Visual Pattern Image Sequence Coding

Peter L. Silsbee, *Member, IEEE,* Alan C. Bovik, *Senior Member, IEEE,* and Dapang Chen, *Member, IEEE*

Abstract—**Visual pattern image sequence coding (VPISC) is a new digital image sequence (video) coding process that possesses significant advantages relative to other technologies; in particular, it is extremely efficient in terms of the computational effort required. It is designed to exploit properties of the human visual systems (HVS), and thus yields high visual fidelity. Visual quality criteria are deliberately chosen over information-theoretic ones on the grounds that, in images intended for human viewing, visual criteria are the most meaningful ones. VPISC yields impressive compression comparable to other recent methods, such as motion-compensated vector quantization.**

VPISC divides the images into spatiotemporal cubes, which are then independently matched with one of a small, predetermined set of visually meaningful three-dimensional space-time patterns. The pattern set is chosen to conform to specific characteristics of the HVS.

Also introduced are two modifications of VPISC: adaptive and foveal VPISC. These are spatiotemporally nonuniform implementations that code different portions of the image sequence at different resolutions, according to either a fidelity criterion (for AVPISC) or a foveation criterion (for FVPISC).

I. Introduction

DIGITAL image and image sequence coding algorithms attempt to represent images and image sequences as compactly as possible. Typically this occurs prior to storage or transmission of the images, and a reverse process must restore them with minimal distortion, according to some criteria, following retrieval or reception. In the case of image sequences, the goal of compact representation (compression) is particularly critical due to the large size of even moderate-length sequences, and the corresponding large time–bandwidth product required for transmission. This goal is easily quantified, and thus the success of an algorithm in terms of compression can be easily and objectively judged.

Manuscript received December 5, 1991; revised September 1, 1992 and December 7, 1992. This work was supported by the National Aeronautics and Space Administration University Grant Program under Grant NAG 9-429. This paper was recommended by Associate Editor John W. Woods.

P. L. Silsbee was with the Department of Electrical and Computer Engineering, University of Texas, Austin, TX. He is now with the Department of Electric and Computer Engineering, Old Dominion University, Norfolk, VA 23529.

A. C. Bovik is with the Department of Electrical and Computer Engineering, University of Texas, Austin, TX 78712-1084.

D. Chen was with the Department of Electrical and Computer Engineering, University of Texas, Austin, TX. He is now with National Instruments, Austin TX.

IEEE Log Number 9209402.

However, the ideal of minimal distortion is considerably more subjective, and in fact is a subject of current debate. Although objective measures such as the mean squared error (MSE) exist, they do not generally correlate well with image degradation as measured by human perception [1]. In this paper, we take the view that it is most appropriate that an error measure reflect the degradation of the information *obtained by the receiver*—in this case, the human visual system (HVS). If an adequate model of the intended receiver is available, an objective, meaningful error measure can be designed. However, for optical images intended for human viewing, no adequate model of the HVS exists for designing such a measure. On the other hand, there is sufficient information to partially characterize the system, and this information can guide the design of image and image sequence coding algorithms. A coding algorithm based on HVS characteristics should preserve information that is most important to the perceptual content of the image (or sequence). In contrast, coding algorithms based on abstract error measures are likely to attempt to preserve information that is relatively unimportant to the receiver.

The family of visual pattern image coding (VPIC) algorithms were designed to take advantage of HVS characteristics for single-image coding [2]–[5]. Specifically, image edges that occur within certain frequency limits are preserved, along with the mean intensity value of more uniform areas. Where edges are present, the mean values may be preserved with less precision, since the edge itself is the most important feature of its immediate neighborhood. These algorithms increase compression by selectively preserving only that information to which the human observer's perceptive system is most sensitive, and throwing away information that, if preserved in the coding and decoding, would be lost or masked in the receiver.

The VPIC algorithms are also very efficient. The simplest form of VPIC uses just a few arithmetic operations per pixel, making it almost as fast as simple DPCM coding. The most complex forms are less than an order of magnitude slower, which is still much faster (when implemented on general-purpose hardware) than other popular techniques, such as vector quantization [6], [7], and comparable to discrete cosine transform coding [8].

In this paper we extend the VPIC paradigm to image sequences by incorporating specific spatiotemporal properties of the HVS [9], [10]. This paper describes the visual pattern image sequence coding (VPISC) algorithms. These

include simple VPISC, hierarchical (multiresolution) VPISC, and adaptive and foveal VPISC. These algorithms use a small set of predetermined 3-D (spatiotemporal) patterns to represent the image sequence. The set of patterns corresponds to visually important features, which may or may not be in motion. Since motion is inherent in the pattern set, *the need for motion compensation is reduced*; it is coded as an integral part of the sequence.

Hierarchical VPISC (HVPISC) is a pyramidal, multiresolution implementation of VPISC. As with many other coding schemes, both quality and compression can be improved by generating a pyramidal representation of the sequence to take advantage of redundancies across scales. 3-D HVPISC differs from other pyramidal strategies, however, in that a spatiotemporal pyramid is created, deriving additional advantage from redundancies over several time scales.

In general, image statistics are not known *a priori*. For this reason, much work has focused on developing *scene-adaptive* algorithms that, in various ways, are capable of adjusting their own parameters to adapt to the statistics of a given image [11]–[13]. AVPISC is a simple extension of VPISC that uses an octree to specify the level of detail at which each region is encoded. It can result in significant further decreases in bit rate without noticeably affecting the image quality. This scheme is scene adaptive in that the level of detail encoded in a given sequence region corresponds to the level of detail actually present in that region. The only parameter adjustment made is a yes/no decision that determines whether or not to encode a given block at all.

Foveal VPISC (FVPISC) also uses an octree to specify how much detail is encoded in each region. Unlike AVPISC, however, the octree is not transmitted but instead is built independently by the receiver and transmitter. Only a few parameters must be communicated from the receiver to the transmitter, to define the shape of the octree. These parameters might be interactively chosen by a person with a mouse, or they might be automatically generated by a computer in an active vision system.

The next section of this paper gives a review of VPIC. Simple VPISC is presented in Section III. The extensions which define AVPISC and FVPISC are discussed in Sections IV and V. Section VI describes the test simulations that were run and the results of those simulations, along with some discussion.

II. Review of VPIC

In this section we review static VPIC. The concepts behind VPIC are fundamentally similar to those behind VPISC, and the smaller dimensionality of the static problem provides a good framework with which to introduce these concepts.

Simple, or static, single-resolution VPIC codes images using a small set of predefined visual patterns or local subimages containing visually important information. Current implementations of VPIC use only uniform areas and single edges of various orientations to code images. Since the pattern set is image independent, unlike those used in well-known strategies such as vector quantization (VQ), there is no codebook construction in VPIC.

