A Real-Time Video Tracking System

ALTON L. GILBERT, MEMBER, IEEE, MICHAEL K. GILES, GERALD M. FLACHS, MEMBER, IEEE, ROBERT B. ROGERS, MEMBER, IEEE, AND YEE HSUN U, MEMBER, IEEE

Abstract—**Object identification and tracking applications of pattern recognition at video rates is a problem of wide interest, with previous attempts limited to very simple threshold or correlation (restricted window) methods. New high-speed algorithms together with fast digital hardware have produced a system for missile and aircraft identification and tracking that possesses a degree of "intelligence" not previously implemented in a real-time tracking system. Adaptive statistical clustering and projection-based classification algorithms are applied in real time to identify and track objects that change in appearance through complex and nonstationary background/foreground situations. Fast estimation and prediction algorithms combine linear and quadratic estimators to provide speed and sensitivity. Weights are determined to provide a measure of confidence in the data and resulting decisions. Strategies based on maximizing the probability of maintaining track are developed. This paper emphasizes the theoretical aspects of the system and discusses the techniques used to achieve real-time implementation.**

Index Terms—**Image processing, intensity histograms, object identification, optical tracking, projections, tracking system, video data compression, video processing, video tracking.**

INTRODUCTION

IMAGE PROCESSING methods constrained to operate on sequential images at a high repetition rate are few. Pattern recognition techniques are generally quite complex, requiring a great deal of computation to yield an acceptable classification. Many problems exist, however, where such a time-consuming technique is unacceptable. Reasonably complex operations can be performed on wide-band data in real time, yielding solutions to difficult problems in object identification and tracking.

The requirement to replace film as a recording medium to obtain a real-time location of an object in the field-of-view (FOV) of a long focal length theodolite gave rise to the development of the real-time videotheodolite (RTV). U.S. Army White Sands Missile Range began the development of the RTV in 1974, and the system is being deployed at this time. Design philosophy called for a system capable of discriminatory judgment in identifying the object to be tracked with 60 independent observations/s, capable of locating the center of mass of the object projection on the image plane within about 2 percent of the FOV in rapidly changing background/foreground situations (therefore adaptive), able to generate a predicted observation angle for the next observation, and required to output the angular displacements of the object within the FOV within 20 ms after the observation was made. The system would be required to acquire objects entering the FOV that had been prespecified by shape description. In the RTV these requirements have been met, resulting in a real-time application of pattern recognition/image processing technology.

The RTV is made up of many subsystems, some of which are generally not of interest to the intended audience of this paper. These subsystems (see Fig. 1) are as follows:

1) main optics;
2) optical mount;
3) interface optics and imaging subsystem;
4) control processor;
5) tracker processor;
6) projection processor;
7) video processor;
8) input/output (I/O) processor;
9) test subsystem;
10) archival storage subsystem;
11) communications interface.

The *main optics* is a high quality cinetheodolite used for obtaining extremely accurate (rms error $\simeq 3$ arc-seconds) angular data on the position of an object in the FOV. It is positioned by the *optical mount* which responds to azimuthal and elevation drive commands, either manually or from an external source. The *interface optics and imaging subsystem* provides a capability to increase or decrease the imaged object size on the face of the silicon target vidicon through a 10:1 range, provides electronic rotation to establish a desired object orientation, performs an autofocus function, and uses a gated image intensifier to amplify the image and "freeze" the motion in the FOV. The camera output is statistically decomposed into background, foreground, target, and plume regions by the *video processor*, with this operation carried on at video rates for up to the full frame. The *projection processor* then analyzes the structure of the target regions to verify that the object selected as "target" meets the stored (adaptive) description of the object being tracked. The *tracker processor* determines a position in the FOV and a measured orientation of the target, and decides what level of confidence it has in the data and decision. The *control processor* then generates commands to orient the mount, control the interface optics, and provide real-time data output. An *I/O pro-*

Manuscript received September 14, 1978; revised November 19, 1978. This work was supported by the U.S. Army ILIR Program and the U.S. Army Research Office.

A. L. Gilbert and M. K. Giles are with the U.S. Army White Sands Missile Range, White Sands, NM 88002.

G. M. Flachs and R. B. Rogers are with the Department of Electrical Engineering, New Mexico State University, Las Cruces, NM 88003.

Y. H. U was with the Department of Electrical Engineering, New Mexico State University, Las Cruces, NM 88003. He is now with Texas Instruments Incorporated, Dallas, TX 75222.

0162-8828/80/0100-0047$00.75 © 1980 IEEE

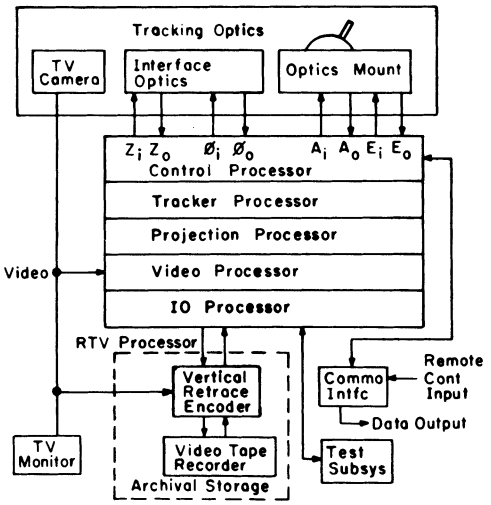


Fig. 1. RTV tracking system.

cessor allows the algorithms in the system to be changed, interfaces with a human operator for tests and operation, and provides data to and accepts data from the *archival storage subsystem* where the live video is combined with status and position data on a video tape. The *test subsystem* performs standard maintenance checks on the system. The *communications interface* provides the necessary interaction with the external world for outputing or receiving data.

The *video processor*, *projection processor*, *tracker processor*, and *control processor* are four microprogrammable bit-slice microprocessors [1], which utilize Texas Instruments' (TIs') new 74S481 Schottky processor, and are used to perform the real-time tracking function.

The four tracking processors, in turn, separate the target image from the background, locate and describe the target image shape, establish an intelligent tracking strategy, and generate the camera pointing signals to form a fully automatic tracking system.

Various reports and papers discuss several of the developmental steps and historical aspects of this project [2]-[7]. In this paper the video, projection, tracker, and control processors are discussed at some length.

