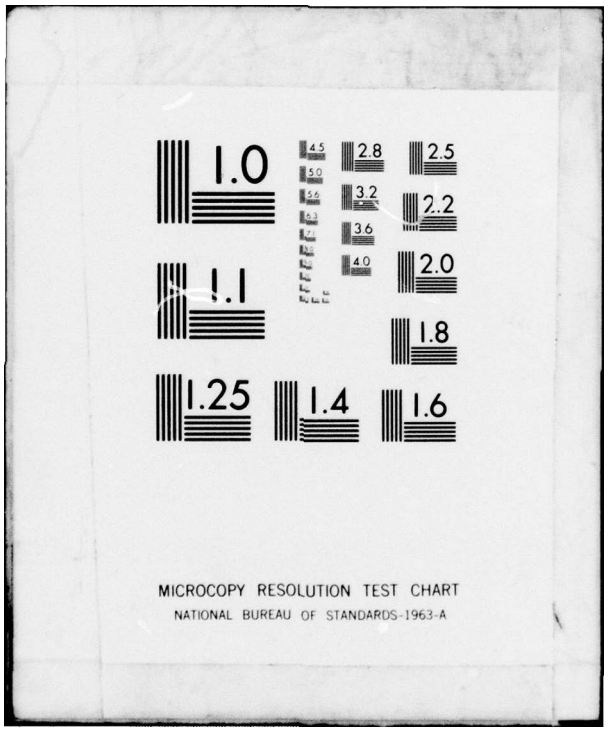
AD-A078 809 VIRGINIA UNIV CHARLOTTESVILLE RESEARCH LABS FOR THE--ETC F/G 17/7
DISCRETE ANALOG PROCESSING FOR TRACKING AND GUIDANCE CONTROL.(U)
AUG 79 E S MCVEY , E A PARRISH , R M INIGO N00014-78-C-0695
UNCLASSIFIED UVA/525350/EE79/102 ONR-CR233-092-1 NL

1 OF 2
AD A078809


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
6.3
7.1
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

REPORT ONR-CR233-092-1

DISCRETE ANALOG PROCESSING FOR TRACKING AND GUIDANCE CONTROL

E. S. McVEY
E. A. PARRISH
R. M. INIGO
R. J. SCHALKOFF

RESEARCH LABORATORIES FOR THE ENGINEERING SCIENCES
UNIVERSITY OF VIRGINIA
CHARLOTTESVILLE
VIRGINIA
22901

CONTRACT N0014-78-C-0695

31 AUGUST 1979

FINAL REPORT FOR PERIOD 1 AUG 78 - 31 AUG 79

PREPARED FOR THE
OFFICE OF NAVAL RESEARCH ● 800 N. QUINCY ST. ● ARLINGTON ● VA ● 22217

SPECIAL NOTICES

Change of Address

Organizations receiving reports on the initial distribution list should confirm correct address. This list is located at the end of the report. Any change of address or distribution should be conveyed to the Office of Naval Research, Code 212, Arlington, VA 22217.

Disposition

When this report is not longer needed, it may be transmitted to othe ıizations. Do not return it to the originator or the monitoring office.

Reproduction

Reproduction in whole or in part is permitted for any purpose of the United States Government.

UNCLASSIFIED (12-12-79)

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

(19) **REPORT DOCUMENTATION PAGE**

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER (18) ONR-CR233-092-1	2. GOVT ACCESSION NO. (9) Final rept. 1 Aug 78-31 Aug 79
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle) (6) DISCRETE ANALOG PROCESSING FOR TRACKING AND GUIDANCE CONTROL	5. TYPE OF REPORT & PERIOD COVERED: Final Report: 8/1/78-8/31/79
	6. PERFORMING ORG. REPORT NUMBER (14) UVA/525350/EE79/102
7. AUTHOR(s) (10) E. S. McVey, E. A. Parrish, R. M. Inigo, R. J. Schalkoff	8. CONTRACT OR GRANT NUMBER(s) (15) N0014-78-C-0695
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Department of Electrical Engineering, School of Engineering and Applied Science, University of Virginia Charlottesville, VA 22901	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: DD Form 1473 "Report Decumentation Page"
11. CONTROLLING OFFICE NAME AND ADDRESS: Office of Naval Research, 800 N. Quincy Street, Arlington, VA 22217	12. REPORT DATE (11) 31 August 1979; 13. NUMBER OF PAGES 162
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): N/A (12) 181	15. SECURITY CLASS. (of this report): Unclassified; 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE: N/A

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

N/A

18. SUPPLEMENTARY NOTES

The findings and conclusions contained in this report are not to be construed as an official Department of Defense or Military Department position unless so designated by other official documents.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

real-time
tracking
algorithm
affine-transformation
model
Taylor series
texture

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The search for and development of algorithms for a real-time, automatic system capable of tracking 2-D targets in complex scenes is discussed. The problems and constraints involved in such an undertaking as well as previous efforts are summarized. No algorithms could be found in the literature that will provide satisfactory results for realistic conditions. A mathematical model of scene spatial and temporal evolution is developed for certain classes of targets and target perturbations. It is shown that for small target →

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

page 1 of 3

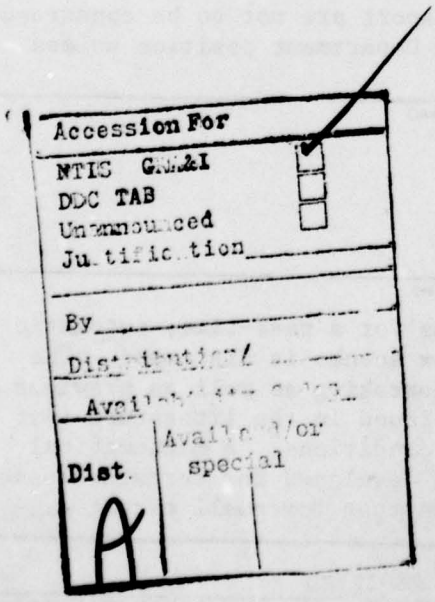
366870

REPORT DOCUMENTATION PAGE

(continued)

20. (continued)

perturbations the 2-D tracking problem may be approximated as a 1-D time-varying parameter estimation problem. A novel and CCD-implementable solution is developed which is superior to previously developed algorithms, particularly when considering target rotation and dilation and target/background separation. Results of system simulation and suggestions for future research are presented.



A Final Report

Contract No. N0014-78-C-0695

DISCRETE ANALOG PROCESSING FOR TRACKING AND GUIDANCE CONTROL

Submitted to:

Office of Naval Research
800 N. Quincy Street
Arlington, Virginia 22217

Submitted by:

E. S. McVey
Professor

E. A. Parrish
Professor

R. M. Inigo
Research Associate

R. J. Schalkoff
Research Associate

Department of Electrical Engineering
RESEARCH LABORATORIES FOR THE ENGINEERING SCIENCES
SCHOOL OF ENGINEERING AND APPLIED SCIENCE
UNIVERSITY OF VIRGINIA
CHARLOTTESVILLE, VIRGINIA

Report No. UVA/525350/EE79/102
August 1979

Copy No. 19

TABLE OF CONTENTS

TABLE OF CONTENTS

(Continued)

TABLE OF CONTENTS

(Continued)

TABLE OF CONTENTS

(Continued)

LIST OF TABLES

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS

(Continued)

LIST OF SYMBOLS AND ABBREVIATIONS

$\underset{\sim}{a}$	6 x 1 or 4 x 1 vector of affine transform parameters (see text)
$\hat{\underset{\sim}{a}}$	TSVIP algorithm estimate of $\underset{\sim}{a}$
$\underset{\sim}{a}_p$	4 x 1 vector of homogeneous affine transform parameters
$\underset{\sim}{a}_c$	2 x 1 vector of constrained affine transform parameters
A	2 x 2 homogeneous affine transform matrix
A_c	2 x 2 matrix of homogeneous affine transform combinations
ΔA	2 x 2 Affine Perturbation Matrix; $\Delta A = A-I$
$\underset{\sim}{b}$	2 x 1 Affine transform translation vector
$\hat{\underset{\sim}{b}}$	TSVIP algorithm estimate of $\underset{\sim}{b}$
$b(\underset{\sim}{x},t)$	3-D process representing scene background
D	N x 6 matrix (see text)
D_c	N x 4 matrix (see text)
$\underset{\sim}{d}(\underset{\sim}{x}_i,t)$	N x 1 scene difference vector
f	Camera lens--sensor plane distance
$f(\underset{\sim}{x},t)$	3-D target textural function
$\underset{\sim}{f}_1, \underset{\sim}{f}_2$	Columns of G
$F(\underset{\sim}{u})$	Multidimensional Fourier transform function
G	N x 2 Matrix of Spatil Partial Derivatives
I	Identity Matrix
i,j,k	Integer indices
$\underset{\sim}{i}$	2 x 1 Index Vector
$n(\underset{\sim}{x},t)$	3-D function (defined in text)
N	Noise vector covariance matrix
p(x,t)	The 3-D scene function
$\underset{\sim}{p}_1, \underset{\sim}{p}_2$	Columns of P_c
$P_s(\underset{\sim}{x},t)$	Segmented, 3-D scene function
$P_n(\underset{\sim}{x},t)$	Noise-corrupted 3-D scene function
P	N x 4 matrix of weighted spatial derivatives

LIST OF SYMBOLS AND ABBREVIATIONS

(Continued)

P_c	N x 2 matrix of weighted spatial derivatives (see text)
Q	Weighting matrix
$s(\underset{\sim}{x},t)$	3-D function (defined in text)
t	Temporal parameter (continuous)
T	Temporal parameter (discrete)
$\underset{\sim}{u}$	N x 1 vector (see text)
$w(\underset{\sim}{x},t)$	3-D window function
$\underset{\sim}{x}$	2 x 1 vector (see text)
X	N x 2 matrix (see text)
z	Camera lens--target distance
α	Magnification (dilation) constant
Δ	Spatial sampling interval
$\lambda(\underset{\sim}{x},t)$	Characteristic function (see text)
$\underset{\sim}{\eta}$	Noise vector or process
θ	Rotational angle
$\underset{\sim}{\theta}$	Camera mount control signal vector
$\dagger$	Denotes matrix pseudoinverse
$\mathcal{F}[\ ]$	Denotes (multidimensional) Fourier transform
$E[\ ]$	Denotes expected value
$\|\|\ \|\|$	Vector norm
$\underset{\sim}{v}$	Denotes vector, e.g. $\underset{\sim}{v} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_N \end{bmatrix}$

ABSTRACT

The search for and development of algorithms for a real-time, automatic system capable of tracking 2-D targets in complex scenes is discussed. The problems and constraints involved in such an undertaking as well as previous efforts are summarized. No algorithms could be found in the literature that will provide satisfactory results for realistic conditions. A mathematical model of scene spatial and temporal evolution is developed for certain classes of targets and target perturbations. It is shown that for small target perturbations the 2-D tracking problem may be approximated as a 1-D time-varying parameter estimation problem. A novel and CCD-implementable solution is developed which is superior to previously developed algorithms, particularly when considering target rotation and dilation and target/background separation. Results of system simulation and suggestions for future research are presented.

