Rapid Estimation of Camera Motion from Compressed Video with Application to Video Annotation

Yap-Peng Tan, Drew D. Saur, Sanjeev R. Kulkarni, *Senior Member, IEEE*, and Peter J. Ramadge, *Fellow, IEEE*

***Abstract*—As digital video becomes more pervasive, efficient ways of searching and annotating video according to content will be increasingly important. Such tasks arise, for example, in the management of digital video libraries for content-based retrieval and browsing. In this paper, we develop tools based on camera motion for analyzing and annotating a class of structured video using the low-level information available directly from MPEG-compressed video. In particular, we show that in certain structured settings, it is possible to obtain reliable estimates of camera motion by directly processing data easily obtained from the MPEG format. Working directly with the compressed video greatly reduces the processing time and enhances storage efficiency. As an illustration of this idea, we have developed a simple basketball annotation system which combines the low-level information extracted from an MPEG stream with the prior knowledge of basketball structure to provide high-level content analysis, annotation, and browsing for events such as wide-angle and close-up views, fast breaks, probable shots at the basket, etc. The methods used in this example should also be useful in the analysis of high-level content of structured video in other domains.**

***Index Terms*— Digital libraries, motion estimation, MPEG, video annotation, video search.**

I. INTRODUCTION

AS DIGITAL video becomes more pervasive, efficient ways of searching and annotating video according to content will be increasingly important [1]. Such tasks arise, for example, in the management of digital video libraries for content-based retrieval and browsing. In this paper, we develop tools based on camera motion for analyzing and annotating a class of structured video using the low-level information available directly from MPEG compressed video. In particular, we show that in certain structured settings it is possible to obtain reliable estimates of camera motion by directly processing data easily obtained from the MPEG format. Working directly with the compressed video greatly reduces the processing time and enhances storage efficiency.

A successful video-annotation system should provide users with a useful abstract of video content in a reasonable processing time. However, given the wide variety of video content, fully automated annotation of generic video is extremely difficult. Current analysis of generic video is usually restricted to examining features such as intensity or color, texture, object shapes, and spatial layout of interframe motion information. For example, many methods have been proposed that use these low-level features to parse video into "shots" (temporal segments of video with smoothly changing content) [2]–[5]. This temporal segmentation is generally considered to be an important initial step in video analysis, since it decomposes the video into fundamental content units. After segmentation, the video shots can be further classified into groups based on their similarities with respect to low-level features. For example, in [6]–[8], low-level features of one or more *key frames* are used to cluster the video into classes of similar shots. Recently, methods for finding interesting temporal patterns within video shots, such as *dialogue-like* or *action-like* sequences, have also been addressed [9]–[11].

Video annotation is often facilitated by prior knowledge of some general structure for the class of video under study. By structure we mean either image structure (what objects are likely to appear and where) or video content (what events are likely to occur and when) or both. Prior knowledge of the video domain is obviously useful in identifying and annotating content. For example, in a study of news programs, Zhang *et al.* [12] employ a syntactic, model-based approach to parse the newscast video based on color histograms and the spatial layouts of key frames into "*anchorperson shot*" and "*news story shot.*" Gong *et al.* [13] consider the problem of parsing soccer video into semantic sections by using the standard layout of a soccer field. They used edge detection and line comparison techniques to classify the soccer video into one of nine categories such as *"in the mid-field," "around the left penalty area,"* and *"near the top right corner."* Recently, Chang *et al.* [14] have developed a prototype system to extract *"touchdown"* sequences from football video by integrating techniques in speech analysis (such as word spotting and detecting cheering) and image analysis (such as edge detection and template matching).

While these examples of structured video analysis have achieved certain goals of interest, they require processing uncompressed video. Analyzing compressed video directly without full-frame decompression is clearly advantageous since it can considerably reduce the processing time and storage requirements. Efficient techniques for extracting useful information directly from compressed video, such as scene-change boundaries and low-resolution images, have recently been considered in [4], [5], and [15].

Manuscript received September 22, 1997; revised August 31, 1998. The research of Y.-P. Tan was supported in part by an IBM Graduate Fellowship. The research of D. D. Saur was supported in part by the New Jersey Center for Multimedia Research. This paper was recommended by Associate Editor R. Stevenson.

The authors are with the Department of Electrical Engineering, Princeton University, Princeton, NJ 08544 USA.

Publisher Item Identifier S 1051-8215(00)01629-3.

We address the analysis and annotation of video by temporal segmentation and camera-motion estimation directly from compressed video. The estimation of camera motion directly from segments of MPEG-compressed video appears to be new. In order for this estimation to provide useful information, it is necessary that the underlying video segment be of a form amenable to analysis using camera motion. In addition, it is necessary that the video be of a form that permits camera-motion estimation to be accomplished if the video was uncompressed. There are numerous examples in the domain of sports video, for example, where important temporal segments of the video satisfy these constraints.

We test our proposed analysis methods using basketball video as a testbed. By examining readily extractable data from MPEG video bit streams, such as macroblock motion vectors and the number of intracoded blocks, we illustrate that it is possible to identify high-level events such as wide-angle and close-up views, fast breaks (FB's), or full court advances (FCA's) of the ball, and shots at the basket within the compressed basketball video. The techniques developed should also be applicable to other domains of structured video. A preliminary version of some of this work appeared in [19].

The principal contributions of the paper are as follows:

1) we show how to obtain useful estimates of camera motion by directly processing MPEG motion vectors;
2) we illustrate the usefulness of the above by application to the analysis and annotation of compressed basketball video and show how useful information about the video content can be obtained directly from the MPEG data.

The remainder of the paper is organized as follows. In Section II, we briefly describe the problem formulation, the MPEG video format, and the data which are used in the proposed system. Section III addresses our first problem: the computation of the underlying camera motion directly from the MPEG data stream. Section IV illustrates how the camera motion algorithm can be used to identify interesting basketball events such as FB's and shots at the basket directly from compressed video. Section V considers an interesting associated problem of segmenting and classifying basketball video into wide-angle and close-up video shots. Although this is not the focus of the paper, such a classification is assumed in Section IV, and this section presents one approach that might be used for this problem using estimates of camera motion.

II. Problem Formulation

Our objective is to rapidly process video stored in MPEG format, without full-frame decompression, to determine with reasonable reliability certain characteristics of the content. For certain types of video, the motion of the camera can be an important feature to aid in this analysis and annotation of content. Hence, it would be useful to be able to estimate camera motion directly from the appropriate MPEG data.

In order to reduce the processing time, we want to use data that can be easily extracted from MPEG video bit streams and that contain useful information about the video content. We focus on MPEG-1 video streams. However, our algorithms should readily extend to MPEG-2 or the forthcoming MPEG-4 video streams. To begin, we briefly describe the relevant parts of the MPEG video compression standard [16] and specify the data which will be used in analyzing video content.

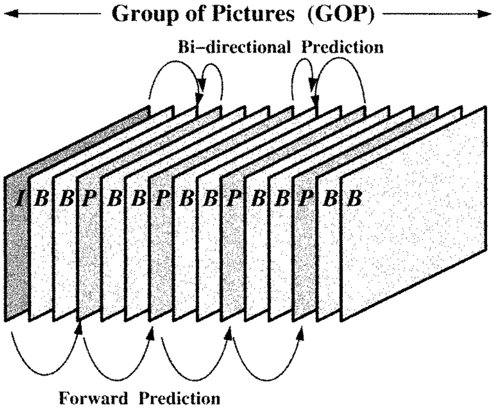


Fig. 1. A typical MPEG encoding GOP.

The syntax of MPEG-1 video defines three main types of coded pictures: intracoded pictures (I-frames), predicted pictures (P-frames) and bidirectionally predicted pictures (B-frames). These pictures are organized into sequences of groups of pictures (GOP) in MPEG video streams. A GOP must start with an I-frame. This may be followed by any number of I and P-frames. The I and P-frames are usually referred to as anchor frames. Between each pair of consecutive anchor frames, several B-frames may appear. In addition, B-frames may appear after the last anchor frame in the GOP and before the I-frame of the next GOP. Fig. 1 shows the typical GOP structure (15–frame sequence of $IBBPBBPBBPBBPBB$) that is used in coding video at a rate of 30 frames/s.

Each video frame is divided into a sequence of nonoverlapping macroblocks. For video coded in 4 : 2 : 0 format [16], each macroblock consists of six 8 × 8 pixel blocks: four luminance (Y) blocks and two chrominance ($CbCr$) blocks. Each macroblock is then either intracoded or intercoded. An I-frame is completely intracoded: every 8 × 8 pixel block in the macroblock is transformed to the frequency domain using the discrete cosine transformation (DCT). The 64 DCT coefficients are then quantized (lossy) and entropy (run length and Huffman, lossless) encoded to achieve compression. Since the coding of an I-frame does not refer to any other video frames, it can be decoded independently and thus provides an entry point for fast random access to the compressed video.

Each P-frame is predictively encoded with reference to its previous anchor frame—the previous I or P-frame. For each macroblock in the P-frame, a local region in the anchor frame is searched for a good match in terms of the difference in intensity. If a good match is found, the macroblock is represented by a motion vector to the position of the match together with the DCT encoding of the difference (or residue) between the macroblock and its match. The DCT coefficients of the residue are quantized and entropy coded while the motion vector is differentially and entropy (Huffman) coded with respect to its neighboring motion vector. This is normally known as encoding with *forward motion compensation*, and we shall refer this type of encoded macroblock as an *intercoded macroblock*. If a good match cannot be found, the macroblock is intracoded using the same method for coding the macroblocks of I-frames. Usually, the residue of an intercoded macroblock can be coded with fewer bits. So, an intercoded macroblock has better compression gain compared to an intracoded macroblock. It is expected that a smaller change of image content between a P-frame and its anchor frame will result in more macroblocks being well matched in the anchor frame, and hence, fewer macroblocks requiring intracoding.

