Bit Allocation for Joint Source/Channel Coding of Scalable Video

Gene Cheung and Avideh Zakhor
University of California, Berkeley

This work was supported by AFOSR contract F49620-96-1-0199, Sun Microsystems, Philips, Hughes Research Laboratories, and California State Programm MICRO and DIMI

Abstract

We propose an efficient bit allocation algorithm for a joint source/channel video codec over noisy channels. Our approach is to distribute the available source and channel coding bits among the subbands in such a way that the expected distortion is minimized. The constructed distortion curves bound the performance degradation should the channel be estimated incorrectly. The algorithm can be used in other similar distortion minimization problems with two constraints, such as power or complexity.

I. Introduction

The advent of wireless personal communication services in recent years has created a number of challenging research problems; a major challenge presented by the wireless channel is its inherent unreliability. This contrasts with wired networks, which have very low error rates. In a large class of wireless video applications, users move at relatively slow speeds, rather than at tens of miles per hour. Consequently, the resulting channels suffer from slow fading and shadowing effects. By estimating the condition of this slowly changing channel, one can adapt any aspect of the transmission scheme to the channel condition. In particular, one can change the source coding and the channel coding algorithms according to the channel condition to minimize the distortion of the received signals. Because the importance of different bits within a bitstream often varies, one can protect different source bits using unequal error protection (UEP) schemes such as Rate Compatible Punctured Convolutional (RCPC) codes [17] to further enhance performance. Indeed, several researchers have applied this idea to speech [1], [2] and image [3], [14], [13] transmission over wireless links. With the exception of [1] which deals with speech, the remaining papers mentioned above explicitly require the source coder to adapt to the channel condition. As an example, in [3] a whole new codebook might have to be designed in order to optimally accommodate each new channel condition.

With highly scalable video compression schemes [15], it is possible to generate one compressed bitstream such that different subsets of the stream correspond to the compressed version of the same video sequence at different rates. Thus, if one uses such a source coder in the wireless scenario, there is no need to change the source coding algorithm as the channel conditions change. This is particularly attractive in heterogeneous multicast networks where the wireless link is only a small part of a much larger network, and the source rate cannot be easily adapted to the individual receiver at the wireless node.

In this paper, we develop an algorithm for optimal partitioning of source and channel coding bits for the scalable video compression algorithm described in [15] and an unequal error protection channel coding scheme. By "optimal", we mean a partitioning which results in minimum expected value of distortion, which we chose to be Mean Squared Error (MSE). We will consider the case where the header files of the encoded bit stream are protected adequately so there is no loss of synchronization, and the channel state is known. Under these conditions, the joint source/channel codec will adapt to the estimated channel and optimally transmit video for the current channel state. In section II, we formalize our optimization problem and relate our approach to the current literature. In section III, we discuss our formulation of the bit allocation problem for joint source/channel codec. In section IV, we describe our proposed bit-allocation algorithm. Section V discusses some implementation issues and results. Section VI provides concluding remarks.

II. Related Work

The study of bit allocation addresses the general problem of finding the optimal distribution of resources (e.g. bits) among a set of competing users (e.g. quantizers) that minimizes the objective function (e.g. distortion), subject to resource constraints. Many existing bit allocation algorithms [4], [5], [6], [7] tackle a common special case when the objective function D is the sum of individual user functions d_k's, and the one resource constraint requires the sum of individual user resources n_k's not to exceed the resource limit R:

$$\min_{\{n_k\}} D = \sum_{k=1}^{K} d_k(n_k) \quad \text{s.t.} \quad \sum_{k=1}^{K} n_k \leq R \tag{1}$$

where K is the number of users. This formulation has practical applications such as bit distribution among quantizers [5], subbands [6] and coders in a classification-based coding scheme.

Recent papers [9], [10] describe a class of optimization problems in signal processing of a similar framework, but with two constraints:

$$\min_{\{n_k,m_k\}} D = \sum_{k=1}^{K} d_k(n_k, m_k) \quad \text{s.t.} \quad \begin{aligned} \sum_{k=1}^{K} n_k &\leq R_1 \\ \sum_{k=1}^{K} m_k &\leq R_2 \end{aligned} \tag{2}$$

In [9], the constraints are power and bandwidth. In [10], the constraints are complexity and bit rate. [9] offers a greedy algorithm that extends the one-dimensional bit allocation algorithm in [6] to another dimension, but the algorithm converges to a local minimum and does not guarantee a global minimum. In this paper, we present a new bit allocation algorithm that solves (2) optimally up to a convex-hull approximation. Our algorithm is an extension of the popular one-dimensional integer programming algorithm in [5]. Similar to [5], our algorithm exploits the discreteness of the user functions d_k's, and as such, it is especially efficient when the user functions d_k's are sparse and the number of users is small. Since user functions are operational RD characteristics in many applications, which are often sparse due to the computational complexity in generating them, our algorithm performs efficiently in these scenarios. In particular, we will present our algorithm in the context of a third practical signal processing problem: optimal distribution of source and channel bits among subbands for transmission of scalable video over noisy channels.

Tradeoff between source and channel coding has been studied from a theoretical standpoint in [11], [12] for vector quantizers. Practical systems for transmission of images based on this optimal tradeoff have been implemented in [13] [14]. With bandwidth being the only constraint, one can solve the optimal source/channel bit distribution problem in the framework of (1). For example in [13], the approach is to first construct the operational distortion-rate curve as function of bits for each subband of a wavelet decomposition, then apply one dimensional bit allocation algorithm in [6]. The optimal distribution of bits within a subband is done by using exhaustive search through all combinations of channel coding rates and quantization step sizes. One common thread among these analyses is that the joint source/channel codec is adaptive to the channel condition, which is assumed to be known perfectly. In the case when the channel is estimated incorrectly, it is difficult to evaluate the performance of the incorrectly adapted system without simulations.

In this paper, we formulate the bit allocation problem for joint source/channel codec in the framework of (2) instead. The added advantage is that one can estimate the performance of the joint source/channel codec during channel mismatch. This is important when evaluating the performance of a codec in scenarios where the channel condition cannot be estimated correctly, or where channel variation is too fast for the codec to adapt.

III. Joint Source/Channel Coding

In scenarios where the transceiving codec has a complexity constraint or the transmitted signal is delay sensitive, the separate source/channel coding theorem does not hold; as such, one must jointly design the source and channel codecs to achieve optimal performance. A common approach for building joint source/channel codecs is to cascade an existing source codec with a channel codec [14]. An important question then is how to distribute the source bits and channel bits between the source and channel codecs so that the resulting distortion is minimized. We formulate this problem formally in the framework of (2) in this section. In section III-A, we discuss the chosen source codec — a rate scalable 3D subband coder described in [15]. The unequal error protection channel codec we use is rate-compatible punctured convolutional codes (RCPC) [17]. The formulation of the problem using the chosen source and channel codecs is presented in section III-B.

A. Scalable Video Coder

The scalable source coder described in [15] has been shown to generate rates anywhere from tens of kilo bits to a few mega bits per second with arbitrarily fine granularity. In addition, its compression efficiency has been shown to be comparable to standards such as MPEG-1 [15]. The fundamental idea is to apply three dimensional subband coding to the video sequence to obtain a set of spatio-temporal subbands. Subsequently, each subband coefficient is successively refined via layered quantization. Finally, conditional arithmetic coding is applied to code different quantization layers. In so doing, the spatial and temporal correlations, as well as correlation between quantization layers are accounted for to maximize compression gain.

We opt to use the 3D scalable video codec in our analysis for a number of reasons. First, the embedded bitstream that the encoder generates is highly scalable. Depending on the estimated channel condition, the bit allocation scheme can easily extract varying subsets of the bitstream representing the signal of varying quality. Second, because the codec is wavelet based, the total distortion is roughly the sum of the distortion of the individual subbands, which fits the optimization framework in (2). Finally, the codec is well documented and readily available at the Video and Image Processing Lab of U.C. Berkeley [16].

B. Source / Channel Bit Allocation

The main problem we solve is: for our chosen source and channel codecs, and for a given total bit budget and channel condition, what is the optimal distribution of source and channel bits among the subbands that minimizes distortion? To solve this optimization problem we must determine: i) how many source bits goes into each subband k, ii) to what extent do we protect each source bit of subband k.

We first assume a total bit budget of B and a memoryless channel with known channel state; we will later consider both a binary symmetric channel (BSC) and an additive white Gaussian noise (AWGN) channel. For each channel state i, we first construct the distortion function D_i as a function of the ratio of source rate to channel rate R_s/R_c, where R_s (R_c) is the source bitrate (channel bitrate) in bits per second[1]. Each point on the curve $(\frac{R_s}{R_c}, D_i)$ represents the minimum distortion attainable from the best distribution of source and channel bits among the subbands such that the sum of source bits (channel bits) does not exceed R_s (R_c). If we enforce $R_c = B - R_s$, then $D_i(\frac{R_s}{B-R_s})$ would merely be a function of one variable, namely R_s, and each point on the curve $D_i(\frac{R_s}{B-R_s})$ will satisfy the bit budget requirement, B. This is shown in Figure 1 for two channel states 1 and 2. By empirically locating the minimum points on these curves, we can find the optimal source to channel bit ratio R_s^*/R_c^*, denoted by r_1 and r_2 for channel states 1 and 2 respectively in Figure 1. The corresponding minimum distortion for the two states are d_1 and d_2.

If the channel estimate is incorrect, we can approximate the performance of the codec using the constructed distortion curves D_i's. Suppose the channel state is incorrectly estimated to be state

[1] Channel code rate is usually defined as a ratio of source bits to channel bits. We decide to use a non-conventional notation for two reasons: i) bit allocation algorithms discussed later can be directly applicable to other optimization problems if the two constraint variables are defined as absolute quantities; and, ii) we are using unequal error protection within a subband; the conventional relative definition may misleadingly give the illusion that all source bits are channel-coded equally at the same rate.

2 when it is in fact state 1. We will erroneously use the source to channel bit ratio r_2 and the distribution of source and channel bits associated with point $D_2(r_2)$. Since the channel state is actually state 1, the distortion of the system is lower bounded by $D_1(r_2) = d_2'$. The reason is that $D_1(r_2)$ represents the best possible performance from the optimal distribution of source and channel bits for source to channel ratio r_2. Similarly, if we assume the channel to be state 1 when it is in fact state 2, then the distortion of the system is lower bounded by $D_2(r_1)$ and upper bounded by $D_1(r_1)$. The performance bounds obtained in this fashion provide a first-order evaluation of the joint source/channel codec during channel mismatch without actual simulations.

In computing each point on the above curves for $R_s = R_s^{\text{target}}$, we must determine how to distribute R_s^{target} source bits and $B - R_s^{\text{target}}$ channel bits among the subbands and quantization layers, such that the distortion, expressed below, is minimized.

$$D_i(\frac{R_s^{\text{target}}}{B - R_s^{\text{target}}}) = \sum_{k=1}^{K} d_k(n_k, m_k) \tag{3}$$

See Table 1 for notations.

To perform the optimization above, we need to construct the subband distortion functions, $d_k(n_k, m_k)$'s. Since each subband has an integer number of quantization layers and there is an integer number of channel protection levels, distortion functions are discrete. Let $\{m_{i,k}\}$ be the distribution of channel bits used to protect n_k source bits in subband k. The total number of channel bits for that subband is:

$$m_k = \sum_{i=1}^{n_k} m_{i,k} \tag{4}$$

For the same number of source bits n_k and same total number of channel bits m_k in a given subband k, there exists many different distributions of channel bits within the subband, corresponding to different levels of protections for different source bits. As a result, for given (n_k, m_k), $d(n_k, m_k)$ can take on more than one value. Since our goal is to minimize distortion, we will take the smallest value of $d(n_k, m_k)$ as the function's value at (n_k, m_k).