Denote the image to be coded as the array $\boldsymbol{I} = \{I_{i,j}\}$ which can be written as the union of disjoint 4×4 blocks of the form $\boldsymbol{b}_{i,j} = [\boldsymbol{I}_{n,m}: 4i \le n \le 4i + 3, 4j \le n \le 4j + 3]$. Prior to assigning patterns to image blocks, the mean block intensities $\{\mu_{i,j}\}$ are coded separately: Visual patterns have zero mean intensities. The local variation in the image $\boldsymbol{I}$ is measured using a discrete approximation to the intensity gradient. The gradient magnitude $|\nabla \boldsymbol{b}|_{i,j}$ and gradient orientation $\angle \nabla \boldsymbol{b}_{i,j}$ within each image block correspond to the contrast and orientation of an intensity change within the image block $\boldsymbol{b}_{i,j}$, and have continuous ranges that must be quantized. Currently, $\angle \nabla \boldsymbol{b}_{i,j}$ is quantized in 45° increments. Patterns are assigned to blocks according to the distribution of positive and negative pixels. Fig. 1 shows one set of eight basic edge patterns $\boldsymbol{p}_j$; this set is the one that is used in current VPIC and VPISC implementations as demonstrated here. For other possible patterns, see [2] and [3]. The positive (light) elements take identical values as do the negative (dark) elements; these are chosen such that each pattern has zero mean and unit gradient magnitude $|\nabla \boldsymbol{p}_j| = 1$ [2].

A. Image Coding and Decoding in VPIC

Image coding in VPIC is a very simple process. The first step consists of determining into which subspace an image block $\boldsymbol{b}_{i,j}$ should be mapped. Once the mean block intensity $\mu_{i,j}$ is computed, the gradient values $(|\nabla \boldsymbol{b}|_{i,j})^2$ and $\tan(\angle \nabla \boldsymbol{b}_{i,j})$ are calculated. If $(|\nabla \boldsymbol{b}|_{i,j})^2 \le (|\nabla \boldsymbol{I}|_{\min})^2$, the block is a uniform block; otherwise it is a pattern block. In either case, the mean intensity $\mu_{i,j}$ is quantized and transmitted using $n_\mu + 1$ bits, where the additional (flag) bit informs the decoder whether the pattern is uniform or a pattern block. The value of n_μ may depend on whether the block is uniform ($n_\mu = n_{\mu u}$) or a pattern block ($n_\mu = n_{\mu e}$). The HVS is quite insensitive to absolute intensity values in the immediate vicinity of large contrasts, as evidenced by, for example, the Mach-band effect [14]; however, insufficient precision in uniform areas can give rise to false contours. Therefore uniform blocks generally require a greater precision ($n_{\mu u} > n_{\mu e}$).

If $\boldsymbol{b}_{i,j}$ is an edge block $(|\nabla \boldsymbol{b}|_{i,j})^2 > (|\nabla \boldsymbol{I}|_{\min})^2$, then the orientation $\angle \nabla \boldsymbol{b}_{i,j}$ maps the image block into a pattern subspace. A unique mapping is achieved by selecting the pattern whose polarity distribution is closest to that of the image block. This is easily accomplished by simply counting the number of (+) and (−) gradient values contained in the image block, and choosing the pattern having the best match of (+) and (−) elements. The index j of the selected pattern $\boldsymbol{p}_j$ is transmitted as $n_p + 1$ bits, where the additional bit indicates the edge polarity. Finally, the (squared) gradient magnitude $(|\nabla \boldsymbol{b}|_{i,j})^2$ is quantized into one of Q levels with index i and transmitted as n_g bits. Thus, $n_u = n_{\mu u} + 1$ bits are required to code a 4×4 uniform block, while $n_e = n_{\mu e} + n_p + n_g + 2$ bits are required to code a 4×4 edge block, where $n_e \ge n_u$. If

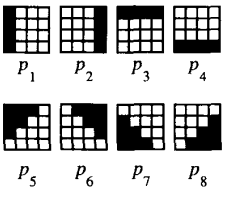


Fig. 1. Set of eight patterns used in VPIC.

uniform and edge blocks are coded with different bit lengths, then the overall range of compression ratios C_{VPIC} obtainable is

$$\frac{128}{n_e}:1 \le C_{\text{VPIC}} \le \frac{128}{n_u}:1.$$

In the worst case, less than three arithmetic operations per pixel are required to achieve this compression; more than two of these are integer additions.

Decoding in VPIC is extremely simple. Each block is decoded independently, and the image is reconstructed by assembling all decoded blocks into a single image. If the block is uniform, then of course the decoder need only place the transmitted block mean value into each pixel of the block. If it is a pattern block, the mean value is placed into each pixel, and then the gradient value, pattern index, and polarity bit are used to access a lookup table containing the pattern, which is added on a pixel-by-pixel basis to the block.

B. Hierarchical VPIC

Despite the efficiency of and coding effectiveness of simple (single-resolution) VPIC, considerable gains can be made in both compression and visual fidelity at marginal additional coding/decoding complexity. These goals are attained via a hierarchical coding framework in which simple VPIC is embedded. Increased accuracy is obtained by first efficiently coding low-resolution versions of the image via VPIC, then using these to augment coding of the high-resolution data. Maximizing the information content of the easily-coded low-resolution subimages yields significant information reduction in the high-resolution coding. HVPIC retains the computational advantages of simple VPIC: Depending on the specific implementation parameters selected, HVPIC requires only 15–25 additions and 1–3 multiplications per pixel.

C. Coding and Decoding in HVPIC

In HVPIC, a dyadic pyramid structure is defined such that the image at any level in the pyramid has twice the resolution of the image at the next lowest resolution level. Assume that the original, full resolution image $\boldsymbol{I} = \{I_{i,j}\}$ to be coded is $2^N \times 2^N$, and that $L \le N$ levels will be constructed in the pyramid representation. The image at the k^{th} resolution level is denoted $\boldsymbol{I}_k = \{I^k_{i,j}\}$ for $k = 1, 2, \cdots, L$, where $\boldsymbol{I}_L = \boldsymbol{I}$; hence the lowest resolution image in the pyramid is the $2^{N-L+1} \times 2^{N-L+1}$ image $\boldsymbol{I}_1$. Lower resolution images are formed from higher resolution images by averaging 2×2 blocks of pixels. This operation is termed Reduce. A second operation, Expand, is defined, which doubles the size of an image by duplicating pixels [3].

In HVPIC the lowest-resolution image $\boldsymbol{I}_1$ is coded using simple VPIC, only with a larger number of bits allocated to both the mean intensity and to the pattern indices than in single-resolution coding. At higher resolutions, it is not the original image which is coded but the residual, or error, image, which represents the difference between the decoded image at the next lower level and the original image at the current level. This process is repeated until the bottom of the pyramid is reached. Thus, the coder transmits a series of coded residual image blocks [3].

