CART algorithm. The columns list the variable name, type of frame for which it is applicable, whether it is an ordinal or categorical variable, and a description of what the variable represents.

Three sequences were encoded by MPEG-2 at a rate of 1.5 Mbits/sec. For each slice that is not the first or last slice of a frame, we considered the loss of that slice, and reconstructed each MB in the slice with each of the candidate methods which could possibly be used for it. For each MB, the method with the lowest reconstructed MSE (over both luminance and chrominance blocks) was considered the "winner" and became the classification associated with that MB. The data set consisting of the

input parameters and the output class was provided to the CART algorithm, and a large tree was grown. At each stage of growth, we are concerned with the size of the tree and its MSE performance. The size of the tree, as measured by the total number of nodes, is directly proportional to the number of bits that will be required as side information to transmit the tree to the decoder. The MSE performance of the tree is measured by choosing, for each MB in the sequence, the EC method dictated by the tree, and reconstructing the missing MB. The average MSE for all MBs is then computed. If the tree is allowed to grow large enough, eventually the classification will be perfect. The MSE will not then be equal to the MSE of the noiseless channel case (no MB loss), but will be the MSE that results from each MB being concealed by its *best* EC method among the set. We call this the "omniscient minimum" MSE, and it could also be obtained by transmitting a couple of bits explicitly for each MB to tell the decoder which EC method to use for that MB. What we consider the "maximum" MSE is the MSE that results from using a single fixed and best method among the first 5 in Table 1. The last 3 methods are excluded as candidates for the best fixed method because they usually can be applied to less than 25% of the MBs. Since certain P,B methods cannot be used next to intra-coded MBs, the use of a single fixed method really means employing one method in all the cases to which it is applicable, and using other pre-determined methods in those cases where it is not. The same pre-determined method is also used when the method dictated by CART is not applicable to the lost MB.

Our goal is to see whether much of this difference between the maximum MSE and the omniscient minimum MSE can be efficiently captured by the use of a decision tree, with significantly less overhead than is required by the explicit specification of EC methods for each MB. We are therefore interested in looking at plots of the MSE reduction versus the number of nodes as the tree grows. Trees were developed for several sequences, including mobile calendar, flower garden, and bicycle, as well as for separate GOPs from these sequences.

Figure 1 shows a CART tree with 7 terminal nodes built for the I frames of the complete flower garden sequence (150 frames). At each node of the tree, the oval lists the splitting test which is applied to split the data of that node. Above the oval is listed for each node the percentage of the node data that has the spatial, panning, and frequency EC methods as their *best* EC method. For example, for the root node of the tree, the spatial wins 22% of the time, panning wins 61%, and frequency wins 15%. The remaining methods make up the remaining 2% of the time. The test applied to this node is to check whether the vertical position of the missing MB is less than or equal to 10 (MBROW $\leq$ 10). The tree branches are labeled with the percentages of the data set that go down each branch. For the terminal nodes, the EC method that has the highest percentage of wins for that node data is selected by the plurality rule as the class for all data in the node.

For long sequences, the overhead of transmitting a tree is amortized, and one can consider larger trees. For the flower and mobile sequences, the plots of distortion reduction versus number of terminal nodes are shown in Figure 3a and b for I and P frames, respectively. In the plots, the maximum MSE is normalized to 1, corresponding to the MSE of the best single EC method out of the methods available.

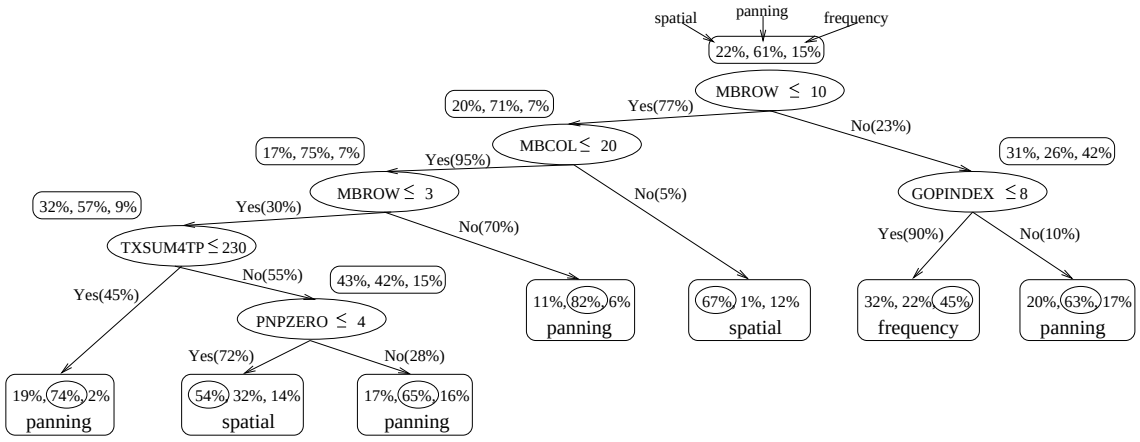

Figure 1: 7-terminal-node tree grown for flower garden I pictures, achieves relative MSE 0.85 compared to an omniscient minimum MSE of 0.68

