Computationally Efficient Operational Rate-Distortion Optimal SNR Scalable Codec

Lisimachos P. Kondi[a]

[a]Dept. of Electrical Engineering, SUNY at Buffalo, Buffalo, NY 14260

ABSTRACT

We present a new scalable video codec which features a computationally efficient decoder as well as an operational rate-distortion optimal encoder. The SNR scalability is accomplished by splitting the Discrete Cosine Transform (DCT) coefficients of the Displaced Frame Difference (DFD) into groups that correspond to the scalable layers. Our Operational Rate-Distortion optimal scalable codec partitions the DCT coefficients of the DFD (or the intensity for intra blocks) into a base layer and one or more enhancement layers. The base layer is constructed by subtracting a value from each quantized DCT coefficient. The subtracted values are then sent as enhancement. If more than two scalable layers are required, the values subtracted for the creation of the base layer are further broken into other values. The partitioning of the DCT coefficients into layers is accomplished by formulating a constrained optimization problem which is then solved using lagrangian optimization. A Dynamic Programming (DP) solution is proposed in order to minimize the Lagrangian cost in a computationally efficient manner. We introduce a new method for finding the required Lagrangian multiplier λ which will meet our target bitrate. The method further reduces the computational complexity of the encoder by minimizing the need for an iterative calculation of the Lagrangian multiplier. Our experimental results show that the proposed codec typically outperforms H.263+ SNR scalability in terms of PSNR while exhibiting a low-complexity decoder. Due to the use of Dynamic Programming and a new method for the estimation of λ, the computational complexity of the operational rate-distortion optimal encoder is competitive with H.263+.

Keywords: Scalable Video Coding, Rate-Distortion Optimization

1. INTRODUCTION

A scalable video codec is defined as a codec that is capable of producing a bit stream which can be divided into embedded subsets. These subsets can be independently decoded to provide video sequences of increasing quality. Thus, a single compression operation can produce bit streams with different rates and reconstructed quality. A small subset of the original bit stream can be initially transmitted to provide a base layer quality with extra layers subsequently transmitted as enhancement layers.

There are three types of scalability supported in version 2 of the H.263 video compression standard (H.263+): SNR, spatial and temporal. In SNR scalability, the enhancement in quality translates in an increase in the SNR of the reconstructed video sequence, while in spatial and temporal scalability the spatial and temporal resolution, respectively, are increased.

A major application of scalability is in video transmission from a server to multiple users over a heterogeneous network, such as the Internet. Users are connected to the network at different speeds, thus, the server needs to transmit the video data at bit rates that correspond to these connection speeds. Scalability allows the server to compress the data only once and serve each user at an appropriate bit rate by transmitting a subset of the original bit stream.

Another important application of scalability is in error resilience. It has been shown[1] that it is advantageous to use scalability and apply stronger error protection for the base layer than for the enhancement layers (Unequal Error Protection). Thus, the base layer will be successfully received with high probability even during adverse channel conditions. Had we not used scalability and protected the whole bit stream equally, there would be a much higher probability of catastrophic errors that would result in a poor quality reconstructed video sequence.

Correspondence Email: lkondi@eng.buffalo.edu

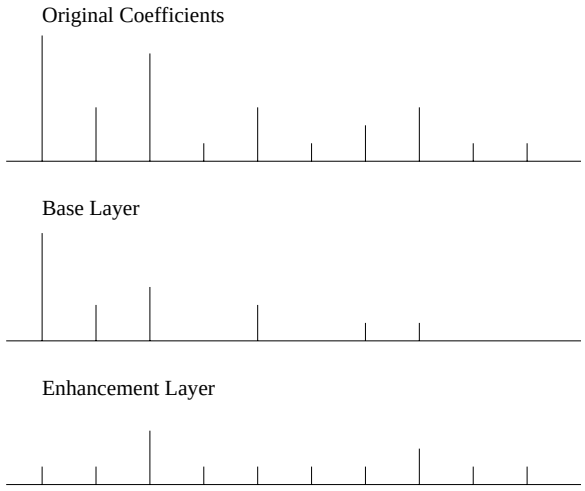

Figure 1: Proposed partitioning of DCT coefficients for SNR scalability.