CHAPTER I

INTRODUCTION

1.1 Background

As the speed, capability and economic advantages of modern signal processing devices continue to increase, there is simultaneously an increase in efforts aimed at developing sophisticated, real-time automatic systems capable of emulating human (or perhaps more generally, biological) functions. The research described herein concerns the development and implementation of algorithms for a real-time video tracking system, in particular, a dedicated discrete-analog computer capable of processing complex, time-varying visual information in order to extract the camera mount control or guidance signals necessary to follow a predetermined visual target.

The attempted automation of this capability results in an extremely complicated problem. This is at least partially due to the fact that "...the perceptual processes involved in...the human visual system ...are not well understood..." [1], and perhaps accounts for the many unsatisfactory empirical and heuristic solutions which have been proposed.

While a more exact mathematical problem description will be undertaken later, basically (see Figure 1) the problem addressed herein is as follows: The target to be tracked is located through a viewing system (a CCD camera and mount with azimuth, elevation and zoom controls is assumed) in such a way that automatic feature extraction is made possible. Once the target is located, it is desired that the automatic system provide control signals to the camera mount to keep the target centered in the camera field of view (FOV), in spite of target translation, rotation (3-D), magnification, and possibly temporary occlusion (partial or total). Thus, the overall problem is to design a "2-D regulator" for a closed loop control system with both

1-D and 2-D "links,"[1] and therefore it should be expected that both control and image processing theory will be employed in the solution. The military and industrial applications for such a sophisticated system appear promising, particularly if the entire CCD processor could be fabricated on a single circuit board. For example, high-level automatic aircraft trackers or assembly line computer vision systems could be implemented.

1.2 Report Organization and Summary

This report summarizes research which led to the development of a set of new, unique, and CCD implementable video processing algorithms comprising what is termed the Taylor Series Video Image Processor (TSVIP). Basically, this approach eliminates many of the severe shortcomings of previous efforts, including the assumption of unrealistically simple targets and scenes, unsuitability for real-time operation, and inability to handle perspective transformations.

The report is organized as follows: Chapter II presents a more detailed view of the system design concept, necessary algorithms for a successful system, desirable system performance features, limitations, and comparison of the desired system with the human visual system (HVS); Chapter III reviews previous work, including shortcomings; Chapter IV discusses the mathematical details of the problem and develops a model for scene spatial and temporal evolution; and Chapter V presents a unique solution, termed the Taylor Series Video Image Processor (TSVIP). Chapter VI summarizes computer simulation undertaken to confirm the validity of the proposed theory. Finally, Chapter VII summarizes the work

---

1) A note here should be made concerning the term "dimension." Unless otherwise stated, this term refers to the number of arguments of a (continuous or discrete) function, not the size of a vector. For example, using this convention, $P(x,t)$ is a 2-dimensional function, whereas $\underset{\sim}{x}(t)$ is a one-dimensional (vector) function (of t), even though $\underset{\sim}{x}$ may be a nx1 vector, which, in referring to its size, is commonly termed an "N dimensional vector."

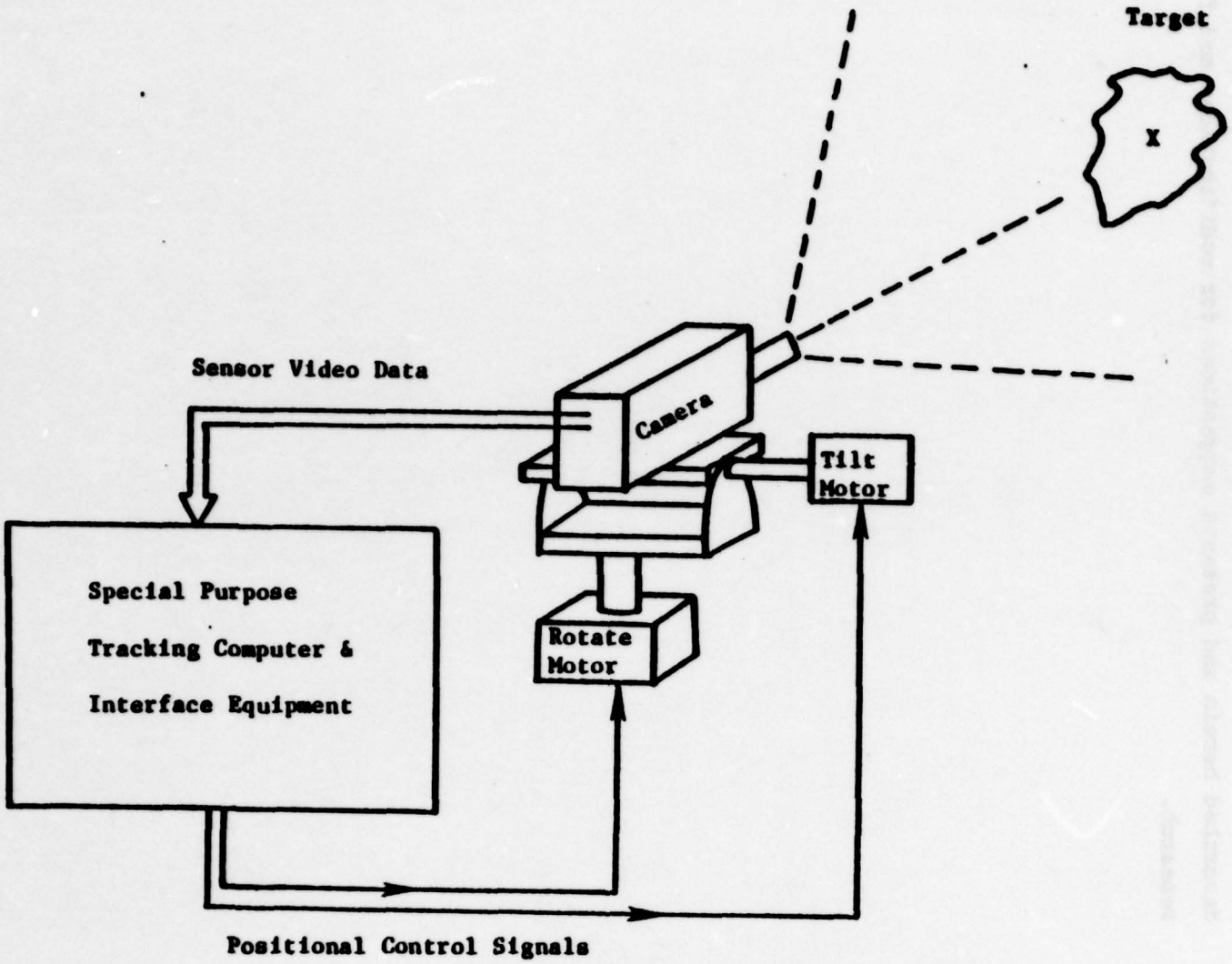
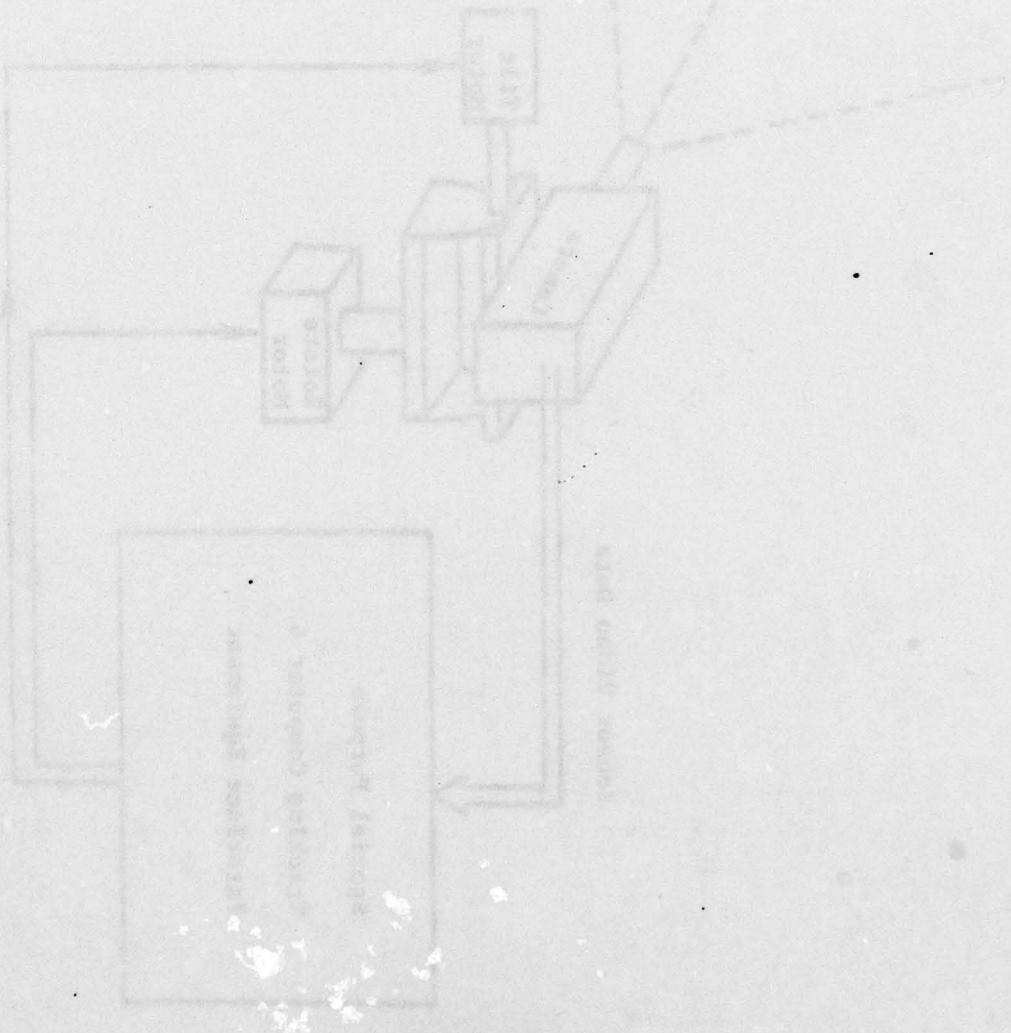


Figure 1 Video Tracking System Example

described herein and presents suggestions for modifications and future research.