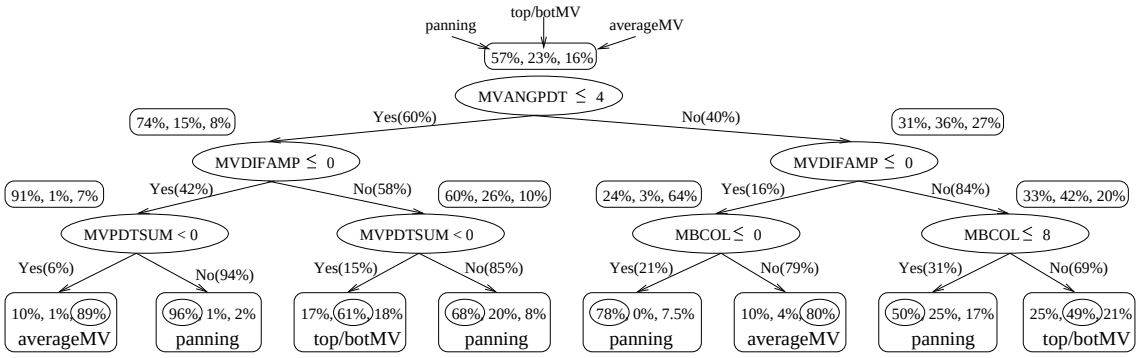

Figure 2: 8-terminal-node tree grown for mobile-calendar P pictures, achieves relative MSE 0.85 compared to an omniscient minimum MSE of 0.72

The misclassification error always decreases as the size of the tree increases; this does not necessarily mean the MSE also decreases because a larger tree may make fewer classification errors but which are more costly in terms of MSE. However as shown in the figures, MSE usually decreases with increasing tree size as well. For the flower garden I pictures, the best fixed method is panning (corresponding to max MSE of 1), and the omniscient minimum has a relative value of 0.67. As shown in Figure 3(a), a tree with 100 terminal nodes (199 total nodes) achieves 0.73. As it takes about 2 bytes to transmit a tree node, this represents an overhead of only 0.04% (for the sequence of length 150 frames, encoded at 1.5 Mbits/sec and 30 frames/sec). The average depth of the 100-node tree is less than 7, so the decoder needs to follow a sequence of only 7 binary tests on the average in order to obtain the concealment method. As shown in Figure 3(b), the available gains for the flower garden P pictures are smaller, since the omniscient minimum is 0.77; here a tree with 110 nodes reaches 0.87. For the flower garden P pictures, the best fixed method was the top/botMV. In the same figures, the mobile sequence had a tree of 107 terminal nodes reaching

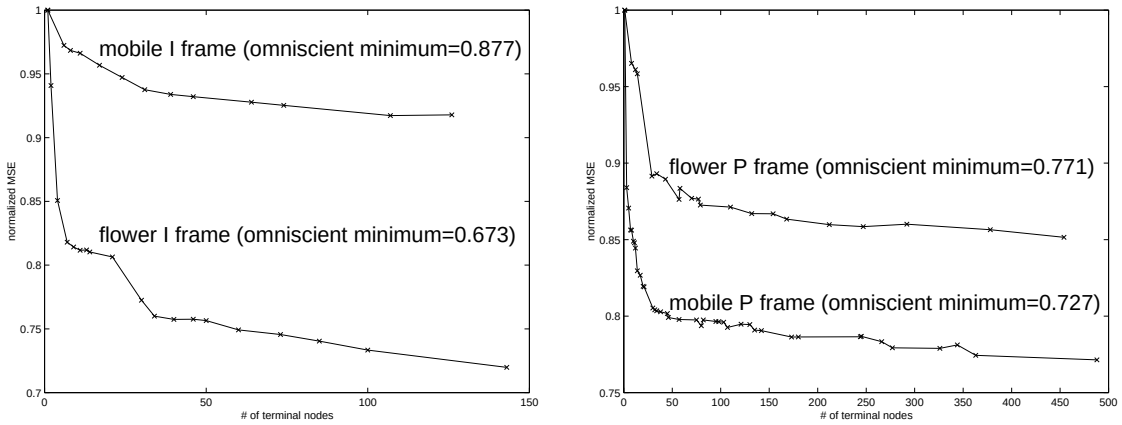

Figure 3: MSE vs. number of tree terminal nodes for (a) I frames, and (b) P frames

0.91 with the omniscient minimum being 0.88 for I frames; for P frames, a tree of 107 terminal nodes reaches 0.79 with an omniscient minimum of 0.73. The best fixed methods for mobile I and P frames were panning and top/botMV also, whereas for the bicycle sequence, the best fixed method for I frames was spatial, and for P frames top/botMV and averageMV were almost tied.

