# Initial Investigations into Joint Source/Channel Video Coding for Nonstationary Channels with Adaptable Residual Vector Quantization

Hezekiah O. Adeyemi
Clark Atlanta University
Atlanta, GA

Christopher F. Barnes
Georgia Institute of Technology
Atlanta, GA

## Abstract[1]

*An adaptable joint source/channel coding system based on residual vector quantization (RVQ) is being developed for channels with nonstationary noise characteristics. The system consists of a residual vector quantizer that uses different numbers of stages for different input vectors. The variable rate RVQ used in this adaptable system is designed with a new design process. This design process generates, in general, different encoder and decoder codebooks. The encoder codebooks provide computationally efficient sequential search tree structures, and the decoder codebooks satisfy conditions necessary for joint optimality.*

## 1 Introduction

The problem addressed in this research is that of developing channel-matched residual vector quantizer codes to improve the quality of video data transmitted over a nonstationary noisy channel. In particular, the communication problem involves transmitting a digital video signal acquired in real time from an infrared sensor on a heliborne platform to a ground based receiving station via a low-rate radio frequency link. The communication link is subjected to both multiple path effects and fading. The research goal is to develop a low rate video coding system that is robust in the presence of severe channel errors.

The source coding subsystem is based on the use of a multiple stage vector quantizer that is formed by cascading a sequence of vector quantizers (VQs) such that each stage quantizes the residual error of the preceding stage. Such residual vector quantizers (RVQs) are subject to two structural constraints: a sequential search encoder constraint and a direct sum codebook constraint. The purpose of the sequential search encoder constraint is to reduce computation requirements. The purpose of the direct sum codebook constraint, which is imposed at both the RVQ encoder and decoder, is to reduce memory requirements. As a result, the composite RVQ structure returns both computation and memory savings over full search VQs, and memory savings over tree structured VQs. For example, if a $k$-dimensional RVQ with output rate $r$ bits per sample consists of $P$ stages with $N = 2^{kr/P}$ codevectors in each stage, then the memory and computation costs are proportional to $P2^{kr/P}$, whereas the memory or computation cost, or both, of most other VQ structures are proportional to $2^{kr}$.

The most efficient RVQs, in terms of memory and computation, are those with many stages and two or four codevectors per stage (the associated implementation costs are the same and are proportional to the expression $2P$ for the two codevector case). However, most empirical evidence suggests that RVQs with many stages can be expected to give unsatisfactory performance [1]. These poor results for RVQs with many stages have proven, at least for some sources, to be more a result of the design method used to generate the stage codebooks, and not necessarily inherent in the RVQ structure itself.

Necessary conditions for the joint optimality of direct sum codebooks were derived and design methods for satisfying these conditions were suggested in [2]. However, one of the major shortcomings of these design methods was the loss of design monotonicity whenever sequential search encoding was used with direct sum codebooks. In this paper, we discuss a design method that is better suited for merging a jointly optimal direct sum structure with a sequential search structure, and discuss ongoing research results where novel RVQ features resulting from the new RVQ design method may lead to performance advantages in joint source/channel coding systems. However, before describing this design method, it is useful to introduce notation.

The $p$th stage of an RVQ is a $k$-dimensional vector quantizer defined by the mapping $\boldsymbol{Q}_p : \mathbf{R}^k \mapsto \boldsymbol{C}_p$, where $\mathbf{R}^k$ is the $k$-dimensional quantizer input space, and $\boldsymbol{C}_p$ is the $p$th stage *codebook* with $p \in \{1, 2, \ldots, P\}$. Residual vector quantizer stage mappings are collectively equivalent to a single stage mapping $\mathbf{Q} : \mathbf{R}^k \mapsto \mathbf{C}$, where $\mathbf{C}$ is the *direct sum codebook* $\mathbf{C} = \boldsymbol{C}_1 + \boldsymbol{C}_2 + \cdots + \boldsymbol{C}_P$. The stage quantizer mappings $\boldsymbol{Q}_p(\cdot)$ are realized as a composition of an encoder mapping $\boldsymbol{E}_p$ and a decoder mapping $\boldsymbol{D}_p$. That is, $\boldsymbol{Q}_p(\boldsymbol{x}) = \boldsymbol{D}_p(\boldsymbol{E}_p(\boldsymbol{x}))$, where $\boldsymbol{x}$ is a realization of $\boldsymbol{X}$, the random source output. The direct sum quantizer also has a functional composition, $\mathbf{Q}(\boldsymbol{x}) = \mathbf{D}(\mathbf{E}(\boldsymbol{x}))$. A $P$-stage RVQ is said to be op-

[1] This research was supported by the Communications and Electronics Command of the U.S. Army.