In Refs. 2, 3 we introduced a new methodology for single pass signal-to-noise ratio (SNR) video scalability based on the partitioning of the DCT coefficients. The DCT coefficients of the Displaced Frame Difference (DFD) for inter-blocks or the intensity for intra-blocks are partitioned into a base layer and one or more enhancement layers, thus, producing an embedded bitstream. Subsets of this bitstream can be transmitted with increasing video quality as measured by the PSNR. Given a bit budget for the base and enhancement layers the partitioning of the DCT coefficients is done in a way that is optimal in the Operational Rate-Distortion sense. The optimization is performed using Lagrangian relaxation and Dymamic Programming (DP).

The scalable codec in Refs. 2, 3 has a decoder with lower computational complexity than H.263+ scalability and generally exhibits higher video quality as measured by the PSNR. However, the use of the bisection algorithm for the determination of the Lagrange parameter λ increases the computational complexity of the encoder since λ is found iteratively. In this paper, we propose a fast method for the determination of the Lagrange parameter. Using this method, the computational complexity of the encoder is competitive with H.263+.[4]

The rest of the paper is organized as follows. In section 2, the Operational Rate-Distortion optimal SNR scalable video codec is presented. In section 5, the fast method for the determination of the Lagrange parameter is discussed. In section 6, experimental results are presented. Finally, in section 7, conclusions are drawn.

2. OPTIMAL SINGLE PASS SCALABLE VIDEO CODEC

Our Operational Rate-Distortion optimal single pass scalable video encoder partitions the quantized DCT coefficients of the DFD (or the intensity for intra blocks) into a base layer and one or more enhancement layers. The base layer is constructed by subtracting a value from each DCT coefficient. The subtracted values are then sent as enhancement (See Fig. 1). If more than two scalable layers are required, the values subtracted for the creation of the base layer are further broken into other values. For example, if we want to transmit a coefficient with magnitude of quantization level 9 using three layers, we can transmit level 5 as base layer, 2 as first enhancement layer and 2 as second enhancement layer. The decoder reconstructs the quantized DCT coefficients by adding the subtracted values (if available to it) to the values it received with the base layer. Thus, a single inverse DCT is required at the decoder as opposed to the n inverse DCT steps required to decode the nth SNR scalable layer in H.263+.[3] Next, we present a formulation and optimal solution to the problem of partitioning the DCT coeficients of this scheme.

It is assumed that the DCT transform of the DFD (or the intensity for intra blocks) is taken and quantized according to the H.263 standard. Then, the proposed algorithm takes over to create the scalable layers. The quantized coefficients are coded as follows in H.263. A triplet (LEVEL,RUN,LAST) is transmitted using suitable

VLC tables, where LEVEL is the quantization level, RUN is the number of zero-valued coefficients that precede it and LAST specifies whether the current coefficient is the last in the block.

The problem is formulated as follows. Given a set of DCT coefficients X and their quantized version $\hat{X}$, find a set $\tilde{X}$ by subtracting a certain value l from each coefficient quantization LEVEL, so that a bit constraint is satisfied. It should be pointed out that $\tilde{X}$ as well as $\hat{X}$ are the "dequantized" values and not the quantization levels. We will call $\tilde{X}$ a *trimmed* version of $\hat{X}$. Let us denote by C the set of the quantization levels that correspond to the quantized coefficients $\hat{X}$ and $\tilde{C}$ the set of quantization levels that correspond to trimmed coefficients $\tilde{X}$. The set of quantization levels $\tilde{C}$ is transmitted as the base layer (along with motion vectors and overhead information). Then, given a bit budget for the base layer, our problem is to find $\tilde{X}$ as the solution to the constrained problem

$$\min[D(X,\tilde{X})|\hat{X}] \text{ subject to } R(\tilde{X}) \leq R_{budget}, \tag{1}$$

where $D(.,.)$ and $R(.)$ are the distortion and rate functions, respectively.