Video Processor

The video processor receives the digitized video, statistically analyzes the target and background intensity distributions, and decides whether a given pixel is background or target [8]. A real-time adaptive statistical clustering algorithm is used to separate the target image from the background scene at standard video rates. The scene in the FOV of the TV camera is digitized to form an $n \times m$ matrix representation

$$P = (p_{ij})\, n, m$$

of the pixel intensities p_{ij}. As the TV camera scans the scene, the video signal is digitized at m equally spaced points across each horizontal scan. During each video field, there are n horizontal scans which generate an $n \times m$ discrete matrix representation at 60 fields/s. A resolution of $m = 512$ pixels per standard TV line results in a pixel rate of 96 ns per pixel. Every 96 ns, a pixel intensity is digitized and quantized into eight bits (256 gray levels), counted into one of six 256-level histogram memories, and then converted by a decision memory to a 2-bit code indicating its classification (target, plume, or background). There are many features that can be functionally derived from relationships between pixels, e.g., texture, edge, and linearity measures. Throughout the following discussion of the clustering algorithm, pixel intensity is used to describe the pixel features chosen.

The basic assumption of the clustering algorithm is that the target image has some video intensities not contained in the immediate background. A tracking window is placed about the target image, as shown in Fig. 2, to sample the background intensities immediately adjacent to the target image. The background sample should be taken relatively close to the target image, and it must be of sufficient size to accurately characterize the background intensity distribution in the vicinity of the target. The tracking window also serves as a spatial bandpass filter by restricting the target search region to the immediate vicinity of the target. Although one tracking window is satisfactory for tracking missile targets with plumes, two windows are used to provide additional reliability and flexibility for independently tracking a target and plume, or two targets. Having two independent windows allows each to be optimally configured and provides reliable tracking when either window can track.

The tracking window frame is partitioned into a background region (BR) and a plume region (PR). The region inside the frame is called the target region (TR) as shown in Fig. 2. During each field, the feature histograms are accumulated for the three regions of each tracking window.

The feature histogram of a region R is an integer-value, integer argument function $h^R(x)$. The domain of $h^R(x)$ is $[0, d]$, where d corresponds to the dynamic range of the analog-to-digital converter, and the range of $h^R(x)$ is $[0, r]$, where r is the number of pixels contained in the region R; thus, there are $r + 1$ possible values of $h^R(x)$. Since the domain $h^R(x)$ is a subset of the integers, it is convenient to define $h^R(x)$ as a one-dimensional array of integers

$$h(0), h(1), h(2), \cdots, h(d).$$

Letting x_i denote the ith element in the domain of x (e.g., $x_{25} = 24$), and $x(j)$ denote the jth sample in the region R (taken in any order), $h^R(x)$ may be generated by the sum

$$h^R(x_i) = \sum_{j=1}^{r} \delta x_i, x(j)$$

where δ is the Kronecker delta function

$$\delta_{i,j} = \begin{cases} 0 & i \neq j \\ 1 & i = j. \end{cases}$$

A more straightforward definition which corresponds to the actual method used to obtain $h^R(x)$ uses Iverson's notation [21] to express $h^R(x)$ as a one-dimensional vector of $d + 1$ integers which are set to zero prior to processing the region R as

$$h \leftarrow (d + 1)\, \rho 0.$$

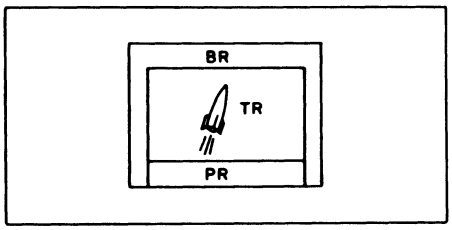


Fig. 2. Tracking window.

As each pixel in the region is processed, one (and only one) element of H is incremented as

$$h[x(j)] \leftarrow h[x(j)] + 1.$$

When the entire region has been scanned, h contains the distributions of pixels over intensity and is referred to as the feature histogram of the region R.

It follows from the above definition that h satisfies the identity

$$r = \sum_{i=0}^{d} h^R(x_i) \quad \text{or} \quad r = +/h.$$

Since h is also nonnegative and finite, it can be made to satisfy the requirements of a probability assignment function by the normalization

$$h \leftarrow h \div +/h.$$

Hereafter, all feature histograms are assumed to be normalized and are used as relative-frequency estimates of the probability of occurrence of the pixel values x in the region over which the histogram is defined.

For the ith field, these feature histograms are accumulated for the background, plume, and target regions and written

$$h_i^{BR}(x): \sum_x h_i^{BR}(x) = 1$$

$$h_i^{PR}(x): \sum_x h_i^{PR}(x) = 1$$

$$h_i^{TR}(x): \sum_x h_i^{TR}(x) = 1$$

after they are normalized to the probability interval $[0, 1]$. These normalized histograms provide an estimate of the probability of feature x occurring in the background, plume, and target regions on a field-by-field basis. The histograms are accumulated at video rates using high-speed LSI memories to realize a multiplexed array of counters, one for each feature x.

The next problem in the formulation of a real-time clustering algorithm is to utilize the sampled histograms on a field-by-field basis to obtain learned estimates of the probability density functions for background, plume, and target points. Knowing the relative sizes of the background in PR, the background in TR, and the plume in TR, allows the computation of estimates for the probability density function for background, plume, and target features. This gives rise to a type of nonparametric classification similar to mode estimation as discussed by Andrews [9], but with an implementation method that allows for real-time realization.

Letting

$$\alpha = \frac{\text{number of background points in PR}}{\text{total number of points in PR}}$$

$$\beta = \frac{\text{number of background points in TR}}{\text{total number of points in TR}}$$

$$\gamma = \frac{\text{number of plume points in TR}}{\text{total number of points in TR}}$$

and assuming that 1) the BR contains only background points, 2) the PR contains background and plume points, and 3) the TR contains background, plume, and target points, one has

$$h_i^{BR}(x) = h_i^B(x)$$

$$h_i^{PR}(x) = \alpha h_i^B(x) + (1 - \alpha) h_i^P(x)$$

$$h_i^{TR}(x) = \beta h_i^B(x) + \gamma h_i^P(x) + (1 - \beta - \gamma) h_i^T(x).$$

By assuming there are one or more features x where $h_i^B(x)$ is much larger than $h_i^P(x)$, one has

$$\alpha = \frac{h_i^{PR}(x)}{h_i^{BR}(x)} - \frac{\epsilon_p}{h_i^{BR}(x)}$$

where $\epsilon_p = (1 - \alpha) h_i^P(x) \ll h_i^B(x)$. Now for all features x where $h_i^P(x) = 0$, one has the solution $\alpha = h_i^{PR}(x)/h_i^{BR}(x)$. For all features x where $h_i^P(x) > 0$, the inequality $h_i^{PR}(x)/h_i^{BR}(x) > \alpha$ is valid. Consequently, a good estimate for α is given by

$$\alpha = \min_x \{h_i^{PR}(x)/h_i^{BR}(x)\}$$

and this estimate will be exact if there exists one or more features where $h_i^{BR}(x) \neq 0$ and $h_i^P(x) = 0$. Having an estimate of α and $h_i^B(x)$ allows the calculation of $h_i^P(x)$.