CHAPTER II

SYSTEM OVERVIEW

Desirable System Features

The system should be designed with the following general performance measures in mind:

(1) Ability for real-time operation in complex monochromatic scenes. This constraint will generally affect temporal and spatial sampling rates, as well as the complexity and implementation of the resulting system algorithms;

(2) Adaptability to time-varying target and (slowly varying) background parameters, in particular, those related to rotation and dilation of the target; and

(3) Minimum probability of loss of target (LOT), if only according to some loosely defined criterion. Later it will be shown that a suitable measure might be to min $\{E[(\hat{\underset{\sim}{b}}-\underset{\sim}{b})^2]\}$, where $\hat{\underset{\sim}{b}}$ and $\underset{\sim}{b}$ are the estimated and actual target translational parameters, respectively.

2.2 Algorithms Necessary for Successful System

Referring to Figure 2, the minimum software necessary for a successful system may be subdivided into four parts:

(1) A target/background (T/B) separation or segmentation algorithm, which segments the scene by classifying pixels (or groups of pixels) as members of either the target or background sets;

(2) A registration algorithm, which processes information from the just-segmented scene as well as memory information to generate raw estimates of the target centroid (or perhaps deviation of target centroid from some nominal value);

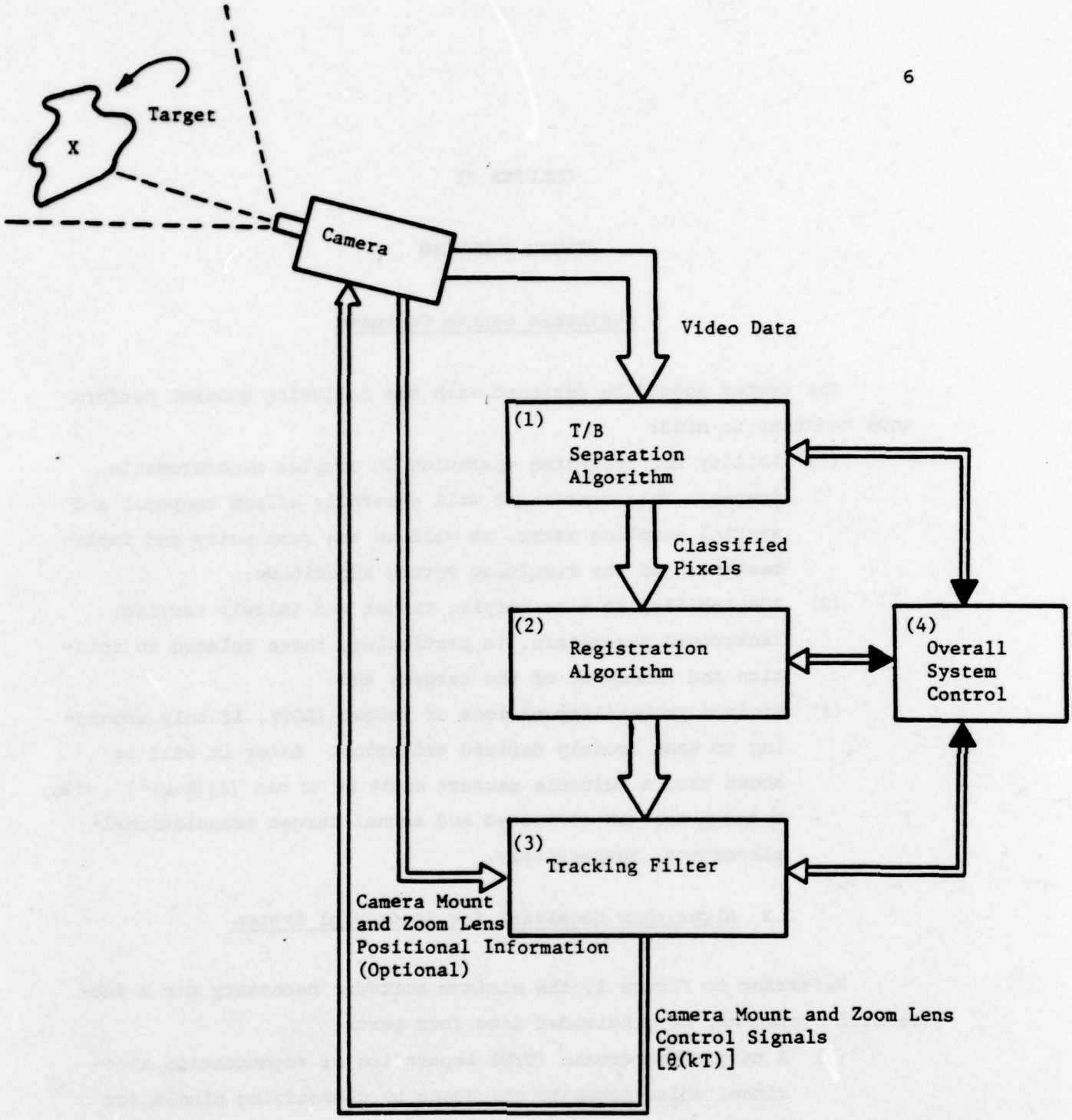


Figure 2 Major Tracking Computer Software Components

(3) A tracking filter, to minimize the effects of noisy data, inexact T/B separation, partial occlusion, and perhaps to predict the target location; and

(4) An overall system control algorithm, to make the major system automatic decisions, and supervise algorithm interaction.

It should be noted that the system's overall success in determining the target position is highly dependent upon the success of algorithms (1) and (2), and that it is possible that both these functions may be accomplished via one algorithm, e.g., a template matching approach. Also, in this report, the major emphasis will be on the development of the first two algorithms, i.e., target/background separation and registration.

2.3 System Limitations and Assumptions

The following restrictions and assumptions are posed for the system developed herein:

(1) The tracking system is a purely passive system, i.e., it processes only 2-D monochromatic visual information resulting from the physics of the scene environment (although any type of information containing texture should be satisfactory).

(2) The image processing algorithms will be generally non-syntactic in approach, thus, according to Tanimato [2], the system employs "...image-driven vision..." rather than "...knowledge-driven vision...." This assumption greatly reduces the systems' need for extensive a priori information.

(3) The system algorithms will be designed to make maximum use of the potential advantages of charge coupled device (CCD) implementation. While detailed explanations of the utility of CCD devices is considered elsewhere [3-8], basically CCD's are potentially useful for the following reasons:

(i) No time-consuming A/D or D/A conversion would be required; and

(ii) Potential processing speed is predicted to

be in excess of 50 MHz [9].

(4) No attempt will be made to handle target re-acquisition or video processing in cases of partial or temporary total occlusion.

(5) The target, even in the case of complex scenes, possesses some attributes or features which make it (automatically) distinguishable from the background. In fact (Section 5.3), it will later be assumed that the target possesses a significant amount of texture.

(6) The target is centered in the camera FOV at the initial or starting time (perhaps by a human operator).

2.4 Comparison of Proposed System with the Human Visual System (HVS)

Obviously, in developing "artificial intelligence" for an automatic video tracking system, one standard of comparison should be the human visual system (HVS). While it is clear that the overall operation of the HVS system is not totally understood [1], certain basic principles of operation, e.g., several of the Gestalt laws, may provide useful design guidelines, and possibly may be automated in real-time. Apparently HVS operation is highly syntactic in approach, i.e., certain pieces of information are extracted from the scene, and together with a great deal of a priori information ("grammar" functions), perhaps acquired over a span of many years, scene analysis proceeds. Two Gestalt laws of particular importance are: (1) the Gestalt law of organization, which roughly says that some areas with common properties (e.g., texture, pattern) are most naturally seen as a unit; and (2) the Gestalt law of common fate, which says that "...if a collection of parts move in unison, one tends to see them as a single figure..." [1]. Researchers have made significant use of the first law, e.g., in texture edge detection, however the second principle has not received nearly as much attention, probably because most scene analysis has been centered on 2-D, time invariant functions. It will be shown later that, for moving targets, automation of the Gestalt law of common fate may be extremely useful for segmentation purposes.

In addition, the HVS, having a biological origin, is highly adaptive, both in real-time, and in the long term. The automatic system under consideration here should also be adaptive, however it is probably not possible or practical to automate long-term adaptivity, i.e., "artificial evolution."

Finally, as mentioned above, the HVS utilizes an extremely large amount of a priori information, as a result of the human learning process. Clearly it is impractical to supply the system considered here with such a large training set of samples, nor would it be practical to incorporate memory sufficient for this data.

CHAPTER III

SUMMARY OF PREVIOUS WORK

3.1 Previous Research Efforts

Due to the extremely current nature of this subject, there are only a handful of specific references. Perhaps the earliest unclassified reference is that of Moskowitz [10] in 1964. This paper outlines the overall problem of visual target tracking, with emphasis on successive scene target registration and the effects of geometric distortion. Two possible template-matching solutions, i.e., correlation and moments, are suggested, and brief 1-D examples of these methods are included.

Yi and Grommes [11] present an analysis and simulation of a video tracking system based upon a simple centroid and second moment algorithm, and have applied this concept to extremely simple targets and scenes (white background, black target, 50x50 pixel scene) at a sampling rate of 60 Hz. This approach is expected to yield poor performance with realistic targets and scenes, primarily due to the lack of an adaptive target/background separation algorithm. A similar shortcoming is found in the simple algorithms of Milstein and Lazicky [12], Baker and McVey [13], Woolard and McVey [14], and Lubinski, et al. [15].