The mean intensity of each residual image block is coded as in the simple VPIC algorithm described in the previous subsection, except that the degree of quantization is made to vary. If the coding at the previous level has been effective, the mean intensity of the residual image block will fall near zero, suggesting a considerable reduction in entropy. This allows significant gains to be made by the use of variable-length codes or quantization levels when coding mean block intensities.

As in simple VPIC, decoding is simpler even than coding. The decoder receives a series of coded image blocks (indices and mean intensities). At the first level, the image is decoded as in simple VPIC. At each successive level, the decoded residual block is simply added to the Expanded image from the previous level. The total coding error is largely a function of the last approximation, which is a significant feature of HVPIC. There is a considerable error compensation capability embedded in the HVPIC structure, since at each level only the error image is encoded.

III. Visual Pattern Image Sequence Coding (VPISC)

While the coding of static images is important for, e.g., image archiving, image sequence coding is a much more desirable and challenging application with enormous application potential if certain technological barriers are surmounted. Static VPIC and HVPIC are very fast static image coding techniques—so fast, in fact, that they are feasible for real-time implementation at little cost. However, it is of interest to consider extending the VPIC concept to image sequence coding for two reasons: 1) by exploiting temporal redundancies, very significant gains in compression may be obtained, leading to significant decreases in the required transmission bandwidths, and 2) characteristics of human motion perception allow a HVS-based image sequence coder to select temporal information that would not be evident to a static algorithm.

The extension of VPIC to image sequence coding is accomplished through the definition of a suitable set of fundamental spatiotemporal visual patterns. These visual patterns reflect the basic philosophy of the VPIC strategy. In this case, localized patterns that evolve over time in an image sequence containing motion or other intensity

changes will be represented by localized spatiotemporal edge patterns.

A. Motivating VPISC

Here we motivate the use of localized spatiotemporal patterns to code image sequences. As in the case of VPIC, it should be remembered that the goal is to design a coding strategy that is consistent with the properties of the receiver—in this case the human eye operating over time or, perhaps, a properly designed robotic eye that makes use of information in the form of image edges and smooth regions.

Widely accepted receptive field models of biological vision such as the highly successful Gabor receptive field model, which have been applied to understanding visual pattern analysis, have recently been extended to visual motion sensing. In particular, Adelson and Bergen [15] and Watson and Ahumada [16] have independently uncovered evidence for the existence of 3-D Gabor-shaped neuronal receptive fields that are directionally sensitive to local image velocities; many of the "optimal" properties associated with the simultaneous localization of spatial patterns in space and frequency can be extended to include temporal localization.

Kelly [17] has published an experimentally obtained approximation to the spatiotemporal threshold surface of the HVS. He demonstrates that the HVS has a maximum sensitivity to features with a spatial frequency of about 2 cycles/degree and a temporal frequency of about 2–3 Hz. His data also show a -3-dB response curve with endpoints at approximately (1.5 cycles/degree, 5 Hz) and (3 cycles/degree, 0.5 Hz) (see Fig. 2). At higher velocities or higher spatial frequencies, features begin to fuse. This characteristic is related to the phenomenon of visual latency: The faster an object is moving, the less detail is perceptible. At very low velocities, features tend to fade. However, in the low-velocity case, the eye's own motion maintains the perceived contrast.

With this in mind, we examine the desired characteristics of an image sequence coder. It should be able to reproduce spatial frequencies of 3 cycles/degree or less, with temporal frequencies of 5 Hz or less. As the spatial frequency, temporal frequency, or velocity of a feature increases, its preservation becomes less important. It must be noted that these quantities are only well defined at the eye. It is therefore, necessary to make some assumptions about the size of the image sequence, and the distance from which it will be viewed. Following Chen and Bovik [2], we assume that a 12-in monitor is being viewed from a distance of about 2 m. This assumption yields a particular set of spatial and temporal frequency constraints; different assumptions would yield different constraints but would not negate the validity of the approach. This viewing geometry gives an approximate viewing angle of 0.05° per $4 \times 4 \times 4$-pixel image block [2]. In the time dimension, at the standard video rate of 30 frames/s, there are 0.13 seconds per image block (four frames), independent of the viewing geometry [see Fig 3(a)].

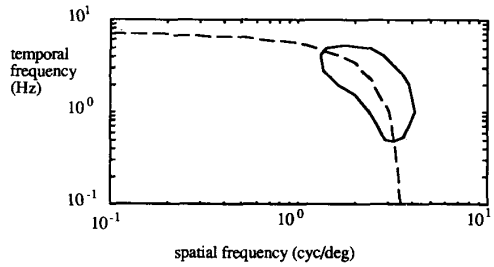


Fig. 2. The solid curve indicates the approximate location in spatial frequency–temporal frequency space of the -3 dB response point of the HVS. The dashed curve indicates peak response at a given velocity. Lines of constant velocity would have a slope of $+1$ on this graph. Adapted from [16].

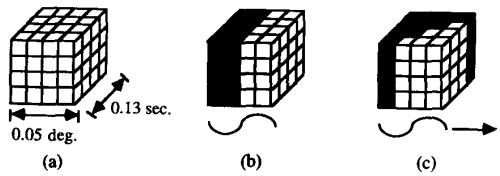


Fig. 3. (a) One 4×4 image block showing its dimensions in terms of visual angle and time. (b) Example of a pattern block used to approximate a 20 cycle/degree intensity modulation. (c) Example of a pattern block used to approximate the same intensity modulation in motion.

Visual patterns including at most a *single* edge per block are sufficient to represent the most important frequencies. 1 edge/block corresponds to 1 cycle/block, if we allow for slight phase errors (i.e., single-pixel errors in edge location). At 0.05 degrees/block this gives 20 cycles/degree, and at 0.13 s/block it gives 7.5 Hz. Note that these values extend well beyond the -3-dB points given above. Thus, patterns with up to a single edge per block are sufficient to encode those features to which the HVS is maximally sensitive. Fig. 3 gives example of pattern blocks that could be used to encode a 20-cycle/degree variation. Lower spatial or temporal frequencies are adequately covered by considering more than one block at a time, while higher frequencies tend to blur and become perceptually unimportant.

Finally, a note about motion compensation is in order. Algorithms have been developed that use complex predictors [18]–[20] along the time axis. If an object is moved relative to a previous frame, a motion compensation algorithm searches in a small neighborhood of the current frame to identify the object or pixels that have changed. If such a correspondence is established, then the frame difference is expected to be very small. The coder then transmits both the frame difference and the displacement vector.