In a similar manner, estimates of β and γ are obtained,

$$\beta = \min_x \frac{h_i^{TR}(x)}{h_i^{BR}(x)}$$

$$\gamma = \min_x \frac{h_i^{TR}(x)}{h_i^P(x)}.$$

Having field-by-field estimates of the background, plume, and target density functions ($h_i^B(x)$, $h_i^P(x)$, $h_i^T(x)$), a linear recursive estimator and predictor [10] is utilized to establish learned estimates of the density functions. Letting $H(i|j)$ represent the learned estimate of a density function for the ith field using the sampled density functions $h_i(x)$ up to the jth field, we have the linear estimator

$$H(i|i) = w \cdot H(i|i-1) + (1 - w) h_i(x)$$

and linear predictor

$$H(i+1|i) = 2H(i|i) - H(i-1|i-1).$$

The above equations provide a linear recursive method for compiling learned density functions. The weighting factor can be used to vary the learning rate. When $w = 0$, the learning effect is disabled and the measured histograms are used by the predictor. As w increases toward one, the learning effect increases and the measured density functions have a

reduced effect. A small w should be used when the background is rapidly changing; however, when the background is relatively stationary, w can be increased to obtain a more stable estimate of the density functions.

The predictor provides several important features for the tracking problem. First, the predictor provides a better estimate of the density functions in a rapidly changing scene which may be caused by background change or sunglare problems. Secondly, the predictor allows the camera to have an automatic gain control to improve the target separation from the background.

With the learned density functions for the background, plume, and target features $(H_i^B(x), H_i^P(x), H_i^T(x))$, a Bayesian classifier [11] can be used to decide whether a given feature x is a background, plume, or target point. Assuming equal *a priori* probabilities and equal misclassification costs, the classification rule decides that a given pixel feature x is a background pixel if

$$H_i^B(x) > H_i^T(x) \quad \text{and} \quad H_i^B(x) > H_i^P(x),$$

a target pixel if

$$H_i^T(x) > H_i^B(x) \quad \text{and} \quad H_i^T(x) > H_i^P(x),$$

or a plume pixel if

$$H_i^P(x) > H_i^B(x) \quad \text{and} \quad H_i^P(x) > H_i^T(x).$$

The results of this decision rule are stored in a high-speed classification memory during the vertical retrace period. With the pixel classification stored in the classification memory, the real-time pixel classification is performed by simply letting the pixel intensity address the classification memory location containing the desired classification. This process can be performed at a very rapid rate with high-speed bipolar memories.

Projection Processor

The video processor described above separates the target image from the background and generates a binary picture, where target presence is represented by a "1" and target absence by a "0." The target location, orientation, and structure are characterized by the pattern of 1 entries in the binary picture matrix, and the target activity is characterized by a sequence of picture matrices. In the projection processor, these matrices are analyzed field-by-field at 60 fields/s using projection-based classification algorithms to extract the structural and activity parameters needed to identify and track the target.

The targets are structurally described and located by using the theory of projections. A projection in the x-y plane of a picture function $f(x, y)$ along a certain direction w onto a straight line z perpendicular to w is defined by

$$P_w(z) = \int f(x, y)\, dw$$

as shown in Fig. 3. In general, a projection integrates the intensity levels of a picture along parallel lines through the pattern, generating a function called the projection. For binary

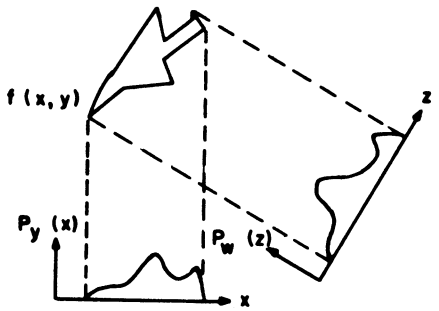


Fig. 3. Projections.

digitized patterns, the projection gives the number of object points along parallel lines; hence, it is a distribution of the target points for a given view angle.

It has been shown that for sufficiently large numbers of projections a multigray level digitized pattern can be uniquely reconstructed [12]. This means that structural features of a pattern are contained in the projections. The binary input simplifies the construction of projections and eliminates interference of structural information by intensity variation within the target pattern; consequently, fewer projections are required to extract the structural information. In fact, any convex, symmetric binary pattern can be reconstructed by only two orthogonal projections, proving that the projections do contain structural information.

Much research in the projection area has been devoted to the reconstruction of binary and multigray level pictures from a set of projections, each with a different view angle. In the real-time tracking problem, the horizontal and vertical projections can be rapidly generated with specialized hardware circuits that can be operated at high frame rates. Although the vertical and horizontal projections characterize the target structure and locate the centroid of the target image, they do not provide sufficient information to precisely determine the orientation of the target. Consequently, the target is dissected into two equal areas and two orthogonal projections are generated for each area.

To precisely determine the target position and orientation, the target center-of-area points are computed for the top section (X_c^T, Y_c^T) and bottom section (X_c^B, Y_c^B) of the tracking parallelogram using the projections. Having these points, the target center-of-area (X_c, Y_c) and its orientation can be easily computed (Fig. 4):

$$X_c = \frac{X_c^T + X_c^B}{2}$$

$$Y_c = \frac{Y_c^T + Y_c^B}{2}$$

$$\phi = \tan^{-1} \frac{Y_c^T - Y_c^B}{X_c^T - X_c^B}.$$

The top and bottom target center-of-area points are used, rather than the target nose and tail points, since they are much easier to locate, and more importantly, they are less sensitive to noise perturbations.