Chow and Aggariwal [16], while also assuming unrealistically simple targets, consider the use of velocity information to aid in template prediction and efficient target registration. Matching invariant to translation and rotation is achieved through the use of area and principal axis as features, and partial occlusion is handled via a simple velocity based prediction algorithm and threshold test.

Uno, et al. [17] devise a simple, real-time implementable algorithm to recognize the shapes of simple objects moving horizontally in the FOV of a TV camera through a simple projection based method of feature extraction. Their work may be significant in that they establish an object-dependent I/O function for the video image processor (V.I.P.).

A similar function is proposed by Baker and McVey [13], and is used in a simple microcomputer-based tracking system. McVey and Woolard [14] make further use of this function together with a perturbation method to determine the proper control signal polarity for simple image tracking.

Armstrong [9] couples a similar I/O control function with a simple signature-based, horizontally and vertically separable, CCD-implementable image processing algorithm to form a closed loop control system, and provides a limited stability analysis for such a system.

It appears that the most current and sophisticated work in this area is due to Flachs, et al. [18-24]. These authors propose a prototype real-time video tracking system which uses extensive parallel processing to enable tracking in 512x512 pixel scenes at a rate of 60 frames/sec. The heart of this system is a Bayesian pixel classifier, aided by fuzzy set logic which forms a binary picture subdivided into the target, background and plume regions. Feature extraction is accomplished via binary projections. A finite-state synchronous sequential machine is used to generate control (pointing) signals, and linear N-point polynomial filters are used for smoothing and predicting target trajectory data.

3.2 Shortcomings of Previous Efforts

The cumulative shortcomings of previous efforts are as follows:

(1) The assumption of unrealistically simple scenes, and therefore the proposal of correspondingly simple algorithms, most often lacking suitable target/background separation or registration algorithms. This generally rules out the successful use of these algorithms with realistic scenes;

(2) Unsuitability for real-time operation. Most previous work has only considered algorithm development, and most often simulation via Fortran programming. Little attempt has been focused on the real-time implementation of even the simplest algorithms;

(3) Inability of the algorithms to handle time-varying scenes;

(4) Inability of the system to handle targets which experience perspective transformations, in particular rotation and dilation; and

(5) Lack of any type of a scene or target model to guide algorithm development. This probably explains why most previous efforts were empirically based.

CHAPTER IV

MATHEMATICAL FORMULATION OF THE VIDEO TRACKING PROBLEM

A logical prerequisite for the development of the system tracking algorithms discussed in Chapter II is a consideration of the underlying processes which produce the system time-varying camera input. Therefore, in this chapter a general 3-D mathematical model for constrained scene-to-scene target motion is developed. Using this model, more precise video tracking system design goals may be formulated. In addition, a closed-loop block diagram for the overall multidimensional process under consideration is developed.

4.1 Block Diagram and Measurement Variables

A first step in the development of a mathematical model for the system described in Section 1.1 is the formulation of a block diagram, and the determination of measurement and control variables. Recalling the assumption of a primary discrete-analog tracking computer, it is obvious that the system will operate in discrete time, i.e., the system measurement and control variables will be obtained at discrete times $t = kT$; $k = 0, 1, \ldots N$, where T is the temporal sampling interval. Also, any 2-D scene at time kT will be a spatially sampled function, for one or both of the following reasons:

(1) A CCD-type camera generates pixel information based upon an NxM discrete sampling lattice; and

(2) The discrete-analog processor necessarily operates with data at discrete instants (note that this data may be discrete in time or space), therefore, even if the scene were continuous in space it would be necessary to spatially sample the information prior to processing. More will be said about sampling in Section 4.3.3.

Looking at Figure 1, two points are noted:

(1) The positional control variables or signals (hereafter denoted by $\underline{\theta}(kT)$ where

$$\underset{\sim}{\theta}(kT) = \begin{bmatrix} \theta_a(kT) \\ \theta_e(kT) \\ z(kT) \end{bmatrix} \qquad (4.1\text{-}1)$$

and

$\theta_a(kT)$ is the camera mount azimuth control at time kT;

$\theta_e(kT)$ is the camera mount elevation control at time kT;

and $z(kT)$ is the camera lens zoom control at time (kT), are a function of the measurement variables or raw video data (hereafter denoted by $p(\underset{\sim}{i},kT)$, the 2-D time varying picture function) perhaps at times (k-1)T, (k-2)T, etc.

(2) $p[\underset{\sim}{i},(k+1)T]$ is a function of $\underset{\sim}{\theta}(kT)$ as well as other variables, especially the maneuvering target trajectory, as discussed in Section 4.3.2. Thus, it should be expected that the overall model used to represent this sytem is of the closed-loop, feedback type.

Figure 3 shows a block diagram based upon the above reasoning. The upper three blocks labelled "T/B Separation and Registration," "Tracking Filter and Camera Mount Dynamics," and "Overall Control" correspond to those algorithms or known constants comprising the tracking computer. The numbers in parenthesis in these blocks correspond to the algorithms described in Section 2.2 and Figure 2. The lower block, labelled "Video Process I/O Relationship," is used to model the evolution of the picture as a function of the camera mount controls and other information, in particular, the time-varying scene data generated by the moving target, time-vary background, etc. In addition, Figure 3 indicates that the system has a unique feature, namely the 2-D "link" existing between the Video Process I/O Relationship and the Tracking Computer. The Tracking Computer apparently can be looked upon as a feedback controller, with the notable feature that it processes 2-D data to form 1-D control signals. The block corresponding to "Video Process I/O

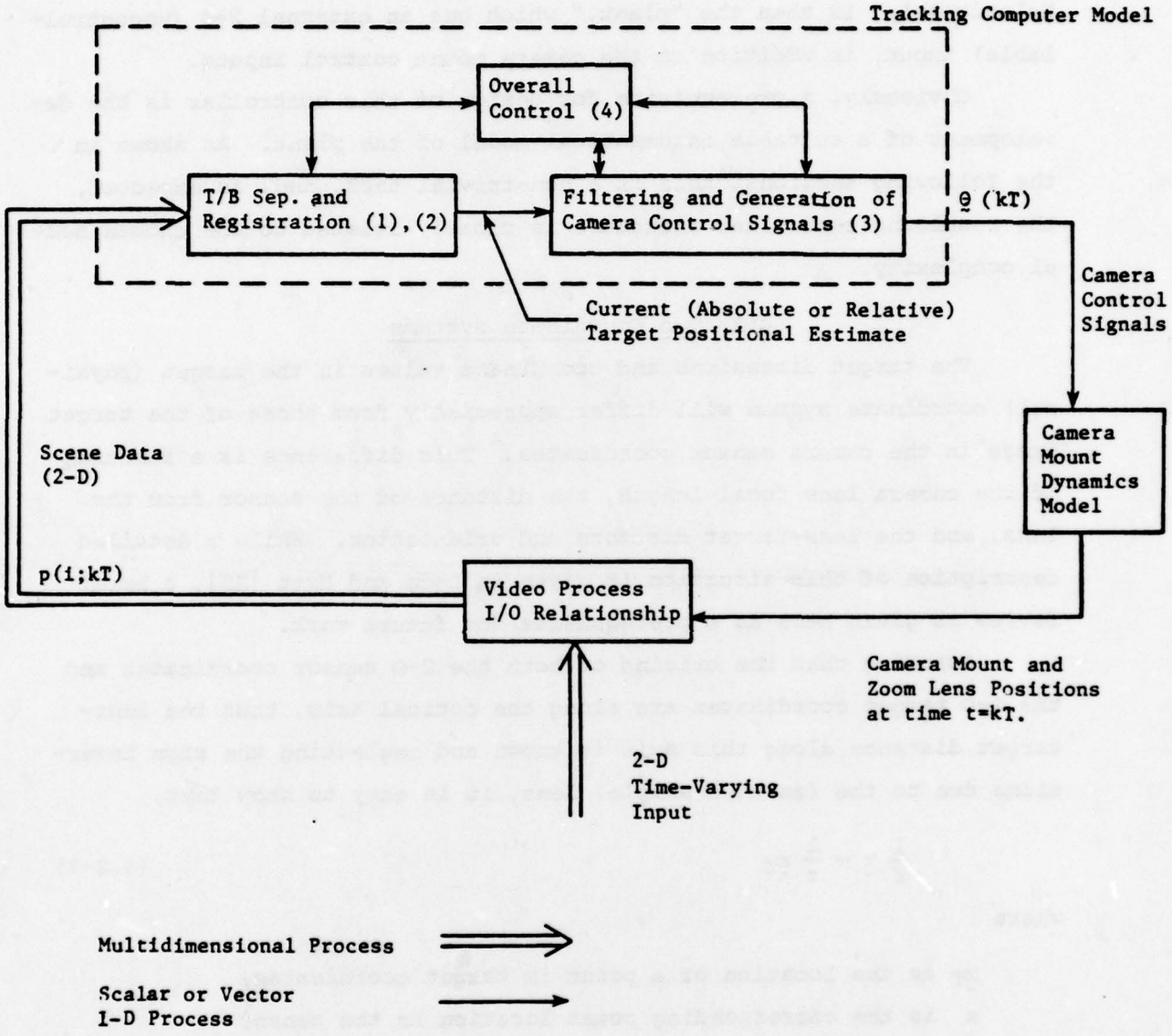


Figure 3. Basic Closed Loop System Model

Relationship" is then the "plant," which has an external 2-D (uncontrollable) input, in addition to the camera mount control inputs.

Obviously, a prerequisite for design of this controller is the development of a suitable mathematical model of the plant. As shown in the following sections, this is a non-trivial task, and, as expected, the resulting controller structure is closely related to the chosen model complexity.

4.2 Two Coordinate Systems

The target dimensions and coordinate values in the target (physical) coordinate system will differ appreciably from those of the target image in the camera sensor coordinates. This difference is a function of the camera lens focal length, the distance of the sensor from the lens, and the lens-target distance and orientation. While a detailed description of this situation is given in Duda and Hart [25], a brief review is given here as a prerequisite for future work.