To achieve further compression, B-frames are bidirectionally predictively encoded with forward and/or backward motion compensation referenced to its nearest past and/or future I- and/or P-frames. Since B-frames are not used as a reference for coding other frames, they can accommodate more distortion, and thus, higher compression gain compared to I or P-frames.

We restrict ourselves to using data which can be easily extracted from MPEG bit streams without full frame decompression. Specifically, we use: the frame number, frame encoding type (I, P, or B), the positions and motion vectors of intercoded macroblocks, the number of intracoded blocks, and the DC coefficient of each DCT-encoded pixel block. This data can be obtained by parsing and entropy (Huffman) decoding the MPEG bit streams. These operations make up less than 20% of the computational load in the full MPEG-1 decoding process [17]. For example, fast algorithms for certain tasks using the DC coefficients extracted from each video frame have recently been reported in [5] and [15].

III. Estimating Camera Motion from MPEG Video

In this section, we present algorithms for estimating camera motion (pan, tilt, and zoom) directly from MPEG compressed video. We assume, of course, that the underlying video is suitable for this task, i.e., if the video was not compressed, then it would be possible to reliably estimate the camera motion from the video data.

A. Preliminaries: Estimation from Uncompressed Video

We first quickly review a method for camera motion estimation from uncompressed video. In this case, if one assumes that the camera is undergoing rotation and zoom but no translation, then the change of image intensity between frames can be modeled by the following 6-parameter projective transformation [18]:

$$\begin{aligned} x' &= \frac{p_1 x + p_2 y + p_3}{p_5 x + p_6 y + 1} \\ y' &= \frac{-p_2 x + p_1 y + p_4}{p_5 x + p_6 y + 1}. \end{aligned} \tag{1}$$

Here, (x, y) and (x', y') are the image coordinates of corresponding points in two neighboring frames and $p_1, \cdots, p_6$ are parameters of the camera motion. The image coordinates are, with respect to a standard orthogonal set of axes, centered in the image, i.e., (0, 0) corresponds to the image center. This model is a very good approximation when there are minimal lens distortion effects, the interframe camera motion is small, and the camera has only zoom and rotation about its three body centered orthogonal axes (X vertically, Y horizontally, and Z through the lens), but no translation.

Suppose that for the first frame, the focal length of the camera is f, and before taking the next frame, the camera undergoes small rotations through angles α, β, and γ about the X, Y, and Z camera axes and the focal length changes to sf. Then, from [18], we know that the parameters $p_1, \cdots, p_6$ satisfy

$$\begin{pmatrix} p_1 & p_2 & p_3 \\ -p_2 & p_1 & p_4 \\ p_5 & p_6 & 1 \end{pmatrix} = \begin{pmatrix} s & s\gamma & -sf\alpha \\ -s\gamma & s & sf\beta \\ \alpha/f & -\beta/f & 1 \end{pmatrix}. \tag{2}$$

Thus, p_1 indicates the interframe camera zoom factor; $p_1 > 1$: zoom in, $p_1 < 1$: zoom out, and $p_1 = 1$: no zoom. The ratio p_2/p_1 is the change in the angle of camera rotation about the lens axis between the two frames, the ratios $-p_3/p_1$ and p_4/p_1 are the changes in the camera pan and tilt angles between the two frames scaled by the first focal length, and p_5 and p_6 capture perspective convergence effects due to camera pan and tilt rotations.

Consider a sequence of frames indexed by $i = 0, \cdots, m$. Let p_j^i denote the value for the parameter p_j, $j = 1, \cdots, 6$, of the projective transformation from frame $i-1$ to frame i. Since p_1^i represents the change in camera zoom from frame $i-1$ to frame i, the net camera zoom after m frames is $z(m) = \prod_{i=1}^m p_1^i$. When there is no zoom, $-p_3^i/p_1^i = -p_3^i$ represents the change in the camera pan angle between frames $i-1$ and i in units scaled by the constant focal length f. In this case, the net change in the pan angle (in scaled units) after m pairs of frames is $p(m) = -\sum_{i=1}^m p_3^i$. When the camera is both zooming and panning, the change in the pan angle between frames $i-1$ and i in constant units scaled by the initial focal length f is $-p_3^i/z(i)$. Hence, the net change in the pan angle (in these scaled units) after m frames is $p(m) = -\sum_{i=1}^m p_3^i/z(i)$. A similar equation can be written for the camera tilt. All three equations can be expressed implicitly by the nonlinear difference equations

$$\begin{aligned} z(m) &= p_1^m z(m-1), \quad z(0) = 1 \\ p(m) &= p(m-1) - p_3^m/z(m), \quad p(0) = 0 \\ t(m) &= t(m-1) + p_4^m/z(m), \quad t(0) = 0. \end{aligned}$$

With these caveats in mind, we henceforth refer to p_1 as the camera zoom factor and the ratios $-p_3/p_1$ and p_4/p_1 as the camera pan and tilt rates, respectively.

Of course, the parameters $p_1, \cdots, p_6$ must be estimated for each consecutive pair of frames in the video. A natural way to do this is by sampling the projective transformation function $(x, y) \mapsto (x', y')$ given by (1), and then solving a least-squares regression problem. The sampling of the transformation can be done by selecting "good" local features in the first frame and finding their corresponding locations in the second frame using feature matching. This yields a set of "noisy" samples $(x_i, y_i) \mapsto (x_i', y_i')$ of the projective transformation function. With $w_i = (x_i, y_i)^T$, the coordinates of the center of the ith feature block in the first image, and $w_i' = (x_i', y_i')^T$, the coordinates of the center of the matched block in the second image, these samples can then be used to form a regression problem that can be solved to determine estimates of the unknown parameters of the transformation. For example, we determine $\theta \triangleq (p_1, \cdots, p_6)$ to minimize

$$Q(\theta) \triangleq \sum_{i=1}^{N} \left\| w_i' \,(p_5 \quad p_6)\, w_i + w_i' - \begin{pmatrix} p_1 & p_2 \\ -p_2 & p_1 \end{pmatrix} w_i - \begin{pmatrix} p_3 \\ p_4 \end{pmatrix} \right\|^2. \tag{3}$$

This is a linear regression that can be readily solved using standard numerical linear algebra methods. See, e.g., [18].

There are two difficulties with this approach. First, the features need to be carefully selected so that they have easily distinguished characteristics that can be accurately located and matched in the second frame, and second, finding the corresponding match to each feature in the second frame is a time-consuming process. As a result, schemes of this form operate far slower than the frame rate of the video, or even the P-frame rate of standard compressed video.

B. Camera-Motion Estimation from the P-frames of MPEG Video

Here, we propose a variation of the above scheme that will estimate the camera motion directly from the MPEG data without the need for feature selection or feature matching. The idea is as follows. To estimate the six-unknown parameters of the projective transformation from the previous anchor frame to the current P-frame, the positions of each intercoded macroblock with nonzero motion and its best match (computed from the MPEG motion vector) are considered as a corresponding feature pair between the P-frame and its anchor frame. This yields a set of (potentially very noisy) samples of the projective transformation between the two frames.

In detail, let the row and column coordinates (i.e., standard image coordinates) of the center of the kth intercoded macroblock in the current P-frame be (i_k, j_k) and its corresponding motion vector (in consistent units) be $(\Delta i_k, \Delta j_k)$. Then, with respect to image-centered Cartesian axes, the center of the kth intercoded macroblock has coordinates

$$w_k' \triangleq \begin{pmatrix} x_k' \\ y_k' \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \left[\begin{pmatrix} i_k \\ j_k \end{pmatrix} - \begin{pmatrix} i_o \\ j_o \end{pmatrix} \right]$$

where (i_o, j_o) are the image coordinates of the center of the image. This macroblock was matched with the point in the previous anchor frame that has image-centered Cartesian coordinates

$$\begin{aligned} w_k \triangleq \begin{pmatrix} x_k \\ y_k \end{pmatrix} &= \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \left[\begin{pmatrix} i_k + \Delta i_k \\ j_k + \Delta j_k \end{pmatrix} - \begin{pmatrix} i_o \\ j_o \end{pmatrix} \right] \\ &= \begin{pmatrix} j_k + \Delta j_k - j_o \\ -i_k - \Delta i_k + i_o \end{pmatrix} = w_k' + m_k \end{aligned}$$

where $m_k = (\Delta j_k, -\Delta i_k)^T$. Hence, we take $w_k \mapsto w_k'$ as a sample of the unknown projective transformation from the anchor frame to the current P-frame. Each intercoded macroblock in the current P-frame thus contributes one (noisy) sample of the projective transformation. These samples are then used in a linear-in-the-parameters least-squares regression of the form (3) to estimate the unknown parameters of the projective transformation. In this case, the cost to be minimized is

$$Q(\theta) \triangleq \sum_{k=1}^{N} \left\| w_k' \,(p_5 \quad p_6)\, (w_k' + m_k) + w_k' - \begin{pmatrix} p_1 & p_2 \\ -p_2 & p_1 \end{pmatrix} (w_k' + m_k) - \begin{pmatrix} p_3 \\ p_4 \end{pmatrix} \right\|^2. \tag{4}$$

Notice that all of the required data can be obtained directly from the current P-frame: $w_k' = (x_k', y_k')$ are the coordinates of the center of the kth intercoded macroblock (with respect to an image centered coordinate system), $m_k = (u_k, v_k)^T \triangleq (\Delta j_k, -\Delta i_k)^T$ is derived from its corresponding motion vector, and N is the number of intercoded macroblocks.

