In Proc. of the Data Compression Conference, DCC'96, Snowbird, UT, April 1996

Optimal bit allocation under multiple rate constraints

Antonio Ortega
Signal and Image Processing Institute
University of Southern California
Los Angeles, CA 90089-2564

Abstract

We present a new Lagrangian-based iterative technique for rate–distortion optimization under multiple rate constraints. We show how for sets of "linear" constraints this technique can be proven to be optimal up to a convex hull approximation. As an application we consider the problem of optimal buffer-constrained bit allocation. Our technique can be used to find an excellent approximation to the solutions achieved using dynamic programming. In cases where the buffer size is relatively large our approach shows a significant reduction in complexity as compared to dynamic programming.

1 Introduction

The problem of bit allocation for known inputs and a discrete set of available quantizers has been studied in the literature [1]. This problem arises in many practical situations where the goal is to minimize the total distortion by properly allocating quantizers to the various blocks or frames in a source, without exceeding a prespecified rate budget (see [2] for an image compression example)[1].

In this work we consider a more general case where a source is subject to additional constraints on the available rate. That is, rather than just limiting the total rate available to code the source, we assume that there exist constraints on the rate available to code subsets of the source blocks. Under simple and realistic conditions on the subsets we will show how to find the optimal solution.

A problem of particular practical interest which fits into this more general formulation is that of bit allocation in a buffer constrained environment [3]. This corresponds to the common situations where a source is to be encoded using some variable bit rate algorithm (i.e. one that assigns a different number of bits to different source blocks) to be transmitted through a constant bit rate channel. The source bits are

[1]While this method could also be used to allocate different quantizers to individual samples the overhead required to transmit the quantizer choice would make this approach impractical.

then buffered prior to transmission. The size of the buffer will determine the end-to-end delay in the system and we will assume it to be given. Our goal will then be to obtain the best performance given the constraint that the buffer should not overflow.

We will prove that optimality can be achieved for the particular case of a simplified buffer constrained optimization where underflow is allowed to occur. We formalize this problem in Section 2 and provide the algorithm in Section 3. We also outline how the algorithm can be used for more general optimization under multiple rate constraints. In Section 4 we will show how the algorithm can be used in the search for the optimal solution of the buffer constrained allocation where no underflow is allowed and will provide experimental results comparing our new approach with the one presented in [3].

2 Problem Definition: Buffer-constrained Bit Allocation

Assume that our goal is to encode N known input blocks[2], $1, 2, \ldots, N$, using a given set $\mathcal{Q}$ of M admissible quantizers, such that, for each choice of quantizer j for a given block i, we incur a distortion cost d_{ij} while requiring a certain rate r_{ij}. Our objective is to find the allocation $\mathbf{x} \in \mathcal{X} = \mathcal{Q}^N$ which assigns a quantizer $x(i)$ to each block i, such that we minimize the total distortion for the given rate constraints. The applicable rate constraints are that a finite buffer, which is used to store the encoder output before transmission over the constant rate channel, should never overflow. More formally, we can define the optimal buffer allocation problem as (see [3] for details):

Problem 1 *Optimal buffer-constrained bit allocation:*
For a given buffer size B_{max} and channel rate per block R, find $\mathbf{x}^*(B_{max})$ such that

$$\mathbf{x}^*(B_{max}) = \arg\min_{\mathbf{x}} \sum_{i=1}^{N} d_{ix(i)} \tag{1}$$

subject to

$$B(r_{ix(i)}) \leq B_{max}, \quad \forall i = 1, \ldots N, \tag{2}$$

and

$$\sum_{i=1}^{N} r_{ix(i)} \leq R \cdot N, \tag{3}$$

where $B(r_{ix(i)})$ is the buffer occupancy corresponding to block i when solution $\mathbf{x}$ is being used. Assuming that the initial buffer occupancy is B_0 we have that $B(r_{1x(1)}) = \max(0, r_{1x(1)} + B_0 - R)$ and in general $B(r_{ix(i)}) = \max(0, r_{ix(i)} + B(r_{i-1x(i-1)}) - R)$. The condition on the total rate of (3) is just used to ensure that the final buffer occupancy is less than or equal to B_0.