Assuming that the origins of both the 2-D sensor coordinates and the 2-D target coordinates are along the optical axis, that the lens-target distance along this axis is known and neglecting the sign inversions due to the (assumed simple) lens, it is easy to show that,

$$\frac{1}{f}\underset{\sim}{x} = \frac{1}{z}\underset{\sim}{x}_T \tag{4.2-1}$$

where

$\underset{\sim}{x}_T$ is the location of a point in target coordinates;

$\underset{\sim}{x}$ is the corresponding point location in the sensor coordinates;

z is the camera-target distance; and

f is the sensor-lens distance.

Thus, in the sensor plane

$$\underset{\sim}{x} = \frac{f}{z}\underset{\sim}{x}_T \tag{4.2-2}$$

or

$$\underset{\sim}{x} = K_d\underset{\sim}{x}_T \tag{4.2-3}$$

Typically, K_d is very small, e.g., for

$f = 50$ mm

$z = 100$ m

$K_d = 5 \times 10^{-4}$

The preceding is important for two reasons:

(1) If K_d changes due to a change in z, for a fixed sensor spatial sampling rate, it may be necessary to adjust K_d via f (zoom lens control) in order to maintain proper spatial resolution; and

(2) The tracking algorithms to be developed will estimate target translation in sensor units, therefore, in designing the camera mount controls, it is necessary to convert this information into physical units (Section 5.4.7).

For the remainder of this work, unless otherwise noted, all variables will be referred to in the sensor coordinates.

4.3 Model Development

In the following sections, models for the static scene and scene-to-scene temporal evolution based upon the characteristic function and time-varying affine transform are presented. In the context of these models, the video tracking problem may then be stated as a (time-varying) affine transform parameter estimation problem.

4.3.1 Basic (Continuous) Scene Model Formulation (2-D)

Following the approach of Nahi [26], it may be shown that (for the case of only one target) the scene may be modelled as follows:

$$p(\underset{\sim}{x};t) = f(\underset{\sim}{x};t)\lambda(\underset{\sim}{x};t) + [1-\lambda(\underset{\sim}{x};t)]b(\underset{\sim}{x};t) \qquad (4.3.1\text{-}1)$$

where the following 3-D functions are defined as

$p(\underset{\sim}{x};t) \triangleq$ picture function intensity at sensor location $\underset{\sim}{x}$ and time t

$b(\underset{\sim}{x};t) \triangleq$ background function intensity at sensor location $\underset{\sim}{x}$ and time t

$f(\underset{\sim}{x};t) \triangleq$ target textural function intensity at sensor location $\underset{\sim}{x}$ and time t

and

$\lambda(\underset{\sim}{x};t) \triangleq$ scene characteristic or replacement function value at sensor location $\underset{\sim}{x}$ and time t, where, at time t

$$\lambda(\underset{\sim}{x};t) = \begin{cases} 1 & \underset{\sim}{x} \text{ where target exists in scene} \\ 0 & \text{elsewhere} \end{cases}$$

Thus, $\lambda(x;t)$ is the binary valued function which, by "turning on" and "turning off" the target and background functions, respectively, gives the target its outline. A simple example of scene modelling, using the above concepts, is shown in Figure 4. Several points should be noted:

(1) The scene model is in general a non-linear equation; and

(2) The estimation of $\lambda(x;t)$, in a complex scene, is a very difficult problem.

Several other useful definitions are:

$$s(\underset{\sim}{x};t) \triangleq f(\underset{\sim}{x};t)\lambda(\underset{\sim}{x};t) \qquad (4.3.1\text{-}2a)$$

$$n(\underset{\sim}{x};t) \triangleq [1-\lambda(\underset{\sim}{x};t)]b(\underset{\sim}{x};t) \qquad (4.3.1\text{-}2b)$$

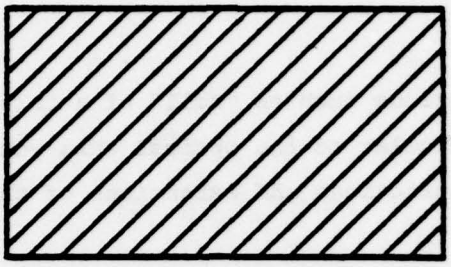
so that

$$p(\underset{\sim}{x};t) = s(\underset{\sim}{x};t) + n(\underset{\sim}{x};t) \qquad (4.3.1\text{-}2c)$$

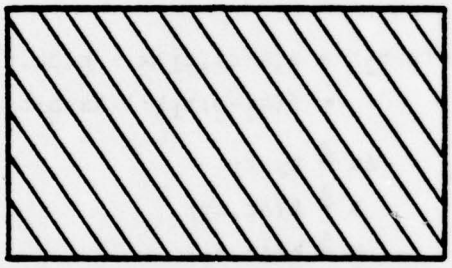
Clearly, $s(\underset{\sim}{x};t)$ is the function that is to be tracked over the time interval of interest.

4.3.2 Continuous Scene Temporal Evolution and 3-D Model

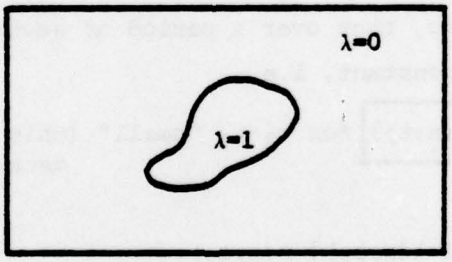
The affine transform is described in Appendix (1). For the remainder of this work, it will be assumed that target perturbations will be restricted to those which can be modelled using a time-varying two-dimensional affine transform (further restrictions are made in Section 5.4.4.2). This means that target motion is constrained to be observed as affine motion in the plane perpendicular to the optical axis, motion along the optical axis (dilation) and translation in the plane perpendicular to this axis. Clearly, this does not limit target motion to 2-D, since an object moving simultaneously to the right and towards the camera may be thought of as experiencing both dilation and translation.



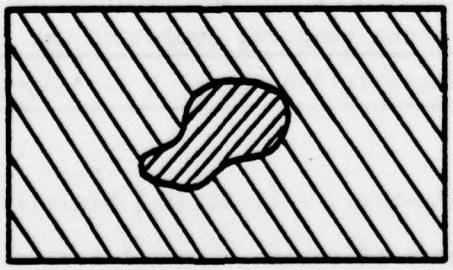
(a) $f(\underset{\sim}{x})$



(b) $b(\underset{\sim}{x})$



(c) $\lambda(\underset{\sim}{x})$



(d) $p(\underset{\sim}{x})$

Figure 4 Basic Scene Formulation Using Model

However, perturbations such as "roll" of a target whose major axis is in the plane perpendicular to the optical axis would be prohibited. Easement of these restrictions and extensions of the model are discussed in Section 4.3.4.

In the continuous (space and time) case, with the above restrictions, the target "outline", λ, must be perturbed in the same manner as the textural function, f, therefore the target function evolves temporally as

$$s(\underset{\sim}{x};t_2) = s[A(\Delta t,t_1)\underset{\sim}{x} + \underset{\sim}{b}(\Delta t,t_1);t_1] \qquad (4.3.2\text{-}1)$$
$$= f(A\underset{\sim}{x}+\underset{\sim}{b};t_1)\cdot\lambda(A\underset{\sim}{x}+\underset{\sim}{b};t_1)$$

where

$$\Delta t \triangleq t_2-t_1$$

and

$$A \triangleq A(\Delta t,t_1)$$

and

$$\underset{\sim}{b} \triangleq \underset{\sim}{b}(\Delta t,t_1) \qquad (4.3.2\text{-}2)$$

are the time-varying affine transform parameter matrix and vector respectively.

At this point also assume that the background function, b, is only slowly time varying, thus over a period of several temporal samples it is approximately constant, i.e.,

$$\boxed{b(\underset{\sim}{x};t_1) = b(\underset{\sim}{x};t_2)} \text{ for } t_2-t_1 \text{ "small" (this is covered in more detail in Section 5.4.6.2)}$$

and therefore

$$n(\underset{\sim}{x};t_2) = [1-\lambda(\underset{\sim}{x};t_2)]\, b(\underset{\sim}{x};t_2) \qquad (4.3.2\text{-}3)$$
$$= [1-\lambda(A\underset{\sim}{x}+b;t_1)]\, \underset{\sim}{b}(\underset{\sim}{x};t_1)$$

Thus, n is only affected by changes in λ.

The overall scene temporal evolution may then be written as

$$\boxed{p(\underset{\sim}{x};t_2) = f(A\underset{\sim}{x}+\underset{\sim}{b};t_1)\lambda(A\underset{\sim}{x}+\underset{\sim}{b};t_1) + [1-\lambda(A\underset{\sim}{x}+\underset{\sim}{b};t_1)]b(\underset{\sim}{x};t_1)}$$

or

$$p(\underset{\sim}{x};t_2) = s[A\underset{\sim}{x}+\underset{\sim}{b};t_1] + [1-\lambda(\underset{\sim}{x};t_2)]b(\underset{\sim}{x};t_1) \qquad (4.3.2\text{-}4)$$

A block diagram of target temporal evolution for two successive scenes is shown in Figure 5.