Let $\hat{p}_1, \cdots, \hat{p}_6$ denote the resultant estimates for the six parameters. Then for each intercoded macroblock we define the residual position errors as

$$e_i^x = \frac{x_i' - (\hat{p}_1(x_i' + u_i) + \hat{p}_2(y_i' + v_i) + \hat{p}_3)}{(\hat{p}_5(x_i' + u_i) + \hat{p}_6(y_i' + v_i) + 1)}$$

$$e_i^y = \frac{y_i' - (-\hat{p}_2(x_i' + u_i) + \hat{p}_1(y_i' + v_i) + \hat{p}_4)}{(\hat{p}_5(x_i' + u_i) + \hat{p}_6(y_i' + v_i) + 1)}.$$

These provide a measure of the accuracy of $\hat{p}_1, \cdots, \hat{p}_6$ in matching the data samples of the projective transformation. From these, we can compute the sample standard deviations σ^x and σ^y of the residuals in the x and y directions. These standard deviations can be used to fine tune the estimation as follows. Those sample pairs which are inconsistent with the estimated parameters, i.e., $e_i^x > c\sigma^x$ or $e_i^y > c\sigma^y$, where c is a constant, can be rejected as outliers. The parameters can then be reestimated from the remaining data. In practice, this outlier rejection can help compensate for the fact that some of the samples obtained from the P-frame motion vectors are highly erroneous. This is discussed further below.

Several comments are in order. First, the purpose of motion compensation in MPEG coding is to remove temporal redundancy by finding a good match in the anchor frame for each intercoded macroblock. Thus, the resultant motion vector does not necessarily represent the “real motion” of the intercoded macroblock. This is not surprising: in the feature-based method, the local block features were carefully selected so that their intensity patterns could be well matched in the next frame but the macroblocks of the P-frames do not necessarily have this property. Thus, although it may be possible to find a good match for a macro block in its anchor frame, this does not mean that it has been matched with its correct location in the anchor frame. This is illustrated by several of the macroblocks in the P-frames from one MPEG GOP shown in Fig. 2. The success of the motion-vector-based estimation method will require that a significant number of the macroblocks do contain intensity patterns that can be accurately matched in the anchor frame. The outlier rejection iteration discussed above can then help reject the data from those macroblocks that did not contain “good features.” In practice, it has been found that the estimation/outlier rejection

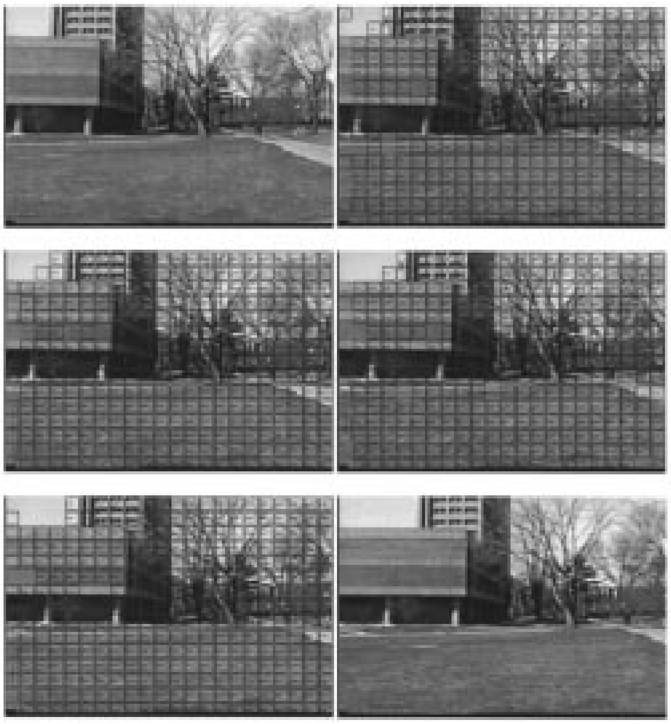


Fig. 2. The initial I-frame and the subsequent P-frames (frames 4, 7, 10, and 13) from one 15-frame GOP with GOP sequence $IBBPBBPBBPBBPBB$ from an MPEG video with almost pure pan to the left. For each P-frame, the intercoded motion blocks and the corresponding motion vectors are shown. The last image at the bottom right is the first I-frame of the next GOP.

iteration usually converges to a satisfactory result with only two or three iterations.

In a P-frame, whether a macroblock is inter or intracoded is determined by several rules [17]. These can be summarized as follows.

1) When a macroblock cannot be well predicted from its anchor frame, i.e., the prediction residual is above a certain threshold, it is intracoded. Otherwise, it is intercoded.
2) Since additional bits are required to code a nonzero motion vector, when the prediction residual of coding a macroblock by using a nonzero motion vector is close to the prediction residual of using a zero motion vector, the motion vector is set to zero and the macroblock is coded without motion compensation. This usually happens, for example, in coding macroblocks within regions of nearly constant intensity.
3) If the quantization coefficients of the prediction residual of an intercoded macroblock are all zero, then the prediction residual is not coded. There are cases where the quantization coefficients of the prediction residual within one or more blocks of a macroblock are all zero. In these cases, only the blocks with nonzero quantization coefficients are required to be coded and they are indicated by a 6-bit coded block pattern.

From these rules, we conclude that intercoded macroblocks with zero motion vectors might be unreliable since they may not be predicted from their true corresponding regions. Thus, they can be excluded from the samples that constitute the parameter estimation problem (4), i.e., the sum is taken only over those intercoded macroblocks with nonzero motion vectors. One exception to this is when the video scene is actually stationary, then almost all of the macroblocks will be intercoded with zero motion vectors. This suggests that when most of the macroblocks of the P-frame are intercoded and the percentage of the intercoded macroblocks with zero motion vector is high, we should set the estimated interframe camera motion to zero, i.e., set $\hat{p}_1 = 1, \hat{p}_2 = \hat{p}_3 = \hat{p}_4 = \hat{p}_5 = \hat{p}_6 = 0$. Conversely, the intercoded macroblocks that have nonzero motion vectors have the potential to provide useful interframe motion information. In particular, an intercoded macroblock with nonzero motion vectors and with one or more of its 8 × 8 blocks not requiring residual coding is potentially more reliable, since a very good match has been found for it in the anchor frame. This information could be used to give extra weight to this sample in the least-squares estimation problem.

Since we are only using the information from P-frames, we are sampling the camera motion. For example, there are four P-frames within the typical GOP shown in Fig. 1. If the MPEG video is coded at a rate of 30 frames/s using this GOP, there are 8 P-frames per second in the video. Thus, by using only the information in each P-frame, we are effectively sampling the pan rate at 8 samples/s. This means that we will require the underlying camera motion rates (per frame) to have a bandwidth of less than 4 Hz. For a great deal of interesting video, this is a reasonable assumption. Using the motion vectors from both the P and B-frames has the potential to yield better accuracy and allow for higher motion bandwidth, but at the cost of increased computation.

Example: Fig. 2 shows the initial I-frame (I_1) and the subsequent 4 P-frames (P_4, P_7, P_{10}, P_{13}) of a single MPEG GOP with structure $IBBPBBPBBPBBPBB$. The last image is the first I-frame of the next GOP. Superimposed on each P-frame, we have indicated the location of the intercoded macroblocks with their motion vectors represented by vectors emanating from the center of each macroblock. The true camera motion is almost pure pan to the left. Notice that the majority of the motion vectors reflect the true camera motion. However, there are some notable exceptions. For example, in P_4 and P_{10}, the two macroblocks in the upper left edge of the building have been swapped. This is a reflection of the fact that they were not good feature blocks. Of course, the matching indicated is adequate for coding purposes even though it gives an erroneous sample of the projective transformation induced by the camera motion. The same phenomenon can be seen to a lesser extent in some of the blocks in the grassy area in the foreground of the scene in P_{10} and in the windows of the building in P_{13}.

We applied our motion-estimation algorithm to the first three GOP from the video indicated in Fig. 2. The relevant frames are

$$I_1B_2B_3P_4B_5B_6P_7B_8B_9P_{10}B_{11}B_{12}P_{13}B_{14}B_{15}$$
$$I_{16}B_{17}B_{18}P_{19}B_{20}B_{21}P_{22}B_{23}B_{24}P_{25}B_{26}B_{27}P_{28}B_{29}B_{30}$$
$$I_{31}B_{32}B_{33}P_{34}B_{35}B_{36}P_{37}B_{38}B_{39}P_{40}B_{41}B_{42}P_{43}B_{44}B_{45}.$$

Of these P_4, P_7, P_{10}, P_{13} from the first GOP, $P_{19}, P_{22}, P_{25}, P_{28}$ from the second GOP, and P_{34}, P_{37}, P_{40}, and P_{43} from the third GOP can be used to estimate the camera motion. Notice that I_1, I_{16} and I_{31}, i.e., the start frame of each