[2]The blocks can be image blocks, video frames, etc.

To simplify things we start by ignoring the effects of underflow so that we have $B(r_{ix(i)}) = r_{ix(i)} + B(r_{i-1x(i-1)}) - R$. Therefore, the constraints defined by (2) are all linear. We will propose an algorithm that achieves the optimal solution when only linear constraints are present. We will later show how this algorithm can be adapted to achieve nearly optimal performance in the case when underflow does not occur (i.e. as in the above formulation). Note that even though the problems for different B_{max} all include the same condition on the total rate of (3), the optimal solutions $\mathbf{x}^*(B_{max})$ *need not be the same.* The optimal solution to Problem 1 can be found in [3] where the problem is formulated as a search for a minimal cost path in a tree representing all the allowable (i.e. non overflowing) solutions. It was shown that the Viterbi algorithm (VA) [4], a form of forward deterministic dynamic programming, can be used to find the optimal solution.

A simple approach that may sometimes provide a good solution consists in assuming that the buffer will be sufficiently large so as not to be a "hard" constraint. In that case one can use the solution of a simpler problem, namely that of minimizing the distortion for a single rate budget constraint, adjusting it as appropriate to avoid overflow situations [5]. This simpler problem can be formulated as follows.

Problem 2 *Optimal budget-constrained bit allocation:*
Find $\mathbf{x}_u^*$ *such that*

$$\mathbf{x}_u^* = \arg\min_{\mathbf{x}} \sum_{i=1}^{N} d_{ix(i)} \tag{4}$$

subject to

$$\sum_{i=1}^{N} r_{ix(i)} \leq R \cdot N = R_{budget} \tag{5}$$

The simpler budget constrained allocation can then be solved using a Lagrangian optimization approach [1]. Details will be given in Section 3. Obviously, we will always have $\mathbf{x}_u^* = \mathbf{x}^*(B_{max} = +\infty)$. Moreover, there exists a finite B_u such that $\mathbf{x}_u^* = \mathbf{x}^*(B_u)$. B_u is the minimum buffer size required for the solution of Problem 2 not to overflow.

While the simpler Lagrangian approach has the advantage of being fast it may not be applicable to situations where the buffer size is relatively small (as is the case in most practical scenarios since the buffer size determines the end to end delay in the system), because the Lagrangian solution could violate the buffer constraint. Conversely, the VA approach will yield the optimal solution at the cost of a complexity that increases linearly with the buffer size. In [3] several heuristic methods were proposed as alternatives to the VA method. In this work we present a novel iterative Lagrangian formulation which guarantees optimality (up to a convex hull approximation) *while including the linear buffer constraints.* Note that we introduce this algorithm in the context of buffer-constrained bit allocation but it would be applicable in other more general scenarios.

3 Lagrangian optimization for multiple rate constraints

In this section we introduce a new iterative technique to solve problems with multiple constraints. We first review the Lagrangian optimization technique which can provide the optimal (up to a convex hull approximation solution) for a given bit budget. We then show the multiple constraint case can be solved by iteratively solving several budget-constrained problems.