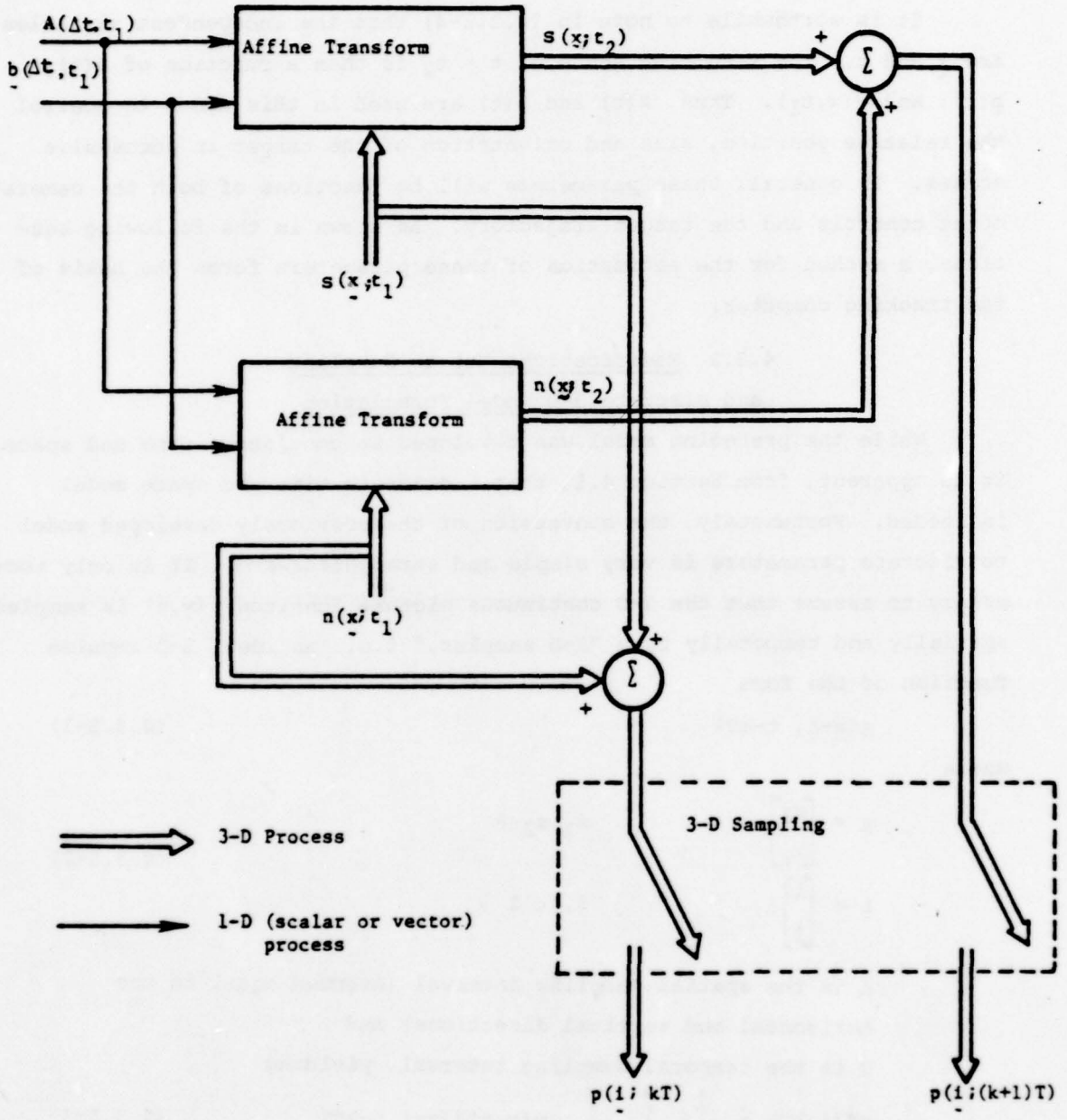


Figure 5 Constrained Scene Temporal Evolution

It is worthwhile to note in (4.3.2-4) that the independent variables are $\underset{\sim}{x}$ and t. The resulting scene at $t = t_2$ is then a function of A(t), $\underset{\sim}{b}(t)$, and $p(\underset{\sim}{x},t_1)$. Thus, A(t) and $\underset{\sim}{b}(t)$ are used in this model to control the relative position, size and orientation of the target in successive scenes. In general, these parameters will be functions of both the camera mount controls and the target trajectory. As shown in the following sections, a method for the estimation of these parameters forms the basis of the tracking computer.

4.3.3 Modifications Due to Sampling and Discrete 3-D Model Formulation

While the preceding model was developed in continuous time and space, it is apparent, from Section 4.1, that a discrete time and space model is needed. Fortunately, the conversion of the previously developed model to discrete parameters is very simple and straightforward. It is only necessary to assume that the 3-D continuous picture funciton $p(\underset{\sim}{x},t)$ is sampled spatially and temporally by a "3-D sampler," i.e., an ideal 3-D impulse function of the form

$$\delta(\underset{\sim}{x}-\underset{\sim}{i},\ t-kT) \tag{4.3.3-1}$$

where

$$\underset{\sim}{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \qquad x_1, x_2 \in R$$

$$\underset{\sim}{i} = \begin{bmatrix} i \\ j \end{bmatrix} \Delta \qquad i,j \in I \tag{4.3.3-2}$$

Δ is the spatial sampling interval (assumed equal in the horizontal and vertical directions) and

T is the temporal sampling interval, yielding

$$p^*(\underset{\sim}{i};kT) = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} p(\underset{\sim}{x};t)\,\delta(\underset{\sim}{x}-\underset{\sim}{i},t-kT) \tag{4.3.3-3}$$

The mathematical considerations and approximations involved in such a formulation are analogous to the well-known 1-D case, e.g., we assume that the discrete-analog tracking computer incorporates a (3-D) zero-order hold device.

Thus, at the output of the ZOH we have the sampled picture function $p(\underset{\sim}{i};\ kT)$, and the corresponding 3-D discrete model temporal evolution may

simply be obtained from (4.3.2-4) as:

$$p[\underset{\sim}{i};(k+1)T] = f[A(kT)\underset{\sim}{i}+\underset{\sim}{b}(kT);kT]\lambda[A(kT)\underset{\sim}{i}+\underset{\sim}{b}(kT);kT] + [1-\lambda[A(kT)\underset{\sim}{i}+\underset{\sim}{b}(kT)];kT]b(\underset{\sim}{i};kT) \quad (4.3.3\text{-}4)$$

In the following sections it will often be useful to switch between the discrete and continuous model formulations, and, in the context of the model developed here, this will not result in any mathematical or practical difficulties. It should be noted, however, that in general, many digital picture processing operations are not "reversible" in the sense that conversion between the discrete and continuous models and/or parameters is straightforward, or even possible. An excellent example of this is the attempted correction of a discretized picture which has undergone perspective distortion (c.f. [1]).

Finally, the effects of the sampling parameters Δ and T, with respect to Fourier sampling constraints, derivative estimation, and the validity of Taylor series approximations are discussed in Sections 5.4.6.1 and 5.4.6.2.

4.3.4 Limitations and Extensions of Model

While many of the constraints and assumptions associated with the previously developed model are not overly restrictive, it is likely that once successful solutions based upon this model have been developed, these restrictions will be gradually eased and new solutions (hopefully) found, until the final model adequately represents a very wide class of real-world situations. For this reason, some obvious extensions of this model are presented.

As mentioned in Section 5.4.4.2, whenever A does not represent strictly rotation and/or dilation, the target is actually being observed through a different viewing angle. Practically, it should also be expected that new target features should appear, particularly around the target edges. One possible way to update the model (4.3.3-4) to handle this information might be to model it as an additional function, e.g.

$$p[\underset{\sim}{i};(k+1)T] = [f(A(kT)\underset{\sim}{i}+\underset{\sim}{b}(kT);kT)+e(kT)]\cdot\lambda[A(kT)\underset{\sim}{i}+\underset{\sim}{b}(kT);kT]$$

$$+ [1-\lambda(A(kT)\underset{\sim}{i}+\underset{\sim}{b}(kT);kT]b(\underset{\sim}{i};kT) \qquad (4.3.4\text{-}1)$$

where e(kT) is primarily "edge oriented."

Closely associated with the above extension is that of handling "roll" of a target whose major axis is in a plane perpendicular to the optical axis. In this case it should be noted that the assumption in (4.3.2-1) is no longer valid, i.e., λ is no longer perturbed in the same manner as f. In fact only f is perturbed in this case, and the scene temporal evolution model will need to be adjusted accordingly.

The model may be adapted to handle multiple targets by defining multiple target and characteristic functions, e.g., f_1, f_2 and λ_1, λ_2. If these targets are allowed to overlap (i.e., pass in front of one another in the FOV) it will be necessary to prioritize the λ_i functions, in order to insure that at any scene point only one λ_i is non-zero. Also note that this adaptation permits the handling of occlusion.

Another important modification of the model might be to permit time-varying scene illumination or brightness variations. This may be accomplished in a rather straightforward manner by simply multiplying the target and background functions by suitable time-varying constants.

4.4 Review of Previous Efforts Using Model

The vast majority of the previously cited research ignores the scene-to-scene temporal link modelled in Section 4.3.2, and discussed in Section 5.1. In the context of the model developed in Section 4.3.1 and the references cited in 3.1, some approaches have been (with temporal indices omitted for simplicity):

(1) Assume $b(\underset{\sim}{x}) = 0 \ \forall \ \underset{\sim}{x}$ so that essentially an all-white (or black) background is present and therefore a T/B separation algorithm is not necessary. Then use simple algorithms to estimate the centroid of $\underset{\sim}{s}(x)$. (e.g., Yi and Grommes [11], Baker/McVey [13], Woolard/McVey [14], Uno [17]).

(2) Edge detection for T/B separation: Constrain $f(\underset{\sim}{x})$ and $b(\underset{\sim}{x})$ and estimate spatial derivatives by differentiation of equation

(4.3.1-1) in both the horizontal and vertical directions as follows:

$$\frac{\partial p(\underset{\sim}{x})}{\partial x_j} = \frac{\partial f(\underset{\sim}{x})}{\partial x_j}\lambda(\underset{\sim}{x}) + \frac{\partial \lambda(\underset{\sim}{x})}{\partial x_j} f(\underset{\sim}{x}) - \frac{\partial \lambda(\underset{\sim}{x})}{\partial x_j} b(\underset{\sim}{x}) + [1-\lambda(\underset{\sim}{x})]\frac{\partial b(\underset{\sim}{x})}{\partial x_j} \tag{4.4-1}$$

$$= \lambda(\underset{\sim}{x})\left[\frac{\partial f(\underset{\sim}{x})}{\partial x_j}\right] - \frac{\partial b(\underset{\sim}{x})}{\partial x_j} + \frac{\partial \lambda(\underset{\sim}{x})}{\partial x_j}[f(\underset{\sim}{x})-b(\underset{\sim}{x})] + \frac{\partial b(\underset{\sim}{x})}{\partial x_j}$$

where $j = 1,2$ (4.4-2)

Then assume that both the target and background have very little texture (see Section 5.2) and that the major abrupt changes in pixel intensity in the scenes are due to the target edges. Thus,

$$\frac{\partial f(\underset{\sim}{x})}{\partial x_j} \simeq 0 \qquad \forall \underset{\sim}{x}$$

$$\frac{\partial b(\underset{\sim}{x})}{\partial x_j} \simeq 0 \qquad \forall \underset{\sim}{x}$$

which leads to

$$f(\underset{\sim}{x})-b(\underset{\sim}{x}) \simeq K \qquad \forall \underset{\sim}{x}$$

so that (4.4-1) becomes

$$\frac{\partial p(\underset{\sim}{x})}{\partial x_j} \simeq \frac{\partial \lambda(\underset{\sim}{x})}{\partial x_j} \cdot K \tag{4.4-3}$$

Thus, spatial derivatives are estimated (perhaps by a simple difference equation) and it is assumed that $\underset{\sim}{x}$'s where $\left|\frac{\partial p(\underset{\sim}{x})}{\partial x_j}\right|$ is "large" are points on the target edge. It should be obvious that if the preceding assumptions are not valid (e.g., in realistic or complex scenes), that it will be very difficult to estimate the target contour (e.g., Chow/Aggarwal [16]).