Because the computational complexity of the to-be discussed bit allocation algorithm depends on the number of points on the distortion functions, we will reduce the number of points in each distortion function by making the following assumptions:

1. The same level of protection will be applied to all the bits within a quantization layer of any given subband.
2. Higher quantization layers, which constitute refinement layers, cannot have higher level of protection than lower layers.

The first assumption assumes that all source bits within a given quantization layer have the same importance. The second assumption does not affect optimality because higher layers cannot be decoded if lower layers are not received correctly. Thus it is intuitively clear that lower layers need to be protected at least as much as the subsequent higher layers. Further details of the distortion function constructions are deferred to section V-A.

Given distortion functions $d_k(n_k, m_k)$'s, the optimization problem is to find the optimal allocation of source and channel bits among subbands:

$$\begin{array}{cc} \min_{\{n_k,m_k\}} & \left\{\sum_{k=1}^{K} d_k(n_k, m_k)\right\} \\ \text{s.t.} & \sum_{k=1}^{K} n_k \leq R_s^{\text{target}} \\ & \sum_{k=1}^{K} m_k \leq B - R_s^{\text{target}} = R_c^{\text{target}} \end{array} \tag{5}$$

In the next section, we will focus on solving (5).

IV. Lagrangian Optimization

There are many approaches to solve (5). If one assumes convexity of distortion functions, one can obtain fast algorithms [8], [7] for bit allocation problems in general. For example, [7] demonstrates that there is a complexity reduction in the generalized BFOS algorithm if one assumes convexity. However, we have found experimentally that the operational distortion surfaces, with respect to the channel rate axis are not convex. An intuitive explanation is the following: in very poor channel conditions, adding a small amount of channel bits to the subband will barely affect the bit error rate, and the subband distortion remains close to the distortion with no channel bits. As the amount of channel bits increases, the effect of the forward error correction kicks in, and the distortion will decrease more dramatically. This results in a non-convex surface. As a result of the non-convexity, we chose not to develop our algorithm relying on this assumption.

Instead of solving the original problem in (5), we will solve the corresponding Lagrangian problem instead:

$$\min_{\{n_k,m_k\}} \left\{ \sum_{k=1}^{K} [d_k(n_k, m_k) + \lambda n_k + \mu m_k] \right\} \tag{6}$$

If there exist multipliers λ and μ such that the source and channel bit budgets in (5) are satisfied with equality, then the optimal solution to the Lagrangian problem is also the optimal solution to the original problem [5].

The appeal of solving the Lagrangian problem is that the problem is now unconstrained. Moreover, for a given λ and μ, we can find the set $\{n_k^o, m_k^o\}$ that minimizes (6) by solving K separate equations individually in the form:

$$\min_{(n_k,m_k)} [d_k(n_k, m_k) + \lambda n_k + \mu m_k] \qquad \text{for} \;\; k = 1, \ldots, K \tag{7}$$

The resulting source and channel rate will be called the *operational* source and channel rate, expressed below:

$$R_s^o = \sum_{k=1}^{K} n_k^o \qquad R_c^o = \sum_{k=1}^{K} m_k^o \tag{8}$$

The problem is to find λ and μ such that the resulting (R_s^o, R_c^o) from (6) will be the same as our target source and channel rate $(R_s^{\text{target}}, R_c^{\text{target}})$. In cases where such multipliers do not exist, then we have to settle for an approximate solution. Although the solution is not optimal, the error is bounded by the following theorem:

Theorem 1: Suppose the approximate solution has operational source rate R_s^o, channel rate R_c^o, and distortion D^o:

$$D^o = \sum_{k=1}^{K} d_k(n_k^o, m_k^o) \tag{9}$$

Let the resulting distortion of the ideal solution be D^*, and the source and channel rate constraint be R_s^* and R_c^* respectively. In addition, let D^1 and D^2 be the distortions of two other Lagrangian solutions resulted from two different sets of multipliers, such that their operational source and channel rates obey the following inequalities:

$$\begin{aligned} R_s^1 &\leq R_s^o, R_s^* \leq R_s^2 \\ R_c^1 &\leq R_c^o, R_c^* \leq R_c^2 \end{aligned} \tag{10}$$

The error of our approximate solution can be bounded by the following:

$$|D^o - D^*| \leq |D^1 - D^2| \tag{11}$$

The proof is a natural extension of the proof of Lemma 7 in [5]. In practice such errors are often negligible.

The distortion functions $d_k(n_k, m_k)$'s are empirically computed discrete non-linear functions; most general bit allocation problems do involve this kind of functions. One approach is to fit analytic continuous functions to these discrete functions, and to perform optimization on the continuous counterpart. In this paper, we provide a method for selecting the Lagrange multipliers without fitting analytic continuous functions to the distortion functions.

To search for the best possible multipliers, we develop two methods in the next two sections. The first method, described in section IV-A, converges very fast when it is far from the solution but performs poorly when near the solution. The second method, described in section IV-B, converges slowly when it is far from the solution but converges to the optimal or near-optimal solution efficiently when near it. We will then discuss a hybrid of the two solutions in the section IV-C, which provides an efficient algorithm for finding an approximate, error-bounded solution in finite time.

A. Linear Approximation of Lagrangian Functions

We will begin with the description of the algorithm in section IV-A.1. Then we will discuss the difficulties with the algorithm in section IV-A.2. See [18] for a proof of the algorithm.

A.1 Development of Algorithm

The first method uses two properties of the rate functions to search for the multipliers: i) the rate function R_s^o (R_c^o) is more sensitive to changes in its primary multiplier λ (μ), ii) the rate functions are convex, non-increasing functions, and they are inversely proportional to their multipliers. The first property is based on empirical observations; the second property is a theoretical result of Lagrange

Multipliers. With these properties, we can approximate operational source rate R_s^o (channel rate R_c^o) as function of multiplier λ (μ) only:

$$R_s^o = A_s \frac{1}{\lambda} + B_s$$
$$R_c^o = A_c \frac{1}{\mu} + B_c \tag{12}$$

where A_s and B_s (A_c and B_c) are chosen constants to fit two empirical data points. The algorithm iteratively selects two points , $(\frac{1}{\lambda_0}, R_{s,0}^o)$ and $(\frac{1}{\lambda_t}, R_{s,t}^o)$, and constructs a linear function in the form of (12). Given the linear function, the algorithm computes the multiplier, λ (μ), that will yield the target rate, R_s^{target} (R_c^{target}). Using the computed multipliers, it finds the corresponding operational rate by solving (6). This gives another data point, and the algorithm repeats. The details of the algorithm are:

Step 0 Initialize iteration index: $t := 0$. Start with initial guess for multipliers, λ_0, μ_0.

Step 1 Find the corresponding operational $R_{s,0}^o$, $R_{c,0}^o$. If the condition:

$$R_{s,0}^0 \leq R_s^{\text{target}} \qquad R_{c,0}^0 \leq R_c^{\text{target}} \tag{13}$$

does not hold, then:

$$\lambda_0 := \frac{1}{\gamma_s}\lambda_0 \qquad \mu_0 := \frac{1}{\gamma_c}\mu_0 \tag{14}$$

where $\gamma_s, \gamma_c < 1$. Repeat step 1. This is point a in Figure 2. Otherwise, update multipliers:

$$\lambda_1 := \gamma_s \lambda_0 \qquad \mu_1 := \gamma_c \mu_0 \tag{15}$$

Step 2 Find the corresponding operational $R_{s,1}^0$ and $R_{c,1}^0$. If the condition:

$$R_{s,1}^0 \geq R_s^{\text{target}} \qquad R_{c,1}^0 \geq R_c^{\text{target}} \tag{16}$$

does not hold, then:

$$\lambda_1 := \gamma_s \lambda_1 \qquad \mu_1 := \gamma_c \mu_1 \tag{17}$$

Repeat Step 2. This is point b in Figure 2. Otherwise, let t:=1.

Step 3 Notice now:

$$R_s^{\text{target}} \in [R_{s,0}^o, R_{s,t}^o] \qquad R_c^{\text{target}} \in [R_{c,0}^o, R_{c,t}^o] \tag{18}$$

Using two source data points, $(\lambda_t, R_{s,t}^o)$ and $(\lambda_0, R_{s,0}^o)$, find A_s and B_s. This is the line that connects points a and b in Figure 2. Use similar procedure to find A_c and B_c.

Step 4 For given A_s and B_s, find λ_{t+1} that yields R_s^{target}. This is $\frac{1}{\lambda_2}$ in Figure 2 in the first iteration. Similarly, find μ_{t+1} that yields R_c^{target}. Get $R_{s,t+1}^o$ and $R_{c,t+1}^o$ by using λ_{t+1} and μ_{t+1} as multipliers to solve (6). This is point c in Figure 2.

Step 5 Check exit conditions:

$$\begin{array}{lll}
\text{if} & |R_{s,t+1}^o - R_s^{\text{target}}| \leq \epsilon_s R_s^{\text{target}}, & \\
\text{and} & |R_{c,t+1}^o - R_c^{\text{target}}| \leq \epsilon_c R_c^{\text{target}}, & \text{exit} \\
\text{else} & & t := t+1 \\
 & & \text{goto step 3}
\end{array}$$

where $\epsilon_s, \epsilon_c < 1$.

A.2 Difficulties of Algorithm

In practice, two problems prevent the algorithm from reaching the optimal solution. The first one is that the assumption of the operational source rate R_s^o (channel rate R_c^o) being function of its primary multiplier only λ (μ), is only a local approximation. R_s^o can be approximated as function of λ only when changes in μ is small. As the algorithm progresses, however, it is inevitable that μ will change. One remedy is to restart the algorithm every N iterations with λ_N and μ_N as initial starting multipliers. This way, $(\lambda_0, R_{s,0}^o)$ is updated with current values of μ. Similar procedure is performed to update $(\mu_0, R_{c,0}^o)$.

The other more serious drawback is that the source and channel rate are in practice discrete rather than continuous functions with respect to their multipliers. As the algorithm approaches the optimal multipliers (λ, μ), the discontinuity of the discrete source and channel functions may create difficulties for it. We address this problem with an improved algorithm presented below.

B. Generalized Shoham-Gersho Algorithm

Our proposed algorithm is a generalization of Shoham-Gersho Integer Programming algorithm [5] which yields optimal or near-optimal solutions to the source bit allocation problem in the framework of (1). We will begin with a brief review of Shoham-Gersho algorithm in section IV-B.1. In section IV-B.2, we will define our generalized algorithm, which we denote as the GSG algorithm. See [18] for proofs of the algorithm.

B.1 Shoham-Gersho Algorithm

The Shoham-Gersho algorithm is an integer programming algorithm for the source bit allocation problem. The algorithm solves the Lagrangian problem by finding the best possible Lagrange multiplier; the algorithm always terminates with an optimal solution or an error-bounded approximate solution. Unlike previous bit allocation algorithms [4], this algorithm addresses a more general class of problems because it does not fit the subband distortion curves to continuous analytic functions, nor does it make any assumptions about the nature of the distortion curves such as convexity.

It is important to understand the geometrical interpretation of the application of Lagrange multiplier to problems of form (1). The Lagrangian problem can be viewed as a minimization of the sum of subband distortions plus a penalty function $Q(\lambda) = \sum_{k=1}^{K} \lambda n_k$:

$$\min_{\{n_k\}} \left\{ \sum_{k=1}^{K} [d_k(n_k) + \lambda n_k] \right\} \tag{19}$$

The penalty function translates into adding a penalty line of slope λ to each of the K distortion functions. Figure 3 shows that for a given multiplier λ_1, we found minima in subband 1 and 2 to be

n_1^* and n_2^*. If these two are the only subbands, then the operational source rate for this multiplier λ_1 is $R_s^o = n_1^* + n_2^*$. If R_s^o is smaller than our budget rate R_s, then we decrease the multiplier value to decrease the effect of the penalty function. Geometrically, that would mean decreasing the slope of the penalty lines. Shoham-Gersho algorithm addresses the problem of exactly how the multiplier, or the slope of the penalty lines, should be adjusted.