3.1 Lagrangian optimization

We first summarize the Lagrangian optimization approach of [1] (see also [6]). Considering Problem 2 above, introduce a Lagrange multiplier λ, a real, nonnegative number, and for each block i introduce the Lagrangian cost associated with a solution $\mathbf{x}$:

$$J_i(\lambda, \mathbf{x}) = d_{ix(i)} + \lambda r_{ix(i)} \tag{6}$$

Then the main result states that for a given value of λ the solution $\mathbf{x}^*_\lambda$ such that

$$\mathbf{x}^*_\lambda = \arg\min_{\mathbf{x}} \sum_{i=1}^{N} J_i(\lambda, \mathbf{x}) \tag{7}$$

is also the solution to Problem 2 when the budget constraint is given by

$$R_{budget} = R^*(\lambda) = \sum_{i=1}^{N} r_{ix^*_\lambda(i)}. \tag{8}$$

Since choices can be made independently for each block, the optimal choice $\mathbf{x}^*_\lambda$ can be easily found by selecting $x^*_\lambda(i)$ such that $x^*_\lambda(i) = \arg\min_q(d_{iq} + \lambda r_{iq})$, where q spans the set of all possible quantizers for block i. Then the problem is to find the appropriate λ to meet the original budget $R \cdot N$. This can be efficiently done using fast convex search techniques (see [2] for example).

The important point to note is that an optimal solution for a given budget is such that each block uses a quantizer which corresponds to the same rate-distortion trade-off, as determined by λ. This minimization intuitively corresponds to finding the operating point that is "first hit" by a line of absolute slope λ. See the example in Figure 1(a). We will call this optimal solution, a "constant slope" solution[3].

Note that so far we have not taken into account the buffering constraint. If the solution obtained using the above method does not produce buffer overflow then it is also the optimal solution to the buffer-constrained optimization problem. However if it does produce overflow then we need some other method to find the optimal buffer-constrained solution. The VA introduced in [3] provides such a solution but has the

[3]We emphasize that even in the budget constrained problem this solution is optimal only up to a convex hull approximation. That is the Lagrangian approach can only find solutions which lie on the convex hull of the combined R-D characteristic. Other optimal, but non convex hull, solutions can be achieved using dynamic programming or with hybrid DP/Lagrangian techniques as in [7].

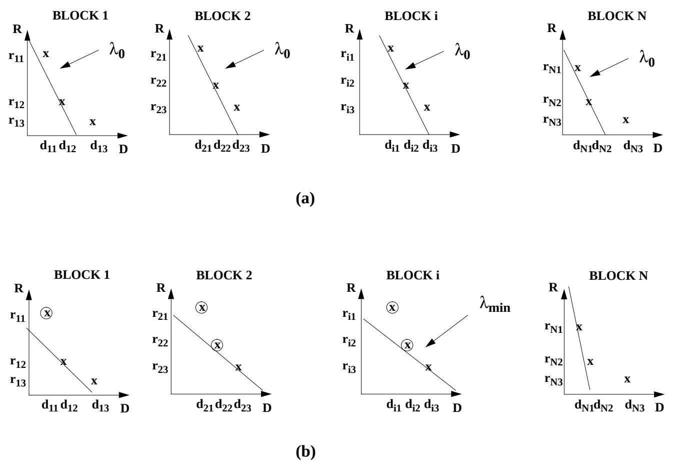
BLOCK 1
BLOCK 2
BLOCK i
BLOCK N
R
D
λ_0
λ_{min}
r_{11} r_{12} r_{13}
d_{11} d_{12} d_{13}
r_{21} r_{22} r_{23}
d_{21} d_{22} d_{23}
r_{i1} r_{i2} r_{i3}
d_{i1} d_{i2} d_{i3}
r_{N1} r_{N2} r_{N3}
d_{N1} d_{N2} d_{N3}
(a)
(b)

Figure 1: Examples of Lagrangian based bit allocation. (a) In the budget constrained case a constant quality slope λ_0 gives the optimal rate-distortion performance for the given blocks. (b) Assume that overflow occurs at block i when slope λ_0 is used. In order to *just* avoid overflow the initial i blocks have to operate at slope equal to $\lambda_{min} > \lambda_0$. This is equivalent to removing those operating points that have been circled. A global optimal solution can then be found using Lagrange multipliers with the reduced set of operating points.

shortcoming of a complexity that increases linearly with the buffer size. Here our goal is to generalize the Lagrangian formulation to solve problems with multiple rate constraints such as in Problem 1.