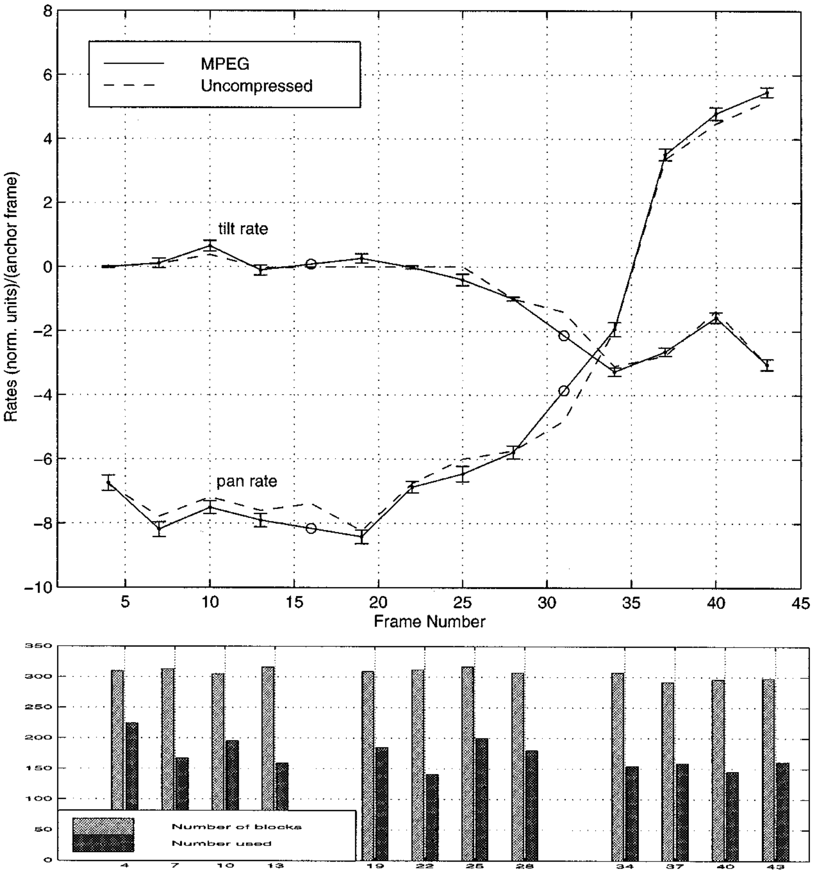


Fig. 3. The estimated pan and tilt rates for the P-frames of the video shown in Fig. 2 estimated by 1) using the motion vectors from the P-frame macroblocks and the estimation scheme proposed in Section III-B and 2) using the uncompressed frames corresponding to the P-frames and a feature-based estimation method. The error bars indicate ±one standard deviation of the residual errors. Data points were rejected if their residual error exceeded 1.5 times the standard deviation of the residual error. The bar chart indicates the number of intercoded blocks with a nonzero motion vector in each P-frame and the number of blocks used in the final estimation after outlier rejection.

GOP, cannot be used since they are entirely intracoded. P_4 permits us to estimate the projective transformation from I_1 to P_4, then from P_7 we estimate the projective transformation from P_4 to P_7, and so on until we use P_{13} to estimate the projective transformation from P_{10} to P_{13}. The next natural step would be to estimate the projective transformation from P_{13} to I_{16}; however, this cannot be done from the compressed data, since I_{16} is entirely intracoded. The next available estimate, based on P_{19}, will be of the projective transformation from I_{16} to P_{19}, and so on. Thus, we obtain regularly spaced samples of the camera motion every three frames, except that every fifth sample is missing. To estimate the missing data, we can interpolate the previous and subsequent samples.

The results of the estimation are shown in Fig. 3. The estimated values are plotted with one standard-deviation error bars and the interpolated values are plotted using “o.” The rates are in focal length normalized units per anchor frame. To plot the estimated rates per frame, divide these quantities by three and use zero-order interpolation between frames. The bar chart below the plot of the estimated rates indicates, for each P-frame, the number of macroblocks with nonzero motion vectors and the number that were used in the estimation after outlier rejection ($c = 1.5$). For comparison, we used a feature-based method operating on the corresponding uncompressed frames to also estimate the camera motion, i.e., we use every third uncompressed frame so as to match the data available from the MPEG scheme. However, in this case there will be no missing samples. If there is no direct measurement of the camera motion, then this is the best estimate of the camera motion that one can reasonably expect. The results of this estimation are also indicated in Fig. 3. As the figure indicates, except at the interpolated points, the results of the two methods are in reasonably good agreement.

Example: Fig. 4 shows the initial I-frame (I_{31}) and the subsequent four P-frames (P_{34}, P_{37}, P_{40}, P_{43}) of the third GOP of an MPEG video with each GOP having the structure $IBBPBBPBBPBBPBB$. The last image is the first I-frame of the next GOP. Superimposed on each P-frame we have indicated the location of the intercoded macroblocks with their motion vectors represented by vectors emanating from the center of each macroblock. In this example, the camera motion is a fast pan to the left (as a player drives to the basket), followed by upward tilt and camera zoom (as the player makes a shot). In addition, there are moving objects in the field of view and their motion is not necessarily in agreement with the camera motion.

We applied our motion-estimation algorithm to four GOP from the video indicated in Fig. 4. The results of the estimation are shown in Fig. 5. The estimated values ($-p_3$, p_4, p_1)

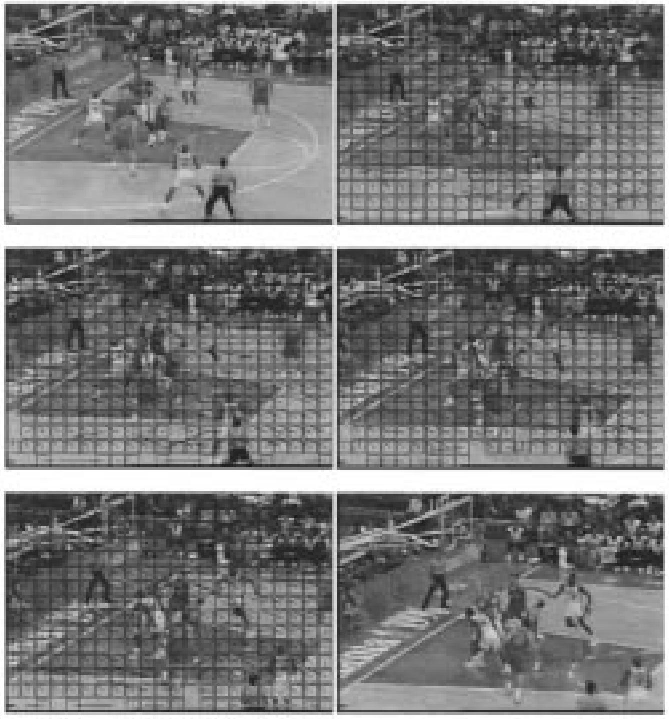


Fig. 4. The initial I-frame and the subsequent P-frames (frames 34, 37, 40, and 43) from the third GOP with GOP sequence $IBBPBBPBBPBBPBB$ from an MPEG video with pan from left to the right, upward tilt, and zoom. For each P-frame, the intercoded motion blocks and the corresponding motion vectors are shown. The last image at the bottom right is the first I-frame of the next GOP.

are plotted directly with the interpolated values indicated by "o." The bar chart below the plot of the parameter estimates indicates for each P-frame, the number of macroblocks with nonzero motion vectors and the number that were used in the estimation after outlier rejection ($c = 1.5$). For comparison, we also estimated the camera-motion parameters using a feature-based method operating on the corresponding uncompressed frames. The results of this estimation are also indicated in the figure. Despite the movement of players and the uniformity of the court area, the results of the two methods are in fairly good agreement. The estimates of camera zoom differ slightly in numerical value but have the same qualitative form.

C. A Faster Estimation Algorithm Using the P-Frames of MPEG Video

In this subsection we develop a simple but faster camera motion estimation algorithm that will operate well under mild additional restrictions on the true camera motion and the geometry of the scene. These assumptions are approximately satisfied for a wide range of interesting video.

To obtain the simplified algorithm, we assume that from one frame to the next: 1) we can set $p_5 = p_6 = 0$, i.e., that perspective distortion effects are minimal and 2) we can set $p_2 = 0$, i.e., that the camera does not rotate about the axis of the camera lens. In this case, the 6-parameter model (1) can be approximated by the three-parameter transformation

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} p_1 & 0 \\ 0 & p_1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} p_3 \\ p_4 \end{pmatrix} \tag{5}$$

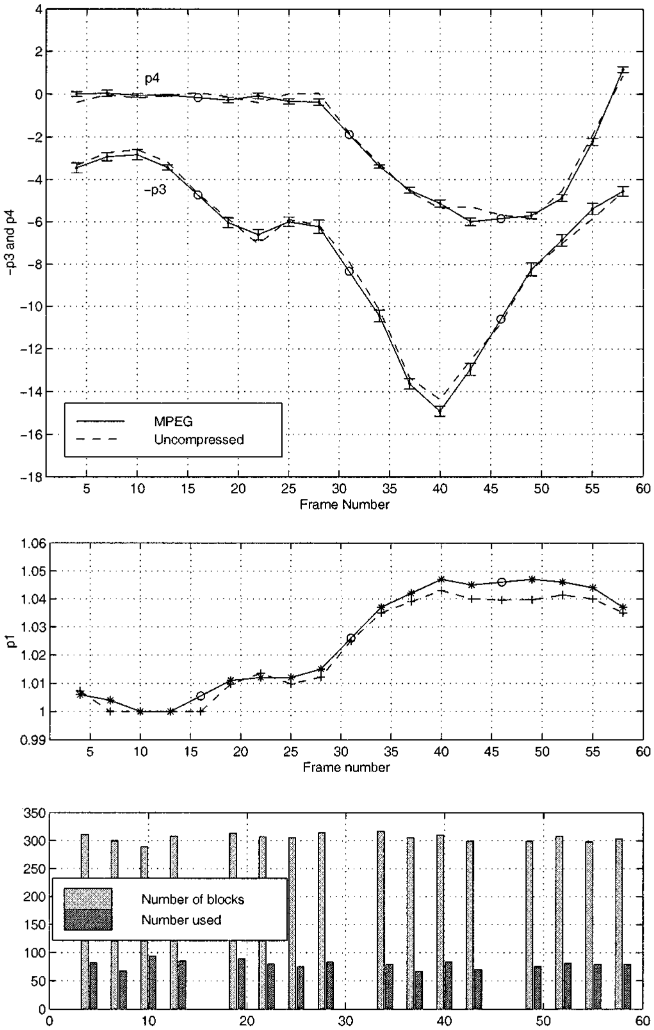


Fig. 5. The estimated pan and tilt rates for the P-frames of the first four GOP from the video shown in Fig. 4 estimated by 1) using the nonzero motion vectors from the P-frame macroblocks and the estimation scheme proposed in Section III-B and 2) using the uncompressed frames corresponding to the P-frames and a feature-based estimation method. The error bars indicate ± one standard deviation of the residual errors. Data points were rejected if their residual error exceeded 1.5 times the standard deviation of the residual error. The bar chart indicates the number of intercoded blocks with a nonzero motion vector in each P-frame and the number used in the final estimation after outlier rejection.