It is necessary to transform the projection functions into

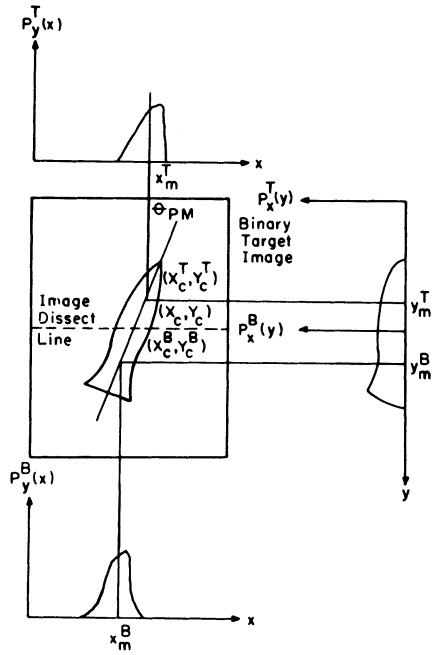


Fig. 4. Projection location technique.

a parametric model for structural analysis. Area quantization offers the advantage of easy implementation and high immunity to noise. This process transforms a projection function $P_w(z)$ into k rectangles of equal area (Fig. 5), such that

$$\int_{Z_i}^{Z_{i+1}} P_w(z)\,dz = \frac{1}{k}\int_{Z_1}^{Z_{k+1}} P_w(z)\,dz$$

$$\text{for } i = 1, 2, \cdots, k.$$

Another important feature of the area quantization model for a projection function of an object is that the ratio of line segments $l_i = Z_{i+1} - Z_i$ and $L = Z_k - Z_2$,

$$S_i = \frac{l_i}{L} \quad \text{for } i = 2, 3, \cdots, k-1$$

are object size invariant. Consequently, these parameters provide a measure of structure of the object which is independent of size and location [13]. In general, these parameters change continuously since the projections are one-dimensional representations of a moving object. Some of the related problems of these geometrical operations are discussed by Johnston and Rosenfeld [14].

The structural parameter model has been implemented and successfully used to recognize a class of basic patterns in a noisy environment. The pattern class includes triangles, crosses, circles, and rectangles with different rotation angles. These patterns are chosen because a large class of more complex target shapes can be approximated with them.

The architecture of the projection processor consists of a projection accumulation module (PAM) for accumulating the projections and a microprogrammable processor for computing the structural parameters. The binary target picture enters the PAM as a serial stream in synchronization

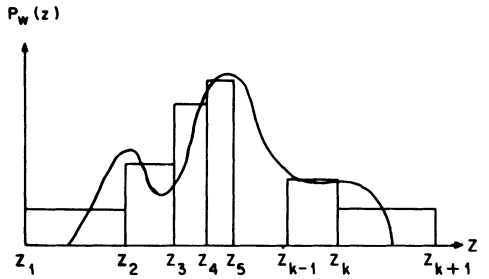


Fig. 5. Projection parameters.

with the pixel classifier of the video processor. The projections are formed by the PAM as the data are received in real time. In the vertical retrace interval, the projection processor assumes addressing control of the PAM and computes the structural parameters before the first active line of the next field. This allows the projections to be accumulated in real time, while the structural parameters are computed during the vertical retrace interval.

Tracker Processor

In the tracking problem, the input environment is restricted to the image in the FOV of the tracking optics. From this information, the tracking processor extracts the important inputs, classifies the current tracking situation, and establishes an appropriate tracking strategy to control the tracking optics for achieving the goals of the tracking system.

The state concept can be used to classify the tracking situations in terms of state variables as in control theory, or it can be interpreted as a state in a finite state automaton [15], [16].

Some of the advantages of the finite state automaton approach are as follows.

1) A finite state automaton can be easily implemented with a look-up table in a fast LSI memory.

2) A finite state automaton significantly reduces the amount of information to be processed.

3) The tracking algorithm can be easily adjusted to different tracking problems by changing the parameters in the look-up table.

4) The finite state automaton can be given many characteristics displayed by human operators.

The purpose of the tracker processor is to establish an intelligent tracking strategy for adverse tracking conditions. These conditions often result in losing the target image within or out of the FOV. When the target image is lost within the FOV, the cause can normally be traced back to rapid changes in the background scene, rapid changes in the target image due to sun glare problems, or cloud formations that obstruct the target image. When the target image is lost by moving out of the camera's FOV, the cause is normally the inability of the tracking optics dynamics to follow a rapid motion of the target image. It is important to recognize these situations and to formulate an intelligent tracking strategy to continue tracking while the target image is lost so that the target image can be reacquired after the disturbance has passed.

To establish an intelligent tracking strategy, the tracker processor evaluates the truthfulness and trackability of the track-

ing data. The truthfulness of the tracking data relates to the confidence that the measured tracking data truly define the location of the target under track. The trackability of the target image relates to the question of whether the target image has desirable tracking properties.

The inputs to the tracker processor are derived from the projection representation of the target image by the projection processor. Area quantization is used to transform each projection function $P(z)$ into $K = 8$ equal area intervals as shown in Fig. 5.

These inputs are:

1) target size (TSZ);
2) target location (TX, TY);
3) target orientation ($T\phi$);
4) target density (TDN);
5) target shape = $\{(SX_i, SY_i) | i = 1, 2, \cdots, 6\}$.

Target size is simply the total number of target points. Target location is given by the center-of-area points of the projections. Where X_i and Y_i are the parameters of Fig. 5 when projected on the x and y axes, respectively,

$$TX = X_5 \quad \text{and} \quad TY = Y_5.$$

The target orientation defines the orientation of the target image with respect to vertical boresight. Target density is derived from the target length (TL), width (TW), and size (TSZ) by

$$TDN = \frac{TL \times TW}{TSZ}.$$

The target shape is described by the ratio of the lengths of the equal area rectangles and the total lengths

$$SX_i = (X_{i+2} - X_{i+1})/(X_8 - X_2)$$

and

$$SY_i = (Y_{i+2} - Y_{i+1})/(Y_8 - Y_2)$$

for $i = 1, 2, \cdots, 6$. Observe that the first and last equal area subintervals are not used in the shape description, since they are quite sensitive to noise.

The tracker processor establishes a confidence weight for its inputs, computes boresight and zoom correction signals, and controls the position and shape of the target tracking window to implement an intelligent tracking strategy. The outputs of the tracker processor are as follows.

Outputs to Control Processor: 1) Target X displacement from boresight (DX), 2) target Y displacement from boresight (DY), 3) desired change in zoom (DZ), 4) desired change in image rotation ($D\phi$), and 5) confidence weight (W).

Outputs to Video Processor: 1) tracking window size, 2) tracking window shape, and 3) tracking window position.