The problem of Eq. (1) can be solved using Lagrangian relaxation. The problem now becomes the minimization of the Lagrangian cost

$$J(\lambda) = D(X,\tilde{X}) + \lambda R(\tilde{X}). \tag{2}$$

In our implementation of the algorithm, we determine a bit budget for the base layer for a whole Group of Blocks (GOB). In H.263 with QCIF-sized frames, one GOB consists of one line of 16×16 macroblocks (11 macroblocks). Each macroblock consists of four luminance and two chrominance 8×8 blocks. Since the encoding of the DCT coefficients is done independently at each block (except for the dc coefficient of intra blocks which is transmitted with the base layer anyway), it is well-known that it is possible to express $J(\lambda)$ as a sum of individual Lagrangian costs (one for each block) and perform the minimization individually for each block. The same λ should be used for all blocks. Then, if the bit budget for the whole GOB is met for a specific λ, we are guaranteed that the minimization of the individual Lagrangian costs results in an optimal bit allocation across the whole GOB.

The problem now reduces to finding the set of quantization levels $\tilde{C}$ and corresponding trimmed DCT coefficients $\tilde{X}$ for every block that would minimize the Lagrangian cost of the block

$$J_{block} = D_{block} + \lambda R_{block} \tag{3}$$

for a given λ. We assume that each admissible candidate set $\tilde{C}$ is constructed as follows. Each non-zero coefficient in the block with quantization level u is either dropped completely or a value l where $l = 0, \ldots, L$ is subtracted from it. Clearly, $l < u$ and L is an appropriate constant. Therefore, there is a finite number of admissible sets $\tilde{C}$ and the minimization of the Lagrangian cost in Eq. (3) could be done using exhaustive search. However, the computational cost would be prohibitive. Luckily, the problem has a structure can be exploited using a Dynamic Programming (DP) solution. The Dynamic Programming algorithm is described next. A similar algorithm has been proposed in Ref. 5 for the quality enhancement in JPEG or MPEG and in Refs. 6, 7 for the optimization of the enhancement layer in SNR scalability as defined in MPEG-2. In this work, however, the same principles are applied to a completely different context.

As noted previously, the 2-D DCT coefficients are ordered in one dimension using the zig-zag scan and encoded using Variable Length Codes (VLCs) that correspond to the triplets
(LEVEL,RUN,LAST). Let us assume for a moment that the coefficients are coded using pairs (LEVEL, RUN), i.e. the same VLC is used whether the coefficient is the last non-zero coefficient in the block or not. We will explain the modifications to the algorithm for (LEVEL,RUN,LAST) later. Then, let us suppose that we consider the problem of minimizing the Lagrangian cost given that coefficient k is the last non-zero coefficient in the block to be coded and coefficients $k+1$ to 63 are all dropped from the base layer. Assuming that we have the solution to this problem where k is the last non-zero coefficient, this solution can be used to help us solve the problem of coefficient k' being the last one, where $k' > k$. Clearly, in the problem where k' is the last coefficient, if we determine that coefficient k should be included, then all coefficients between 0 and k would be trimmed as in the problem where k was the last coefficient. Therefore, the solution to the smaller problem can

be used as part of the solution to a larger problem. This is a characteristic of problems which can be solved using Dynamic Programming techniques. We next describe the basic characteristics of the algorithm.

The proposed algorithm uses incremental Lagrangian costs $\Delta J_{j,k}$ which are defined as

$$\Delta J_{j,k}^{l} = -E_k^l + \lambda R_{j,k}^l \text{ for } j < k \text{ and } l = 0, \ldots, L. \tag{4}$$

In the above equation,

$$E_k^l = X_k^2 - (X_k - \tilde{X}_k^l)^2 \tag{5}$$

where X_k is the original kth unquantized coefficient and $\tilde{X}_k^l$ is the quantized coefficient which corresponds to quantization level $\tilde{C}_k^l = C_k - l$ where C_k is the original quantization level. $R_{j,k}^l$ is the rate (in bits) that would be required to encode quantization level $\tilde{C}_k^l$ given that the previous non-zero coefficient was coefficient j. $R_{j,k}^l$ can be found from the VLC table and is simply the length of the corresponding code.