The crux of the algorithm is the notion of *singular points* – a special set of multiplier values such that the optimal set of solutions is non-unique. Geometrically, such multipliers create a slope on the subband distortion curves such that a pair of adjacent convex-hull data points on a particular distortion curve are simultaneously minimum. Figure 4a shows the original distortion function with respect to source bits. To make n_1 and n_2 simultaneously minimum, we first find slope $\lambda_1 = -\frac{d(n_1)-d(n_2)}{n_1-n_2}$. Figure 4$b$ shows the effect of adding a line of slope λ_1 to $d(n)$ – there are now two minima in the distortion function. This implies that a singular point such as λ_1, has more than one operational source rate.

An important property of singular points is that neighboring singular points always share one solution. Figure 4c shows an adjacent singular point to the one in 4b, and they share one solution, namely n_2. Another property is that there can be no additional solutions in between the neighboring singular points. We see that as we decrease λ_1 to λ_2, the only possible minimum is n_2, which is the common solution for the two adjacent singular points. Another interpretation of this property is that the set of non-singular multipliers in between two neighboring singular multipliers does not yield any more solutions that is not covered by the two singular multipliers. Therefore, the set of singular points leads to the entire set of solutions to the Lagrangian problem in (6) for all possible values of multipliers. We can now make the following important conclusion:

Instead of sweeping the multiplier value λ from zero to infinity continuously in search of an operational rate that is close to our target rate, it is sufficient to look only at the singular points, since they alone lead to all possible Lagrangian solutions anyway.

Figure 5 shows the operational rate as a function of multiplier λ. Notice the singular points, λ_*, λ_2, λ_1 etc, each have multiple solutions, denoted by circles. Notice also that non-singular point multipliers do not lead to solutions that is not already covered by the singular points.

Since operational source rate is monotonically non-increasing with respect to the multiplier, at any given point on the operational rate plot, we only need to search neighboring singular points iteratively in the direction towards our target rate instead of traversing all of them. Geometrically, to go to a neighboring multiplier, we are decreasing (increasing) the slope of the penalty lines gradually in all subbands until non-unique solutions appear in a subband. With the new non-unique solutions in hand, we compute the new operational source rates and check against the target. In

Figure 5, we start at multiplier λ_1, then move to neighboring λ_2, then to neighboring λ_*; each time we drive our operational rate closer to our target rate R_s^*. Upon reaching λ_*, we notice that the two associated operational rates encloses our target rate. These are the closest solutions we can find by solving the Lagrangian; we will settle with the solution set corresponding to $R_{s,1}$ to be our approximate solution. See [5] for more details.

B.2 Development of GSG Algorithm

Our algorithm extends Shoham-Gersho algorithm to another dimension. Similar to the 1-D case, the multiplier problem in the form of (6) can be viewed as minimizing the sum of subband distortions plus a penalty function $Q(\lambda, \mu) = \sum_{k=1}^{K} \lambda n_k + \mu m_k$. With the added dimension, the penalty function now translates into adding elevated penalty planes in n and m axes to all subbands with slopes λ and μ respectively. Our goal is to iteratively make adjustments to the slopes of the penalty planes such that our operational rate pair (R_s^o, R_c^o) converges to our target rate pair $(R_s^{\text{target}}, R_c^{\text{target}})$. Similarly, we would like to make these adjustments using singular points. The notion of singular points for two dimensions, however, is slightly more complicated and needs to be explained further.

Singular Points: A non-singular set of multipliers, (λ, μ), similar to the 1-D case, implies that there is a corresponding set, $\{(n_k^o, m_k^o)\}$, that uniquely minimizes (6). If the set of multipliers is singular, then there is at least one subband, called the *pivot subband*, where there is more than one point in the subband that minimizes it. For example, there are two points, $\mathbf{x}_p^{o,1} = (n_p^{o,1}, m_p^{o,1})$ and $\mathbf{x}_p^{o,2} = (n_p^{o,2}, m_p^{o,2})$, that are co-minimum of subband p:

$$\begin{aligned} \exists p &\in \{1, \ldots, K\} \quad \text{s.t.} \\ \mathbf{x}_p^{o,1} &= \arg\left[\min_{(n_p,m_p)} \{d_p(n_p, m_p) + \lambda n_p + \mu m_p\}\right] \\ \mathbf{x}_p^{o,2} &= \arg\left[\min_{(n_p,m_p)} \{d_p(n_p, m_p) + \lambda n_p + \mu m_p\}\right] \\ \mathbf{x}_p^{o,1} &\neq \mathbf{x}_p^{o,2} \end{aligned} \tag{20}$$

Points that satisfy (20), $\mathbf{x}_p^{o,1}$ and $\mathbf{x}_p^{o,2}$ in the example, are each called *pivot point*. *Two-point pivoting* is the case where the multiplier pair yields only one pivot subband, having only two pivot points.

Note that we have two degrees of freedom in choosing the multiplier pair, (λ, μ). To satisfy (20) for two pivot points in a subband, we essentially have one equation, and only one degree of freedom is needed to satisfy it. To exploit the additional degree of freedom in selecting our multiplier pair, we have two alternative. First, we can find another point within the pivot subband such that together with the original two pivot points, we have three pivot points that are simultaneously minimum for a multiplier pair. We call this case *triangular*. This implies that we can move from a two-point pivoting case to a triangular case by using up the remaining degree of freedom in multiplier selection. Second, we can use up the remaining degree of freedom by finding two pivot points in a different subband. We again have two equations, each in the form of (20). We denote the case as *quadrangular*. We can move from a two-point pivoting case to a quadrangular case by using up the remaining degree of freedom.

Geometrically, a two-point pivoting case means adjusting slopes of the penalty planes in the n and m axes of all subbands such that there are two minimum points in a subband. Figure 6a shows that originally there is one unique minimum in subband i. In Figure 6b, we create slopes in the two axes such that there are two minima. As indicated by the arrow, we can continue to decrease the tilt of the plane surface by changing λ and μ while keeping these two points minimum. In Figure 6c, we reach a triangular case; there are three pivot points that are simultaneously minimum for subband i. In Figure 6d, we reach the other alternative; instead of finding a third pivot point in subband i, we find two pivot points in another subband j.

Note also that if the case is two-point pivoting, then the resulting operational source channel rate pair, called *pivot rate pair*, are in two distinct pairs:

$$
\begin{aligned}
(R_s^{o,1}, R_c^{o,1}) &= (n_p^{o,1} + \sum_{k=1, k\neq p}^{K} n_k^o,\ m_p^{o,1} + \sum_{k=1, k\neq p}^{K} m_k^o) \\
(R_s^{o,2}, R_c^{o,2}) &= (n_p^{o,2} + \sum_{k=1, k\neq p}^{K} n_k^o,\ m_p^{o,2} + \sum_{k=1, k\neq p}^{K} m_k^o)
\end{aligned}
\tag{21}
$$

where p denotes the index of the pivot subband. Similarly, if the case is triangular or quadrangular, there will be three or four corresponding pivot rate pairs of (R_s^o, R_c^o) respectively.

Similar to the 1-D case, singular points lead to all possible solutions to the Lagrangian problem. Therefore, instead of sweeping multiplier pair (λ, μ) continuously for all possible values, it is sufficient just to look at the singular points. Instead of considering all singular values, however, we will only need to step through a sequence of singular points such that we iteratively approach the best possible solution.

Lines and Regions of Eligibility: We now introduce the notion of lines and regions of eligibility for the R_s-R_c plane and the n_k-m_k plane of any subband k. Suppose we are in a two-point pivoting case with singular pair (λ^o, μ^o), resulting in two pivot points, $\mathbf{x}_p^{o,1}$ and $\mathbf{x}_p^{o,2}$ in pivot subband p, and unique optimal points $\{\mathbf{x}_k^o\}$ for non-pivot subbands. For the R_s-R_c plane, *line of eligibility* is the line passing through pivot rate pairs $\mathbf{X}^1 = (R_s^{o,1}, R_c^{o,1})$ and $\mathbf{X}^2 = (R_s^{o,2}, R_c^{o,2})$. The line divides the plane into two regions. The region that contains the target rate pair $\mathbf{X}^{\text{target}} = (R_s^{\text{target}}, R_c^{\text{target}})$ is the *region of eligibility*, as shown in Figure 7c. We draw corresponding line of eligibility on n_k-m_k plane of each subband k as well: the line goes through the currently optimal point(s) with the same slope as the line on R_s-R_c plane. We identify the regions of eligibility in n_k-m_k planes as the same corresponding side of the line as the one in R_s-R_c plane. Figure 7b illustrates this for non-pivot subband k. Figure 7a illustrates this for pivot subband p. Note that by definition of pivot rate pairs for two-point pivoting in (21), line with slope of line of eligibility in R_s-R_c plane and passes through optimal point $\mathbf{x}_p^{o,1}$ of pivot subband p must necessarily pass through $\mathbf{x}_p^{o,2}$ as well.

Description of Algorithm: Armed with the definitions above, we can now sketch the outline of our algorithm in words as follow:

1. Start with initial multiplier λ^o and μ^o that yield a two-point pivoting subband p. This uses up one degree of freedom.
2. Use the remaining degree of freedom by searching through regions of eligibility of all subbands to find either (a) the next pivot point in the 2-point pivoting subband (triangular), or, (b) two new pivot points in a new subband (quadrangular).
3. Use pivot selection scheme of section IV-B.4 to choose two pivot points out of the three pivot points in triangular case, or out of the four pivot points in quadrangular case.
4. Repeat step 2 and 3 until we get sufficiently close to $(R_s^{\text{target}}, R_c^{\text{target}})$ in the R_s-R_c plane.

The essence of the algorithm is to choose the new pivot point(s) in step 2 in such a way that the resulting pivot rate pair(s) are in some sense closer to the target rate $(R_s^{\text{target}}, R_c^{\text{target}})$ then previous pivot rate pairs. For this to happen, the new pivot rate pair(s) must be in the region of eligibility of the R_s-R_c plane. This means that for each subband, to search for the next potential pivot point, we only need to search among the points that are in the region of eligibility.

Finding Candidate Pivot Points: Geometrically, the next pivot point is the first point that, as the tilt of the planes is being decreased (increased) gradually, becomes co-minimum of its subband together with the original minimum point(s): $\mathbf{x}_p^{o,1}$ and $\mathbf{x}_p^{o,2}$, in the case of pivot subband (Triangular), or, $\mathbf{x}_k^o$, in the case of non-pivot subband (Quadrangular). Keep in mind that we are doing so while keeping the pivot points co-minimum in the pivot subband, thus we are decreasing the tilt in one dimension only. This is illustrated in Figure 8a. We first define a *tilted distortion function* for each subband k as:

$$\bar{d}_k(n_k, m_k) = d_k(n_k, m_k) + \lambda^o n_k + \mu^o m_k \tag{22}$$

where (λ^o, μ^o) are the pivoting singular pair. In Figure 8a, l_e denotes the line of eligibility of this subband. Suppose we perform a change of basis to l-z axes, as shown in Figure 8a, such that l is parallel to line l_e and z is perpendicular to it. Consider the set of all the planes passing through the l axis, each having a different tilt angle with respect to the l-z plane. Since l_e is parallel to l in all these planes, the two pivot points will evaluate to the same value no matter how large the tilt of the plane is. Therefore, if we search for the next pivot point by gradually changing the tilt of the plane passing through l axis, we will be doing so while keeping the original pivot points co-minimum.