The outputs to the control processor are used to control the target location and size for the next frame. The boresight correction signals are used to control the azimuth and elevation pointing angles of the telescope. The desired zoom is used to control the zoom lens, keeping the target visible within the FOV. The desired image rotation controls the image rotation element to keep the target image vertical. The confidence weight is used by the control processor much like a Kalman weight to combine the measured and predicted values. When the confidence weight is low, the control processor relies more heavily on the recent trajectory to predict the location of the target on the next frame.

The outputs to the video processor define the size, shape, and position of the tracking window. These are computed on the basis of the size and shape of the target image and the amount of jitter in the target image location. There is no loss in resolution when the tracking window is made larger; however, the tracking window acts like a bandpass filter and rejects unwanted noise outside the tracking window.

A confidence weight is computed from the structural features of the target image to measure the truthfulness of the input data. The basic objective of the confidence weight is to recognize false data caused by rapid changes in the background scene or cloud formations. When these situations are detected, the confidence weight is reduced and the control processor relies more heavily on the previous tracking data to orientate the tracking optics toward the target image. This allows the control processor to continue tracking the target so that the target image can be reacquired after the perturbation passes.

The confidence weight measures how well the structural features of the located object fit the target image being tracked. A linear recursive filter is used to continually update the structural features to allow the algorithm to track the desired target through different spatial perspectives. Experimental studies have indicated that the structural parameters $S = \{(SX_i, SY_i) | i = 1, 2, \cdots, 6\}$ and the target density are important features in detecting erratic data. Let $TDN(k)$ and $(SX_i(k), SY_i(k))$ for $i = 1, 2, \cdots, 6$ represent the measured target density and shape parameters, respectively, for the kth field, and let $(\overline{SX}_i(k), \overline{SY}_i(k))$ represent the filtered values for the target shape parameters. The linear filter is defined by

$$\overline{SX}_i(k+1) = \frac{(k-1)\,\overline{SX}_i(k) + SX_i(k)}{K}$$

$$\overline{SY}_i(k+1) = \frac{(k-1)\,\overline{SY}_i(k) + SY_i(k)}{K}$$

for $i = 1, 2, \cdots, 6$ and a positive integer K. The confidence weight for the kth field is given by

$$W(k) = \alpha_1 \max\{1 - C(k), 0\} + \alpha_2 \min\{1, TDN(k)\}$$

where

$$C(k) = \sum_{i=1}^{6} SX_i(k) - \overline{SX}_i(k) + \sum_{i=1}^{6} SY_i(k) - \overline{SY}_i(k)$$

and

$$0 \leqslant \alpha_1, \quad \alpha_2 \leqslant 1, \quad \alpha_1 + \alpha_2 = 1.$$

This formulation for the confidence weight has been experimentally tested, and it has demonstrated the ability to measure the truthfulness of the tracking data in the tracking environment. The filtered shape parameters are not updated

when the confidence weight falls below a given lower threshold. This prevents the shape parameters from being updated incorrectly during periods when the target image is lost or badly perturbed by the background noise.

To formulate an intelligent tracking strategy, the tracking algorithm needs the ability to respond to a current input based upon the sequence of inputs that lead to the current state (or situation). A finite state sequential machine possesses such a property because each state represents the collection of all input sequences that take the machine from the initial state to the present state (Nerode's tape equivalence [17]). By defining an equivalence relation R on the tape set as

$$XRy \quad \text{if} \quad \delta(s_0, X) = \delta(s_0, y) \quad \forall x,\ y \in \sum *$$

the tape set $\Sigma *$ can be partitioned into equivalent classes

$$[x] = s_i = \{y \mid xRy \ \forall y \in \sum *\}.$$

Consequently, a state represents all input sequences that produce a given tracking situation. This interpretation of input sequences transforms the development of the tracking algorithm into a problem of defining a finite state sequential machine

$$TA = (S, I, Z, \delta, W).$$

The states of the machine $S = \{s_i, s_2, s_3, \cdots, s_n\}$ define the different tracking situations that must be handled by the tracking algorithm. The inputs to the finite state machine are derived from the image parameters that characterize the size, shape, and location of the present target image. The output set Z defines a finite set of responses that the tracking algorithm employs for maintaining track and retaining high resolution data. The next state mapping δ: $S \times I \rightarrow S$ defines the next state $\delta(s_i, i_j) = s_k$ when an input i_j is applied to state s_i. The output mapping W: $S \rightarrow Z$ is a Moore output that defines the proper tracking strategy (response) for each state.

The inputs to the sequential machine are chosen to give the sequential machine a discrete measure of the trackability of the target image. These inputs are:

1) target size (TSZ);
2) confidence weight (W);
3) image orientation ($T\phi$);
4) image displacement from boresight (TX, TY);
5) image movement (ΔTX, ΔTY);
6) rate of change in target size (ΔTSZ);
7) rate of change in confidence weight (ΔW).

The image size is clearly an important measure of the trackability since the image is difficult to track when the image is either too small or too large. The confidence weight provides a measure of whether the image is the target under track. Image displacement from boresight gives a measure of whether the image is leaving the FOV. Image movement in the FOV measures the amount of jitter in the target image. The rate of change in the target size and confidence weight allows a prediction of what is likely to happen on subsequent fields. These inputs are quantized into an 8-bit binary input vector for the sequential machine.

The states of the sequential machine are chosen to define the major tracking situations. These states are:

1) target acquisition = S_1;
2) normal tracking = S_2;
3) abrupt change in FOV = S_3;
4) leaving or out of FOV = S_4.

S_1 corresponds to the situation where the tracker is trying to acquire the target. This situation occurs during initial acquisition and during periods of time when the target image is lost. S_2 corresponds to the situation where the target image is under track with the normal tracking algorithm and no special tracking strategy is required. S_3 corresponds to the situation where the target image undergoes erratic and abrupt changes in its shape or size within the FOV and S_4 corresponds to the situation where the target image is leaving or has left the FOV of the camera. These states only categorize the major tracking situations. State parameters are used to further refine the tracking situation by providing a parametric description of the tracking situation.

The tracking algorithm views the target structure and activity within the tracking window on a field-by-field basis. Based upon these observations, the tracking algorithm must classify the present tracking situation and enter the appropriate state whose outputs best handle the situation. The motivation of the next state mapping, δ: $I \times S \rightarrow S$, is to provide a tracking strategy to keep the tracker in the normal tracking state. Much experimental effort has been devoted to establish a next state mapping with desirable tracking characteristics. A simplified state diagram is given in Fig. 6 which defines the essential properties of the state transition behavior.

The output set Z defines a finite set of responses that the tracking algorithm can make to maintain track while retaining high resolution data. Several of these responses control the tracking window, and they are performed at electronic speed. However, the zoom lens and tracking optics operations are mechanical in nature and require more time to be performed.