If coefficient k is dropped completely, the increase in mean squared error is X_k^2 where X_k the unquantized coefficient. If the quantized and trimmed coefficient is transmitted instead, the increase in mean squared error is $(X_k - \tilde{X}_k^l)^2$. Therefore, E_k^l represents the difference in mean squared error between dropping coefficient k from the base layer and transmitting the quantized coefficient trimmed by l.

$\Delta J_{j,k}^l$ represents the incremental Lagrangian cost of going from coefficient j to coefficient k (dropping the coefficients between them) and subtracting l from quantization level C_k. The algorithm keeps track of the minimum Lagrangian cost for each coefficient k assuming that it is the last coefficient to be coded in the block. We will denote this cost as J_k^*. For intra blocks we assume that the dc coefficient is transmitted intact with the base layer while for inter blocks, it is treated like every other coefficient. Therefore, for intra blocks, J_0^* is the Lagrangian cost of encoding only the dc coefficient and dropping all other coefficients. Since the dc coefficient is not encoded using the same VLCs as the rest of the coefficients, we will not count the bits used for the encoding of the base layer in the minimization. Therefore, if we drop all ac coefficients of a block, the rate will be zero and the distortion will be equal to

$$J_0^* = E_{intra} = \sum_{i=1}^{63} X_i^2, \tag{6}$$

since the DCT transform is unitary and we can calculate the mean squared error in either the spatial domain or the frequency domain. For inter blocks, we allow for the possibility of dropping all coefficients, including the dc. Then, we define

$$J_{-1}^* = E_{inter} = \sum_{i=0}^{63} X_i^2. \tag{7}$$

As mentioned earlier, we need to take into account the fact that different VLCs are used depending on whether the coefficient to be encoded is the last one in the block or not. Therefore, we define a second incremental cost

$$\Delta J_{j,k,last}^l = -E_j^l + \lambda R_{j,k,last}^l, \tag{8}$$

where $R_{j,k,last}^l$ is the number of bits that are required to encode quantization level $\tilde{C}_k^l$ given that j was the previous non-zero coefficient and coefficient k is the last one to be encoded in the block. We also keep the minimum Lagrangian costs $J_{k,last}^*$ for each coefficient k given that it is last coefficient to be encoded in the block.

3. PROPOSED RATE DISTORTION OPTIMAL SNR SCALABLE CODER

We are now ready to give the details of the algorithm. The algorithm is recursive and stores the minimum Lagrangian costs for the block when coefficient k is the last non-zero coefficient in the block, where $k = 0, \ldots, 63$ for inter blocks, and $k = 1, \ldots, 63$ for intra blocks. In the following discussion, we consider the case for inter blocks. For intra blocks, the only difference is that the recursion starts at $k = 1$ instead of $k = 0$.

The recursion begins with "coefficient" -1 which means that the imaginary coefficient -1 is the last coefficient in the block to be coded and all coefficients between 0 and 63 are dropped from the base layer. The cost of dropping all coefficients is stored as J^*_{-1} and is given by Eq. (7). Then, we proceed to find the minimum cost path that ends in coefficient 0 ($k = 0$). Clearly, this means that coefficient 0 will be kept and all others will be dropped but we need to find what value l_0 will be subtracted from its quantization level. That l_0 is the one which minimizes the expression $J^*_{-1} + \Delta J^{l_0}_{-1,0}$. The resulting cost is the minimum cost when coefficient 0 is the last one to be encoded in the block and is equal to J^*_0. We also need to perform the same procedure using $\Delta J^{l'_0}_{-1,0,last}$ instead of $\Delta J^{l_0}_{-1,0}$. Thus, we compute $J^*_{0,last} = \min_{l'_0}\{J^*_{-1} + \Delta J^{l'_0}_{-1,0,last}\}$ where l'_0 is not necessarily the same as l_0.