Such techniques have been successful in achieving greatly increased compression, and improved image quality at the new compression levels. However, the motion detection process can be complex or may operate under unrealistic assumptions, such as object regularity. It is highly desirable, then to reduce the need for this computationally intensive process. In VPISC, motion is an inher-

ent part of the coding structure, and thus moving structures that satisfy the assumptions outlined above can be encoded in a very simple and robust way. It is possible that this will enable simplified (and comparatively inexpensive) motion compensation algorithms to be used in conjunction with VPISC.

B. Defining VPISC

Image sequences may be sensed and transmitted on an ongoing (real-time) basis, in which case a causal coding strategy is required, or the images to be coded may first be recorded for later coding, transmission, and archiving. The VPISC algorithms presented here are inherently noncausal, requiring delays (for causal implementation) ranging from four frames (0.13 s) to a few seconds. In the 3-D hierarchical version of VPISC, very low-resolution image cubes may be computed over many frames. The coding/transmission occurs in real time, but with an inserted delay. Hybrid schemes could be developed that would allow shorter delays using predictors, at the cost of some of the extreme simplicity that makes VPISC so attractive.

Assume that the image sequence to be coded is represented as an indexed 3-D array $\boldsymbol{I} = \{I_{i,j,k}\}$, where (i, j) represents samples in image space and k indexes samples of the image sequence taken at equally spaced time instants $t = t_k$. For simplicity, assume further that the images are $2^N \times 2^N$ and that the image sequence is of length 2^M; this is without loss of any generality since sequences of other lengths can be decomposed into subsequences of length $2^{M'}$, $M' < M$ and coded separately by VPISC; odd-length subsequences can be treated as a special case that we will not explore here but that should present little difficulty.

Similarly to simple VPIC, we represent I as the union of disjoint $4 \times 4 \times 4$ space-time image cubes of the form $\boldsymbol{b}_{i,j,k} = [I_{n,m,t}: 4i \le n \le 4i + 3, 4j \le m \le 4j + 3, 4k \le t \le 4k + 3]$, and the mean cube intensities $\{\mu_{i,j,k}\}$ are extracted and coded separately prior to assigning spatiotemporal visual patterns to the image cubes. The calculations performed are similar in nature to the 2-D case; however, in addition to the spatial gradient (averaged over four frames), it is necessary to calculate a temporal gradient. For a $4 \times 4 \times 4$ block,

$$\begin{aligned}
\Delta_x \mathbf{b}_{i,j,k} = {} & \mathrm{AVE}\{I_{n,m,t}: 4i + 2 \le n \le 4i + 3, \\
& 4j \le m \le 4j + 3, 4k \le t \le 4k + 3\} \\
& - \mathrm{AVE}\{I_{n,m,t}: 4i \le n \le 4i + 1, \\
& 4j \le m \le 4j + 3, 4k \le t \le 4k + 3\}, \\
\Delta_y \mathbf{b}_{i,j,k} = {} & \mathrm{AVE}\{I_{n,m,t}: 4i \le n \le 4i + 3, \\
& 4j + 2 \le m \le 4j + 3, 4k \le t \le 4k + 3\} \\
& - \mathrm{AVE}\{I_{n,m,t}: 4i \le n \le 4i + 3, \\
& 4j \le m \le 4j + 1, 4k \le t \le 4k + 3\}, \\
\Delta_t \mathbf{b}_{i,j,k} = {} & \mathrm{AVE}\{I_{n,m,t}: 4i \le n \le 4i + 3, \\
& 4j \le m \le 4j + 3, 4k + 2 \le t \le 4k + 3\} \\
& - \mathrm{AVE}\{I_{n,m,t}: 4i \le n \le 4i + 3, \\
& 4j \le m \le 4j + 3, 4k \le t \le 4k + 1\}.
\end{aligned}$$

Here, $\Delta_x \boldsymbol{b}_{i,j,k}$ is an estimate of the local derivative (with respect to x) of the image intensity at block (i, j, k), and $\Delta_y \boldsymbol{b}_{i,j,k}$ and $\Delta_t \boldsymbol{b}_{i,j,k}$ are analogous quantities for the y and t directions respectively, Furthermore,

$$|\nabla \boldsymbol{b}|_{i,j,k} = \sqrt{(\Delta_x \boldsymbol{b}_{i,j})^2 + (\Delta_y \boldsymbol{b}_{i,j})^2}$$

and

$$\angle \nabla \boldsymbol{b}_{i,j,k} = \tan^{-1}(\Delta_y \mathbf{b}_{i,j} / \Delta_x \mathbf{b}_{i,j})$$

provide estimates of the gradient magnitude and the gradient orientation, respectively.

The type of gradient magnitude information transmitted through the communication channel depends on the type of pattern. If the pattern is non-time-varying, that is, $|\nabla \boldsymbol{I}|_{\min} \le |\nabla \boldsymbol{b}|_{i,j,k}$ and $|\nabla_t \boldsymbol{b}|_{i,j,k} \le |\nabla \boldsymbol{I}|_{\min}$, then the gradient is transmitted as defined above. For a time-varying pattern, with $|\nabla \boldsymbol{I}|_{\min} \le |\nabla_t \boldsymbol{b}|_{i,j,k}$, the gradient is recalculated to include the time variation:

$$|\nabla \boldsymbol{b}|_{i,j,k} = \sqrt{(\Delta_x \mathbf{b}_{i,j,k})^2 + (\Delta_y \mathbf{b}_{i,j,k})^2 + (\Delta_t \mathbf{b}_{i,j,k})^2}.$$

We now define spatiotemporal image blocks for use in VPISC coding. Figs. 4–6 depict a set of patterns that can be used to code image sequences in a VPISC framework. We briefly consider the types of spatiotemporal structures that this set of patterns is intended to code, and the specific advantages derived. The patterns are subdivided into three groups, which are labeled static patterns, moving patterns, and changing patterns. Each group is intended to encode a particular subset of the temporal frequency–spatial frequency continuum. For a given temporal frequency range, the HVS response peaks at a certain spatial frequency range, as indicated in Fig. 2; this observation guides the design of the pattern blocks.

The set of eight static patterns (Fig. 4) used in the current implementation of VPISC have x- and y-dependency only; they do not vary with time. These patterns are derived from the VPIC pattern set, which has already been shown to work well for static image coding in [2]–[4]. These patterns, along with uniform blocks, cover the low end of the temporal frequency scale (less than 1 Hz), at which the HVS response as a function of spatial frequency peaks around 3–4 cycles/degree [17].

The eight moving patterns (Fig. 5) are used to encode intermediate temporal frequencies, including most of the peak response area mentioned in Section III-A (about 1–2 Hz). They all consist of a single intensity edge, which moves across the block over the course of the four frames. These patterns have a velocity of just over 1 degree/s.