(3) Thresholding for T/B separation:

Assume

$$E\{f(\underset{\sim}{x})\} \geq K_1$$

$$K_1 > E\{b(\underset{\sim}{x})\} \geq K_2$$

Then design a pixel classifier such that;

$\underset{\sim}{x}_k$ is a target pixel if $p(\underset{\sim}{x}_k) \geq K_1$

$\underset{\sim}{x}_k$ is a background pixel otherwise.

This approach is also questionable in realistic scenes, since it is likely that both the target and background functions will have significant intensity variations, and therefore, if this approach is taken, the resulting segmented scene will have "measles" [1].

(4) Statistical segmentation: Assume that $f(\underset{\sim}{x})$ and $b(\underset{\sim}{x})$ may be distinguishable by some statistical test (variance, pdf estimate, etc.). Then, using this, test, classify pixels, or groups of pixels, as either originating in $f(\underset{\sim}{x})$ or $b(\underset{\sim}{x})$ (e.g., Flachs [18-24]; "texture" edge detection).

In summary, approaches (1-3) are of very limited use in typical scenes, and the accuracy and efficient implementation of (4) is questionable.

4.5 Problem Statement Using Model

Assumption 6 in Section 2.3 assumed that the target is centered in the camera FOV (perhaps by a human operator) at $t = kT = 0$. Therefore, using the previously developed model, and assuming that the effect on the scene due to camera mount controls is known (this is addressed in Section 5.4.7), (4.3.3-4) indicates that any sequence of M scenes,

$$\{p(\underset{\sim}{i};kT)\}; \qquad k = 1,2, \ldots M$$

is related by a sequence of 6M affine parameter variables $A(kT)$, $\underset{\sim}{b}(kT)$; $k = 1,2,\ldots M$.

Recall from Appendix 1 that $\underset{\sim}{b}(kT)$ represents the target translation during the k^{th} sampling interval (i.e., the period from $t = kT$ to $t = (k+1)T$). If $\underset{\sim}{b}(kT)$ were known over the range of k of interest, the tracking problem would be simplified considerably, since this data could be combined with other (assumed) known system parameters (Section 5.4.7) to generate $\theta_a(kT)$ and $\theta_e(kT)$ over this interval. Similarly, if $A(kT)$

were also known, the formulation of z(kT) would be simple. These parameters are not known, however; therefore the basic image tracking problem may be stated as:

> Given $p(\underset{\sim}{i};0)$ and the M succeeding scenes, $p(\underset{\sim}{i};kT)$; k = 1,2,. . .M, generate "good" estimates of A(kT) and $\underset{\sim}{b}(kT)$; k = 1,2,. . .M, and use these estimates to form $\underset{\sim}{\theta}(kT)$; k = 1,2,. . .M, such that the target tends to remain centered in the camera FOV, and (perhaps) approximately the same overall size.

Thus, the algorithms for the estimation of A(kT) and $\underset{\sim}{b}(kT)$ form the basis of the tracking computer. A composite model for the overall video tracking problem is shown in Figure 6. While there probably exist a large number of possible solutions, the majority of the remainder of this work involves the novel solution of this problem using data samples from two successive scenes and an assumption concerning scene-to-scene temporal dependence. This solution has some very important and desirable properties, in particular, the potential for real-time operation and relatively simple CCD implementation. Alternate solutions based upon this model are discussed in Chapter VII.

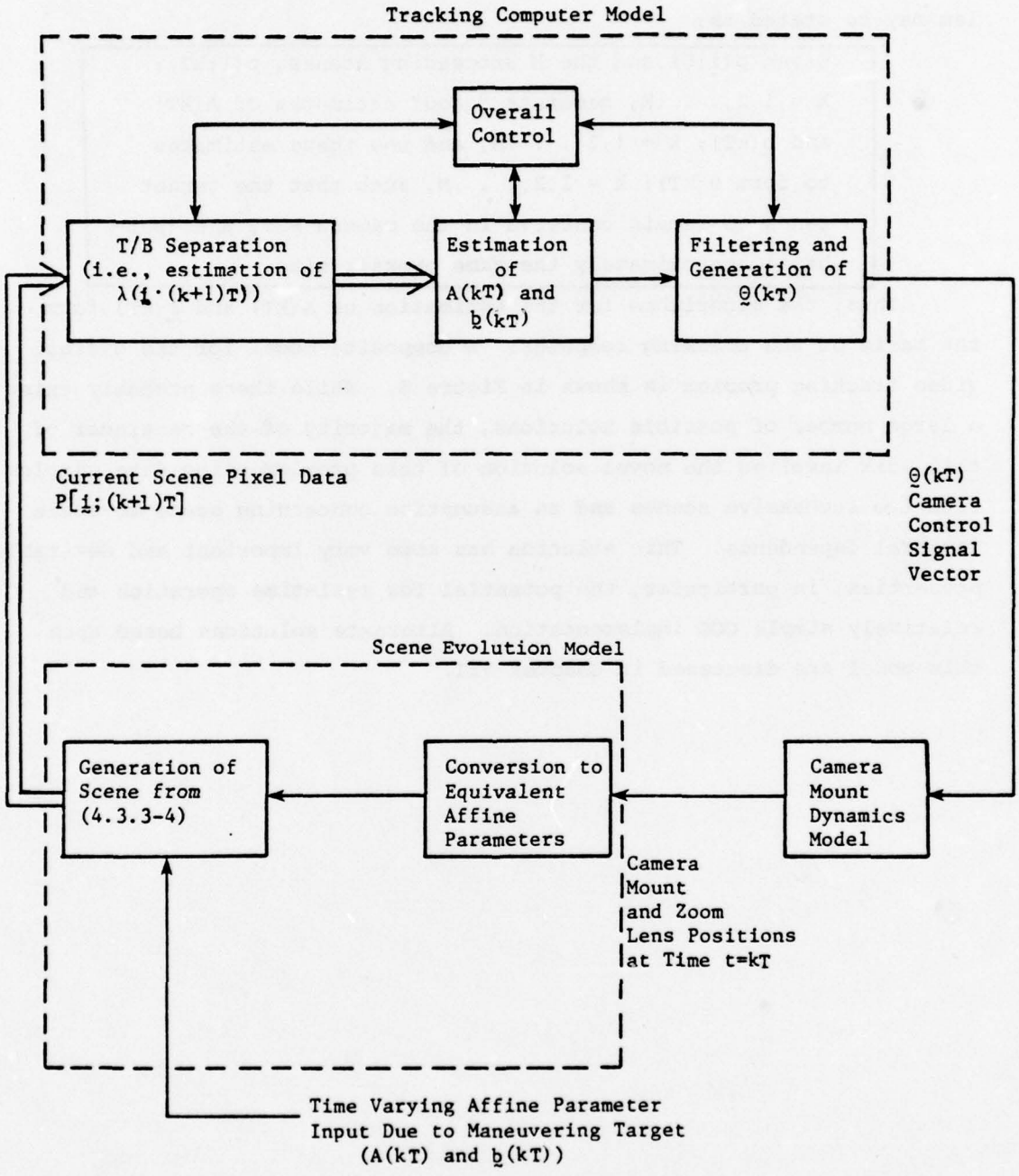


Figure 6 Composite Model for Video Tracking System

CHAPTER V

DEVELOPMENT OF THE TAYLOR SERIES VIDEO IMAGE PROCESSOR (TSVIP) ALGORITHM

This chapter presents a novel and ideally CCD-implementable solution to the video tracking problem as posed in Section 4.5. By assuming a sufficiently fast system temporal sampling rate, it is shown that the scene-to-scene temporal evolution modelled in Section 4.3.3 may be approximated using a linear model. CCD implementable estimation algorithms are then developed for this model to provide estimates of target translation, rotation and dilation.

5.1 Scene Temporal Dependence, Temporal Sampling Rate and Algorithm Complexity, and Time/Space Computational Trade-Offs

The 3-D model developed in the previous section clearly shows the scene-to-scene temporal dependence expected when viewing a moving target. It is expected that as the temporal sampling rate increases, this scene-to-scene temporal dependence also increases, i.e., as $T \to 0$ there is very little change in successive scenes. Conversely, for a relatively large temporal sampling interval, T, it is expected that consecutive scenes would have relatively little information in common, and therefore the problems of extracting target features invariant to expected affine perturbations and the estimation of these perturbations become significantly more difficult, and lead to increasingly complicated algorithms. Therefore, if T may be chosen sufficiently "small" (this is defined more precisely in Section 5.4.6.2), the above reasoning may be employed to design a set of efficient and accurate tracking algorithms, hereafter referred to as the Taylor Series Video Image Processor (TSVIP) algorithms.

The concept of system time/space computational tradeoffs is also noteworthy. As described in Blaauw [27], the total system cost to achieve specified data processing is a function of "time" and "space," where "time" is expressed as a function of the delay times of the physical

components that realize the desired logic function, and "space" refers to the number of physical components employed. For the system considered here, it is clearly desirable to minimize the temporal cost, in order to achieve real-time operation, therefore it will be assumed that the spatial cost of the system (i.e., the number of components per IC chip and the total number of chips) is relatively insignificant. Thus, the system should be designed to operate with a temporal sampling rate sufficiently high to enable real-time operation, and make maximum use of computational complexity in "space." The TSVIP algorithms to be presented make significant use of parallel processing, and also inherently enable temporally efficient "pipelining," since smapled information from the present scene may be processed while the next scene is being sampled simultaneously.