3.2 The multiple constraint case

A first Lagrangian approach which can be used in the multiple constraint case is to introduce a vector Lagrange multiplier, such that a different multiplier corresponds to each of the constraints. A general description of this approach can be found in [8]. Applications to rate allocation problems were first introduced in [9]. Also, in [10] an iterative procedure with multiple Lagrange multipliers has been introduced for Problem 1. The approach we present here is simpler and more intuitive than that in [10]. We also prove here that our approach is optimal within a convex hull approximation.

The key observation for our approach is to note that even though the Lagrangian formulation does not provide the optimal solution for the buffer-constrained case, it *does provide an operating point where rate and distortion are traded off optimally for the given budget.* If the solution is not admissible because it produces overflow it can nonetheless be used as the starting point to achieve an optimal a buffer-constrained solution.

We propose to use an iterative approach. We first solve the problem without the buffer constraint using the Lagrangian method. This will result in a Lagrange multiplier λ_0 such that the corresponding solution $\mathbf{x}^*_{\lambda_0}$ will meet the budget $N \cdot R$. Then we check if overflow occurs when using $\mathbf{x}^*_{\lambda_0}$ under a finite buffer. Clearly, if the solution $\mathbf{x}^*_{\lambda_0}$ does not overflow it is the solution to both Problem 1 and Problem 2 and we are done.

Now assume that overflow does occur at least once and that the first block for which overflow occurs is block i_1. This means that the first i_1 blocks use too many bits at the current quality slope λ_0. The best way to reallocate the available bits among those blocks to prevent overflow is to perform another Lagrangian optimization among only those first i_1 blocks, with a budget such that overflow is *just* avoided. Since the solution will give fewer bits to those initial blocks it will require a new Lagrangian slope $\lambda_{min}(i_1)$ such that $\lambda_{min}(i_1) > \lambda_0$. As outlined in Section 3.1 this new allocation will be optimal for the first i_1 blocks. The next question is then: is this approach globally optimal as well?

Let $R(1, i_1)$ be the rate used for the initial segment of blocks 1 through i_1 at slope λ_0, and let $R_o(1, i_1)$ be the maximum rate that can be used by the initial segment in order to avoid overflow. At this point we do not know if the optimal solution will use $R_o(1, i_1)$ or fewer bits in the initial segment but we know that no more than $R_o(1, i_1)$ bits can be used. We can thus define an effective quality slope $\lambda_{eff}(\lambda, i)$ for every block i in the sequence, such that

$$\lambda_{eff}(\lambda, i) = \max(\lambda, \lambda_{low}(i)), \tag{9}$$

where λ_{low} is the lower bound on the quality slope to be used for each block. We let

$$\lambda_{low}(i) = \lambda_{min}(i_1), \quad \forall i = 1, \ldots, i_1 \tag{10}$$

and we use the effective slope $\lambda_{eff}(\lambda, i)$ instead of the quality slope λ in the optimization. This means that when we are globally operating at a constant slope we cannot decrease the slope, i.e. increase the quality, below $\lambda_{min}(i_1)$ in the initial segment and therefore overflow can be avoided on that segment. In this manner we iteratively introduce lower bounds on the quality factors for each block every time an overflow is detected and then run again the usual Lagragian based optimization. This approach is equivalent to elimininating from the set of operating points those quantization choices which, for the initial segment, would result in total rates which violate the overflow constraints (see Figure 1(b)).