Features with much higher velocities than this must also be represented, but as velocity increases, HVS response declines, and peak response takes place at lower spatial frequencies. The three changing patterns (Fig. 6) are used in this case. Fine detail is no longer preserved, and if it were it would not be perceived by the visual system.

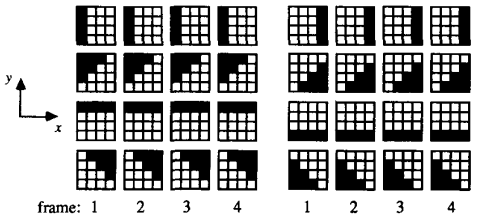


Fig. 4. The set of eight static patterns used in our VPISC simulations. Note that these are the same patterns from Fig. 1, repeated over four frames.

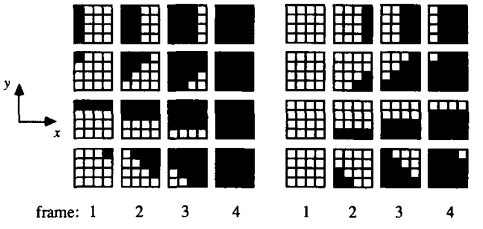


Fig. 5. The eight moving blocks used in VPISC.

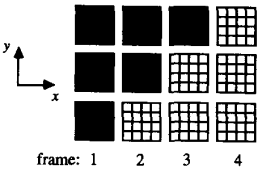


Fig. 6. The three changing blocks used in VPISC.

This gives a base set of 19 patterns which, based on our results, are adequate to code a wide variety of image sequences with low perceptual distortion. Other pattern sets could, of course, be used. For example, patterns could be optimized for a particular application, using, e.g., *a priori* upper limits on edge velocity (which might allow elimination of the changing blocks), or different viewing angle assumptions, resulting in a different set of frequency constraints at the image (and perhaps yielding smaller block sizes and less compression, or larger block sizes and more compression).

C. Coding Efficiency

The attainable coding efficiency using a simple VPISC depends upon the resolution taken along the time axis. All the implementations described here use $4 \times 4 \times 4$ edge and uniform image sequence cubes. We have (using the same notation developed earlier) $n_u = n_{\mu u} + 1$ bits used to code uniform cubes (unchanged from the 2-D case) and $n_e = n_{\mu e} + n_p + n_g + 2$ bits used to code an edge cube, where only n_p changes with the increased dimensionality. We have used $n_p = 6$ in our simulations. Since each of the 19 patterns developed above has a reversed-polarity version as well (obtained by substituting black pixels for white, and vice versa), there are 38 unique patterns that must be encoded. Since the number of patterns (38) is not a power of two, n_p (which should be approximately equal to $\log_2(38)$ but must be an integer) varies from 5 to 6; that is, some patterns are encoded with 5 b, and others with 6 b. In the following development, the pessimistic value of 6 is used for simplicity. Uniform and edge cubes are coded with different bit lengths, so the overall range of compression C_{VPISC} obtainable is

$$\frac{514}{n_e}:1 \le C_{\text{VPISC}} \le \frac{512}{n_u}:1.$$

For the values $n_u = 7$, $n_e = 13$ (used in the simulations described in later sections) a compression range $39{:}1 \le C_{\text{VPISC}} \le 73{:}1$ is obtained. While this range of compression is not unimpressive, far better gains can be obtained by casting VPISC within a hierarchical framework, as described next.

D. Hierarchical Coding in VPISC

By increasing the overhead over a set of low-resolution (spatio or spatiotemporal) subimages of the 3-D sequence, highly increased compression can be obtained. Higher spatial frequencies require longer response times to be well perceived; thus, at high frame rates, more information should be allocated to lower (time) resolution images, coded as uniform cubes and pattern cubes, since they will contribute the most to the visual fidelity of the coded image sequence.

The structure of the resolution hierarchy is similar to that of static VPIC; however, the decomposition may take place in two or three dimensions. A 3-D decomposition yields better compression and better quality, and is even slightly cheaper computationally. However, it entails more of a processing delay because many frames need to be considered simultaneously. This is a handicap only for interactive, real-time situations; used for recorded video, for example, this delay could be made completely transparent.

For a 2-D resolution hierarchy, we operate almost exactly as in simple HVPIC (Section II-C), the only difference being that the coding and decoding steps operate on four-frame groups. The Expand and Reduce operations, and all image subtractions, take place as before.

For the 3-D case, we must redefine Expand and Reduce as follows. $\text{Expand}_{3\text{-D}}$ duplicates each pixel in both spatial dimensions and in time; that is, given an $n \times n$ frame, $\text{Expand}_{3\text{-D}}$ yields two identical $2n \times 2n$ frames. $\text{Reduce}_{3\text{-D}}$ averages 2×2 blocks of pixels from two successive frames to yield a single pixel (see Fig. 7). Thus, given two successive $2n \times 2n$ frames, $\text{Reduce}_{3\text{-D}}$ yields a single $n \times n$ frame. As before, note that $\text{Expand}_{3\text{-D}}$, following $\text{Reduce}_{3\text{-D}}$, does not in general restore the original image. Thus, if we are using, say, a four-level resolution pyramid, we begin with 32 frames and apply $\text{Reduce}_{3\text{-D}}$ three times to obtain the lowest-resolution four-frame group for initial coding.

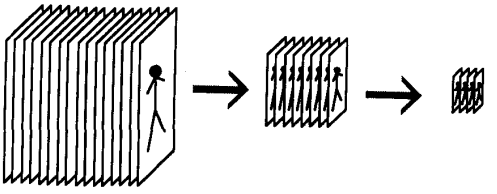


Fig. 7. The $\text{Reduce}_{3\text{-D}}$ operation applied twice to 16 successive frames.

The mean value of each block, at all resolutions except the lowest, can be expected to fall near zero. Although this will vary from image to image (i.e., we leave "near" undefined), it is a trivial consequence of the redundancy in images that all coding algorithms hope to exploit. The compression can be significantly increased by using a variable-length code for the block mean values at all but the top pyramid level. For values near zero, a two-bit code is used. For larger magnitude values a longer code is used, which again varies with the type of block.

Depending on the image characteristics, a wide range of compression ratios will occur. For the 19-pattern set that we have used, the worst case block is a pattern block with a large mean offset. This type of block may contribute up to 13 b to the total. The best case, on the other hand, is a uniform block with a small mean error. This will contribute only 3 b to the total. Now, each full-resolution block requires bits for itself, plus a fraction of those allocated to each of the lower-resolution blocks that contain it. For a 2-D, L-level hierarchy, this amounts to $1 + 4^{-1} + 4^{-2} + \cdots + 4^{-L} \approx 1.33$ blocks. Thus $512/(13 \times 1.33) \leq$ compression $\leq 512/(3 \times 1.33)$, or $29.5 \leq$ compression ≤ 128. For a 3-D hierarchy, we have $1 + 8^{-1} + 8^{-2} + \cdots + 8^{-L} \approx 1.14$ blocks. This gives $512/(13 \times 1.14) \leq$ compression $\leq 512/(3 \times 1.14)$, or $34.5 \leq$ compression ≤ 149.