For $k = 1$, we can either keep coefficients 0 and 1 or just coefficient 1. Again, we need to determine the value to be subtracted from C_1. Now, the minimum cost will be $J^*_1 = \min_{i,l_1} J^*_i + \Delta J^{l_1}_{i,1}$, for $i = -1, 0$. We also need to calculate $J^*_{1,last}$ in a similar manner.

For a general k, the minimum costs are found as

$$J^*_k = \min_{i,l_k}\{J^*_i + \Delta J^{l_k}_{i,k}\}, \text{ for } i = -1, \ldots, k-1, \tag{9}$$

and

$$J^*_{k,last} = \min_{i,l'_k}\{J^*_i + \Delta J^{l'_k}_{i,k,last}\}, \text{ for } i = -1, \ldots, k-1. \tag{10}$$

The algorithm calculates J^*_k and $J^*_{k,last}$ for all $k = 0, \ldots, 63$ and also stores the last non-zero coefficients (predecessors) i and the subtracted values l_k and l'_k which minimize Eqs. (9) and (10), respectively. The k which results in the minimum $J^*_{k,last}$ will be denoted as k^*. Clearly, $J^*_{k^*,last}$ is equal to the minimum Lagrangian cost J_{block} for the whole block. Therefore, we know that coefficient k^* will be included in the base layer and we look up the value to be subtracted from it. Then, we look up the optimal predecessor i which resulted in $J^*_{k^*,last}$. Let us denote this coefficient as k_1. Then, k_1 will be included in the base layer and the value to be subtracted from it is looked up. Then we look up the predecessor that resulted in $J^*_{k_1}$ and continue recursively in the same fashion until we arrive at the imaginary coefficient -1.

"Pruning" of non-optimal predecessors i in Eqs. (9) and (10) can be performed, that is based on the observation that in the standard H.263 VLC table (and in practice, most custom made VLC tables), the number of bits $R^{l_j}_{i,j}$ required for the encoding of coefficient j given that the previous non-zero coefficient was coefficient i is monotonically increasing with the zero run length $j - i - 1$. Therefore, for $i < j$, $\Delta J^{l_k}_{i,k} \geq \Delta J^{l_k}_{j,k}$ and $\Delta J^{l'_k}_{i,k,last} \geq \Delta J^{l'_k}_{j,k,last}$. Thus, if $J^*_i > J^*_j$, then $J^*_i + \Delta J^{l_k}_{i,k} > J^*_j + \Delta J^{l_k}_{j,k}$ and $J^*_i + \Delta J^{l'_k}_{i,k,last} > J^*_j + \Delta J^{l'_k}_{j,k,last}$, for every l_k and l'_k. Thus, i cannot be an optimal predecessor to k. We will denote the set of coefficients to be considered as optimal predecessors of coefficient k as S_k.

It is interesting to point out that the proposed algorithm is equivalent to finding the shortest path in an Directed Acyclic Graph (DAG). Fig. 2 shows a DAG for the case of just three DCT coefficients (instead of 64). The vertices of the DAG correspond to the Lagrangian costs J^*_k while the edges correspond to the differential costs $\Delta J_{i,k}$. For simplicity, in this graph we assume that coefficients can either be included or dropped from the base layer, i.e., no "trimming" is involved. The first vertex takes the value $J^*_{-1} = E$, where E is equal to E_{intra} or E_{inter}, depending on the type of the macroblock. The last vertex designated as "end" is needed to show that the last coefficient for the block has been encoded. Clearly, $\Delta J_{i,end} = 0$ for all i. The solution of finding the shortest path of the DAG is exactly the algorithm we described.

The computational complexity of the proposed algorithm heavily depends on how successful the pruning is and how many iterations are needed to find the appropriate λ for each block for a particular video sequence. In this paper, we propose a new method for determining the appropriate λ, as described in section 6.