Theorem 1 *This approach is globally optimal for Problem 1 (with linear constraints), that is,*

(a) *none of the quantizers that are eliminated by using $\lambda_{eff}(\lambda, i)$ instead of λ can ever be part of the globally optimal solution, and*

(b) *the Lagrangian method using $\lambda_{eff}(\lambda, i)$ results in an optimal solution once all the successive overflow points have been taken into account. For each successive overflow point $i_1, i_2, \ldots$ we generate the λ_{eff} lower bounds for the quality slope of all blocks up to the overflow point.*

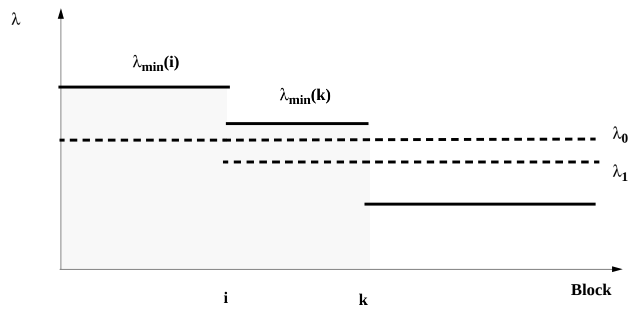

Figure 2: Example of utilization of the λ_{eff} parameter. The shaded areas depict the slope values that cannot be used in order to prevent overflow. i and k are the first and second observed overflow points, respectively. λ_1 is the overall slope selected after the first overflow has been removed. Note that each time overflow occurs in the current solution a new lower bound is given for blocks up to the point where overflow occurred.

Proof:
(a) Assume that arbitrary rates R_1 and R_2 are chosen for the segments $(1, i_1)$ and $(i_1 + 1, N)$, respectively. Once we have selected R_1 and R_2, the quantizers in each segment can be allocated independently. Clearly, for a given choice of rates, the minimum distortion solution comes from reallocating R_1 and R_2 bits to each segment via the Lagrangian method. The goal of the overall optimization will be then to find which R_1 and R_2 are to be selected[4]. The additional constraint that $R_1 < R_o(1, i_1)$ means that the slope for the first segment has to be *at least* $\lambda_{min}(i_1)$, for any choice of $R_1 < R_o(1, i_1)$. This is true for any rate $R_1 < R_o(1, i_1)$ and is independent of how bits are allocated for the second segment. Therefore, the quantizers that are eliminated by using $\lambda_{eff}(1, i_1)$ instead of λ will never be part of the optimal solution.
(b) Given the above, is it still optimal to use Lagrangian optimization to find the overall solution? Using $\lambda_{eff}(\lambda, i)$ guarantees that the overflow constraint is not violated. Thus we have eliminated the operating points that could potentially produce a problem, i.e. some of the blocks operate now with a reduced set of available quantizers in order to prevent overflow. We now have the same problem as initially (i.e. budget constrained allocation) with a reduced set of possible operating points. The Lagrangian optimization technique is thus applicable as usual.

Note that once we have eliminated the conflictive operating points using the $\lambda_{eff}(\lambda, i)$ rule we can consider that we are re-starting with a new budget constrained allocation problem, and therefore the above arguments can be used recursively for successive overflow points. As soon as a solution is reached where no overflow occurs we have achieved the optimal buffer-constrained solution. Note that if the constraints are non-linear, as in Problem 1, part (a) no longer holds[5] and thus we can only approximate the optimal solution. □

[4]Note that the same is true in Problem 2 for any two segments

[5]Reducing the rate of blocks before an underflow point $i_0 < i_1$, i.e., where the buffer occupancy is zero, does not help in reducing the buffer occupancy at the overflow point i_1!

3.3 Algorithm

We now describe the iterative algorithm in more detail. We just need to recursively apply the above explained method, i.e. use the successive overflow points to eliminate sets operating points from the optimization. We first describe the Lagrangian optimization to be used at each iteration for a given set of lower bounds on the quality slope $\lambda_{low}(i)$, we then explain how to update those lower bounds (refer to Fig. 2.)

