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.

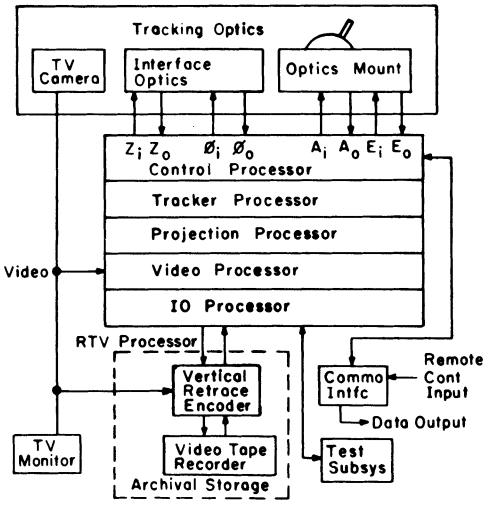

Fig. 1. RTV tracking system.

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. 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

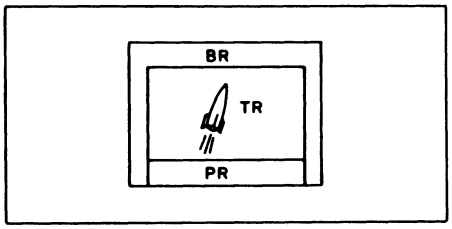

Fig. 2. Tracking window.

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

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

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

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

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

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

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

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

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

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

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

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 $\alpha = h_i^{\mathrm{PR}}(x)/h_i^{\mathrm{BR}}(x)$. For all features x where $h_i^P(x) > 0$, the inequality $h_i^{\mathrm{PR}}(x)/h_i^{\mathrm{BR}}(x) > \alpha$ is valid. Consequently, a good estimate for α is given by

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

and this estimate will be exact if there exists one or more features where $h_i^{\mathrm{BR}}(x) \neq 0$ 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,

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

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

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

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 pat-

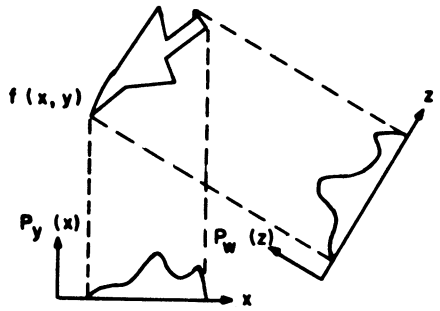

Fig. 3. Projections.

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.

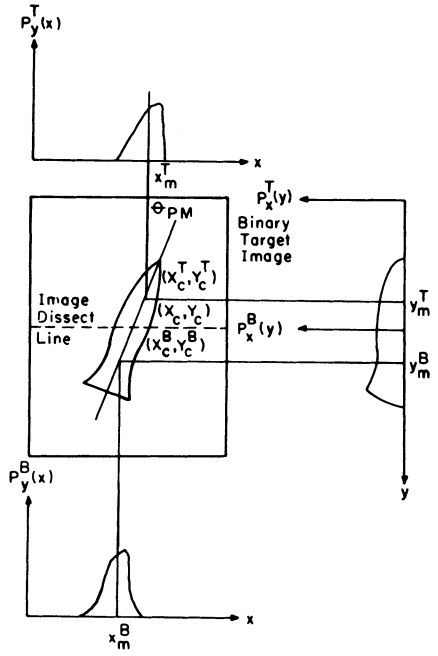

Fig. 4. Projection location technique.

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

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

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 $L = Z_k - Z_2$,

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

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

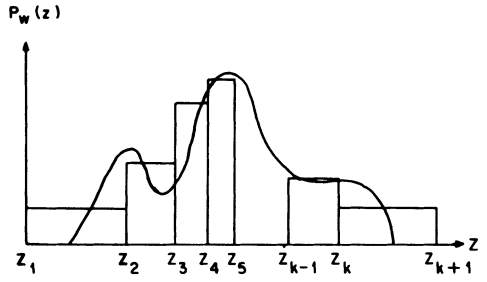

Fig. 5. Projection parameters.

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 pro-

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.