E. Additional Processing

The VPISC technique, as described so far, can produce a few objectionable artifacts. At low resolutions, patterns are added to the decoded image; these patterns are step functions (restricted to the domain on which the pattern is defined), and thus contain higher frequencies than those they are meant to represent. When these patterns are expanded, the step characteristic remains intact, even at the finest resolution; the effect of this can be visually objectionable. Fortunately, these can be easily dealt with. Filtering each frame with a simple 3×3 smoothing filter prior to adding the patterns at each resolution effectively removes the problem. The nature of the filter kernel is not critical; applying an FIR filter with the impulse response $h_{i,j} = \{h_{-1,0}, h_{0,0}, h_{1,0}\} = \{1/4, 1/2, 1/4\}$ followed by the 90°-rotated filter with impulse response $h_{i,j}$ works quite well and is easily implemented with nothing but integer additions and shifts. With this method, low-resolution patterns are eventually filtered by a kernel with quite a large extent, while the highest resolution patterns are not filtered at all (recall that their higher frequency components are visually negligible).

F. Coding Complexity

The computation involved in VPIC can be shown to be linear with the number of image pixels considered. Thus, an increase in the dimensionality entails no increase in computation per pixel. In the simulations discussed here, for example, simple VPISC requires 2.3 integer additions, 1 integer comparison, 0.06 integer multiplications, and 0.09 floating point operations per pixel. The most complex algorithm, which uses a 3-D resolution hierarchy, requires 15 integer additions, 1.1 integer multiplications, 1.3 integer comparisons, and 0.1 floating point operations per pixel. These figures compare very favorably to VQ-based coders and, for the worst case, are comparable to DCT-based coders with no motion compensation [1]. Decoding in all cases is of similar or less complexity.

IV. Adaptive VPISC

It can be observed that many, probably most, image sequences contain some regions with fine detail and some relatively uniform regions. Here we take regions in the 3-D sense, observing that some regions contain motion and others do not. These uniform regions are well represented by a single intensity value. Thus, very few bits should be required to encode them. Furthermore, the hierarchical structure of VPISC (and other pyramidal coding schemes) is well suited to the task of finding these areas. Conceptually, all we need to do is apply a test after the encoding of each pyramid level to determine if each block is adequately represented. If it is, we need not consider that block in any finer detail.

In practice, a useful and easy-to-implement test is to simply encode the entire image using a nonadaptive algorithm and examine the output. Areas that required little adjustment at high resolutions must have been adequately coded at some lower resolution. Since VPISC is a very fast algorithm, there is little computational sacrifice in doing this.

Our implementation is very simple. First, the image is encoded, using a standard VPISC algorithm, and a record is kept of the coding at each level. Then, an octree is built from the bottom up by examining the coder's output at each block. The octree tells the coder which blocks to transmit, and the decoder which to expect.

The octree is built in the following manner. Each bit in the octree represents one block at some pyramid level. Let O_n denote the octree level corresponding to the image I_n. Then, O_n has $2^N/2^{(L-n+2)} \times 2^N/2^{(L-n+2)} \times 2^M/2^{(L-n+2)}$ elements, where the subsequence under consideration consists of 2^M frames of $2^N \times 2^N$ pixels each. Let $O^{(n)}_{i,j,k}$ denote the i^{th} bit in the j^{th} column of the k^{th} frame in the n^{th} level of the tree. We first mark each bit at the lowest level as terminal if no edge was detected and no large correction of the mean occured at the corresponding block. Here, a large correction is a correction with magnitude greater than 4. This value was chosen

as a compromise, where a lower threshold gives less image degradation but less compression, and a higher value gives better compression but false contours start to appear. Thus, set $O^{(0)}_{i,j,k} = 0$ if there is no significant detail at that block and $O^{(0)}_{i,j,k} = 1$ otherwise.

Note that a large correction of the mean indicates that a block differs from its immediate spatial or temporal neighbors. That is, the mean value of the current block is substantially different than the mean value of the larger block of which it is a part. This can be true only if at least one of its neighbors also has a significantly different mean value. Thus, one can say that there is local detail or motion at the scale presently under consideration. At lower resolution levels, each bit is then processed as follows:

$O^{(n)}_{i,j,k} = 0$ if $O^{n+1}_{n,m,t} = 0$
for all $(n,m,t) \in \{(2i,2j,2k), (2i,2j,2k+1), (2i,2j+1,2k), (2i,2j+1,2k+1), (2i+1,2j,2k), (2i+1,2j,2k+1), (2i+1,2j+1,2k), (2i+1,2j+1,2k+1)\}$
and if no edge or large mean value was found at this block,

$O^{(n)}_{i,j,k} = 1$ otherwise.

If $O^{(n)}_{i,j,k}$ is set to 0, it is considered terminal and its descendants are discarded. A terminal block is considered to be adequately encoded.

The coder sends, and the decoder processes, only information pertaining to those blocks whose corresponding octree nodes have no terminal ancestor. Sub-blocks of a terminal block are simply ignored; they are guaranteed to contain, at most, minor corrections.

The octree is a very inexpensive structure to calculate and transmit. The calculation requires only one to ten bit comparisons per node, which may be combined into a single integer comparison. The transmission of the octree requires just 1 b per node, with the exception of the lowest, most expensive level, which is calculated but not transmitted. The lowest level of the octree has 1 b for every 64 pixels; the size of each successive level is reduced by a factor of eight. Since there are 2^{2N+M} pixels in the image sequence (or subsequence), it is necessary to calculate

$$2^{2N+M-6} \cdot (2^0 + 2^{-3} + \cdots + 2^{-3L}) \approx 2^{2N+M-6} \cdot (\tfrac{8}{7}) = 2^{2N+M}/56 \text{ nodes}$$

and to transmit

$$2^{2N+M-6} \cdot (2^{-3} + 2^{-6} + \cdots + 2^{-3L}) \approx 2^{2N+M-9} \cdot (\tfrac{8}{7}) = 2^{2N+M}/448 \text{ b.}$$

This is a maximum computational cost of less than 0.02 operations per pixel, and a transmission cost of just 0.002 b per pixel.

For some sequences, the gains can be considerable. Obviously, the more uniform a sequence is in space and/or time, the greater the advantage to be derived by this technique.