where $p_1 = s$, $p_3 = -sf\alpha$ and $p_4 = sf\beta$. Set $q_3 = -f\alpha$ and $q_4 = f\beta$. Then the above relationship can also be written as

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = s\left[\begin{pmatrix} x \\ y \end{pmatrix} + b\right] \qquad b = \begin{pmatrix} q_3 \\ q_4 \end{pmatrix}. \tag{6}$$

The advantage of the approximate model (6) is that the three parameters s, q_3 and q_4 can be quickly and easily estimated. Note that in our existing terminology, s is the zoom factor, $-q_3$ is the pan rate, and q_4 is the tilt rate. To estimate these from the P-frames of an MPEG video we assume, as before, that each intercoded macroblock of a P-frame provides a (noisy) sample $w_k \mapsto w'_k$ of the transformation (6). Then we estimate s and b by finding the values that minimize

$$J(s, b) \triangleq \sum_{k=1}^{N} \|w'_k - s(w_k + b)\|^2. \tag{7}$$

Notice that if we set $c = sb$, then

$$\begin{aligned} J(s, b) &= \sum_{k=1}^{N} \|w'_k - s(w_k + b)\|^2 \\ &= \sum_{k=1}^{N} \|w'_k - sw_k - c)\|^2 = K(s, c). \end{aligned}$$

The function $K(s, c)$ is clearly a convex function of the three parameters s, c_1, c_2 where $c = (c_1, c_2)^T$. Provided our samples form a sufficiently rich set, $K(s, c)$ will be strictly convex, and hence, have a unique minimizing solution (s^*, c^*). It follows that under these conditions, $J(s, b)$ also has a unique minimizing solution $(s^*, b^*) \triangleq (s^*, c^*/s^*)$. We assume henceforth that this holds.

Now for any $s \neq 0$ and b

$$\begin{aligned} J(s, b) &= \sum_{k=1}^{N} s^2 \left\| \frac{w'_k}{s} - w_k - b \right\|^2 \\ &\geq \sum_{k=1}^{N} s^2 \left\| \frac{w'_k}{s} - w_k - b(s) \right\|^2 \end{aligned}$$

where

$$\begin{aligned} b(s) &\triangleq \frac{1}{N} \sum_{k=1}^{N} \frac{w'_k}{s} - w_k = \frac{\overline{w}'}{s} - \overline{w} \\ \overline{w}' &\triangleq \frac{1}{N} \sum_{k=1}^{N} w'_k \\ \overline{w} &\triangleq \frac{1}{N} \sum_{k=1}^{N} w_k. \end{aligned}$$

It follows that $J(s^*, b^*) \geq J(s^*, b(s^*))$, and hence that $J(s^*, b^*) = J(s^*, b(s^*))$. So, the unique minimizing solution has $b^* = b(s^*)$.

Now for any s

$$\begin{aligned} J(s, b(s)) &= \sum_{k=1}^{N} \|w'_k - \overline{w}' - s(w_k - \overline{w})\|^2 \\ &\geq \sum_{k=1}^{N} \|w'_k - \overline{w}' - s^o(w_k - \overline{w})\|^2 \end{aligned}$$

where

$$s^o \triangleq \frac{\sum_{k=1}^{N} (w'_k - \overline{w}')^T (w_k - \overline{w})}{\sum_{k=1}^{N} \|w_k - \overline{w}\|^2}.$$

Thus, $J(s^*, b^*) = J(s^*, b(s^*)) \geq J(s^o, b(s^o))$. This implies that $J(s^*, b^*) = J(s^o, b(s^o))$, and hence by uniqueness that

$$\begin{aligned} s^* &= \frac{\sum_{k=1}^{N} (w'_k - \overline{w}')^T (w_k - \overline{w})}{\sum_{k=1}^{N} \|w_k - \overline{w}\|^2} \\ b^* &= \frac{\overline{w}'}{s^*} - \overline{w}. \end{aligned}$$

These equations give closed-form expressions for estimates of the camera zoom factor (s) and the pan and tilt rates ($-\hat{q}_3$ and $\hat{q}_4$). These closed-form expressions can be quickly evaluated from the MPEG data using $14N + 6$ floating-point operations where N is the number of intercoded macroblocks with nonzero motion vectors. By contrast, the previous 6-parameter scheme required the solution of a linear least-squares problem of size $2N \times 6$. This is a task requiring greater than $2N \times 6 \times 6 = 72N$ flops. Hence, we expect the approximate 3-parameter scheme to be roughly five times faster per iteration. Typically, the robust estimation takes 3–4 iterations; thus, the 3-parameter scheme will be typically 15–20 times faster than the 6-parameter scheme.

Example: We applied our fast motion estimation algorithm to the first four GOP from the video indicated in Fig. 4. The results of the estimation are shown in Fig. 6. The estimated values $(-q_3, q_4, s)$ are plotted with the interpolated values indicated by o. The bar chart below the plot of the parameter estimates indicates for each P-frame the number of macroblocks with a nonzero motion vector and the number that were used in the estimation after outlier rejection ($c = 1.5$). For comparison, we have also plotted on the same graph the estimated camera motion parameters using a feature-based method operating on the corresponding uncompressed frames and the estimates obtained using the six parameter MPEG-based estimation method of Section III-B. As the figure indicates, the results of all three methods are in reasonable agreement.

IV. Application to the Annotation of Basketball Video

To test the proposed camera-motion estimation algorithm on realistic compressed video, we have applied the algorithm to four MPEG basketball video sequences. These sequences, listed in Table I, were coded from three different recorded basketball video programs using two different commercial MPEG hardware encoders with a frame rate 30 frames/s, a bit rate 1.5 Mb/s (including audio), and the GOP pattern shown in Fig. 1.

The basketball video consists of a sequence of wide-angle shots of a relatively long duration which capture the flow of the play, interspersed with close-up shots of short duration, which capture interesting highlights. The wide-angle shots are captured by a mid-court camera mounted on one side of the central region of the court. The top row of Fig. 7 shows some typical images from several different wide-angle video shots. The bottom row of Fig. 7 shows some sample images from different close-up shots.

During a wide-angle shot, the mid-court camera follows the ball. Thus, when the ball is being advanced from one end of

TABLE I
NUMBER OF FRAMES, VIDEO SHOTS, WIDE-ANGLE (WA) SHOTS, AND CLOSE-UP (CU) SHOTS OF FOUR TEST BASKETBALL VIDEO SEQUENCES

Sequence	No. of frames	No. of shots	No. of WA shots	No. of CU shots
NCAA96	5462	8	3	5
NBA96	5654	20	5	15
Olympic96-1	6800	21	7	14
Olympic96-2	7052	23	7	16

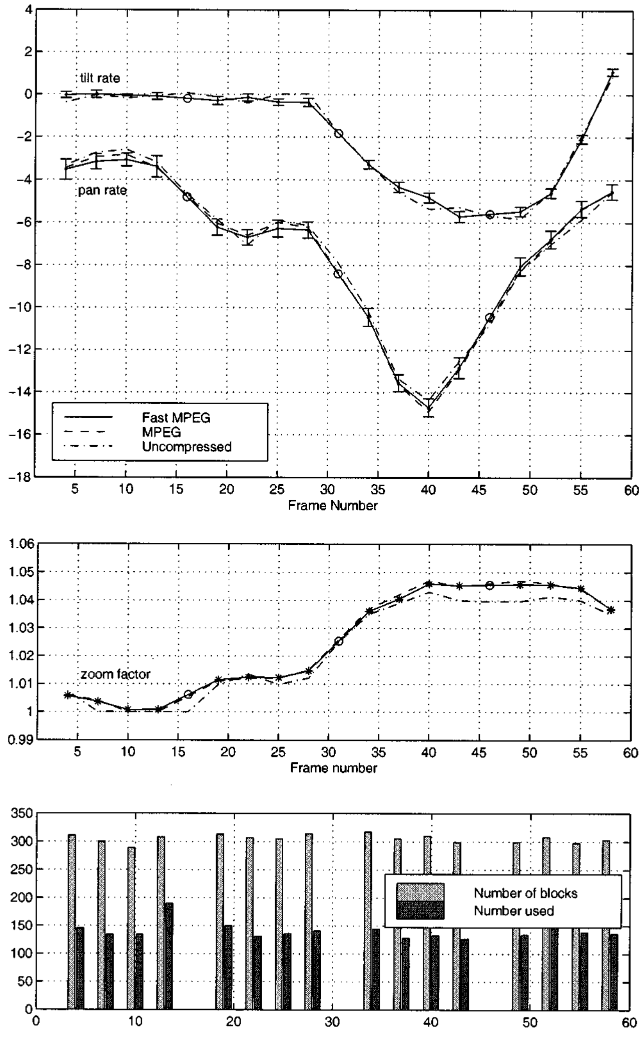


Fig. 6. The estimated pan and tilt rates for the P-frames of the first four GOP from the video shown in Fig. 4 estimated by: 1) using the motion vectors from the P-frame macroblocks and the fast estimation scheme proposed in Section III-C; 2) using the MPEG estimation scheme proposed in Section III-B; and 3) using the uncompressed frames corresponding to the P-frames and a feature-based estimation method. The error bars indicate ±one standard deviation of the residual errors. Data points were rejected if their residual error exceeded 1.5 times the standard deviation of the residual error. The bar chart indicates the number of intercoded blocks with a nonzero motion vector in each P-frame and the number used in the final estimation after outlier rejection.