1058-6393/95 $4.00 © 1995 IEEE

timal if it gives at least a locally minimum value of the average distortion $\mathcal{D}(\mathbf{E},\mathbf{D}) = E\{d(\boldsymbol{X},\mathbf{D}(\mathbf{E}(\boldsymbol{X})))\}$. Although an abuse of notation, we find it useful in the following discussions to represent the direct sum encoder by $\mathbf{E} = \boldsymbol{E}_1 + \boldsymbol{E}_2 + \cdots + \boldsymbol{E}_P$, and to represent the direct sum decoder by a similar expression.

The remainder of this paper is outlined as follows. Section 2 reviews methods previously used to design the stage codebooks of RVQs and presents the improved design method. Section 3 describes a type of successive approximation RVQ that uses different numbers of stages on different input blocks and shows the results of preliminary experimental investigations. Section 5 describes how the new RVQ structures will be generalized to permit robust joint source/channel coding in nonstationary, severely noisy environments.

## 2 RVQ Design Methods

The first method suggested for designing RVQs was sequential application of the Generalized Lloyd Algorithm (GLA) [3]. However, this design method is increasingly suboptimal as the number of stages increases, and has resulted in the recommendation that such RVQs be limited to only a few stages [1].

Once necessary conditions for the joint optimality of RVQ encoders and decoders were discovered [2], application of the GLA paradigm to develop a joint design approach for RVQs was straight forward (assuming the complexity of optimal encoding is tolerated). However, the joint GLA RVQ design method encounters difficulties when the complexity of optimal encoding is not allowed. Optimal sequential search partitioning for an arbitrary set of stage codebooks often requires stage decision regions that are not connected nor convex, while efficient sequential search encoders generally require well behaved stage partition cells. Using the joint design process with suboptimal encoding rules can lead to nonmonotonic design behavior, and occasionally, catastrophically poor performance results, especially if the RVQ has a high output rate, many stages, or if a small training set is used.

The monotonicity problem of joint design methods has led to the acceptance of various procedures to stabilize the design process [4]. One approach for solving this problem is the use of separate, and in general, different stage codebooks at the RVQ encoder and decoder [5]. However, to use this approach, a modified design method for RVQ is required that is capable of providing separate encoder and decoder codebooks, but where the two sets of codebooks work together to provide good overall RVQ performance. Such a design method is described next.

To correct the lack of monotonicity in the joint GLA RVQ design process, a modified design paradigm based on the following principle can be used. Instead of seeking a new encoder that is optimal for the new (just updated and fixed) decoder (as required by the GLA algorithm), when designing RVQs with sequential search encoders, it is easier to find a new successive approximation encoder that is improved relative to the old successive approximation encoder. This less ambitious design goal is more easily obtained if the encoder and decoder codebooks are permitted to be different.

To describe this design approach, assume that we have a set of fixed direct sum encoder codebooks, then using the procedure of [2], new direct sum decoder codebooks are selected to be jointly optimal for the fixed partition induced by the sequential search structure imposed on the fixed encoder codebooks. Given this new fixed set of decoder codebooks, and the fixed set of old encoder codebooks, a combination of some subset of the old encoder stage codebooks and some subset of the new decoder stages codebooks is selected to produce a new sequential search encoder with improved performance relative to the performance of the previous sequential search encoder. The trick to this design method is finding efficient rules for choosing different stages from the different systems. One such rule is described next, along with a more detail description of this separate encoder/decoder codebook design process.