To find out which point of which subband will be the next pivot point mathematically, we do the following. For each subband k, we elect a candidate pivot point $\mathbf{x}_k^*$: For each point in the region of eligibility, we find out how much the plane passing through l axis must tilt for that point to become a minimum. The point that requires the minimum tilt in order to become minimum is the first point that will become co-minimum of the subband. In the new coordinate system, finding the minimum tilt point is equivalent to finding the minimum slope point in the 1D case. Figure 8b shows the 2D view of the tilted distortion function of pivot subband p in the new coordinate system; l axis is pointing out of the page, and z axis is pointing along the page. We first observe that the two pivot points are on top of each other: they evaluate to the same value, $\bar{d}_p^{\min}$, and they have the same z coordinate, z_p^o. Note also the region of eligibility in this coordinate system is the set of points whose z coordinate values, z_p's, are larger then the minimum points' z_p^o. To find the point with the minimum slope, we evaluate the slopes of all points $z_p > z_p^o$, where slope is:

$$\text{slope} = \frac{\bar{d}_p(l_p, z_p) - \bar{d}_p^{\min}}{|z_p - z_p^o|} \tag{23}$$

Note that in the original coordinate system, $|z_p - z_p^o|$ is the projected distance of the point to the line of eligibility. The point with the minimum slope is the candidate pivot point of the subband. The same procedure applies for non-pivotal subbands. To choose a pivot point among the K candidates, we pick the point that requires the smallest tilt. This corresponds to the first point that would become co-minimum if we actually change the tilt of the plane passing through l axis gradually as mentioned before.

Selecting Pivot Points: If the new pivot point is found in the pivot subband, then we have a triangular case. We now have three pivot rate pairs, points $\mathbf{X}^1$, $\mathbf{X}^2$ and $\mathbf{X}^*$, as shown in Figure 9*a*. If the new pivot point is found in a subband other than the pivot subband, then the original minimum point of that subband becomes a pivot point as well, and we have a quadrangular case. We have four pivot rate pairs, points $\mathbf{X}^1$, $\mathbf{X}^2$, $\mathbf{X}^{*,1}$ and $\mathbf{X}^{*,2}$,as seen in Figure 9*b*. In the triangular case, a *pivot point selection rule* picks two of three pivot points in the pivot subband as new pivot points. In the quadrangular case, it picks one of two pivot subbands, each of which contains two pivot points, as the new pivot subband. The planes tilt again in the direction of the target pair using the new pivots, and the process continues. The algorithm stops when an enclosed area, whose corners are denoted by the pivot rate pairs, includes our target rate; this indicates that we have reached the closest convex-hull surface to the target, whose corners are the pivot rate pairs.

Details of Algorithm (see Table 2 for notations):

1. **Multiplier Initialization**: Initialize multipliers λ^o and μ^o that yields a two-point pivoting case, say in subband p (soon to be discussed). Find the set $\{(n_k^o, m_k^o)\}$ that minimizes (6), and the corresponding operational source and channel pairs, denote $\mathbf{X}^1 = (R_s^{o,1}, R_c^{o,1})$, $\mathbf{X}^2 = (R_s^{o,2}, R_c^{o,2})$.
2. **Definition of Tilted Distortion Function**: For each subband k, introduce a *tilted distortion function* as defined in (22).
3. **Definition of Eligibility Region**: Let the *slope of eligibility line*, shown in Figure 7*b*, be:

$$M = \frac{R_s^{o,2} - R_s^{o,1}}{R_c^{o,2} - R_c^{o,1}} \tag{24}$$

For each subband k , construct a *line of eligibility*, $n_k = Mm_k + b$, that goes through the optimal point(s) $\mathbf{x}_k^o = (n_k^o, m_k^o)$. Identify a region of eligibility, C_k, in each subband k that corresponds to the region of eligibility in the R_s^o-R_c^o plane.

4. **Identification of Candidate Pivot Point**: For each subband k, find a point $\mathbf{x}_k^* = (n_k^*, m_k^*)$ such that:

$$\mathbf{x}_k^* = \arg\left[\min_{(n_k,m_k)\in C_k} \frac{\bar{d}_k(n_k, m_k) - \bar{d}_k(n_k^o, m_k^o)}{|\text{proj } \ (n_k, m_k) \ \text{ on line of elig.}|}\right] \tag{25}$$

Among these K points from K subbands, find one that yields the minimum slope (the minimum value in (25)). Call it $\mathbf{x}^*$.

5. **Pivot Selection**: If $\mathbf{x}^*$ is in the pivot subband, then it is a triangular case. Define a point $\mathbf{X}^* = (R_s^*, R_c^*)$ as :

$$\mathbf{X}^* = (n_p^* + \sum_{k=1, k\neq p}^{K} n_k^o, \quad m_p^* + \sum_{k=1, k\neq p}^{K} m_k^o) \tag{26}$$

where p denotes the index of the pivot subband. We choose two of three points among the set $\{\mathbf{X}^1, \mathbf{X}^2, \mathbf{X}^*\}$ in the R_s^o-R_c^o plane, and specify the new region of eligibility, based on the pivot point selection rule (to be discussed).

If $\mathbf{x}^*$ is not in the original pivot subband, then it is a quadrangular case. Define points $\mathbf{X}^{*,1} = (R_s^{*,1}, R_c^{*,1})$ and $\mathbf{X}^{*,2} = (R_s^{*,2}, R_c^{*,2})$ as:

$$\begin{aligned} \mathbf{X}^{*,1} &= \left(n_p^{o,1} + n_j^* + \sum_{k=1, k\neq p,j}^{K} n_k^o, \quad m_p^{o,1} + m_j^* + \sum_{k=1, k\neq p,j}^{K} m_k^o \right) \\ \mathbf{X}^{*,2} &= \left(n_p^{o,2} + n_j^* + \sum_{k=1, k\neq p,j}^{K} n_k^o, \quad m_p^{o,2} + m_j^* + \sum_{k=1, k\neq p,j}^{K} m_k^o \right) \end{aligned} \tag{27}$$

where p is the index of the original pivot subband, and j is the index of the new pivot subband. We use the pivot point selection rule to select two points in the set $\{\mathbf{X}^1, \mathbf{X}^2, \mathbf{X}^{*,1}, \mathbf{X}^{*,2}\}$, and then specify the region of eligibility. The algorithm stops if pivot point selection rule signals termination.

6. **Multiplier Re-initialization**: For the newly selected pivot points in the pivot subband, find any multiplier pairs (λ^o, μ^o) such that the resulting tilted distortion function for the pivot subband evaluated at the pivot points will have the same value. Goto step 2.

B.3 Initialization

For step 1 of the algorithm, the goal is to initialize multipliers λ^o and μ^o such that it results in a two-point pivoting case. We will assume we already have optimal set $\{(n_k^o, m_k^o)\}$ (optimal solution to (6)) for a given non-singular point multiplier pair (λ^l, μ^l). A simple method is the following: first define a slightly different tilted function than (22) for each subband as:

$$\bar{d}_k(n_k, m_k) = d_k(n_k, m_k) + \mu_l m_k \tag{28}$$

For each subband, find $\mathbf{x}_k^*$ such that:

$$\mathbf{x}_k^* = (n_k^*, m_k^*) = \arg \left[\min_{n_k > n_k^o} \frac{\bar{d}_k(n_k, m_k) - \bar{d}_k(n_k^o, m_k^o)}{n_k - n_k^o} \right] \tag{29}$$

Geometrically, this is first point in subband k that will become co-minimum with the present minimum $\mathbf{x}_k^o$ if the slope λ of the penalty plane on the n axis is gradually changed. Among these K points, find one that minimizes (29). Call it $\mathbf{x}^*$. This is the first point in all subbands that will become co-minimum if the slope λ is gradually changed. The subband of $\mathbf{x}^*$ is the pivot subband. Let $-\gamma = \frac{\bar{d}_k(n_k, m_k) - \bar{d}_k(n_k^o, m_k^o)}{n_k - n_k^o}$ evaluated at $\mathbf{x}^*$. The 2-point pivoting multiplier values for step 1 of the algorithm can now be expressed as: $(\lambda^o, \mu^o) := (\gamma, \mu^l)$.

B.4 Pivot Point Selection Rule

In either the triangular or quadrangular case, we must select two pivot rate pairs in the R_s-R_c plane, and hence the corresponding pivot points in subbands, out of three or four possible pairs for step 5 of the algorithm. The selection rule must select pivot rate pairs in such a way that it ensures the algorithm will make progress, and therefore will converge to the best possible solution.

We will begin with a few necessary definitions. Define *pivot line segment*, labeled a in Figure 10, as the line segment between two current pivot rate pairs, $\mathbf{X}^1, \mathbf{X}^2$. Define *distance line segment*, labeled b in Figure 10, as the line segment that minimizes the distance between the target $\mathbf{X}^{\text{target}}$ and the pivot line segment. The selection rule is as follows:

- **Triangular:** To avoid stagnation, the new pivot rate pair $\mathbf{X}^*$ must be one of the two pairs selected. To choose between the two original pivot rate pairs, $\mathbf{X}^1$ and $\mathbf{X}^2$, we select one that yields a new pivot line segment that crosses the current distance line segment. If such point does not exist, then we select the pivot rate pair that yields a pivot line segment that touches the current distance line segment. In Figure 10a, by selecting pivot rate pairs $\mathbf{X}^*$ and $\mathbf{X}^2$, the new pivot line segment, a', crosses the current distance line segment, b.
- **Quadrangular:** Again, to avoid stagnation, one of the two new pivot rate pairs, $\mathbf{X}^{*,1}$ and $\mathbf{X}^{*,2}$, must be selected. Similarly, we select two pairs that yield a new pivot line segment that crosses the current distance line segment. If such pivot rate pairs do not exist, then we select the two that yield a pivot line segment that touches the current distance line segment. In Figure 10b, there are no two pivot rate pairs that yield a pivot line segment that crosses the distance line segment b. So we select $\mathbf{X}^{*,2}$ and $\mathbf{X}^2$ to yield pivot line segment a' which touches b.
- **Ending Condition:** When the triangle or the parallelogram whose corners are denoted by the three or four pivot rate pairs encloses the target rate, we terminate the algorithm.

Using this selection rule, one can show that the length of the distance line segment decreases during the procession of the algorithm, meaning the algorithm is making progress at each iteration. See Appendix A for convergence of the algorithm using this pivot point selection rule.

B.5 Time-sharing of Operational Points

Instead of settling for an approximate solution $\mathbf{X}^1$ without using up the available bandwidth, i.e. $R_s^1 < R_s^*$ and $R_c^1 < R_c^*$, we can use time-sharing to divide time among the neighboring pivot rate pairs so that average rate equals the target rate, and the distortion is lowered. This is permissible in coding schemes where the increase in overhead for time-sharing does not over-burden the implementation complexity of the system.

Suppose we exit the algorithm with the set of operational points shown in Figure 11. Instead of settling with the solution associated with operational rate pair $\mathbf{X}^1$, we do time-sharing among

$\mathbf{X}^1$, $\mathbf{X}^2$ and $\mathbf{X}^{*,1}$. In general, we select three of the possible four pairs using the following criteria: among the groups of three points that enclose the target, select the group that minimizes the total distance between the target and individual points of the group. Denote the three selected pairs as $\{\mathbf{X}^1, \mathbf{X}^2, \mathbf{X}^3\}$. Suppose we spend fraction α of time at $\mathbf{X}^1$, β of time at $\mathbf{X}^2$, and $1-\alpha-\beta$ of time at $\mathbf{X}^3$. The distortion is the weighted average of the three:

$$D^{\text{t.s.}} = \alpha D^1 + \beta D^2 + (1-\alpha-\beta)D^3 \tag{30}$$

where t.s. stands for time-sharing. The fractions α, β must satisfy the following equations so that the average source and channel rate equal the constraints:

$$\begin{aligned} \alpha R_s^1 + \beta R_s^2 + (1-\alpha-\beta)R_s^3 &= R_s^{\text{target}} \\ \alpha R_c^1 + \beta R_c^2 + (1-\alpha-\beta)R_c^3 &= R_c^{\text{target}} \end{aligned} \tag{31}$$

Rewriting the above equation, we can solve for α, β as follows:

$$\begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} R_s^1 - R_s^3 & R_s^2 - R_s^3 \\ R_c^1 - R_c^3 & R_c^2 - R_c^3 \end{bmatrix}^{-1} \begin{bmatrix} R_s^{\text{target}} - R_s^3 \\ R_c^{\text{target}} - R_c^3 \end{bmatrix} \tag{32}$$

C. Hybrid Algorithm

The GSG algorithm has one major disadvantage that discourages its exclusive use: because it painstakingly searches through every singular multiplier pair on the search trajectory toward to the optimal solution, the algorithm is slow if the initial operational source-channel pair is very far from the optimal. To remedy this problem, we propose a hybrid algorithm that first uses the linear approximation algorithm in section IV-A to find an approximate solution, then uses the GSG algorithm to refine the estimate into an optimal or near-optimal solution. The linear approximation algorithm has efficient convergence until it encounters the discreteness of the rate functions near the optimal. The GSG algorithm performs poorly when far from the solution, but finds the optimal solution efficiently when near it. Therefore the hybrid algorithm combines the advantages of both algorithms while avoiding their respective pitfalls.