the court to the other, the camera is panned consistently in that direction. Following the pan, the camera may have immediate zoom-in if a shot is attempted, or the camera will follow the ball as it is passed around the key area and zoom-in when a shot at the basket is attempted. Thus, estimating the camera pan rate and zoom factor during the wide-angle shot should provide useful information about the content. Although the players are also in motion, they usually only take up a small area of the field of view, and their movement can be regarded as an additional source of "noise" in the camera-motion estimation problem.

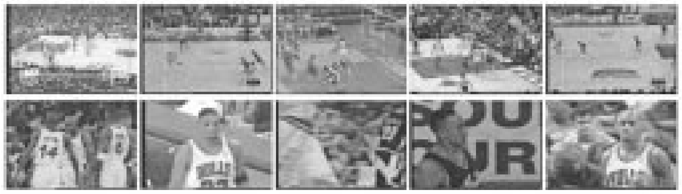


Fig. 7. Sample images from different wide-angle shots (top row) and close-up shots (bottom row) of basketball video.

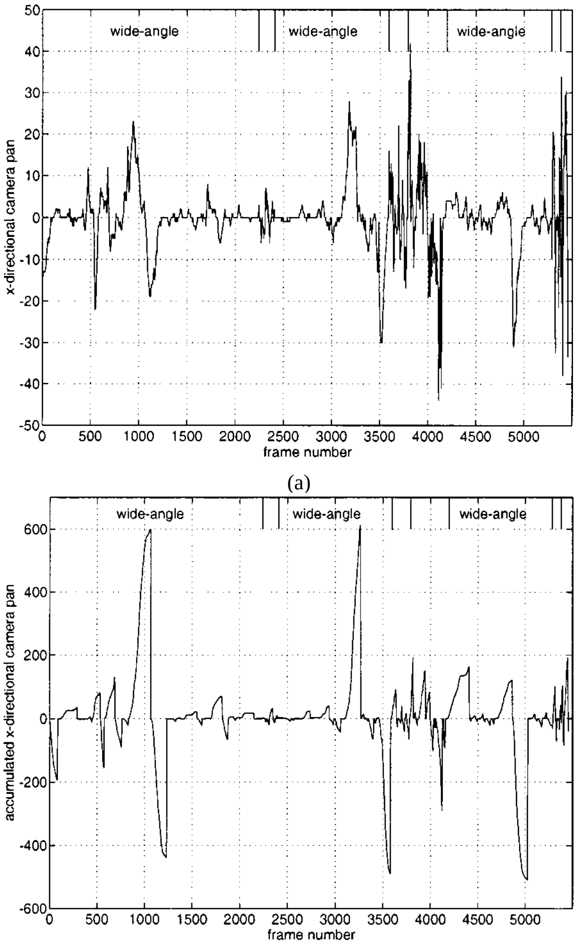
By contrast, the close-up shots are usually of short duration and the motion of the camera does not necessarily dominate the change in image intensity from frame to frame. As a result, computing reliable camera motion from the close-up shots is generally difficult, and in any case, of questionable value.

This prior knowledge of the structure of the video suggests that to analyze the content directly from the MPEG data: 1) we temporally segment the video into wide-angle and close-up shots; 2) for each wide-angle shot, we use the camera-motion estimation algorithm to determine the actions of the camera; and 3) we then use the estimated camera motion over wide-angle shots to make useful annotations on content.

By "useful annotation," we mean a correct identification of interesting events such as full court advances (FCA's), Fast breaks (FB's), shots at the basket, etc., within each wide-angle shot. By nature, the assignment of content annotations to the basketball video is often subjective. What constitutes a "fast-break," for example, is a subjective decision that could be made under a variety of different criteria. It is not our objective to study this issue. We simply wish to indicate that the estimates of camera motion from the MPEG data are sufficiently accurate to enable correct qualitative decisions about the video content, and hence, can serve as a useful component in an annotation system.

Since the motion of the mid-court camera is dominated by zoom and horizontal pan, only the estimated parameters p_1 (zoom rate) and $-p_3/p_1$ (pan rate) will be used to analyze the video content. In addition, we will assume for the moment that a temporal segmentation and classification of the video into wide-angle and close-up shots is available. An automatic means of obtaining this segmentation and classification directly from the MPEG video is discussed in a later section.

To clearly see persistent camera motion, we can accumulate, i.e., sum, the estimated pan rate. A slight variation on this is often useful: we sum the magnitude of the pan rate when the pan



(a)

(b)

Fig. 8. (a) Estimated camera pan rate of the NCAA96 basketball sequence. (b) Its directional accumulated pan.

is in a consistent direction and reset the sum to zero when the pan direction is changed. We call this the *directional accumulated pan*.

A. Detection of FB's and FCA's

FB's and FCA's of the basketball can serve as pivots in parsing the basketball video into semantic sections. They usually indicate the start of a new offense, a change of ball possession, and the positions of the players and the ball, etc. For example, the players and the ball are at the left side of the court after an FB or FCA from right to left.

Fig. 8(b) shows the directional accumulated pan computed from the MPEG estimated camera pan rate [Fig. 8(a)] of the NCAA96 basketball sequence. The five large slopes in the directional accumulated pan correspond to five long and persistent camera pans in one direction that occur when the camera is tracking an FB or FCA of the basketball.

To automatically detect FB's and FCA's, we proceed as follows. For each video sequence, we estimate the camera motion from the MPEG video using either of the MPEG-based algorithms proposed in Section III. Then we compute the directional accumulated pan from the estimated camera motion. With high reliability, we can declare an FB/FCA to be detected when the magnitude of the directional accumulated pan exceeds a preset threshold. The starting and ending point of the detected slope are then considered as the locations of the beginning and end of the corresponding FB/FCA. The magnitude of each slope actually indicates the rate of camera pan, and hence, the rate of the advance of the players between each pair of P–P frames or I–P frames. To decide whether an FCA of the ball is an FB we threshold these slopes. The decision on what constitutes an FB is a qualitative one; our experience was that when the magnitude of the average slope is greater than eight, classification as an FB was justified.

TABLE II
RESULTS OF DETECTING FB'S AND FCA'S FROM THE FOUR TEST BASKETBALL SEQUENCES

Sequence	No. of FB/FCA	Missed detection	False detection
NCAA96	5	0	0
NBA96	5	0	2
Olympic-1	10	1	0
Olympic-2	6	1	0

Table II shows the experimental results of detecting FB/FCA from the four test basketball sequences. Using only the estimated pan rate, the algorithm correctly detected 24 of 26 FB's (92%) and gave false detections for 2 of the 24 detected segments (8%).

B. Detection of Shots at the Basket

Examination of the basketball video indicated that most shots at the basket occur in the following locations:

1) An FB or FCA followed immediately by camera zoom-in.
2) A FB/FCA followed by a short camera still and then another FB/FCA in the opposite direction. In this case a shot at the basket was probably made after the end of the first FB/FCA.
3) A FB/FCA followed by a close-up shot and then another FB/FCA. In this case a shot at the basket was probably made before the close-up shot.
4) A FB followed immediately by another FB in the opposite direction. In this case a shot was probably attempted but missed after the first FB.
5) Whenever there is a camera zoom, especially when the players are at one end of the court.

Although none of these rules is absolute, they manage to limit the locations of probable shots at the basket to a much smaller subset of the video than the entire video sequence. Once these events have been detected, the user of the annotation system or a more sophisticated algorithm can examine the uncompressed video in these segments to identify the locations of the real shots at the basket.

Based on these observations, we designed an algorithm using the estimated camera motion to detect the following video segments as locations of probable shots at the basket.

1) Video segments containing camera zoom-in right after an FB/FCA or when the camera is pointing at one end of the court.
2) Video segments at the end of an FB/FCA that are followed by a close-up shot.
3) Video segments at the beginning and end of the region between two consecutive FB/FCA's in a wide-angle shot.

TABLE III
RESULTS OF DETECTING SHOTS AT THE BASKET FROM THE FOUR TEST BASKETBALL SEQUENCES

Sequence	Number of frames	Number of shots at the basket	Number of extracted frames	Missed detection	False detection
NCAA96	5462	7	678	1	0
NBA96	5654	4	1096	0	6
Olympic-1	6800	12	1055	0	2
Olympic-2	7052	8	900	0	2

Fig. 9. Sample images of six detected shots at the basket from the four test basketball sequences.