Suppose that during the design process we have a given fixed encoder $\mathbf{E}^i$ (where $i$ is an index for the iterative design loop) and the joint decoder optimization process has just been completed to yield the decoder $\mathbf{D}^i$. Since the joint decoder process yields codebooks that satisfy conditions necessary for minimum distortion, this decoder optimization step is guaranteed to deliver $\mathcal{D}(\mathbf{D}^i) \leq \mathcal{D}(\mathbf{E}^i)$. Next, as a first step towards finding an improved encoder, set

$$\mathbf{E}^{i+1} = \mathbf{D}^i = \boldsymbol{D}_1^i + \boldsymbol{D}_2^i + \cdots + \boldsymbol{D}_P^i, \qquad (1)$$

and evaluate $\mathcal{D}(\mathbf{E}^{i+1})$. Since the use of the updated direct sum structure $\mathbf{D}^i$ in the sequential search encoder $\mathbf{E}^{i+1}$ is not guaranteed to be effective, it is not guaranteed that $\mathcal{D}(\mathbf{E}^{i+1})$ will be equal to or less than $\mathcal{D}(\mathbf{D}^i)$. (Remember that $\mathcal{D}(\mathbf{D}^i)$ is evaluated under the partition associated with the old $\mathbf{E}^i$, and $\mathcal{D}(\mathbf{E}^{i+1})$ is evaluated under the partition that would be associated with the use of the new $\mathbf{D}^i$ in a sequential search encoder.) However, considering the fact that the sequential search structural constraint is imposed only at the RVQ encoder and not at the decoder, a more appropriate design goal would be to seek for an encoder $\mathbf{E}^{i+1}$ such that $\mathcal{D}(\mathbf{E}^{i+1}) \leq \mathcal{D}(\mathbf{E}^i)$. That is, we seek a new sequential search encoder that performs no worse than the last sequential search encoder. This modified design objective is reasonable in that, in this case, knowledge of a necessary condition does not exist for finding a sequential search encoder that is *both optimal and efficient* given a fixed direct sum decoder.

Now, what if the new encoder is such that $\mathcal{D}(\mathbf{E}^{i+1}) > \mathcal{D}(\mathbf{E}^i)$? In this case we suggest the following additional modification to the above design process. Based on the conjecture that it is the decoder design changes at the first few stages of the RVQ that causes tree structure entanglement at the latter stages [2], it is reasonable to try a sequential search encoder based on the first few stages of the old encoder $\mathbf{E}^i$ and the last few stages of the new decoder $\mathbf{D}^i$. That is, a new encoder is tentatively selected as

$$\mathbf{E}^{i+1}(s) = \boldsymbol{E}_1^i + \boldsymbol{E}_2^i + \cdots + \boldsymbol{E}_{s-1}^i + \boldsymbol{D}_s^i + \cdots + \boldsymbol{D}_P^i, \qquad (2)$$

where $s$ is the index of what we choose to call the pivot stage. The new $\mathbf{E}^{i+1}(s)$ is then tested for improved performance. If $\mathcal{D}(\mathbf{E}^{i+1}(s)) < \mathcal{D}(\mathbf{E}^{i})$, then an improved encoder has been discovered and the design process can proceed to the next decoder optimization step. Otherwise, if $\mathcal{D}(\mathbf{E}^{i+1}(s)) \geq \mathcal{D}(\mathbf{E}^{i})$, then $s$ is incremented to $s'$ and a new $\mathbf{E}^{i+1}(s')$ is tested to see if $\mathcal{D}(\mathbf{E}^{i+1}(s')) < \mathcal{D}(\mathbf{E}^{i})$. This process of incrementing the pivot stage and of testing the resulting direct sum structure as a sequential search encoder is continued until either $\mathcal{D}(\mathbf{E}^{i+1}(s)) < \mathcal{D}(\mathbf{E}^{i})$ or until $s = P$. If $s = P$, then $\mathbf{E}^{i+1}(s) = \mathbf{E}^{i}$ and $\mathcal{D}(\mathbf{E}^{i+1}(s)) = \mathcal{D}(E^{i})$, and the entire process can either be stopped, or a new stage can be added to the RVQ. Experiments have shown that there usually exist combinations of old encoder and new decoder stage codebooks with $s < P$ that form a new set of RVQ encoder codebooks that give improved sequential search performance (relative to the previous set of old encoder codebooks).

## 3 VR-RVQ Video Coding

Most of the implementation efficiency of an RVQ over a full search VQ is obtained in going from one stage to two stages. However, there are RVQ coding systems and applications where the use of more than two stages is advantageous. One such system is a variable block rate residual vector quantizer (VR-RVQ) that uses different numbers of stages on different input blocks to achieve enhanced localized fidelity [8]. In this case, the use of a large number of stages permits fine-grain rate control.