V. Implementation

A. Operational Distortion-Rate Functions

In practice, there are two possible approaches for arriving at the operational distortion functions for various subbands. In the first approach, we analyze each video clip individually off-line in order to compute its exact distortion curves. In the second approach, we can arrive at a set of distortion curves for a class of video sequences, say large motion or head & shoulder. To obtain an accurate estimate of the class distortion curves, we take the ensemble average of a long representative sequence. The computations of these distortion curves are performed off-line, and therefore do not introduce excessive delay in the transmission system. Clearly the second approach is more suitable for real time, interactive application than the first approach.

The distortion metric we chose is mean square error (MSE). While we realize MSE may not the best metric for video quality evaluation, due to its wide use in the literature and the lack of a well accepted metric for video in the research community, we decide to use it for our application. Note that our algorithm can be applied to any metric, given the distortion can be expressed as the sum of individual distortions as shown in (2). The distortion functions can now be expressed as:

$$d_k(n_k, m_k) = E[|\hat{X}_k(n_k, m_k) - X_k|^2] \tag{33}$$

where X_k is the kth subband component of the original signal, and $\hat{X}_k(n_k, m_k)$ is the quantized and channel corrupted kth subband component of the signal given n_k source bits and m_k channel bits. We can first expand the expected distortion of subband k as the sum of conditional distortions for a collection of error events. According to the Total Probability Theorem, the events in the collection are disjoint, and they collectively span the sample space.

$$\begin{aligned} & E[|\hat{X}_k(n_k, m_k) - X_k|^2] = \\ = & \sum_{i=1}^{n_k} E[|\hat{X}_k(n_k, m_k) - X_k|^2 \mid e'_{1,k}..e'_{i-1,k}e_{i,k}]P(e'_{1,k}..e'_{i-1,k}e_{i,k}) \\ & +E[|\hat{X}_k(n_k, m_k) - X_k|^2 \mid e'_{1,k}..e'_{n_k,k}]P(e'_{1,k}..e'_{n_k,k}) \end{aligned} \tag{34}$$

where $e_{i,k}$ denotes the event that bit i of subband k is received incorrectly, and $e'_{i,k}$ denotes the corresponding complement event. We will assume the usage of conditional arithmetic coding in the coding of subband coefficients. Since conditional arithmetic coding is a variable length code, it is a good approximation to assume that if bit i is corrupted, all bits in the remaining codeword are rendered useless due to loss of synchronization. So we can assume that the resulting error when bit i and some other bits after bit i are flipped is approximately equivalent to the resulting error when only bit i is flipped.

We will now define three functions to ease our notations. Let $h_k(i)$ be the distortion function of subband k if i source bits are used under noiseless condition. Let $f_k(i)$ be the resulting distortion function of subband k if only bit i is flipped. To obtain this function, we experimentally inject an error at bit i of the corresponding subband, and average out the error over 200 frames to get an approximate value. Let $g(m)$ be the resulting error probability of a source bit if, on average, m channel bits are used to protect it. This function will obviously depend heavily on the particular implementation of the channel codec and the channel condition. In our experiment, we use RCPC for our unequal error protection codec. Since RCPC is a convolutional code, bit error can be bounded using [17]:

$$g(m) \leq \frac{1}{P} \sum_{d=d_{\text{free}}}^{\infty} c_d(m) P_d \tag{35}$$

where P is the puncturing period, P_d is the probability that the wrong path at distance d is selected, and c_d is the distance spectra. Depending on the channel coding rate, c_d will be different. For BSC, P_d is simply $P_d = {P_e}^d$, where P_e is the cross over probability. For AWGN channel with BPSK modulation and soft decoding, P_d is:

$$P_d = \frac{1}{2}\mathrm{erfc}\sqrt{\frac{dE_s}{N_0}} \tag{36}$$

where $\frac{E_s}{N_0}$ is the signal-to-noise ratio of the channel. With the above two function definitions, we can write[2]:

$$E[|\hat{X}_k(n_k, m_k) - X_k|^2 | e'_{1,k} ... e'_{i-1,k} e_{i,k}] = f_k(i) \tag{37}$$

$$P(e'_{1,k} ... e'_{i-1,k} e_{i,k}) = g(m_{i,k}) \prod_{j=1}^{i-1} [1 - g(m_{j,k})] \tag{38}$$

where $g(m_{j,k})$ is the resulting error probability if $m_{j,k}$ channel bits are used to protect bit j of subband k. Substituting into the previous equation:

$$\begin{aligned} d_k(n_k, m_k) &= E[|\hat{X}_k(n_k, m_k) - X_k|^2] \\ &= \sum_{i=1}^{n_k} g(m_{i,k}) f_k(i) \prod_{j=1}^{i-1} [1 - g(m_{j,k})] \\ &\quad + h_k(n_k) \prod_{j=1}^{n_k} [1 - g(m_{j,k})] \end{aligned} \tag{39}$$

B. Results

To demonstrate the effectiveness of the GSG algorithm, we construct Figure 12. The two plots represent two subband distortion functions as functions of source bits n and channel bits m. Our goal is to to minimize the sum of the two distortions in the form of (5), such that the sum of source bits and sum of channel bits do not exceed $(R_s^{\text{target}}, R_c^{\text{target}}) = (6, 7)$. Figure 13$a$ shows the pivot rate pairs we obtain in each iteration. For each iteration, Figure 13b maps the corresponding three or four pivot rate pairs onto the R_s-R_c plane. If the algorithm yields a triangular case for one iteration, Figure 13b draws a triangle whose corners are denoted by the three rate pairs. If it yields a quadrangular case, Figure 13b draws a quadrangle whose corners are denoted by the four rate pairs. The target rate pair is denoted by * symbol. We see in Figure 13b that iteration 1 yields a triangular case. When we move to iteration 2 (another triangular case), we found pivot $(2, 4)$, which is in the direction of the target rate. Iteration 6 is a quadrangular case, denoted by the parallelogram 6. We see that the next cluster of pivots is closer to the target pair that the previous. In Figure 13b, we see that the algorithm terminates after 15 iterations. In this case, we found the optimal solution.

To test the overall algorithm numerically, we combined the 3D scalable video codec [15] and Rate-Compatible Punctured Convolutional Codes [17] to build our joint source/channel codec. For

[2]The resulting error pattern after Viterbi decoding is correlated. To precisely calculate the probability of long sequence of correlated bit errors is difficult, and as such, we estimate the probability that the first i bits of subband k received error-free to $\prod_{j=1}^{i-1}[1 - g(m_{j,k})]$.

source coding, we used 3 levels of spatial and 2 levels of temporal subband decomposition as shown in Figure 14. We used 200 frames of the digitized video "Raiders of the Lost Ark" to construct the operational distortion-rate surfaces, $d_k(n_k, m_k)$'s, and applied our bit allocation strategy to compute the distortion curve as function of source to channel coding ratio $D_i(\frac{R_s}{R_c})$ for various BSC i with $P_e = 0.001$ to 0.05. The total bit budget is $250kbits/s$. We see in Figure 15a that there exist unique distortion minima for various channel state. The distribution of source and channel bits among subbands for channel state $P_e = 0.05$ and source to channel ratio 0.6 is shown in Table III.

We now compare our bit allocation algorithm with the one in [9]. The Lervik & Fischer algorithm solves the 2-dimensional optimization problem in two steps: first, keeping channel bits for each subband fixed, it finds the optimal distribution of source bits by tracing the convex-hull along the source axis; then, keeping source bits fixed, it finds the optimal distribution of channel bits by tracing the convex-hull along the channel axis. The resulting operational source rate, channel rate and distortion for BSC $P_e = 0.05$ and $\frac{R_s}{R_c} = 0.6$ is shown in Table IV. Notice that while the operational source and channel bits for the two algorithms are similar, the distortion is much higher for the Lervik & Fischer algorithm. There are two reasons: first, by solving the 2-dimensional problem in two steps, the algorithm converges to a local minimum instead of a global minimum; second, since the algorithm solves the problem one axis at a time, it can only find solutions that are on a rectangular grid, a subset of all possible solutions.

To show that our optimization strategy is essential in poor channel condition, $P_e = 0.05$, we compare its performance with other codecs that uses ad-hoc bit allocation strategies in Figure 15b. Curve a in Figure 15b, shows the PSNR of the scalable codec under ideal noiseless conditions for 100 frames. The average PSNR in this case is 29.7 dB. Curve b in Figure 15b shows the PSNR of our proposed optimized codec operating at the optimal $R_s/R_c = 0.6$, with unequal error protection as described in earlier sections. The average PSNR in this case is about 2 dB lower than the ideal noiseless case. Curve c in Figure 15b shows the performance of a codec operating at $R_s/R_c = 0.571$ using equal error protection. This codec distributes R_s source bits using one-dimensional bit allocation algorithm that assumes a noiseless channel, then channel codes these source bits with R_c channel bits equally. As seen, the PSNR is about the same as case b for most frames, except for occasional drops of 25 dB. These drops are a direct consequence of the fact that important source bits not adequately protected. Finally curve d in Figure 15b shows the performance of the same equal error protection codec as in c but operating at $R_s/R_c = 0.44$. In this case, the source bits are protected adequately, but the insufficiency of source bits causes the quantization error of source coding to dominate the resulting error. Curve d has a 3 dB drop from curve b. The main conclusion to be drawn from Figure 15b is that optimal source/channel bit distribution does make a significant difference in poor channel condition scenarios.

VI. Conclusion

In this paper, we have presented a methodology to optimally allocating source and channel bits for video transmission over noisy channels. In particular, an optimal bit allocation strategy that is efficient and yields near-optimal solutions is presented. Our development of the theory shows that our solution is very close to optimal, and our results prove that in poor channel conditions, an optimal bit allocation scheme is essential to maintain good visual quality. Although we have discussed our algorithm in the context of video transmission over noisy channels, we feel that our strategy can be applied to other optimization problems with two constraints, such as power or complexity.

VII. Acknowledgments

The authors are grateful for the reviewers' valuable comments on the earlier versions of this manuscript.

Appendix

I. Proof of Pivot Point Selection Rule

We now prove that the using the pivot point selection rule we described in in section IV-B.4 enables the algorithm to converge to the ending conditions. We accomplish that by showing a special metric, which tracks the progress of the algorithm at each iteration, decreases or remains the same at each iteration. This indicates that the algorithm yields a cluster of points that are at least as close to the target as the previous set.

We will start by showing that the selection rule is feasible: until we reach the ending condition, there always exists two pivot rate pairs that yields a pivot line segment that crosses or touches the distance line segment at every iteration. We will divide the proof into two cases: 1) when the distance line segment is a perpendicular drop from the target X^{target} to the interior of the pivot line segment; and, 2) when the distance line segment is a line connecting the target and one of the current pivot pairs X^1, X^2. Lemma 1 proves the first case; Lemma 2 proves the second.