Table III shows the experimental results of detecting shots at the basket from the four basketball sequences by using the proposed algorithm. A shot at the basket is considered to be successfully detected when the identified video segment contains at least one video frame in which the ball is in the air, i.e., the ball has already left the player's hand. As we can see from the table, the algorithm has a high rate of successful detection (30/31 = 97%) and managed to reduce the number of frames that need to be examined from 24 968 to 3729, an 85% reduction. Unfortunately, the false detection rate was rather high (10/30 = 33%), which means that further examination of the selected video would be necessary. Each row of Fig. 9 shows sample images of a detected shot at the basket from the four basketball sequences. The first and last image in each row are the first and last frames of the detected shot at the basket.

V. TEMPORAL SEGMENTATION AND CLASSIFICATION OF VIDEO SHOTS

In this section, we address the auxiliary problem of classifying the basketball video into wide-angle and close-up video shots. We present an algorithm based on estimated camera motion and other data easily extracted from the MPEG format to automatically segment and classify the basketball video into these two types of video shots.

The first step in such an algorithm is to detect scene-change boundaries within the video. To do so without full-frame decomposition, we use a scene-change detection algorithm that operates directly on the MPEG video DC sequence. This is the sequence of reduced size images constructed from the DC coefficients of the DCT transformed image blocks in the MPEG video stream. Such a DC sequence usually retains most of the dominant content of the original video and suppresses small intensity changes between successive images. In addition to reducing the computation time, use of the DC sequence can make the scene change detection more robust to small object and camera motions. The scene-change detection algorithm used is similar to those proposed by Zhang *et al.* [3] and Yeo *et al.* [5]. However, instead of using the intensity histogram of the entire DC image, we divide each DC image into blocks (12 image blocks in this paper) and the intensity histograms of corresponding image blocks of successive frames are compared to each other. This allows us to capture some spatial characteristics of the image intensity. For the basketball video analyzed, this method was able to detect most of the significant scene changes.

TABLE IV
MAPPING OF THE THREE QUANTITATIVE MEASURES TO ORDINAL SCALES

V_{var}	V^o_{var}	N_{intra}	N^o_{intra}	P_{pan}	P^o_{pan}
				[100,∞)	-2
[0,3)	1	[0,5)	1	[80,100)	1
[3,4)	2	[5,10)	2	[60,80)	2
[4,10)	3	[10,20)	3	[20,60)	3
[10,∞)	4	[20,∞)	4	[0,20)	4

A. Classification of Wide-Angle and Close-up Shots

The four sequences listed in Table I were segmented into video shots by our scene-change detector, and the video shots were *manually* examined and labeled as wide-angle and close-up shots. These manual classifications are used as ground truth for testing the automatic classification algorithm.

Our classification algorithm is based on the following three quantities:

1) variation of the estimated camera motion;
2) number of intracoded macroblocks;
3) persistence of estimated camera motion.

We measure the variation of the estimated camera motion by the mean absolute difference of camera pan rate between successive P-frames, i.e.,

$$V_{\text{pan}} = \frac{1}{K-1} \sum_{i=1}^{K-1} |V_P(i+1) - V_P(i)| \tag{8}$$

where K is the total number of P-frames within the video shot and $V_p(i)$ is the estimated camera pan rate from the ith P-frame. We expect that the variation of estimated camera motion (V_{pan}) of a wide-angle shot is much smaller than that of a close-up shot.

Let N_{intra} denote the average number of intracoded macroblocks per P-frame of the video shot. We expect N_{intra} to

TABLE V
RESULTS OF CLASSIFYING THE FOUR TEST BASKETBALL SEQUENCES INTO WIDE-ANGLE SHOTS AND CLOSE-UP SHOTS BY USING DIFFERENT THRESHOLDS

Sequence	No. of WA shots	No. of CU shots	*THD*	False positive	False negative
NCAA96	3	5	1	0	1
			3	0	0
			5	0	0
			7	0	0
			9	0	0
NBA96	5	15	1	0	3
			3	0	2
			5	1	0
			7	3	0
			9	8	0
Olympic96-1	7	14	1	0	3
			3	0	2
			5	0	0
			7	1	0
			9	2	0
Olympic96-2	7	16	1	0	3
			3	0	3
			5	1	1
			7	2	0
			9	6	0

be roughly proportional to pan rate for wide-angle video shots but significantly larger for close-up shots.

Each wide-angle shot normally contains one or more events which have motion of significant duration in one consistent direction, e.g., FB's or half-court advances. Therefore, the camera motion estimated from these wide-angle shots should have intervals with this kind of persistent motion. We measure the persistence of the estimated camera motion of a video shot by summing the number of P-frames in the three largest persistent camera pans within the video shot. For instance, if the sequence

$$4(L)\ 2(R)\ 30(L)\ 2(R)\ 2(L)\ 3(R)$$
$$5(L)\ 10(R)\ 20(L)\ 7(R)\ 5(L)$$

represents the number of P-frames in each persistent camera pan in the video shot, i.e., 4 consecutive P-frames panning left followed by two consecutive P-frames panning right, followed by 30 consecutive P-frames panning left, etc., then the three largest estimated camera pans occur in the second panning left (30 P-frames), the fifth panning left (20 P-frames), and the fourth panning right (10 P-frames). Hence

$$P_{\text{pan}} = 30 + 20 + 10 = 60. \tag{9}$$

It is expected that wide-angle video shots, which normally contain the interesting events, such as FB's, will have larger values of P_{pan} than a close-up video shots.

To deal with the potential variations of the measured values between different video sequences, the quantitative values of the measures N_{intra}, V_{pan}, and P_{pan} are mapped into ordinal (or qualitative) scales N^o_{intra}, V^o_{pan}, and P^o_{pan}, as shown in Table IV. The ordinal scales reveal the likelihood of a video shot being a wide-angle shot, i.e., the smaller the values of the ordinal scales, the more likely the video shot is a wide-angle shot. It should be noticed that when $P_{\text{pan}} > 100$, it is mapped to a negative ordinal value (–2) to increase the likelihood of the video shot being a wide-angle shot. This is because when the persistence of the camera pan (P_{pan}) is that large (more than 100 P-frames), then at least one of the persistent camera pans within the video shot has more than $100/3 \approx 33$ P-frames—which is about 4.125 s of persistent camera pan in the sequence. Such a long camera pan must be due to the result of an FB or the ball advancing in a constant direction. Therefore, the video shot is most likely a wide-angle shot captured by the mid-court camera.

Once the ordinal values of the three measures have been obtained, the overall likelihood is computed as their sum, i.e.,

$$\mathcal{R} = V^o_{\text{var}} + N^o_{\text{intra}} + P^o_{\text{pan}}. \tag{10}$$

A video shot is then classified as a wide-angle or close-up shot by using the decision rule

$$\delta(\mathcal{R}) = \begin{cases} \text{wide-angle shot,} & \text{if } \mathcal{R} \le THD \\ \text{close-up shot,} & \text{otherwise} \end{cases} \tag{11}$$

where THD is a suitable threshold which is decided from experimental study.

Define a *false positive* to be a close-up shot that is incorrectly classified as a wide-angle shot and a *false negative* as a wide-angle shot that is incorrectly classified as a close-up shot. Table V shows the number of false positives and false negatives that result from classifying the four different basketball sequences by using the decision rule (11) with different thresholds. As expected, if the value of the threshold is increased, the number of false positives increases (i.e., more close-up shots are incorrectly classified as wide-angle shots) and the number of false negatives decreases (i.e., less wide-angle shots are incorrectly classified as close-up shots). For these particular video sequences, when the value of threshold is between 3–8, the false positives and false negatives can both be kept small. However, we have no confidence that this range will be suitable in general. The value of the threshold may need to be adjusted based on the particular video and the particular MPEG encoder.

As an alternative to setting a fixed threshold, the decision rule can be used in several passes through the compressed video to identify wide-angle video shots in order of increasing likelihood. Each time we scan the data, we gradually increase the threshold THD. When the threshold THD is small, only

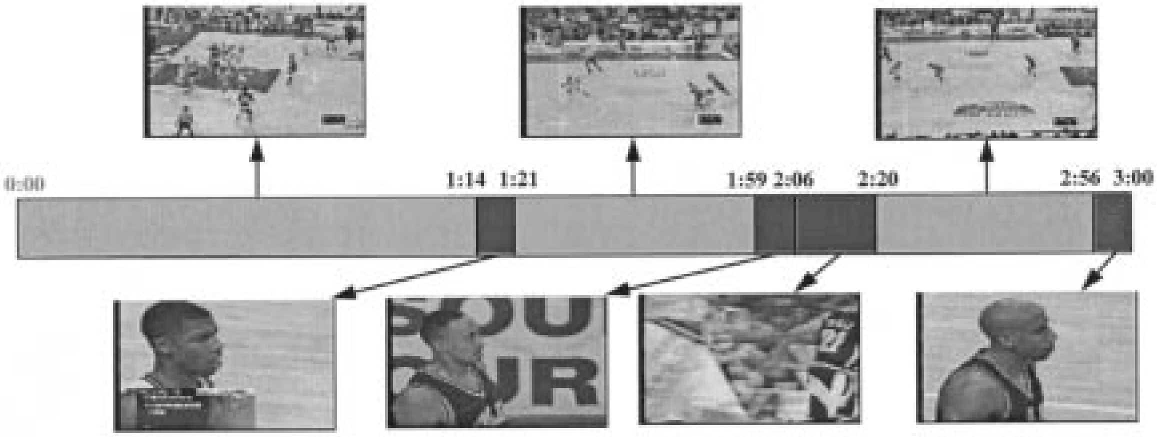


Fig. 10. The result of classifying the NCAA96 basketball sequence into wide-angle shots and close-up shots.

those video shots which are highly likely to be wide-angle will be selected as wide-angle shots. By gradually increasing the threshold THD, more and more shots with "wide-angle like" characteristics will be classified as wide-angle shots. Thus, using this approach, the wide-angle video shots can be identified in order of increasing likelihood.