When transmitting compressed video data through a noisy channel, prediction techniques such as motion compensation are highly susceptible to channel bit errors. Residual vector quantizers, on the other hand, have been shown to be inherently immune to channel bit errors [10]. To extend the noise-immunity of the RVQ coding system to the coding of video motion information, we have expanded the conventional 2-dimensional (2-D) spatial image blocks used by vector quantizers to 3-dimensional (3-D) video cubes. The video cubes consist of 2-spatial dimensions and 1-temporal dimension, and thus allow the RVQ to exploit temporal redundancy.

By applying a VR-RVQ to 3-D video cubes, the variable rate control of VR-RVQ provides enhanced localized fidelity in temporal-spatial subregions of the video sequence data volume that contain either spatial high contrast boundaries (image edges) or localized movement. A simple rate allocation algorithm can be used with VR-RVQ to assign different numbers of RVQ stages to different vectors formed from mean-removed 3-D video cubes. For example, the sum-of-squared-pixel values of the mean-removed video cube elements, and a set of sum-of-squared-pixel value thresholds similar to those employed in [6] were used in this paper to determine which RVQ stages to use on each 3-D video cube.

In our initial VR-RVQ video coding experiments, the video cube dimensions were selected to be $16 \times 16$ pixels in spatial extent, and to be 1 second long in temporal duration. Since VR-RVQ codecs were simulated that operated at different video frame rates, this definition of the video cube size implies that the number of samples in the VR-RVQ vectors varies as a function of the frame rate. For example, for frames rates of 5, 10, 15, and 30 frames per second (fps), the corresponding VR-RVQ vector sizes are as follows:

$$\underbrace{(5\text{ Frames})}_{\text{One Second}} \times \underbrace{(16 \times 16)}_{\text{2-D Block Size}} = 1,280 \text{ Pixels per Vector}$$

$$\underbrace{(10\text{ Frames})}_{\text{One Second}} \times \underbrace{(16 \times 16)}_{\text{2-D Block Size}} = 2,560 \text{ Pixels per Vector}$$

$$\underbrace{(15\text{ Frames})}_{\text{One Second}} \times \underbrace{(16 \times 16)}_{\text{2-D Block Size}} = 3,840 \text{ Pixels per Vector}$$

$$\underbrace{(30\text{ Frames})}_{\text{One Second}} \times \underbrace{(16 \times 16)}_{\text{2-D Block Size}} = 7,680 \text{ Pixels per Vector}$$

The implementation efficiencies of the VR-RVQ and the design stability of the new RVQ design process permit the use of such large vector sizes. However, the use of 3-D cubes requires that video frames be stored in a buffer. The length of the buffer is determined by the temporal extent of the video cube.

Computer simulations were conducted to determine the effectiveness of the adaptive RVQ coding algorithm. For an input data format of (160 × 120 pixels/frame) × (8 bits/pixel) = 153.6 Kilobits per frame, VR-RVQ systems with coded output rates of 19.2 Kilobits per second (Kbps), 9.6 Kbps, and 4.8 Kbps were simulated and evaluated. For a frame rate of 30 fps, these target output rates mandate RVQ compression systems that provide, respectively, 240:1 (bits in:bits out), 480:1, and 960:1 compression ratios. For a frame rate of 5 fps, these target output rates mandate RVQ compression systems that provide, respectively, 40:1, 80:1, and 160:1 compression ratios, or equivalently, output data rates of 0.2 bits per pixel (bpp), 0.10 bpp, and 0.05 bpp. The VR-RVQ codec frame rate was controlled by discarding video frames.

The RVQ system used 32 codevectors per stage, and the total number of stages was 128. However, only a subset of the 128 stages were used to encode each of the input source data vectors. Local and global rate allocation algorithms were used to determine exactly which and how many stages were used on each input vector [6]. Example compression results are shown in Figure 1.

As a side note, preliminary experiments indicate that it is possible for the RVQ encoder and decoder to operate at different frame rates. In some situations, it may be advantages to have the RVQ encoder operate at a low frame rate to reduce encoder computation complexity, and the RVQ decoder to operate at a high frame frame to increase the quality of coded video movement. The use of larger vectors at the decoder seems to incur only a slight loss in single frame image quality but provides a smoother representation of motion without an increase in the transmitted bit rate.