Lemma 1 Given the relative locations of the pivot pairs and the target are as in case 1. Then either we have reached ending condition, or there exists one set of pivot pairs such that it creates a pivot line segment that crosses the current distance line segment.

Proof 1 Let suppose the two pivot pairs, X^1 and X^2, are on the R_c axis, as shown in Figure 16ia. We can do that without loss of generality because a simple change of basis and a linear translation can move any set of points to this configuration. We first divide the search space of new pivot rate pair(s) into three disjoint subspaces, A, B and C, as seen in Figure 16ib. These subspaces resulted from lines that connect the pivot rate pairs and the target and the distance line segment.

We can discover one or two pivot rate pairs, resulting in Triangular or Quadrangular case. Let us suppose it is the former first. If the next pivot pair is found in region A, as in Figure 16ic, then it is clear that we have reached the ending condition. If the next pivot pair is in region B, as in Figure 16id, then selecting pivot pairs X^* and X^2, we have a new pivot line segment, a', that crosses the current distance line segment b. By symmetry, we can also conclude that if pivot pair is found in Region C, by selecting X^* and X^1, we have a new pivot line segment that crosses the current distance line segment as well.

Suppose two pivot pairs are found, resulting in the Quadrangular case. If either one of the left or right pivot pairs, $X^{*,1}$ or $X^{*,2}$, is found in region A (Figure 16ii$a$), then we have reach the ending condition. If $X^{*,1}$ is found in region B with $X^{*,2}$ in region C, such that the line connecting them is above the target (Figure 16ii$b$), we have again reached ending condition. If the line is below the target (Figure 16ii$c$), then that line is the new pivot line segment and it crosses the distance line segment. If $X^{*,1}$ and $X^{*,2}$ are both in Region B, then by selecting $X^{*,2}$ and X^2 as pivot pairs, we have a new pivot line segment that crosses the distance line segment. Finally, by symmetry, if $X^{*,1}$ and $X^{*,2}$ are both in Region C, then then by selecting $X^{*,1}$ and X^1 as pivot pairs, we have a new pivot line segment that crosses the distance line segment. □

Lemma 2 Given the relative locations of the pivot pairs and the target are as in case 2. Then at least one of three cases must be true: i) we have reached ending condition; ii) there exists at least one set of two pivot pairs that yields a pivot line segment that crosses the current distance line segment; and iii) there exists at least one set of two pivot pairs that yields a pivot line segment that touches the current distance line segment at the end point.

Proof 2 We follow similar procedure as in Proof 1, and divide the space into three subspaces, A, B and C. First suppose it is Triangular case, as seen in Figure 17i. If pivot pair lands in region A, we have reached ending condition. If pivot pair lands in region B, by selecting X^* and X^2, the new pivot line segment a' touches distance line segment b at X^2. If pivot pair lands in region C, then by selecting X^* and X^1, the new pivot line segment a' crosses b.

Suppose it is Quadrangular case, as seen in Figure 17ii. If either $X^{*,1}$ or $X^{*,2}$ lands in region A, then we have reached the ending condition. If $X^{*,1}$ is found in region B with $X^{*,2}$ in region C, such that the line connecting them is above the target (Figure 17ii$a$), we have again reached ending condition. If the line is below the target (Figure 17ii$c$), then that line is the new pivot line segment and it crosses the distance line segment. If $X^{*,1}$ and $X^{*,2}$ are both in Region B (Figure 17ii$b$), then by selecting $X^{*,2}$ and X^2 as pivot pairs, we have a new pivot line segment that touches the distance line segment. Finally, if $X^{*,1}$ and $X^{*,2}$ are both in Region C (Figure 17ii$d$), then then by selecting $X^{*,1}$ and X^1 as pivot pairs, we have a new pivot line segment that crosses the distance line segment. □

Using Lemma 1 and 2, we can now prove that the selection rule ensures the algorithm is making progress.

Theorem 3 Suppose there exists a cluster of points that satisfy the ending condition. The selection method stated in section IV-B.4 terminates in that cluster.

Proof 3 We first define a special metric, d_i, as the length of the distance line segment at iteration i. It measures how close a cluster of points is from the target. By definition of distance line segment, if the pivot line segment of the next cluster is closer to the target, then it will have a smaller metric than previous cluster.

If the next pivot line segment crosses the current distance line segment, then length of the new distance line segment must be smaller than the previous one. This is best illustrated geometrically. In Figure 18a, we see pivot pairs P^* and X^2 yields a pivot line segment that crosses the current distance line segment. We denote the point of intersection as I. Clearly distance between target X^{target} and I, e_{i+1}, is strictly smaller than the length of distance line segment, d_i. Further, The next distance line segment, b', must have length d_{i+1} smaller than or equal to length of e_{i+1}. We can conclude the following: if next pivot line segment crosses the current distance line segment, then $d_{i+1} \leq e_{i+1} < d_i$. By Lemma 1, we know such pivot line segment always exists if we are in case 1. Therefore, we can conclude that the metric must strictly decrease for the next iteration if we are in case 1.

Notice in case 2, the only time there is no pivot line segment that crosses the distance line segment is when the new pivot(s) is(are) in region B, shown in Figure 17ic, 17ii$b$. In such cases, the distance metric might remain the same between iterations. In Figure 18b, by selecting P^* and X^2, the metric remains the same. However, as the algorithm continues to progress, this situation cannot remain. Since the search space is continually being rotated, it will eventually reach a pivot pair such that the metric will decrease. In Figure 18b, the new pivot pair goes from P^* to Q^* to R^*. When we reach R^*, the distance line segment is a perpendicular drop (case 1) and the metric is decreased. Therefore we can conclude that the metric must eventually decrease if we are in case 2.

Since the metric continues to decrease as the algorithm progresses, the cluster moves closer to the target. The cluster that can make no more progress is the one with the ending condition. Therefore, the algorithm converges to the cluster with the ending condition. □

References

[1] R.Cox, J.Hagenauer, N.Seshadri and C.Sundberg, "Subband Speech Coding and Matched Convolutional Channel Coding for Mobile Radio Channels,"*IEEE Trans. SP*, vol.39, pp.1717-1731, August 1991.

[2] H. Shi, P. Ho, and V. Cuperman, "Combined Speech and Channel Coding for Mobile Radio Communications", *IEEE Transactions on Vehicular Technology*, Vol. 43, No. 4, pp. 1078- 1087, November 1994.

[3] N. Tanabe, N. Farvardin, "Subband Image Coding Using Entropy-Coded Quantization over Noisy Channel," *IEEE Jour. Comm.* vol. 10, No. 5, pp.926-942, June 1992.

[4] J. J. Huang and P. M. Schultheiss, "Block quantization of correlated Gaussian random variables," *IEEE Trans. Commun. Sys.*, vol. CS-11, pp.289-296, Sept, 1963.

[5] Y.Shoham and A.Gersho, "Efficient Bit Allocation for an Arbitrary Set of Quantizers," *IEEE Trans. ASSP*, vol.36, pp.1445-1453, September 1988.

[6] P. H. Westerink, J. Biemond, and D.E. Boekee, "An Optimal Bit Allocation Algorithm for Subband Coding," *ICASSP*, pp.757-760, 1988.

[7] Eve A. Riskin, " Optimum Bit Allocation via the Generalized Breiman, Friedman, Olshen, and Stone Algorithm." *IEEE Trans. on Info. Theory.* 37(2):400-402, March 1991.

[8] D. Bertsekas, *Nonlinear programming*, Belmont: Athena Scientific, 1995.

[9] J. Lervik and T. Fischer, "Robust Subband Image Coding for Waveform Channels with Optimum Power- and Bandwidth Allocation," *ICASSP 97*, pp.3089-3092, 1997.

[10] V. Goyal and M. Vetterli, "Computation-Distortion Characteristics of Block Transform Coding," *ICIP 97*, pp.2729-2732, 1997.

[11] B. Hochwald, K. Zeger, "Tradeoff Between Source and Channel Coding," *IEEE Trans. on Info. Theory*, vol.43, No.5, pp.1412-1424, September 1997.

[12] H. Jafarkhani, P. Ligdas, N. Farvardin, "Adaptive Rate Allocation in a Joint Source/Channel Coding Framework for Wireless Channel," *IEE 46th Vehicular Tech. Conf 96*, 1996.

[13] J. Garcia-Frias, D. Benyamin and J. Villasenor, "Rate-distortion-optimal Parameter Choice in a Wavelet Image Communications System," *ICIP 97*, pp.25-28, 1997.

[14] G. Sherwood and K. Zeger, "Progressive Image Coding for Noisy Channels," *IEEE Signal Processing Letter*, vol.4, No.7, pp.189-191, July 1997.

[15] D.Taubman and A.Zakhor, "Multirate 3-D Subband Coding of Video," *IEEE Trans. Image Proc.*, vol.3, pp.572-88, September 1994.

[16] http://www-video.eecs.berkeley.edu/download/scalable

[17] J.Hagenauer, "Rate-Compatible Punctured Convolutional Codes (RCPC Codes) and their Applications," *IEEE Trans. Comm.*, vol.36, pp.389-399, April 1988.

[18] G. Cheung, Master's thesis, UC Berkeley, 1998.

K	number of subbands
i	Channel State Information for channel i
$D_i(\alpha)$	signal distortion for channel i when R_s/R_c is α
n_k	source bits used for kth subband
m_k	channel bits used for kth subband
$m_{i,k}$	channel bits used to protect bit i of subband k
$d_k(n_k, m_k)$	distortion function of kth subband given n_k, m_k

TABLE I

Table of Notations

$\mathbf{x}_k^o = (n_k^o, m_k^o)$	optimal point for subband k, given multiplier set (λ, μ)
(R_s^o, R_c^o)	operational source-channel pair, given multiplier set (λ, μ)
$\mathbf{x}_p^{o,1} = (n_p^{o,1}, m_p^{o,1})$	pivot point 1 in pivot subband p
$(R_s^{o,1}, R_c^{o,1})$	operational source-channel pair 1 using pivot point 1 as subband minimum, and multiplier set (λ, μ)
$\mathbf{x}_k^* = (n_k^*, m_k^*)$	candidate pivot point of subband k
C_k	region of eligibility of subband k
$\bar{d}_k(n_k, m_k)$	$= d_k(n_k, m_k) + \lambda n_k + \mu m_k$; tilted distortion function of kth subband

TABLE II

Table of Notations for GSG Algorithm

spatial	temporal	source n_k	channel m_k
1	0	8211.84	21075.36
1	1	4101.12	7611.12
1	2	6342.72	12546.48
1	3	4322.88	8436.00
2	0	5645.76	11361.60
2	1	2304.96	3772.80
2	2	3613.44	6345.60
2	3	1850.88	2921.52
3	0	4911.36	9591.60
3	1	2882.88	4750.80
3	2	4179.84	7608.48
3	3	4017.60	3841.44
4	0	2873.28	4906.56
4	1	528.00	792.00
4	2	2501.76	3990.72
4	3	780.48	1170.72
5	0	8339.52	16181.04
5	1	780.48	1170.72
5	2	5527.68	2765.76
5	3	1389.12	2083.68
6	0	4907.52	8925.84
6	1	876.48	1314.72
6	2	4581.12	1846.32
6	3	1055.04	1582.56
7	0	1432.32	2198.64
7	1	0.0	0.0
7	2	0.0	0.0
7	3	0.0	0.0
8	0	4244.16	7541.04
8	1	0.0	0.0
8	2	0.0	0.0
8	3	0.0	0.0
9	0	564.48	846.72
9	1	0.0	0.0
9	2	0.0	0.0
9	3	0.0	0.0
10	0	360.96	0.0
10	1	0.0	0.0
10	2	0.0	0.0
10	3	0.0	0.0

TABLE III

DISTRIBUTION OF SOURCE AND CHANNEL BITS USING GSG ALGORITHM (IN BITS/S)

algorithm	R_s^o	R_c^o	D^o
Lervik & Fischer	94281 bps	157757 bps	262.14
proposed GSG	93128 bps	157180 bps	170.39