Method Name	I frames				P frames			
	flower garden		mobile calendar		flower garden		mobile calendar	
	A-mse	R-mse	A-mse	R-mse	A-mse	R-mse	A-mse	R-mse
spatial	1648	1.12	2043	2.85	1806	1.68	2218	2.60
frequency	1640	1.11	2520	3.51				
panning	1475	1.00	718	1.00	1624	1.51	853	1.00
top/botMV					986	0.917	784	0.920
averageMV					1030	0.958	847	0.993
no MB loss	134	0.091	162	0.226	144	0.133	176	0.207
omnisc. min	992	0.673	630	0.877	829	0.771	620	0.727
CART-tree	1076	0.733	655	0.912	936	0.870	676	0.793

Table 3: MSE results (averaged over 4 luminance and 2 chrominance blocks) for various concealment methods, and for no MB loss (noiseless channel). All CART-trees have 100 to 110 terminal nodes. (A-mse: Absolute MSE; R-mse: Relative MSE)

Some numerical results appear in Table 3. The attained MSE is listed for the reconstructed MPEG-2 sequence for a noise-free channel (no MB loss) for I and P frames for the flower garden and mobile sequences. The MSE for each fixed method is listed, as well as the omniscient minimum MSE, and the MSE from using the CART-tree. These are all given in the column A-mse. Relative MSEs are provided in the column R-mse, where the best of the single methods is normalized to 1. The MSEs for all the concealment approaches are much worse than the MSE of the decompressed MPEG sequence with no MB loss; this is because all concealment approaches are calculated based on *every* macroblock being individually lost and concealed, which is

useful for comparing the concealment approaches against each other.

When implemented at the level of individual GOPs, tree success rates varied. In our experiment each GOP was composed of 15 frames, including 1 I frame, 4 P frames and 10 B frames. For P frames, in mobile-calendar, trees with about 40 terminal nodes capture 40% to 80% of the available MSE reduction, capturing on average about 70% and providing MSEs typically in the range of 75% to 90% of the best fixed EC method. In the flower garden sequence, often trees with only 20 terminal nodes can capture 75% to 95% of the available MSE reduction, capturing on average about 85% and providing MSEs in the range of 50% to 70% of the best fixed EC method. The overhead rate for sending one tree of size 20 terminal nodes for each GOP is 0.08%.

In conclusion, we have presented two new temporal EC methods, based on estimating global pan parameters, and separating a MB into top and bottom halves for separate use of MVs from above and below. These new EC methods were often the best choices among the fixed methods. The use of a decision tree to choose adaptively among the various methods consistently provided lower distortion than any of the fixed methods alone. With a reduced set of input parameters, decision trees could be designed in real time for individual GOPs, or for individual frames; or decision trees could be designed for variable-length groups of data as the previous concealment strategy becomes outdated. Requiring only a small and adjustable level of overhead that depends on the tree size, the method can provide an attractive alternative to the transmission of CMVs, which in any case are only applicable to intra-coded blocks.

Acknowledgment: This work was supported by the National Science Foundation and by the Center for Wireless Communications at UCSD.

References

[1] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth, Belmont, CA, 1984.

[2] P. Cuenca, A. Garrido, F. Quiles, L. Orozco-Barbosa, T. Olivares, and M. Lozano. Dynamic error concealment technique for the transmission of hierarchical encoded mpeg-2 video over atm networks. In *Proc. 1997 IEEE Pacific Rim Conf. on Comm., Computers and Sig. Proc., PACRIM*, vol. 2, pp. 912–915, Victoria, BC, Canada, Aug. 1997. IEEE.

[3] W. Luo and M. El Zarki. Analysis of error concealment schemes for MPEG-2 video transmission over ATM based networks. In *Visual Communications and Image Processing '95*, volume 2501, pages 1358–1368, Taipei, Taiwan, May 1995. SPIE.

[4] J.-W. Suh and Y.-S. Ho. Error concealment based on directional interpolation. *IEEE Transactions on Consumer Electronics*, 43(3):295–302, Aug. 1997.

[5] Y. Wang and Q.-F. Zhu. Error control and concealment for video communication: A review. *Proc. IEEE*, 86(5):975–775, May 1998.

[6] Q.-F. Zhu, Y. Wang, and L. Shaw. Coding and cell-loss recovery in DCT-based packet video. *IEEE Trans. on Circuits and Systems for Video Tech.*, 3(3):248–258, June 1993.