## 4 Combined Source/Channel Coding

A primary objective of this research effort is to further generalize the RVQ design process to incorporate joint source/channel coding into the RVQ codebooks. Such channel-matched multiple stage VQ codes have been shown to be rather immune to very noisy channels [10]. We anticipate that with the capability of using different encoder and decoder RVQ codebooks, it should be possible to develop a channel-matched RVQ coding system for nonstationary channels. For example, assume that a nonstationary channel can be adequately described by a small set of channel states and that a channel state estimator is employed at the receiver. The use of a the channel state estimator and a set of channel-match RVQ decoder codebooks would allow the joint source/channel coding system to adapt to changing channel characteristics. The development of this adaptive combined coding system is facilitated by the new RVQ design process: the encoder codebooks will be designed under the assumption of a noiseless channel (or, perhaps, under a model that describes the long-time average characteristics of the channel), and the set of RVQ decoder codebooks will be designed such that each codebook is matched to a particular channel state. Furthermore, the product code structure of channel-matched residual vector quantizers with many stages (and sufficiently large stage codebook sizes) is expected to provide an additional degree of noise immunity.

Work has begun on an adaptive channel-matched VR-RVQ coding system that operates in the following way. First, the vector means and the numbers of stages used for each RVQ vector are channel coded and transmitted to the receiver. The mean-residual vectors are then encoded with a sequence of RVQ encoder stages designed for a noiseless channel. The RVQ codevector indexes output by the RVQ stages are then transmitted to the receiver without channel coding. At the receiver, the overhead information is channel decoded, and the frequency of detected and corrected errors is used to estimate the channel state (bit error probability). The channel state estimate is then used to select one of a set of RVQ decoder codebooks that is matched to the estimated channel state.

By applying forward error correction channel coding to the important overhead information (comprised of vector mean values and the number of bits resulting from entropy encoding the RVQ codevector index stream), the channel noise immunity of the RVQ system is enhanced by essentially eliminating the possibility of loosing source code synchronization.

## 5 Summary

A new design process for RVQs has been described and the extension of RVQ to video coding with the use of 3-D video cubes has been introduced. The advantages of the 3-D data cube method for forming VQ vectors is that 1) improved noise immunity compared to conventional motion compensation video coding techniques is achieved, and 2) preliminary experiments indicate that the decoded frame rate can be increased without a corresponding increase in the transmitted bit rate (decoder complexity is increased). Ongoing work in developing an adaptive RVQ video coding system for joint source/channel coding was also discussed.

Future work in relating the prioritized successive approximation bit stream of RVQ codevector indices to multiresolution transmission and modulation schemes may lead to image compression and transmission systems that can tolerate severe fading on very noisy nonstationary channels.

### Acknowledgement

The authors would like to thank Arnold M. Moore of Clark Atlanta University for technical support and computer simulations.

## References

[1] A. Gersho and R. M. Gray. *Vector Quantization and Signal Compression.* Kluwer, Boston, 1992.

[2] C. F. Barnes and R. L. Frost. Vector quantizers with direct sum codebooks. *IEEE Transactions on Information Theory*, 39(2):565–580, Mar. 1993.

[3] B. H. Juang and A. H. Gray. Multiple stage vector quantization for speech coding. In *ICASSP,* 1, 596–600, 1992.

[4] W.-Y. Chan, S. Gupta and A. Gersho. Enhanced multistage vector quantization by joint codebook design. In *IEEE Transactions on Communications,* 40(11):1693–1697, Nov. 1992.

[5] C. F. Barnes and E. J. Holder. Successive approximation quantization with generalized decoding for wavelet transform image coding. In *Proc. Asilomar on Signals, Systems and Computers,* 1, 533–537, 1993.

[6] C. F. Barnes and E. J. Holder. Classified variable rate residual vector quantization applied to image subband coding. In *Proc. DCC,* 272–281, 1993.

[7] Y. Linde, A. Buzo, and R. M. Gray. An algorithm for vector quantizer design. *IEEE Transactions on Communications*, 28:84–95, Jan. 1980.

[8] F. Kossentini, M. J. T. Smith and C. F. Barnes. Image coding with variable rate RVQ. In *ICASSP,* 3, 369–372, 1992.

[9] W. H. R. Equitz and T. M. Cover. Successive refinement of information. *IEEE Transactions on Information Theory*, 37(2):269–275, Mar. 1991.