Algorithm 1 *Lagrangian optimization with given quality slope lower bounds*

Given $\lambda_{low}(i)$ for all i we use a modified version of the Lagrangian optimization of [1]. We search for λ such that we meet the desired budget $N \cdot R$ with a rate $R(\lambda)$ corresponding to a solution $\mathbf{x}^*_\lambda$ such that

$$\mathbf{x}^*_\lambda = \arg\min_{\mathbf{x}} \sum_{i=1}^{N} J'_i(\lambda, \mathbf{x}) \tag{11}$$

with

$$J'_i(\lambda, \mathbf{x}) = d_{ix(i)} + \lambda_{eff}(\lambda, i) r_{ix(i)}. \tag{12}$$

We just replace λ by $\lambda_{eff}(\lambda, i)$ in the Lagrangian formulation of [1].

Algorithm 2 *Iterative lagrangian optimization with multiple rate constraints:*

Initialize $\lambda_{low}(i) = 0$ for all $i = 1, \ldots, N$.

Step 0 *: Find the optimal solution for the budget constrained problem with budget $N \cdot R$.*

Step 1 *: Does the current solution overflow the buffer? If not, then this is the optimal solution. If it does overflow, let k be the first overflow point. Add k to a list of overflow points $\mathcal{L}$, there are now l points in that list.*

Step 2 *: Find the minimum slope $\lambda_{min}(k)$ which* just *prevents overflow up to block k. We now update the lower bound on the slope for all the blocks up to k, i.e. for all $i = 1, \ldots, k$ we let $\lambda_{low}(i) \leftarrow \max(\lambda_{min}(k), \lambda_{low}(i))$.*

Step 3 *: Find the optimal budget constrained allocation using the slope lower bounds $\lambda_{low}(i)$ with Algorithm 1 and go to* **Step 1**.

3.4 Generalization

We have concentrated so far in the case of buffer constrained optimization. Consider now the case where we have k partial rate constraints each affecting a set of blocks $C_k = \{k_1, k_2, \ldots\}$, where the k_i are the indices of the blocks affected by the constraint. This could be the case if, say, for storage efficiency reasons we need to impose a restriction on partial rate counts. Each rate constraint has the form

$$\sum_{i \in C_k} R_i \leq B(k), \tag{13}$$

where $B(k)$ is the corresponding partial rate budget. It can be shown that if the constraints are embedded, i.e. $C_k \subset C_{k+1}$ for all k, then we can use the same technique as in the buffering case. Also it can be easily seen that the same holds if the constraints are disjoint, i.e. $C_k \cap C_{k+1} = \emptyset$.

The optimization procedure would be the same. We first find the solution to the problem without the partial constraints. We then consider all the partial constraints that have been violated by that solution. Considering one constraint at a time we find the lower bound on the quality slope λ which guarantees that the constraint is not violated. We then update λ_{low} and run the global optimization again.

4 Experimental Results

In this section we present our experimental results for Problem 1. Recall that in our developments we have thus far considered that underflow does not occur. When underflow occurs as in the exact formulation of Problem 1 we can no longer claim optimality, however we can achieve a good approximation to the optimal solution provided by the VA [3].

To achieve this solution we use a modified iterative procedure. Now, we keep track of both overflow and underflow points (the latter are those points where the buffer occupancy $B(r_{ix(i)})$ is equal to zero.) We modify Algorithm 2 so that now we replace **Step 2** by:

Step 2': *Let k' be the last underflow point prior to block k. Find the minimum slope $\lambda_{min}(k)$ which just prevents overflow up to block k by reallocating bits among blocks k' through k. We now update the lower bound on the slope for all the blocks from k' to k, i.e. for all $i = k', \ldots, k$ we let $\lambda_{low}(i) \leftarrow \max(\lambda_{min}(k), \lambda_{low}(i))$.*

Note that if there is no underflow point before k **Step 2** remains as before. This modification of the algorithm is needed to take into account underflow. It is clear that to prevent overflow one has to reduce the overall rate of the blocks leading up to the overflow point. However, reducing the rate for those blocks that occur before or at the underflow point is useless since further reductions in rate will not change the buffer state (i.e. buffer occupancy will still be zero!).