The output set Z contains the following responses:

1) location of tracking window;
2) shape and size of tracking window;
3) target location (boresight corrections);
4) confidence weight;
5) desired zoom;
6) control shape parameter update procedure.

These output responses give the tracking algorithm the ability to respond to the tracking situations defined by tracking states.

The motivation for the output mapping, W: $I \times S \rightarrow Z$, is to associate with each state an intelligent strategy to maintain track with high image resolution. The outputs to the image decomposition algorithm control the size and position of the tracking window. The outputs to the control processor control the zoom and pointing angle of the tracking optics. The tracking algorithm must make many compromises between

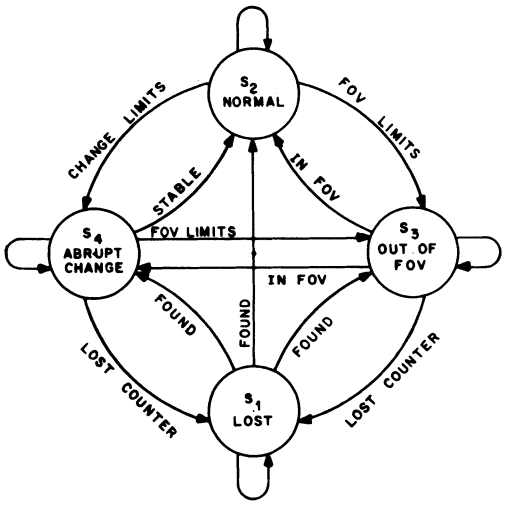


Fig. 6. Next-state mapping.

tracking ability and image resolution. For example, when the target image is about to leave the FOV, the zoom can be reduced to keep the target in the FOV. Increasing the FOV, however, decreases the target size and image resolution. These compromises have been formulated and solved with standard optimization techniques. The solutions are precomputed and stored in a look-up table for the tracking algorithm.

By implementing the tracking algorithm as a finite state machine where the outputs are obtained from the look-up tables, it is possible to realize the tracking algorithm with a standard microprogrammable processor.

Control Processor

The purpose of the control processor is to generate the control signals that drive the tracking optics. The processor receives inputs from the tracker processor on the target location (TX, TY), orientation $(T\phi)$, and length (TL), measured relative to boresight and on a confidence weight (W) that measures the reliability of the measured data. Using these inputs, the control processor predicts the desired tracker azimuth and elevation pointing angles $(\Theta_A(k+1), \Theta_E(k+1))$, the zoom lens setting $(Z(k+1))$ and the image rotation setting $(\phi(k+1))$ for the next field denoted by $(k+1)$.

The control processor provides the estimated and predicted values of the variables necessary to orientate the tracking optics toward the target. Using the confidence weight to combine the predicted variables with current measurements, the processor provides estimates of current target position variables. These estimates are combined with previous estimates to predict the target position variables for the next control interval.

Since the form of the estimator equations is similar for the azimuth, elevation, zoom, and image rotation variables, the following notation is used in the development:

$\Theta(k)$ variable to be estimated or predicted (azimuth, elevation, rotation, or zoom);
$\Theta_m(k)$ measured values of $\Theta(k)$;
$\hat{\Theta}(k)$ estimated value of $\Theta(k)$ using measurements through kth frame;
$\hat{\Theta}_j(k+1|k)$ predicted value of $\Theta(k+1)$ using measurements through kth frame and the j index type predictor;
$\hat{\Theta}(k+1|k)$ predicted value of $\Theta(k+1)$ using a combination of the set of predictors;
$w(k)$ confidence weight.

Using the confidence weight in a manner similar to that of a Kalman gain, the estimated value is obtained from the measured and predicted values by

$$\hat{\Theta}(k) = (1 - W(k))\,\hat{\Theta}(k|k-1) + W(k)\,\Theta_m(k)$$

where the confidence weight is normalized such that $0 \leqslant W(k) \leqslant 1$. Consequently, the predictors having the functional form

$$\hat{\Theta}(k+1|k) = F(\hat{\Theta}(k), \hat{\Theta}(k-1), \cdots, \hat{\Theta}(k-r))$$

rely more heavily on the estimated values when the weight is low. This is very important for continuing track when the target passes through clouds or noisy backgrounds such as the mountain/sky interface.

A simple filter-predictor that has demonstrated the ability to track highly maneuverable targets uses a linear convex combination of a linear-two-point and a quadratic-five-point polynomial filter-predictor. This scheme is based on the reasoning that the linear-two-point polynomial can better anticipate a rapid maneuver from the missile, while the quadratic-five-point polynomial is used to reduce the variance of error. The coefficients of the linear convex combination are chosen to obtain an unbiased minimum variance of error estimate.

In formulating the linear convex combination estimate of the linear $(\hat{\Theta}_l)$ and quadratic $(\hat{\Theta}_q)$ predictors

$$\hat{\Theta} = \alpha_1 \hat{\Theta}_l + \alpha_2 \hat{\Theta}_q,$$

it is assumed that the predictors $\hat{\Theta}_l$ and $\hat{\Theta}_q$ are independent; hence, the minimum variance error estimate [18] is given by the coefficients

$$\alpha_1 = \frac{\sigma^2(\hat{\Theta}_q)}{[\sigma^2(\hat{\Theta}_l) + \sigma^2(\hat{\Theta}_q)]}$$

$$\alpha_2 = \frac{\sigma^2(\hat{\Theta}_l)}{[\sigma^2(\hat{\Theta}_l) + \sigma^2(\hat{\Theta}_q)]}$$

and

$$\hat{\Theta}(t_{k+1}|t_k) = \frac{\sigma^2(\hat{\Theta}_q)\,\hat{\Theta}_l(t_{k+1}|t_k) + \sigma^2(\hat{\Theta}_l)\,\hat{\Theta}_q(t_{k+1}|t_k)}{\sigma^2(\hat{\Theta}_l) + \sigma^2(\hat{\Theta}_q)}$$

is the desired prediction of the chosen parameter.

Simulation studies [19] were conducted to compare the performance of this simple and somewhat primitive filter-predictor with the more robust extended Kalman filter [20]. The results of these studies indicated the total performance of the tracking system with the simple filter-predictor compared favorably with the extended Kalman filter. In the light of the real-time computational constraints, the simple filter-predictor is suitable for more real-time tracking problems.