4. EXTENSION OF THE ALGORITHM TO A MULTIPLE NUMBER OF LAYERS

We have thus far presented an optimal algorithm for partitioning a set of quantized DCT coefficients into two layers. A bit budget is set for the base layer and the outside rate control is responsible for maintaining the

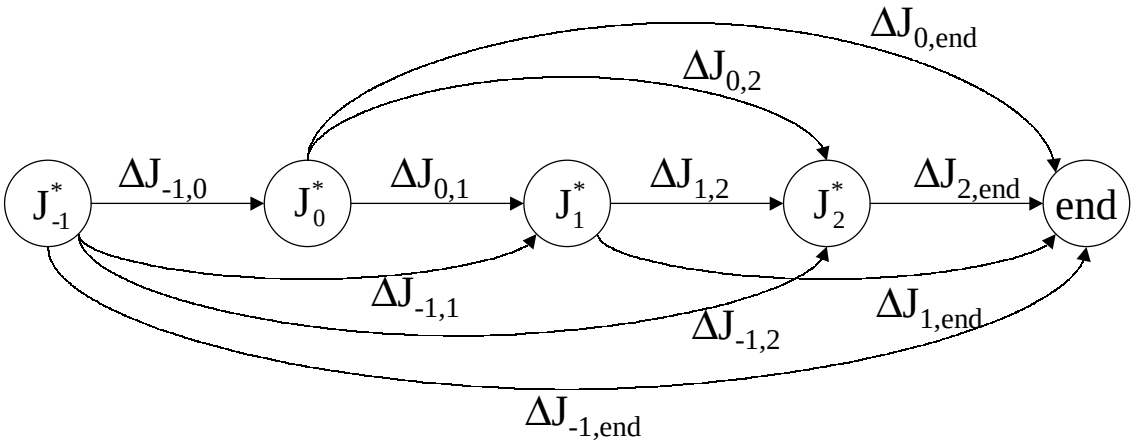

Figure 2: Directed Acyclic Graph representation of the optimal DCT coefficient partitioning problem.

total target bit rate for all layers. We now extend the algorithm to more than two layers. As described earlier, in order to partition the DCT coefficients into more than two layers, we first perform the partitioning into two layers as described in the previous section. We then partition the enhancement layer into two layers making the total number of layers equal to three. Now, the problem is formulated as follows. Given the quantized DCT coefficients and the partitioning into two layers, partition the coefficients of the enhancement layer (of the original partitioning) into two layers such that the distortion between the original unquantized coefficients and the coefficients reconstructed using the new base layer and the new first enhancement layer is minimized. The minimization is subject to a bit budget constraint for the first enhancement layer. Let us assume that we have already partitioned the DCT coefficients into two layers and the set of coefficient quantization levels for the second layer is C_{enh}. We now want to partition C_{enh} into two sets of coefficients, namely C_2 and C_3. The coefficients of the base layer C_1 have already been selected during the partitioning of the coefficients into two layers. Let X_{1+2} be the "dequantized" DCT coefficients when the first two of the three layers are utilized. Then, our problem is to choose the coefficient quantization levels C_2 that will make up scalable layer 2 (first enhancement layer) such that

$$\min_{C_2}[D(X, X_{1+2})|C_{enh}] \text{ subject to } R(C_2) \leq R_{budget,2}, \tag{11}$$

where $R_{budget,2}$ is the bit budget for scalable layer 2 (first enhancement layer).

The above problem can be solved using the algorithm already described. The only difference is in the definition of E_k^l and J_{-1}^*. The value of J_{-1}^* should be equal to the distortion that occurs when no coefficients are selected for layer 2 (the first enhancement layer). Therefore, the distortion will be the distortion incurred when including only the base layer, that is,

$$J_{-1}^* = E = \sum_{i=0}^{63}(X_i - \tilde{X}_i)^2, \tag{12}$$

where $\tilde{X}$ is the coefficients selected for the base layer as determined by the partitioning of the DCT coefficients into two layers. It should be pointed out that since in the case of intra blocks the dc coefficient goes to the base layer, for this part of the algorithm there is no distinction between intra and inter blocks. Thus, the first Lagrangian cost is J_{-1}^* and the recursion starts at $k = 0$ and not $k = 1$.