V. Foveal VPISC

Foveal VPISC (FVPISC) was developed in mimicry of the human eye, which is capable of great resolution in only a small central portion of the viewing field (the fovea), while resolution falls off quickly in peripheral areas [14]. FVPISC is intended for applications involving a fixation point or region of interest over which detail is important that is also surrounded by a contextual background that contains useful information at a lower resolution. The implicit assumption is that it is adequate to present the background at a lower resolution. Such a concept is likely to be useful in active vision systems, where computational speedup can be obtained by performing variable-resolution processing, with high-resolution processing occurring only at specific, limited locations [21], [22]. Another use is in applications involving fixation-point-controlled imaging, where a human user coordinates the location of high-resolution coded/decoded image quality in order to make it coincident with his/her fixation.

FVPISC is identical to AVPISC except in the generation of the octree. In FVPISC, a 2-D region of interest is specified by some means, which is not important to this discussion. For example, a person might specify the region with a mouse, or it might be automatically specified by a computer in an active vision system. Let us represent the region as a circle with center (i_0, j_0) and radius r. Then we generate the octree according to

$$O^{(n)}_{i,j,k} = 0 \quad \text{if } \left[(4 \times 2^{L-n})i - i_0\right]^2 + \left[(4 \times 2^{L-n})j - j_0\right]^2 > [(L-n+1)r]^2$$

$O^{(n)}_{i,j,k} = 1$ otherwise.

Thus, the information within the region of interest is encoded at the highest resolution, but outside that region is a series of annular regions that are coded at successively lower resolutions. This results in image sequences that are coded at significantly lower bit rates, but with preserved high fidelity in the most interesting portions of the scene and lower fidelity in the contextually rich background. Note that in FVPISC, $O^{(n)}_{i,j,k}$ does not depend on k; however, the definition given above allows the use of the same decoder for FVPISC as is used for AVPISC.

VI. Simulations

The following algorithms were applied to the standard test image sequences Miss America and Salesman (a frame from each is shown in Fig. 8): simple VPISC, 2-D and 3-D hierarchical VPISC, and adaptive and foveal VPISC using a 3-D hierarchy. Both image sequences were

TABLE I
CODING PARAMETERS USED IN SIMULATIONS

Algorithm	# of Gradient Levels	# of Patterns	Bits for Mean (uniform)	Bits for Mean (pattern)	Bits per Uniform Block	Bits per Pattern Block
Simple VPISC	8	19	6	4	7	12–13
2-D HVPISC	8/4	19	6/2	4/2	9/3	9–13
3-D HVPISC	8/4	19	6/2	4/2	9/3	9–13
AVPISC	4	19	6/2	5/2	10/4	10–14
FVPISC	4	19	6/2	5/2	9/3	1–14

(a) (b)

Fig. 8. Frames from original sequences. (a) Miss America. (b) Salesman.

(a) (b)

Fig. 9. VPISC coded frames. (a) Miss America. (b) Salesman.

digitized at 30 frames/s. The sequences were 512 pixels × 512 pixels, digitized to 256 (nominal) gray levels. They were created using linear interpolation from 256 × 256 image sequences (while this may seem suspect, it does not alter what, in our view, are the important judgement criteria: transmission bit rate and visual quality, given a viewing geometry). 64 frames of each sequence were used in each simulation.

In all simulations presented here, a set of 19 patterns (and their complements) was used, classified as static edges, moving edges, and changing blocks. Patterns from Figs. 4–6 were used. Uniform blocks were coded with 7 b (6 for the mean block value and one indicator bit), while the coding for pattern blocks varied according to the algorithm and pyramid level (see Table I).

Simulation results are presented in Figs. 9–13. Fig. 9 shows the frames from Fig. 8, after encoding with simple VPISC. Note that some areas appear blocky. This effect is lessened with any of the hierarchical versions; however, it should also be noted that, when the sequence is viewed at full speed, many of the defects are simply not apparent. Figs. 10 and 11 show examples of 2- and 3-D HVPISC, while AVPISC and FVPISC results are shown in Figs. 12 and 13, respectively. The image quality is clearly best when 3-D HVPISC is used. AVPISC results are virtually indistinguishable from 3-D HVPISC results, as expected, since the two algorithms encode almost the same information. FVPISC results are identical to 3-D HVPISC results within the region of interest.

Table II shows a numerical summary of the results. Salesman, which contains considerably more detail than Miss America, does not benefit significantly from AVPISC coding; however, it does not suffer either. Miss America gains considerably in compression.

(a) (b)

Fig. 10. 2-D hierarchical VPISC coded frames. (a) Miss America. (b) Salesman.

(a) (b)

Fig. 11. 3-D hierarchical VPISC coded frames. (a) Miss America. (b) Salesman.

VII. CONCLUSION

VPISC is a highly promising image compression technique yielding compression ratios of well over 100:1 with good image quality. It is extremely efficient, suitable for use in real-time telecommunications applications. Motion

(a) (b)

Fig. 12. AVPISC coded frames. (a) Miss America. (b) Salesman.

Fig. 13. FVPISC coded Salesman.

TABLE II
SUMMARY OF CODING RESULTS

Image Sequence	Algorithm	Bits per Pixel	Compression Ratio	Transmission Rate (kb/s)
Miss America	Simple VPISC	0.124	64	977
	HVPISC	0.080	100	630
	3-D HVPISC	0.068	118	534
	AVPISC	0.048	166	378
Salesman	Simple VPISC	0.163	49	1278
	HVPISC	0.129	62	1011
	3-D HVPISC	0.112	71	884
	AVPISC	0.108	74	849
	FVPISC	0.034	234	268

is incorporated directly into the coding structure; thus, in many cases, motion compensation is not required. The development of simplified motion compensation algorithms that take advantage of this property would be an interesting area for further study.

AVPISC is a useful extension of VPISC that can yield lower bit rates for many image sequences without noticeably affecting the quality of the result. It has a very low computational cost, and, though it is not guaranteed to give lower bit rates, the risk is very slight. As demonstrated by the Salesman sequence, however, it does not help very much for busy sequences. A guaranteed reduction in bit rate can be obtained through the use of FVPISC, provided a region of interest can be defined.

REFERENCES

[1] M. Rabbani and P. W. Jones, *Digital Image Compression Techniques*. Bellingham, WA: SPIE, 1991.

[2] D. Chen and A. C. Bovik, "Visual pattern image coding," *IEEE Trans. Commun.*, vol. 38, pp. 2137–2146, Dec. 1990.

[3] D. Chen and A. C. Bovik, "Hierarchical visual pattern image coding," *IEEE Trans. Commun.*, vol. 4, pp. 671–675, Apr. 1993.

[4] D. Chen and A. C. Bovik, "Hierarchical visual pattern image coding," in *Proc. Int. Conf. Pattern Recognition*, Atlantic City, NJ, June 1990.

[5] A. C. Bovik and D. Chen, "Method and apparatus for processing visual pattern images," U. S. Patent No. 5144688, 1992.