Simulation and Hardware

A computer simulation of the RTV tracking system, incorporating the algorithms used by the four processors described in this paper, was developed and used to evaluate the system under realistic tracking conditions. The simulation model includes dynamic models for the target trajectory and the tracking optics and mount in addition to the RTV processor algorithms in order to verify the complete tracking system.

The simulation displays the target and plume points as they are identified by the video processor. These points are superimposed on a cross hair which locates the boresight of the tracking optics. In addition to the digitized images, the simulation displays the target and plume tracking windows and the projections accumulated for each video field. These outputs provide a direct measure of the performance of the four processors as well as an overall measure of the tracking accuracy of the system.

Two types of input data are available to the simulation—simulated digitized video fields and actual digitized video fields from video tape recordings of typical tracking sequences. Fig. 7 contains two representative simulation outputs selected from the first 100 fields of simulated digitized video. The RTV tracker performs well during this simulated tracking sequence. All processors function properly, and track is maintained throughout the tracking sequence.

Fig. 8(a) is a halftone graphics display of a field of actual digitized video which was injected into the RTV simulation. The simulator repeatedly processed the same video field superimposed on the simulated target trajectory in order to test the static and dynamic responses of the RTV processors. The result after six processing frames, shown in Fig. 8(b), verifies the effectiveness of the processing algorithms used in the RTV tracker. Since no plume is present, both windows are tracking the target. Several important conclusions may be deduced from this result: the video processor successfully identifies most of the target pixels; the projection data is accumulated and used effectively by the projection and tracker processors to obtain the target orientation and the displacement of the target from boresight; and both windows are tracking the target and closing down on it to reduce noise.

Subsequent to simulation, a hardware model was developed and is being deployed at the present time. Bench tests of all subsystems indicate that the projected performance of each processor has been achieved.

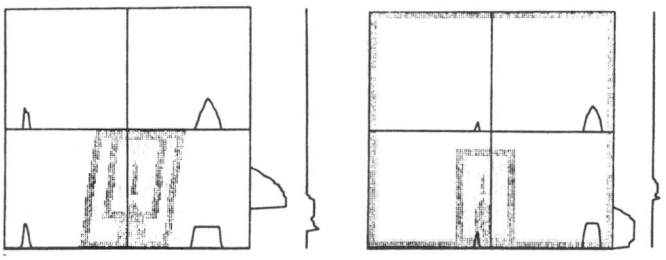
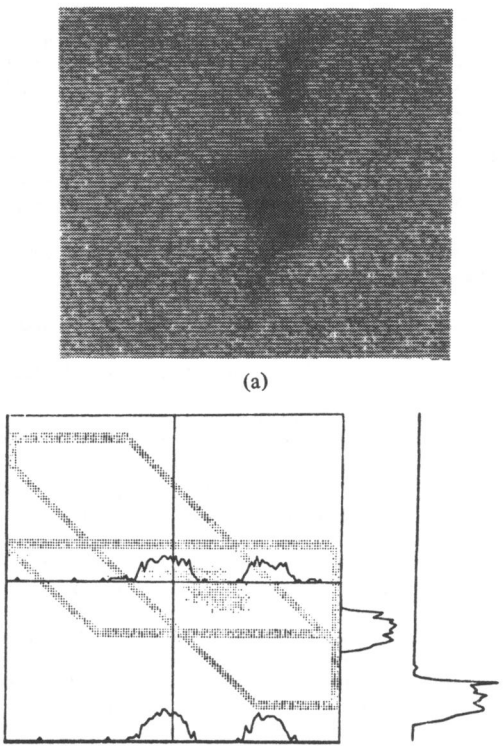


Fig. 7. Simulation outputs.



(a)

(b)

Fig. 8. (a) Digitized video. (b) Simulation output.

Conclusions

These four subsystems form only a part of the RTV; many of the other subsystems are of equal complexity. Changes currently taking place in digital technology coupled with aggressive research in real-time pattern recognition/image processing algorithms are now making possible highly sophisticated hardware for recognition and tracking purposes. While RTV is an example of this type of a system, work at WSMR and NMSU (and elsewhere) is continuing with the purpose of finding faster and better ways of processing video intelligently.

A lot of work remains before a truly versatile video analysis system exists. RTV is but a step in that direction. Research directed toward the solution of this class of problems needs to have two fundamental premises as a starting point. The first is that algorithms should be general, not specifically created for a particular scene. Too often pattern recognition methods have depended upon extensive "tweaking" of parameters to yield good results. This approach is not allowable for real-time processing. Secondly, the algorithms should not require extensive data storage, since manipulation of the data consumes resources that exceed those available for most applications.

As is evidenced by RTV, progress can be made; the future of video processing is bright. The applications in the nonmilitary community possibly exceed the military applications in importance. RTV is a forerunner of a large class of intelligent video processing machines that will perform a wide variety of tasks of importance to the human family.

Acknowledgment

Important contributions to this project were made by a number of persons not named herein, including faculty and students at universities and staff members at WSMR. The project has enjoyed many participants and cosponsors whose contributions are acknowledged.

REFERENCES

[1] P. I. Perez, "A multiprocessor architecture with a master shared control store," Ph.D. dissertation, New Mexico State Univ., 1978 (available from University Microfilms, Ann Arbor, MI).

[2] G. M. Flachs and A. L. Gilbert, "Automatic boresight correction in optical tracking," in *Proc. 1975 Nat. Aerospace and Electronics Conf.*, June 1975.

[3] G. M. Flachs, W. E. Thompson, Y. H. U, and A. L. Gilbert, "A real-time structural tracking algorithm," in *Proc. IEEE 1976 Nat. Aerospace and Electronics Conf.*, May 1976.

[4] G. M. Flachs and J. A. Vilela, "A structural model for polygon patterns," in *Proc. IEEE 1976 Nat. Aerospace and Electronics Conf.*, May 1976.

[5] G. M. Flachs, W. E. Thompson, J. M. Taylor, W. Cannon, R. B. Rogers, and Y. H. U, "An automatic video tracking system," in *Proc. 1977 Nat. Aerospace and Electronics Conf.*, May 1977.

[6] R. B. Rogers and G. M. Flachs, "Mathematical modeling and simulation in a programmed design methodology," in *Proc. 1st Int. Conf. on Mathematical Modeling*, Aug. 1977.

[7] A. L. Gilbert and M. K. Giles, "Novel concepts in real-time optical tracking," in *Proc. Army Sci. Conf.*, June 1978.

[8] N. J. Nilsson, *Learning Machines.* New York: McGraw-Hill, 1965, ch. 4.