The second difference is in the definition of $E_k^{l_k}$. Now we have

$$E_k^{l_k} = (X_k - \tilde{X}_k)^2 - (X_k - X_{1+2,k}^{l_k})^2, \tag{13}$$

where $X^{l_k}_{1+2,k}$ is the "dequantized" coefficient k when including layers 1 and 2 (the base layer and the first enhancement layer). Layer 2 has been constructed by subtracting value l_k from the quantization index of C_{enh}.

The partitioning of the DCT coefficients into more layers is done by repeatedly partitioning the enhancement layer into two layers. If we want to partition the coefficients into n layers assuming that they are already partitioned into $n-1$ layers, we need to solve the problem of finding the set of quantization levels for layer $n-1$, C_{n-1} such that

$$\min_{C_{n-1}}[D(X, X_{1+2+\ldots+n-1})|C_{enh}] \text{ subject to } R(C_{n-1}) \leq R_{budget,n-1}, \tag{14}$$

where now C_{enh} is the (last) enhancement layer of the partitioning into $n-1$ layers and $X_{1+2+\ldots+n-1}$ is the set of "dequantized" DCT coefficients when the first $n-1$ layers are utilized.

Similarly to the case of three layers, we define J^*_{-1} and $E^{l_k}_k$ as

$$J^*_{-1} = E = \sum_{i=0}^{63}(X_i - X_{1+2+\ldots+n-1,i})^2, \tag{15}$$

and

$$E^{l_k}_k = (X_k - X_{1+2+\ldots+n-2,k})^2 - (X_k - X^{l_k}_{1+2+\ldots+n-1,k})^2 \tag{16}$$

where $X_{1+2+\ldots+n-2,k}$ is the "dequantized" coefficient k when layers up to $n-1$ are used.

5. DETERMINATION OF THE LAGRANGE PARAMETER

In our previous work,[2,3] the determination of the Lagrangian parameter λ in Eq. (1) is done as follows. For a Group of Blocks (GOB), the number of bits required for its encoding (R_{total}) is calculated. Then, the target number of bits for the base layer R_{budget} is calculated as a fixed percentage of R_{total}. This percentage is determined by the target bit rates for each scalable layer. Them, the bisection algorithm is used. The minimization of $J(\lambda)$ using the DP algorithm is performed using a low and a high value of λ (λ_l and λ_h). A low λ results in a high rate $R(\tilde{X})$ while a high λ results in a low $R(\tilde{X})$. Thus, the bisection algorithm adjust the λ and performs iterations of the DP algorithm until the desired bit rate R_{budget} is reached. This increases the computational complexity of the encoder since the DP algorithm has to be performed a number of times until the bit rate target is met.

In this paper, we propose a formula which was derived experimentally for the determination of the Lagrange parameter without the use of iterations. Thus, the Lagrange parameter λ is selected as

$$\lambda = A\frac{R_{total}}{R_{budget}}\exp(-R_{total}/B). \tag{17}$$

The constants A and B need to be determined. Our experiments show that values of $A = 500$ and $B = 1000$ work well in most cases. Using the above formula for the selection of λ, the obtained bit rates for the whole video sequence are met within a margin of 5% from the target. If even more accuracy in meeting the target bit rates is desired, Eq. (17) can be used to yield an initial estimate of λ followed by a limited number of iterations to find the λ that meets the target bit rate with the desired accuracy. In either case, the computational complexity of the encoder is reduced.