Our experiments are conducted with a series of JPEG-coded 8 by 8 pixel blocks from an image. There are 4 admissible quantizers and we show our results when encoding the image at 100 bits/block with several different buffer sizes. Our results indicate a very good approximation to the performance of the VA, with differences in PSNR of less than 0.1dB.

Note that the complexities of the Lagrangian approach and the VA have opposite behaviors. The VA is faster for small buffer sizes and may become impractical for large B_{max}. Conversely the Lagrangian approach requires the least complexity for large buffers because in that case the initial (budget constrained) solution is more likely to be close to the buffer-constrained one.

Acknowledgement: The author wishes to thank Kannan Ramchandran from University of Illinois at Urbana-Champaign for his comments on an earlier version of the manuscript.

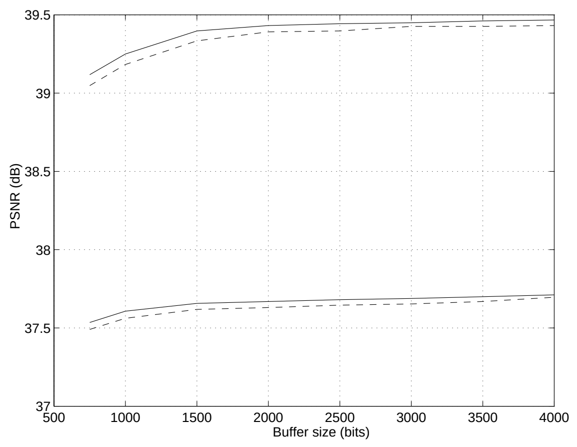

Figure 3: Experimental results. The dashed lines represent the result of our algorithm and the solid lines that of the VA. Results are given for channel rates of 64 and 100 bits/block (bottom and top, resp.) and for several values of the buffer size.

References

[1] Y. Shoham and A. Gersho, "Efficient bit allocation for an arbitrary set of quantizers," *IEEE Trans. on Signal Proc.*, vol. 36, pp. 1445–1453, Sept. 1988.

[2] K. Ramchandran and M. Vetterli, "Best wavelet packet bases in a rate–distortion sense," *IEEE Trans. on Image Proc.*, vol. 2, pp. 160–175, Apr. 1993.

[3] A. Ortega, K. Ramchandran, and M. Vetterli, "Optimal trellis-based buffered compression and fast approximations," *IEEE Trans. on Image Proc.*, vol. 3, pp. 26–40, Jan. 1994.

[4] G. D. Forney, "The Viterbi algorithm," *Proc. of the IEEE*, vol. 61, pp. 268–278, Mar. 1973.

[5] P. A. Chou, T. Lookabaugh, and R. M. Gray, "Entropy-constrained vector quantization," *IEEE Trans. on Signal Proc.*, vol. 37, pp. 31–42, Jan. 1989.

[6] H. Everett, "Generalized Lagrange multiplier method for solving problems of optimum allocation of resources," *Operations Research*, vol. 11, pp. 399–417, 1963.

[7] Y. Yoo, A. Ortega, and K. Ramchandran, "A novel hybrid technique for discrete rate-distortion optimization with applications to fast codebook search for SVQ," in *Proc. of ICASSP'96*, (Atlanta, GA), May 1996. To appear.

[8] M. L. Fisher, "The Lagrangian relaxation method for solving integer programming problems," *Management Science*, vol. 27, pp. 1–18, Jan. 1981.

[9] D. W. Lin, M.-H. Wang, and J.-J. Chen, "Optimal delayed-coding of video sequences subject to a buffer-size constraint," in *Proc. of VCIP'93*, (Cambridge, MA), Nov. 1993.

[10] J.-J. Chen and D. W. Lin, "Optimal coding of video sequences over ATM networks," in *Proc. of ICIP-95*, vol. I, (Washington, D.C.), pp. 21–24, Oct. 1995.