[9] H. L. Andrews, *Introduction to Mathematical Techniques in Pattern Recognition.* New York: Wiley-Interscience, 1972, pp. 148–150.

[10] B. P. Lathi, *Random Signals and Communication Theory.* Scranton, PA: International Textbook, 1968, pp. 275–279.

[11] I. Selin, *Detection Theory.* Princeton, NJ: Princeton Univ. Press, 1965, pp. 11–13.

[12] S. K. Chang, "The reconstruction of binary patterns from their projections," *Commun. ACM*, vol. 14, Jan. 1971.

[13] Y. H. U, "Projection theory for real-time vision tracking," Ph.D. dissertation, New Mexico State Univ., Aug. 1978 (available from University Microfilms, Ann Arbor, MI).

[14] E. G. Johnston and A. Rosenfeld, "Geometrical operations on digitized pictures," *Picture Processing and Psychopictorics*, B. S. Lipkin and R. Rosenfeld, Eds. New York: Academic, 1970.

[15] R. Romovic and R. B. McGhee, "A finite static approach to the synthesis of bioengineering control systems," *IEEE Trans. Hum. Factors Electron.*, vol. HFE-7, pp. 65–69, June 1966.

[16] E. S. Angel and G. A. Bekey, "Adaptive finite state models of manual control systems," *IEEE Trans. Man-Mach. Syst.*, pp. 15–20, Mar. 1968.

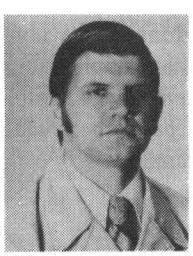
[17] T. L. Booth, *Sequential Machines and Automata Theory.* New York: Wiley, 1968.

[18] A. Papoulis, *Probability Random Variables and Stochastic Processes.* New York: McGraw-Hill, 1965, pp. 385–426.

[19] W. E. Thompson, G. M. Flachs, and T. R. Kiang, "Evaluation of filtering and prediction techniques for real-time video tracking of high performance targets," *Trans. 1978 Nat. Aerospace and Electronics Conf.*, June 1978.

[20] C. B. Chang, R. H. Whitney, and M. Atham, "Application of adaptive filtering methods to maneuvering trajectory estimation," Lincoln Lab. Tech. Note 1975-59, Nov. 1975.

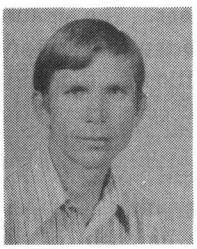
[21] K. E. Iverson, *A Programming Language.* New York: Wiley, 1962.



Alton L. Gilbert (S'68–M'73) was born in Elmira, NY, on April 13, 1942. He received the B.S.E.E., M.S.E.E., and Sc.D. degrees, all from the New Mexico State University, Las Cruces, in 1970, 1971, and 1973, respectively.

From 1961 to 1968 he served with the U.S. Navy in the Polaris missile program. From 1973 to the present he has been a Researcher and Research Manager with the Advanced Technology Office of the Instrumentation Directorate at U.S. Army White Sands Missile Range, NM. As Principle Investigator and Manager of the Real-Time Videotheodolite program, he has become widely known for his interests in video processing methods.

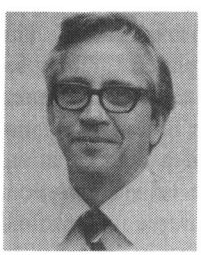
Dr. Gilbert is a member of the Technical Review Committee of the Joint Services Electronics Program, of the Electronics Coordinating Group for the U.S. Army, the Information Theory Group of the IEEE, Tau Beta Pi, Eta Kappa Nu, Phi Kappa Phi, and Sigma Xi.



Michael K. Giles was born in Logan, UT, on October 24, 1945. He received the B.E.S. and M.S.E.E. degrees from Brigham Young University, Provo, UT, in 1971, and the Ph.D. degree from the University of Arizona, Tucson, in 1976.

He was an Electronics Engineer with the U.S. Naval Weapons Center, China Lake, CA, from 1971 to 1977, where he specialized in the design and evaluation of electrooptical and optical systems. At China Lake he developed low-noise photodiode preamplifiers, photoparametric amplifiers, and rapidly tunable dye lasers for Navy applications. Since 1977 he has been with the Instrumentation Directorate, U.S. Army White Sands Missile Range, NM, where he is applying image-processing and pattern-recognition techniques to the development of real-time optical tracking systems. His research interests also include optical and electrooptical image and signal processing.

Dr. Giles is a member of the Optical Society of America, the Society of Photo-Optical Instrumentation Engineers, Tau Beta Pi, and Eta Kappa Nu.



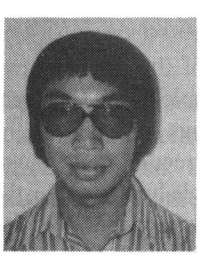
Gerald M. Flachs (S'68–M'68) received the B.S., M.S., and Ph.D. degrees, all from Michigan State University, East Lansing.

He is a Professor of Electrical and Computer Engineering at New Mexico State University, Las Cruces. His current research interests include image-processing techniques, distributive computer systems, and digital system design with microprogrammable processors.

Robert B. Rogers (M'77) was born in China Lake, CA on April 13, 1952. He received the B.S. degree in physics from the New Mexico Institute of Mining and Technology (NMIMT), Socorro, in 1973, and the M.S. and Ph.D. degrees in electrical engineering from New Mexico State University (NMSU), Las Cruces, in 1976 and 1978, respectively.

While at NMIMT, he performed research on cross-spectral analysis of thunder and lightning channel reconstruction. He is now at NMSU, where his primary research area is real-time video filtering with bit-slice microprogrammable processor systems. He is currently with the New Mexico Solar Energy Institute's Modeling and Analysis Section at NMSU.

Dr. Rogers is a member of the American Geophysical Union and the IEEE Computer Society.



Yee Hsun U (S'72–M'79) was born in Hong Kong, China, on May 30, 1949. He received the B.S.E.E. degree from Purdue University, West Lafayette, IN, in 1972, and the M.S.E.E. and the Ph.D. degrees in electrical engineering from New Mexico State University, Las Cruces, in 1974 and 1978, respectively.

From 1976 to 1978, he was a staff engineer with the Real-Time Video Tracker project at New Mexico State University. He joined Texas Instruments Incorporated, Dallas, in September 1978 as a member of technical staff of the Corporate Research, Development & Engineering Division. His research interests are in real-time tracking, vision systems, machine intelligence, and the application of pattern recognition to industrial automation.