6. EXPERIMENTAL RESULTS

We next compare the performance of our SNR scalable video codec with the H.263+ implementation by the University of British Columbia.[4] The "Foreman" sequence was used for the simulations. The results are shown in Table 1 for a base layer bit rate of 14 kbps and an enhancement layer of 4 kbps (making the total rate equal to 18 kbps) and in Table 2 for a base layer bit rate of 28.8 kbps and an enhancement layer of 27.2 kbps (making the total rate equal to 56 kbps). The original frame rate of these sequences is 30 frames per second and the original length is 300 frames (10 seconds). The resulting encoded frame rate is close to 8 frames per second in all

Bit rate (kbps)	14	18
Proposed Algorithm PSNR (dB)	28.83	29.41
H.263 Algorithm PSNR (dB)	28.49	28.97

Table 1. Comparison of the average PSNR of the proposed algorithm and the H.263 standard SNR scalability algorithm at 14-18 kbps.

Bit rate (kbps)	28.8	56
Proposed Algorithm PSNR (dB)	30.20	33.02
H.263 Algorithm PSNR (dB)	30.29	31.98

Table 2. Comparison of the average PSNR of the proposed algorithm and the H.263 standard SNR scalability algorithm at 28.8-56 kbps.

cases. The Peak Signal-to-Noise Ratio (PSNR) reported was calculated as the average PSNR of all components (Y, C_b, C_r) and all encoded frames. It can be seen from these experimental results that the video quality of our SNR scalable codec as measured by the PSNR is comparable or better than the quality of H.263.

Figs. 3 and 4, show representative frames of the "Foreman" sequence encoded using the Optimal Single-Pass Codec at 28.8-56 kbps.

7. CONCLUSIONS

In this paper we presented a method for SNR scalable video coding which is based on the partitioning of the DCT coefficients. Our decoder has a lower computational complexity than the H.263 decoder since it only requires a single inverse DCT while the H.263 decoder requires n inverse DCT stages to decode n scalable layers. Using the proposed selection of the Lagrange parameter λ, the need for iterations of the DP algorithm at the encoder is reduced or eliminated. Thus, the complexities of the proposed encoder and the H.263 encoder are now comparable, although the H.263 encoder is generally faster. The video quality of the proposed encoder is generally better than that of H.263. The proposed SNR scalable video codec would be especially useful in applications where a fast decoder is required. Examples would include handheld devices such as cellular phones and Personal Digital Assistants (PDSs).

REFERENCES

1. R. Aravind, M. R. Civanlar, and A. R. Reibman, "Packet loss resilience of MPEG-2 scalable video coding algorithms," *IEEE Trans. on Circuits and Systems for Video Technology* **6**, pp. 426–435, Oct. 1996.

Figure 3. Frame 81 of the "Foreman" sequence encoded using the Optimal Single-Pass Codec at 28.8-56 kbps (28.8 kbps layer).

Figure 4. Frame 81 of the "Foreman" sequence encoded using the Optimal Single-Pass Codec at 28.8-56 kbps (56 kbps layer).

2. L. P. Kondi and A. K. Katsaggelos, "An optimal single pass SNR scalable video coder," in *Intern. Conf. on Image Processing*, (Kobe, Japan), 1999.
3. L. P. Kondi and A. K. Katsaggelos, "An operational rate-distoriton optimal single-pass SNR scalable video coder," *IEEE Trans. on Image Processing* **10**, pp. 1613–1620, Nov. 2001.
4. U. of British Columbia, "TMN version 3.2." H.263+ Public domain implementation.
5. K. Ramchandran and M. Vetterli, "Rate-distortion optimal fast thresholding with complete JPEG/MPEG decoder compatibility," *IEEE Trans. on Image Processing* **3**, pp. 700–704, Sept. 1994.
6. D. Wilson and M. Ghanbari, "Optimisation of two-layer SNR scalability for MPEG-2 video," in *Intern. Conf. on Accoustics, Speech and Signal Processing*, pp. 2637–2640, 1997.
7. D. Wilson and M. Ghanbari, "Optimization of MPEG-2 SNR scalable codecs," *IEEE Trans. on Image Processing* **8**, pp. 1434–1438, Oct. 1999.