Fig. 10 shows, in the video display order, the result of classifying the NCAA96 video into wide-angle and close-up shots.

VI. Concluding Remarks

This paper has presented two main results. First, we have introduced two methods of estimating camera motion directly from MPEG video without the need for full-frame decompression. These methods use the MPEG motion vectors from P-frames in an intelligent way as point samples of the corresponding projective transformation between the anchor frame and the P-frame. By solving an associated least-squares problem, these noisy samples can used to estimate the parameters of the projective transformation, and hence, the camera motion.

Second, we have explored the use of this technique in the analysis and annotation of MPEG basketball video. We have demonstrated that estimated camera motion obtained directly from the MPEG format can be used to detect interesting content, such as FB's or FCA's of the ball, shots at the basket, and possessions of the ball. These algorithms can be used to select a small subset of interesting video segments that can then be further decompressed for more detailed analysis.

Using the presented analysis methods, an annotation file of basketball video content can be generated by gathering data such as video shot boundaries, wide-angle and close-up shots, camera motion (pan and zoom), the locations of full court pan, FB's, rebounds, shots at the basket, etc. This annotation file can be used for browsing the basketball video content by jumping easily from one event of interest to the next. The annotation file can also be used to assist the clustering of basketball video sequence by similar events of interest; for example, clustering the video segments in which the team of interest possesses the ball or clustering the video segments in which a shot at the basket may have occurred. This may be useful in assisting, for example, a coach to analyze the defensive or offensive strategy of a team of interest, or a film producer to edit and compose a highlight video sequence with interesting basketball events.

Extending these techniques to other domains of structured video seems to be a promising and interesting direction for further research.

References

[1] D. Anastassiou, "Digital television," *Proc. IEEE*, vol. 82, pp. 510–519, Apr. 1994.

[2] P. Aigrain and P. Joly, "The automatic and real-time analysis of film editing and transition effects and its applications," *Comput. and Graphics*, vol. 18, no. 1, pp. 93–103, 1994.

[3] H. J. Zhang, C. Y. Low, and S. W. Smoliar, "Video parsing and browsing using compressed data," *Multimedia Tools and Applicat.*, vol. 1, no. 1, pp. 89–111, Mar. 1995.

[4] J. Meng, Y. Juan, and S.-F. Chang, "Scene change detection in a MPEG compressed video sequence," in *Proc. IS&T/SPIE Symp.*, vol. SPIE-2419, 1995, pp. 14–25.

[5] B.-L. Yeo and B. Liu, "Rapid scene analysis on compressed videos," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 5, pp. 533–544, Dec. 1995.

[6] F. Arman, R. Depommier, A. Hsu, and M.-Y. Chiu, "Content-based browsing of video sequences," in *ACM Int. Conf. Multimedia*, Oct. 1994, pp. 97–103.

[7] D. Zhong, H.-J. Zhang, and S.-F. Chang, "Clustering methods for video browsing and annotation," *Storage and Retrieval for Still Image and Video Databases IV*, vol. SPIE-2670, pp. 239–246, Feb. 1996.

[8] M. M. Yeung and B.-L. Yeo, "Time-constrained clustering for segmentation of video into story units," *Proc. Int. Conf. Pattern Recognition*, vol. C, pp. 375–380, Aug. 1996.

[9] P. Aigrain, P. Joly, and V. Longueville, "Medium knowledge-based macro-segmentation of video into sequences," *Intell. Multimedia Inform. Retrieval*, 1996.

[10] H. Aoki, S. Shimotsuji, and O. Hori, "A shot classification method of selecting effective key-frames for video browsing," *ACM Multimedia*, pp. 1–10, 1996.

[11] M. M. Yeung and B.-L. Yeo, "Video content characterization and compaction for digital library applications," *Storage and Retrieval for Still Image and Video Databases IV*, vol. SPIE-3022, pp. 45–58, Feb. 1997.

[12] H.-J. Zhang, S. Y. Tan, S. W. Smoliar, and G. Yihong, "Automatic parsing and indexing of news video," *Multimedia Syst.*, vol. 2, pp. 256–266, 1995.

[13] Y. Gong, L. T. Sin, C. H. Chuan, H. Zhang, and M. Sakauchi, "Automatic parsing of TV soccer programs," in *Proc. Int. Conf. Multimedia Computing and Systems*, May 1995, pp. 167–174.

[14] Y.-L. Chang, W. Zeng, I. Kamel, and R. Alonso, "Integrated image and speech analysis for content-based video indexing," in *Proc. IEEE Multimedia*, 1996, pp. 306–313.

[15] S.-F. Chang and D. G. Messerschmitt, "Manipulation and composition of MC-DCT compressed video," *IEEE J. Select. Areas Commun.*, vol. 13, pp. 1–11, Jan. 1995.

[16] *Information technology—Coding of moving pictures and associated audio for digital storage media at up to 1.5 Mbit/s*, ISO/IEC 11 172-1, 1993.

[17] V. Bhaskaran and K. Konstantinides, *Image and Video Compression Standards—Algorithms and Architectures*. Norwell, MA: Kluwer, 1995.

[18] Y.-P. Tan, S. R. Kulkarni, and P. J. Ramadge, "A new method for camera motion parameter estimation," in *Proc. 1995 IEEE Int. Conf. Image Processing*, vol. 1, Oct. 1995, pp. 406–409.

[19] D. D. Saur, Y.-P. Tan, S. R. Kulkarni, and P. J. Ramadge, "Automated analysis and annotation of basketball video," *Storage and Retrieval for Still Image and Video Databases IV*, vol. SPIE-3022, pp. 176–187, Feb. 1997.

Yap-Peng Tan was born in Johor, Malaysia, in 1969. He received the B.S. degree in electrical engineering from National Taiwan University, Taiwan, ROC., in 1993, and the M.A. and Ph.D. degrees in electrical engineering form Princeton University, Princeton, NJ, in 1995 and 1997, respectively.

In 1997, he joined the Digital Imaging and Video Division of Intel Corporation, Chandler, AZ, as a Senior Systems Architect. He was also the Intel Representative to the International Color Consortium (ICC). Since January 2000, he has been a faculty member at the Nanyang Technological University of Singapore. His current research interests include signal and image processing, video analysis and manipulation, computer vision, and color imaging.

Dr. Tan is a Member of SPIE, the Inter-Society Color Council (ISCC), and Phi-Tau-Phi Scholastic Honor Society. He was the recipient of an IBM Graduate Fellowship from IBM T. J. Watson Research Center, Yorktown Heights, NY, from 1995 to 1997.

Drew D. Saur received the B.S.E. degree in electrical engineering from Princeton University, Princeton, NJ, in 1996.

He joined Andersen Consulting in 1996 and is currently a Consultant, specializing in performance and benchmark analysis of client/server and web-based e-commerce applications.

Mr. Saur is a Member of Sigma Xi Research Society, Tau Beta Phi Honor Society, and Phi Beta Kappa Scholastic Honor Society.

Sanjeev R. Kulkarni (M'91–SM'96) received the B.S. degree in mathematics, the B.S. degree in electrical engineering, and the M.S. degree in mathematics, all from Clarkson University, Potsdam, NY, in 1983, 1984, and 1985, respectively. He also received the M.S. degree from Stanford University, Stanford, CA, in 1985, and the Ph.D. degree from M.I.T., Cambridge, MA, in 1991, both in electrical engineering.

From 1985 to 1991, he was a Member of the Technical Staff at M.I.T. Lincoln Laboratory, working on modeling and processing of laser radar measurements. In the spring of 1986, he was a part-time Faculty Member at the University of Massachusetts, Boston. Since 1991, he has been with Princeton University, Princeton, NJ, where he is currently Associate Professor of Electrical Engineering. He spent January 1996 as a Research Fellow at the Australian National University, and in 1998 was with Susquehanna Investment Group. His research interests include statistical pattern recognition, nonparametric estimation, adaptive systems, information theory, and image/video processing.

Dr. Kulkarni served as an Associate Editor for the IEEE TRANSACTIONS ON INFORMATION THEORY. He received an ARO Young Investigator Award in 1992, a National Science Foundation Young Investigator Award in 1994, and several teaching awards at Princeton University.

Peter J. Ramadge (S'79–M'82–SM'92–F'96) was born in Sydney, Australia, in 1954. He received the B.Sc., B.E. (electr.), and M.E. degrees from the University of Newcastle, Australia, and the Ph.D. degree from the Department of Electrical Engineering, University of Toronto, Canada.

He joined the faculty of Princeton University in September 1984, where he is currently a Professor of electrical engineering. He has been a visiting Professor at M.I.T. and a Visiting Research Scientist at IBM's Tokyo Research Laboratory. His research interests are in video and image processing, digital libraries, multimedia, signal processing, and computer-based control.

Dr. Ramadge served as an Associate Editor for the journals *Systems and Control Letters, Mathematics of Control, Signals and Systems, Discrete Event Dynamic Systems: Theory and Applications,* and the IEEE TRANSACTIONS ON AUTOMATIC CONTROL. He was Program Chair of the 32nd IEEE Conference on Decision and Control in San Antonio, TX and has served on the board of governors of the IEEE Control Systems Society. He is a Member of SIAM.