[6] R. M. Gray, "Vector quantization," *IEEE Acoust., Speech, Signal Processing Mag.*, vol. 1, pp. 4–29, Apr. 1984.

[7] M. Goldberg and H. Sun, "Image sequence coding using vector quantization," *IEEE Trans. Commun.*, vol. COM-34, pp. 703–710, July 1986.

[8] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *IEEE Trans. Comput.*, vol. C-23, pp. 90–93, Jan. 1974.

[9] P. L. Silsbee, A. C. Bovik, and D. Chen, "Visual pattern image sequence coding," in *Proc. SPIE Symp. Visual Communication Image Processing*, Laussanne, Switzerland, Oct. 1990.

[10] P. L. Slisbee and A. C. Bovik, "Nonuniform visual pattern image sequence coding," in *Proc. SPIE Symp. Electronic Imaging Science Technology*, San Jose, CA, Feb. 1991.

[11] W. H. Chen and W. K. Pratt, "Scene adaptive coder," *IEEE Trans. Commun.*, vol. COM-32, pp. 225–232, Mar. 1984.

[12] W. A. Pearlman, "Variable block rate and blockwise spectral adaptation in cosine transform image coding," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, pp. 773–776, New York, NY, Apr. 1988.

[13] P. L. Silsbee and A. C. Bovik, "Adaptive visual pattern image coding," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, Toronto, CA, May 1991.

[14] T. N. Cornsweet, *Visual Perception*. New York: Academic, 1970.

[15] E. H. Adelson and J. R. Bergen, "Spatiotemporal energy models for the perception of motion," *J. Opt. Soc. Am. A*, vol. 2, pp. 284–299, 1985.

[16] A. B. Watson and A. J. Ahumada, Jr., "Model of human visual-motion sensing," *J. Opt. Soc. Amer. A.*, vol. 2, pp. 322–341, 1985.

[17] D. H. Kelly, "Motion and vision, II. Stabilized spatio-temporal threshold surface," *J. Opt. Soc. Amer.*, vol. 69, pp. 1340–1349, Oct. 1979.

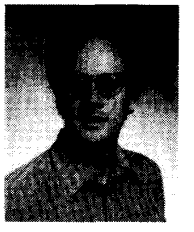
[18] A. N. Netravali and J. A. Stuller, "Motion-compensated transform coding," *Bell Syst. Technol. J.*, vol, 58, pp. 381–391, Sept, 1979.

[19] T. S. Huang, Y. P. Hsu, and R. Y. Tsai, "Interframe coding with general two-dimensional motion compensation," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, 1982.

[20] R. Srinivasan and K. R. Rao, "Motion-compensated coder for videocoferencing," *IEEE Trans. Commun.*, vol. COM-35, pp. 297–303, Mar. 1987.

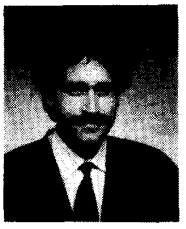
[21] A. Califano, R. Kjeldsen, and R. M. Bolle, "Data and model driven foveation," in *Proc. 10th Int. Conf. Pattern Recognition*, Atlantic City, NJ, June 1990.

[22] P. J. Burt, "Multiresolution techniques for image representation, analysis, and 'smart' transmission," in *Proc. SPIE Conf. Visual Communication Image Processing*, Philadelphia, PA, Nov. 5–10, 1989.



Peter L. Silsbee (S'92-M'93) received the A.B. degree in engineering science from Dartmouth College, Hanover, NH, in 1987, and the M.S.E. and Ph.D. degrees in electrical and computer engineering in 1989 and 1993, respectively, both from the University of Texas, Austin.

Since 1993 he has been Assistant Professor of Electrical and Computer Engineering at Old Dominion University, Norfolk, VA. His current research interests include speech recognition and image processing, with special emphasis on strategies for adapting speech recognition technologies to cope with the variability inherent in real-world applications. He holds one U. S. patent.

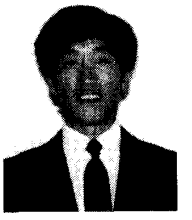


Alan C. Bovik (S'80–M'84–SM'90) was born in Kirkwood, MO, on June 25, 1958. He received the B.S. degree in computer engineering in 1980, and the M.S. and Ph.D. degrees in electrical and computer engineering in 1982 and 1984, respectively, from the University of Illinois, Urbana-Champaign, IL.

He is currently the Hartwig Endowed Fellow and Professor in the Department of Electrical and Computer Engineering, the Department of Computer Sciences, and the Biomedical Engineering Program at the University of Texas, Austin, TX, where he is the Director of the Laboratory for Vision Systems. During the spring of 1992, he held a visiting position in the Division of Applied Sciences, Harvard University, Cambridge, MA. His current research interests include image processing, computer vision, three-dimensional microscopy, and computational aspects of biological visual perception. He has published over 160 technical articles in these areas.

He has been involved in numerous professional society activites, including the following: Associate Editor, *IEEE Transactions on Signal Processing*, 1989–present; Associate Editor, *Pattern Recognition*, 1988–present; Steering Committee, *IEEE Transactions on Image Processing*, 1991–present; General Chairman, *1st IEEE International Conference on Image Processing*, to be held in Austin, Texas, November, 1994; Local Arrangements Chairman, *IEEE Computer Society Workshop on the Interpretation of 3-D Scenes*, October 1989; Program Chairman, *SPIE/SPSE Symposium on Electronic Imaging*, February 1990; and Conference Chairman, *SPIE Conference on Biomedical Image Processing*, 1990 and 1991. He is a frequent consultant to local industrial and academic institutions.

Dr. Bovik is a Registered Professional Engineer in the state of Texas. He is a winner of the University of Texas Engineering Foundation Halliburton Faculty Excellence Award, an Honorable Mention winner of the International Pattern Recognition Society Award for Outstanding Contribution, and was a National Finalist for the 1990 Eta Kappa Nu Outstanding Young Electrical Engineer Award.



Dapang Chen (M'90) was born in Shanghai, China, in June 1958. He received the B.S. degree in computer engineering from the University of Science and Technology of China in 1982, and the M.S. degree in biomedical engineering and the Ph.D. degree in electrical engineering fromthe University of Texas, Austin, TX, in 1985 and 1990, respectively.

Since 1986 he has been with National Instruments Corporation, Austin, Texas, first as a DSP Engineer and later as Analysis Software Manager. He is currently responsible for the research, development, and implementation of signal processing algorithms. His primary research interests include image processing, image coding, computer vision, and signal processing.

He recieved the MCC Outstanding Student Research Award in 1989 for the work in image coding and has also received one U.S. Patent. In 1992, he was a finalist of the EDN's Innovator of the Year Award.