TABLE IV

COMPARISON OF BIT ALLOCATION ALGORITHMS: $(R_s^*, R_c^*) = (94744$ BPS, 156240 BPS$)$

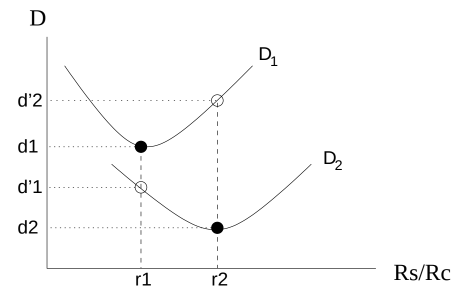


Fig. 1. Distortion as Function of Source to Channel Rate

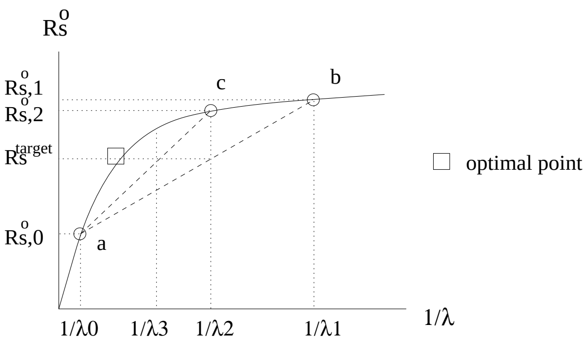


Fig. 2. Example of Linear Lagrange Multiplier Search

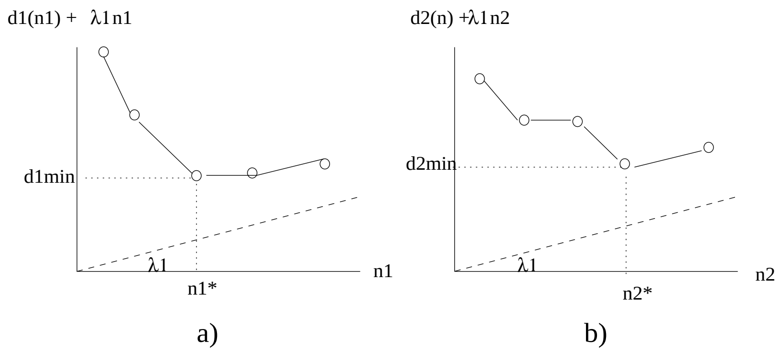


Fig. 3. Example of 1-D Lagrange Multiplier Problem

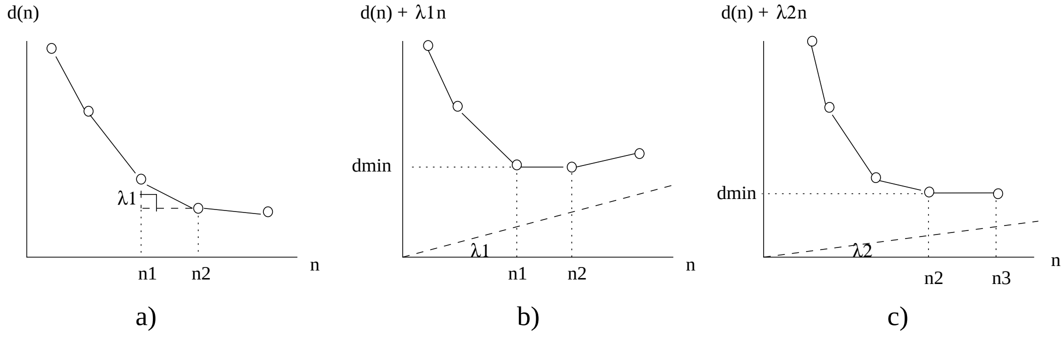


Fig. 4. Example of 1-D Singular Point Multipliers

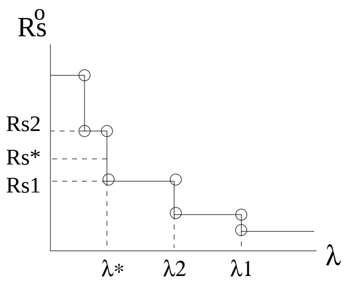


Fig. 5. Operational Source Rate vs. Multiplier

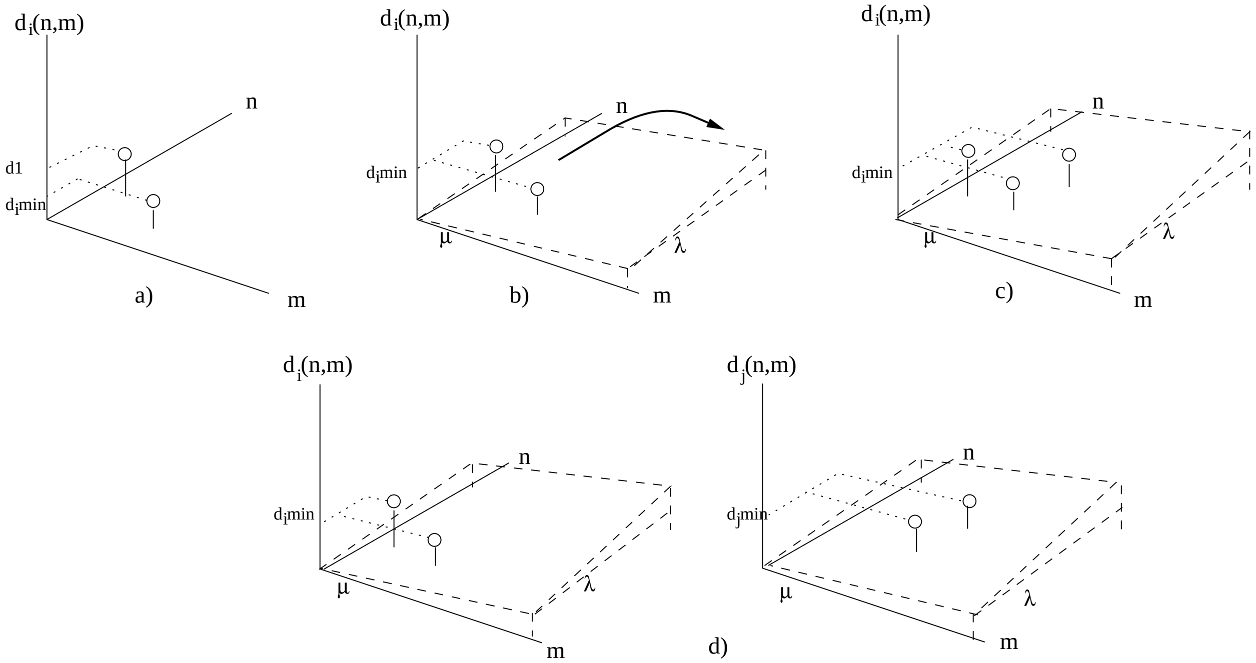


Fig. 6. Geometrical Interpretation of Singular Points in 2-D

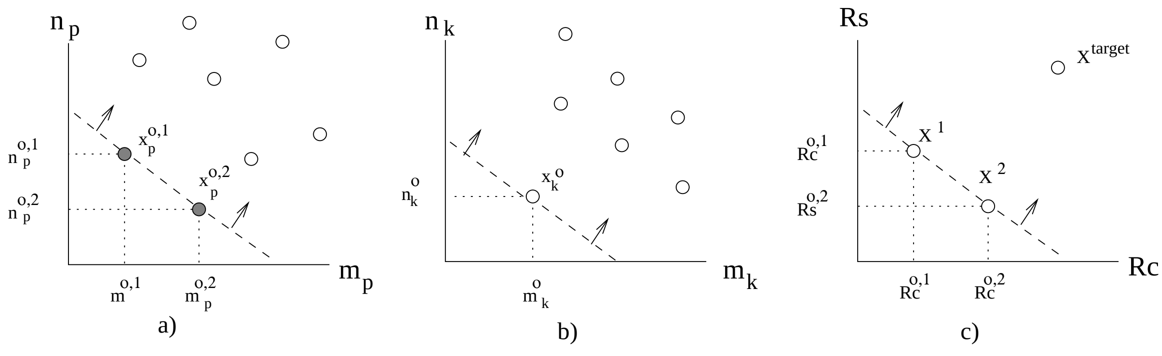


Fig. 7. Line and Region of Eligibility for GSG Algorithm

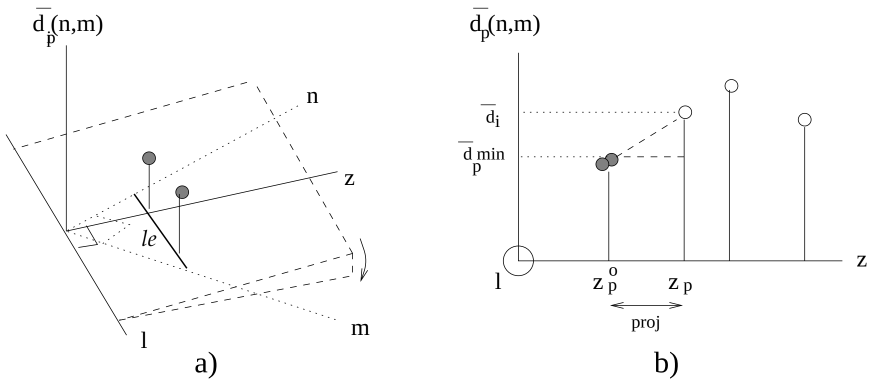
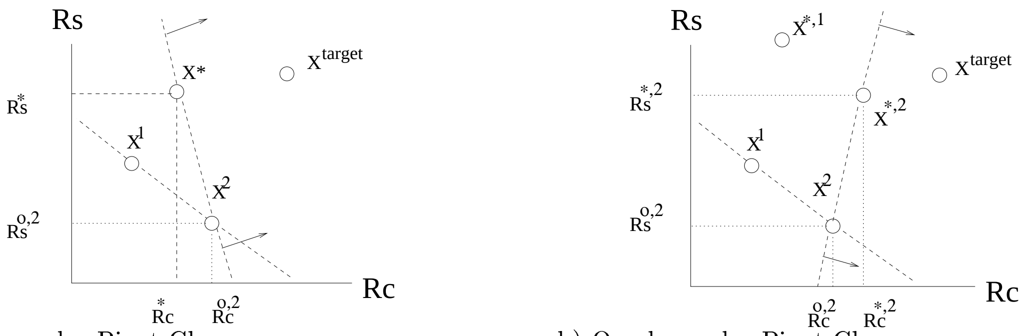


Fig. 8. Change of Basis for GSG Algorithm



a) Triangular Pivot Change

b) Quadrangular Pivot Change

Fig. 9. Example of Pivot Change in Triangular and Quadrangular Case

a) Triangular

b) Quadrangular

Fig. 10. Pivot Point Selection Algorithm

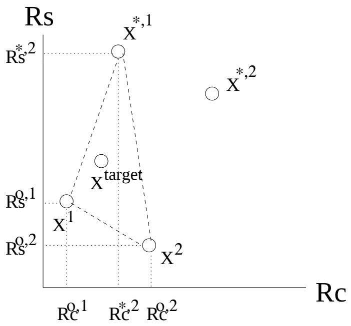
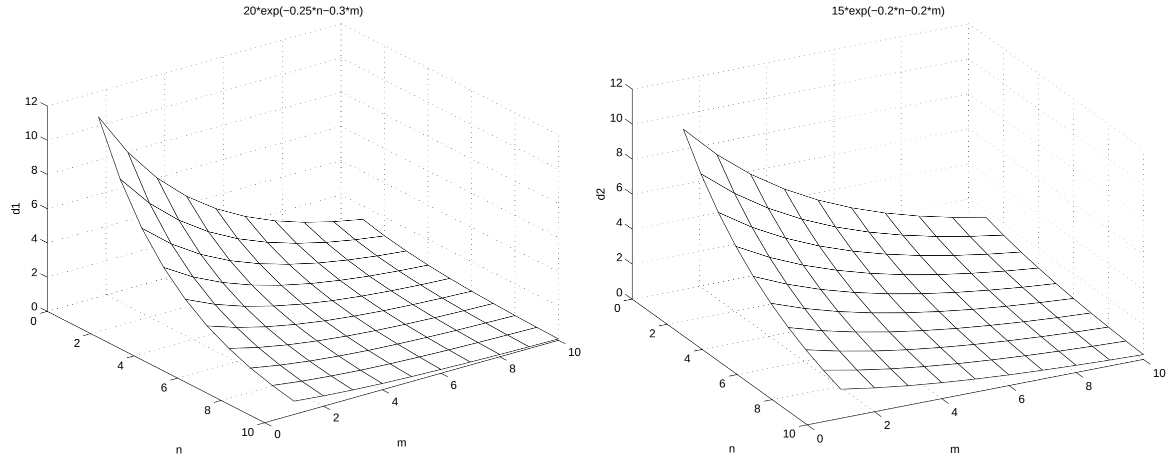