[10] N. Phamdo, N. Farvardin and T. Moriya. A unified approach to tree-structured and multistage vector quantization for noisy channels. *IEEE Transactions on Information Theory*, 39(3):835–850, May 1993.

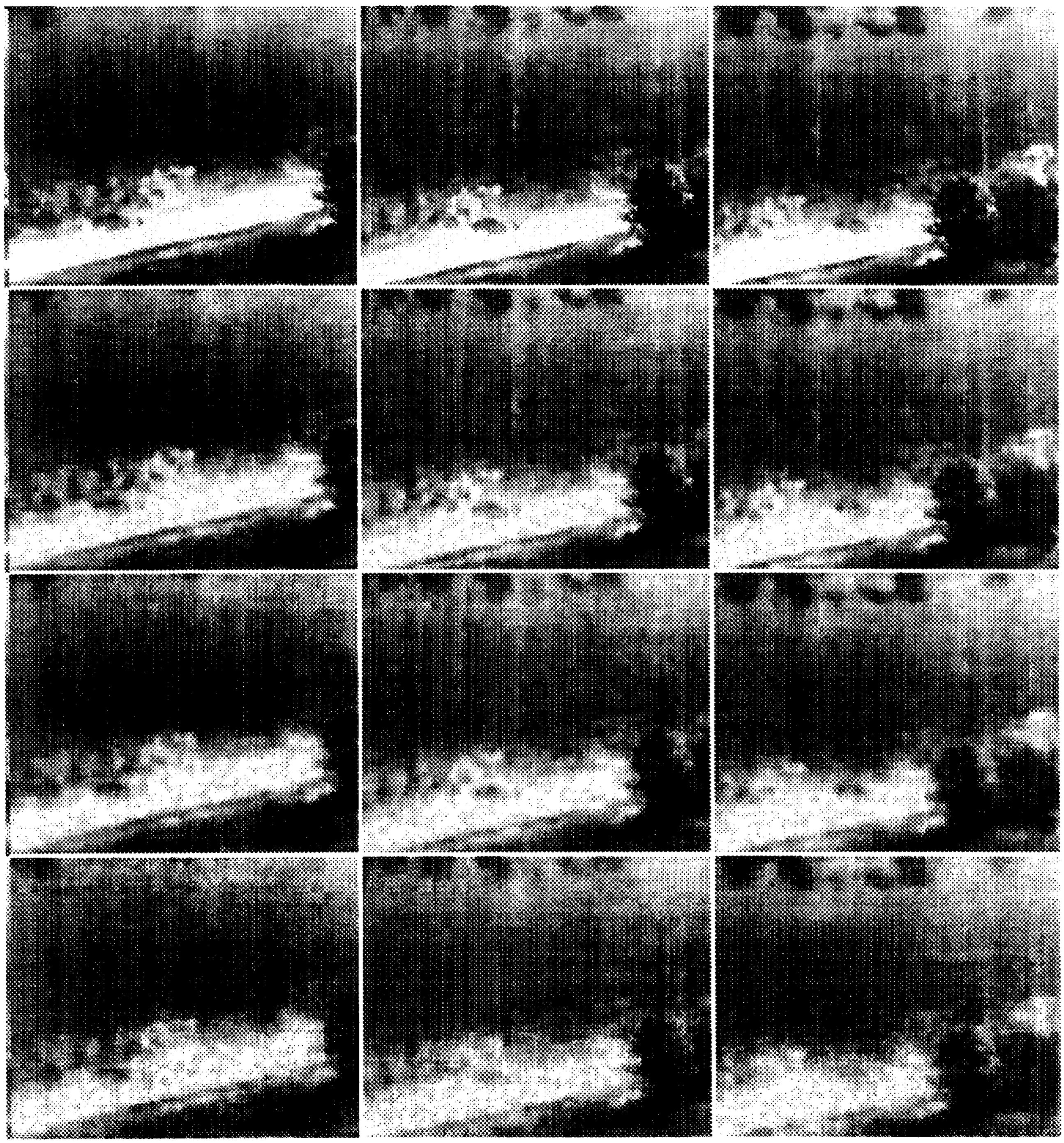

Figure 1: Three example infrared frames: first row original, second row coded at 0.20 bpp (PSNR= 32.54), third row coded at 0.10 bpp (PSNR= 29.33), fourth row coded at 0.05 bpp (PSNR= 26.68.