Fig. 11. Time Sharing between Neighboring Approximate Solutions

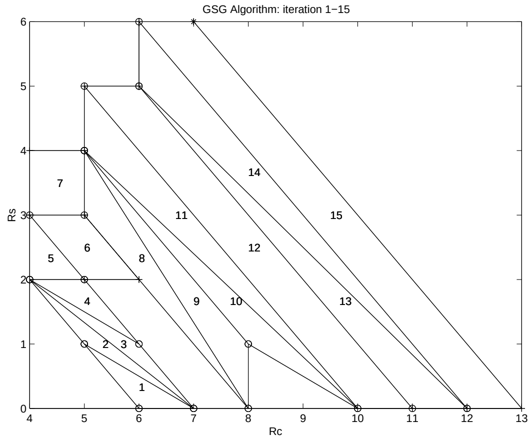


a) Subband 1 Distortion Function   b) Subband 2 Distortion Function

Fig. 12. Original Distortion Functions

iteration	$(R_s^{o,1}, R_c^{o,1})$	$(R_s^{o,2}, R_c^{o,2})$	$(R_s^{*,1}, R_c^{*,1})$	$(R_s^{*,2}, R_c^{*,2})$
1	(0,6)	(1,5)	(0,7)	
2	(0,7)	(1,5)	(2,4)	
3	(0,7)	(2,4)	(1,6)	
4	(1,6)	(2,4)	(2,5)	
5	(2,5)	(2,4)	(3,4)	
6	(2,5)	(3,4)	(2,6)	(3,5)
7	(3,4)	(3,5)	(4,4)	(4,5)
8	(3,5)	(4,5)	(0,8)	
9	(4,5)	(0,8)	(1,8)	
10	(4,5)	(1,8)	(0,10)	
11	(4,5)	(0,10)	(5,5)	
12	(0,10)	(5,5)	(0,11)	(5,6)
13	(0,11)	(5,6)	(0,12)	
14	(5,6)	(0,12)	(6,6)	
15	(0,12)	(6,6)	(0,13)	(6,7)

a) Table Representation of Rate Pairs



b) Graphical Representation of Rate Pairs

Fig. 13. Procession of GSG Algorithm

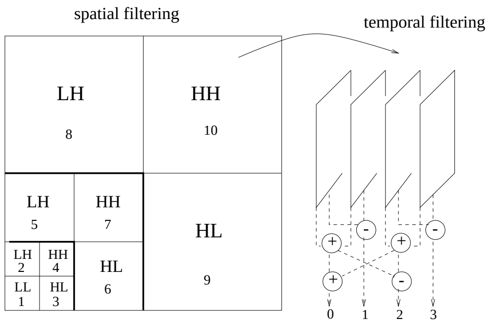
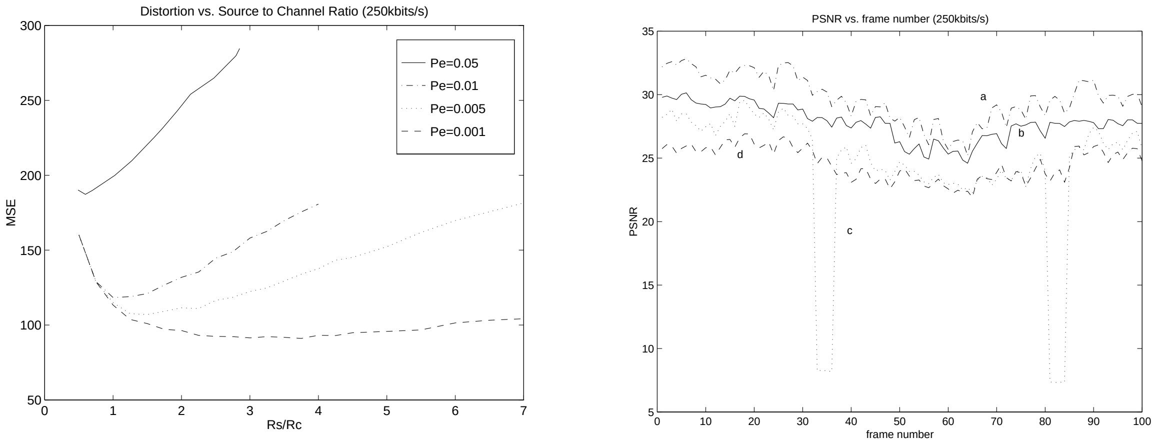


Fig. 14. Spatial and Temporal Subband Decomposition



a) MSE vs. R_s/R_c for diff. Channel State

b) PSNR vs. Frame Number

Fig. 15. Experimental Results

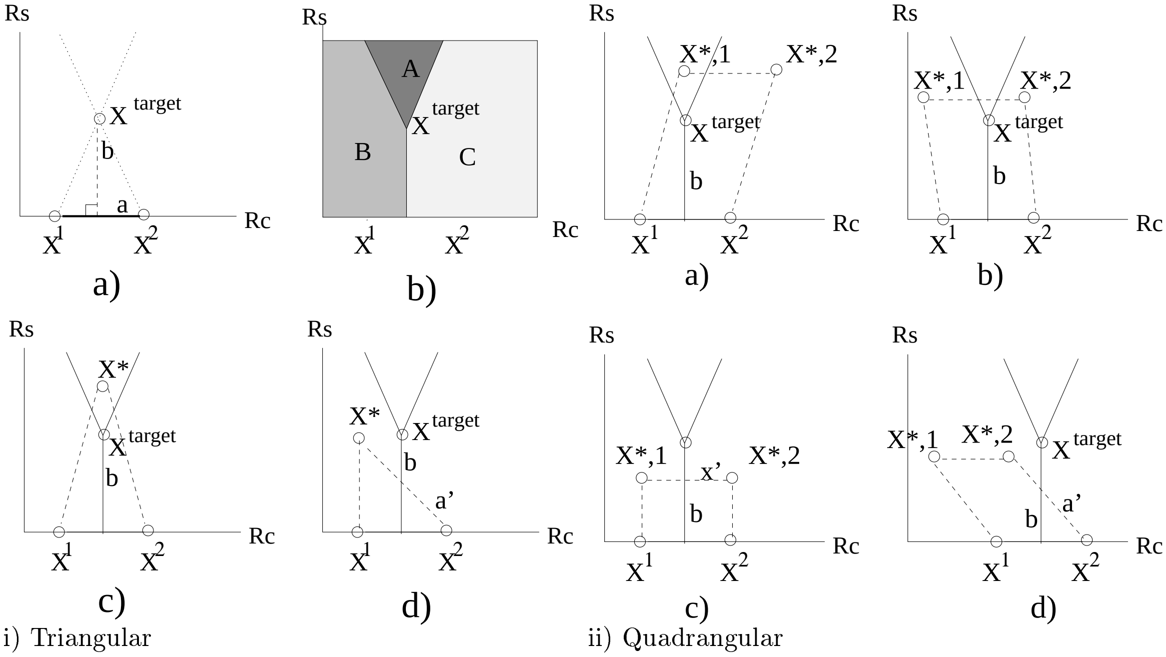


Fig. 16. Proof of Pivot Point Selection Rule: case 1

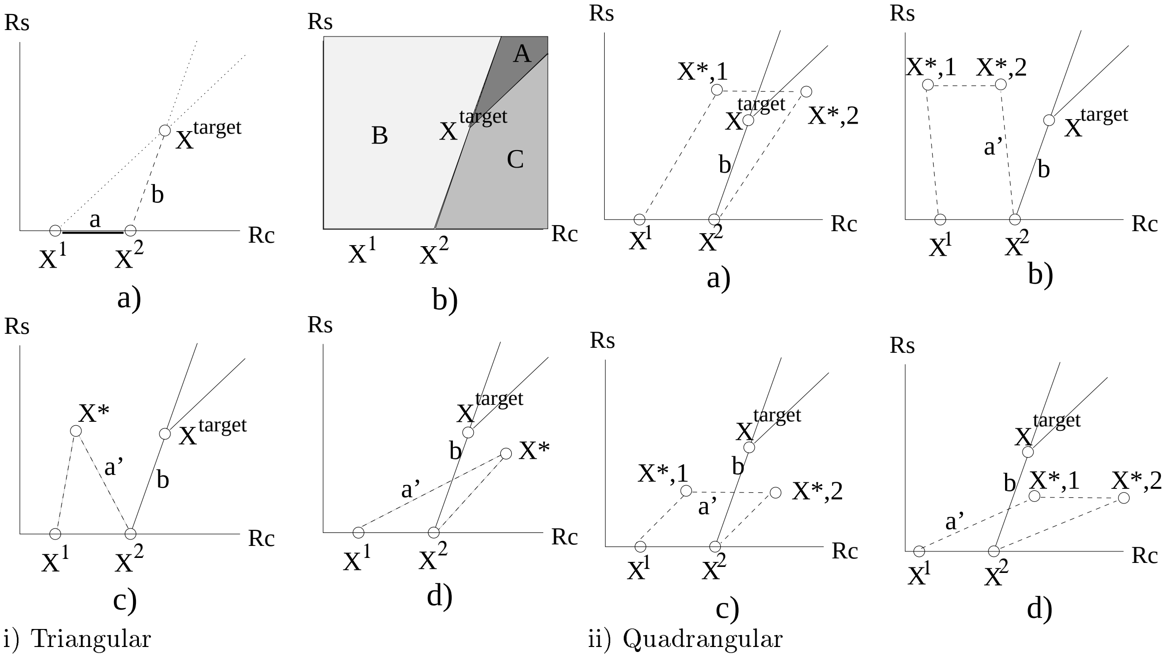


Fig. 17. Proof of Pivot Point Selection Rule: case 2

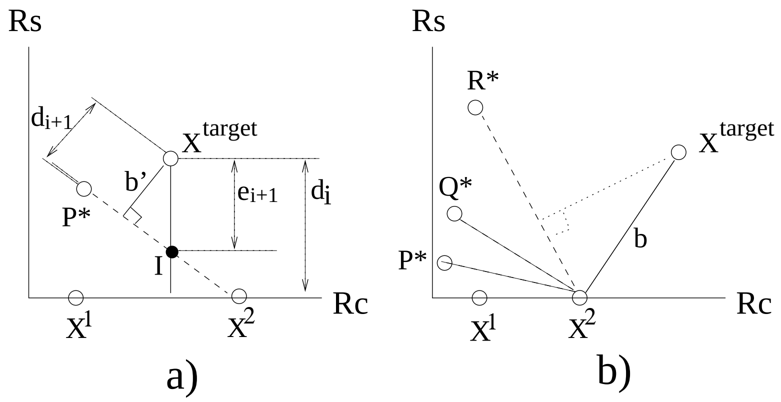


Fig. 18. Proof of Theorem 3