5.2 Target Texture Definition and Relationship to $[\frac{\partial f(\underset{\sim}{x})}{\partial \underset{\sim}{x}}]^T$

Image "texture" is a perceptual phenomenon. Often this term has been used to denote some particular ad-hoc measure of the spatial variations in light intensity of an image, e.g., a standard bar pattern would be thought of as a highly textured pattern. This research views texture from a more mathematical point of view, particularly when considering target texture. By definition, the textured target function is represented by $f(\underset{\sim}{x};t)$, where $||\frac{\partial f(\underset{\sim}{x};t)}{\partial \underset{\sim}{x}}||$ is nonzero for some set of $\underset{\sim}{x}_k$'s, and for all t of interest. Thus, the greater the value of $||\frac{\partial f(\underset{\sim}{x}_k t)}{\partial \underset{\sim}{x}}||$ at a particular $\underset{\sim}{x}$, and the larger the set of $\underset{\sim}{x}_k$'s where this derivative-based function is nonzero, the more "textured" the target is said to be. Texture is thus a target spatial feature. Equivalently, a target is textured if the 2-D function

$$\mathcal{F}\,[f(\underset{\sim}{x};t)]_{t=t_p} \triangleq F(\underset{\sim}{u})\,|_{t=t_p} \qquad (5.2\text{-}1)$$

has nonzero components at spatial frequencies other than $\underset{\sim}{u} = \underset{\sim}{0}$.

Texture has been shown to be very useful for scene segmentation, however, in the work which follows it will be shown that target texture may also be used to aid tracking algorithms. Thus, it is necessary to assume that the target of interest is textured (perhaps even after filtering and preprocessing). This is not an overly restrictive assump-

tion, however, for two reasons:

(1) Most complex targets of interest exhibit this feature; and

(2) The class of textureless targets, which is typically assumed in previous research, enables much simpler processing algorithms for both segmentation and registration, therefore they are not of interest in the present work.

5.3 Taylor Series Expansion of Model and Corresponding Scene Temporal Evolution

In the following two sections the scene temporal evolution model (4.3.3-4) is approximated using a truncated Taylor series expansion. By restricting the tracking processor input to data from the target interior (temporarily assuming T/B separation has been accomplished), the model may be simplified even further, and this simplification enables the affine parameter estimation techniques of Section 5.4.

5.3.1 Direct Approximation for Small A, $\underset{\sim}{b}$

Recall the assumption that

$b(\underset{\sim}{x};t_1) = b(\underset{\sim}{x};t_2)$ for t_2-t_1 "small." Also note from the formulation of scene evolution in (4.3.2-4) that temporal changes in f and λ are only due to the time varying A and $\underset{\sim}{b}$ parameters, therefore

$$\frac{\partial f(\underset{\sim}{x};t)}{\partial t} = \frac{\partial(\underset{\sim}{x};t)}{\partial t} = 0 \quad \forall \underset{\sim}{x} \text{ and } \forall t$$

We can expand $s(\underset{\sim}{x};t_2)$ and $\lambda(\underset{\sim}{x};t_2)$ about $\underset{\sim}{x}$ and keep only the first order terms. Dropping the temporal indices for simplicity:

$$s(A\underset{\sim}{x}+\underset{\sim}{b}) \simeq s(\underset{\sim}{x}) + \left(\frac{\partial s(\underset{\sim}{x})}{\partial \underset{\sim}{x}}\right)^T (A-I)\underset{\sim}{x} + \left(\frac{\partial s(\underset{\sim}{x})}{\partial \underset{\sim}{x}}\right)^T \underset{\sim}{b} \qquad (5.3.1\text{-}1)$$

$$= s(\underset{\sim}{x}) + \left(\frac{\partial s(\underset{\sim}{x})}{\partial \underset{\sim}{x}}\right)^T \Delta A\underset{\sim}{x} + \left(\frac{\partial s(\underset{\sim}{x})}{\partial x}\right)^T \underset{\sim}{b}$$

where $\Delta A(\Delta t,t_1) \triangleq A(\Delta t,t_1) - I$ (5.3.1-2)

clearly $A(0,t) = I$

and $\underset{\sim}{b}(0,t) = \underset{\sim}{0}$

so that

$$\Delta A(0,t) = 0 .$$

Similarly

$$\lambda(A\underset{\sim}{x}+\underset{\sim}{b}) \approx \lambda(\underset{\sim}{x}) + (\frac{\partial\lambda(\underset{\sim}{x})}{\underset{\sim}{x}})^T \Delta A\underset{\sim}{x} + (\frac{\partial\lambda(\underset{\sim}{x})}{\underset{\sim}{x}})^T \underset{\sim}{b} \qquad (5.3.1\text{-}3)$$

Two remarks are in order:

(1) Clearly the above equations are only valid for $||(A-I)\underset{\sim}{x}+\underset{\sim}{b}||$ "small." This will be addressed in Section 5.4.6.2; and

(2) Due to the binary nature of $\lambda(\underset{\sim}{x};t)$, some modification(s) will be necessary to handle $(\frac{\partial\lambda}{\partial\underset{\sim}{x}})^T$. As will be shown later, this is not a serious problem.

Thus, with the above assumptions, it is easy to show that the total scene evolves from (4.3.2-1) as:

$$p(\underset{\sim}{x};t_2) \simeq s(\underset{\sim}{x};t_1) + [\frac{\partial s(\underset{\sim}{x};t_1)}{\partial\underset{\sim}{x}}]^T(\Delta A\underset{\sim}{x}+\underset{\sim}{b}) + b(\underset{\sim}{x};t_1)[1-\lambda(\underset{\sim}{x};t_1) - (\frac{\partial\lambda(\underset{\sim}{x};t_1)}{\partial\underset{\sim}{x}})^T(\Delta A\underset{\sim}{x}+\underset{\sim}{b})] \qquad (5.3.1\text{-}4)$$

Recalling $s = f\lambda$, expanding $\left[\frac{\partial s(\underset{\sim}{x};t_1)}{\partial\underset{\sim}{x}}\right]^T$ yields

$$p(\underset{\sim}{x};t_2) \simeq f(\underset{\sim}{x};t_1)\lambda(\underset{\sim}{x};t_1) + [(\frac{\partial f(\underset{\sim}{x};t_1)}{\partial\underset{\sim}{x}})^T\lambda(\underset{\sim}{x};t_1) + f(\underset{\sim}{x};t_1)(\frac{\partial\lambda(\underset{\sim}{x};t_1)}{\partial\underset{\sim}{x}})^T](\Delta A\underset{\sim}{x}+\underset{\sim}{b}) + b(\underset{\sim}{x};t_1)[1-\lambda(\underset{\sim}{x};t_1)] - b(\underset{\sim}{x};t_1)(\frac{\partial\lambda(\underset{\sim}{x};t_1)}{\partial\underset{\sim}{x}})^T(\Delta A\underset{\sim}{x}+\underset{\sim}{b}) \qquad (5.3.1\text{-}5)$$

so there are 3 primary scene regions:

(1) $\lambda = 1$; $(\frac{\partial\lambda}{\partial\underset{\sim}{x}}) = \underset{\sim}{0}$ (Interior of target)

(2) $\lambda = 0$; $(\frac{\partial\lambda}{\partial\underset{\sim}{x}}) = \underset{\sim}{0}$ (Background)

(3) $\lambda = 0$ or 1; $(\frac{\partial\lambda}{\partial\underset{\sim}{x}})$ = "large" (Target contour)

For the remainder of the discussion in this section, assume the system is only operating on points where $\lambda = 1$ and $\frac{\partial\lambda}{\partial\underset{\sim}{x}} = \underset{\sim}{0}$. Thus, only target texture points are being considered and the segmentation problem temporarily is assumed solved. The topic of segmentation is covered in detail in Section 5.4.5. The T/B separation algorithm provides an esti-

mate of $p(\underset{\sim}{x};t)$ for $\lambda(\underset{\sim}{x};t) = 1$ (i.e., $f(\underset{\sim}{x};t)$ on the target interior). Denote this estimate by $p_s(\underset{\sim}{x};t)$ where $p_s(\underset{\sim}{x};t) = p(\underset{\sim}{x};t)$ $\forall \underset{\sim}{x}$ where $\lambda(\underset{\sim}{x};t) = 1$. The remaining system is therefore "nearsighted" in that it operates strictly on points in the interior of the target.

Now define the scene difference function based upon target textural points as:

$$d(\underset{\sim}{x};t_2) \triangleq p_s(\underset{\sim}{x};t_2) - p_s(\underset{\sim}{x};t_1) \qquad (5.3.1\text{-}6)$$

Recalling past assumptions, it is easy to show that:

$$d(\underset{\sim}{x};t_2) \simeq \left(\frac{\partial f(\underset{\sim}{x};t_1)}{\partial \underset{\sim}{x}}\right)^T [\Delta A(\Delta t,t_1)\underset{\sim}{x} + \underset{\sim}{b}(\Delta t,t_1)] \quad \forall \underset{\sim}{x} \in \{R_i\} \qquad (5.3.1\text{-}7)$$

where $\{R_i\}$ is the set of all points on the interior of both the original and perturbed targets.

Remark--This probably seems obvious intuitively, however the preceding work is useful in that it enables handling the more general case, i.e., we can study the effect of inexact segmentation on the algorithms to be developed (Section 5.4.6.3).

If (5.3.1-7) holds, then a very nebulous 3-D problem may be reduced to one of 1-D (time-varying) parameter estimation. A note should be made that, while the preceding derivations were for the continuous case, these ideas also apply to the discretized version, as explained in Section 4.3.3. Before proceeding to estimation considerations, it is useful to consider model temporal evolution from a different (but equivalent) point of view.

5.3.2 Using the Scene Temporal Differential Equation and Difference Approximations

Recall, from (4.3.2-1) that, for the target textural function alone (i.e., where $\lambda(\underset{\sim}{x};t) = 1$)

$$f(\underset{\sim}{x};t_2) = f[A(t_2-t_1,t_1)\underset{\sim}{x} + \underset{\sim}{b}(t_2-t_1,t_1);t_1] \qquad (5.3.2\text{-}1)$$

Expanding (5.3.2-1) in a Taylor Series expansion and retaining only the

linear terms yields (temporal indices are understood):

$$f(A\underset{\sim}{x}+\underset{\sim}{b}) \simeq f(\underset{\sim}{x}) + (\frac{\partial f(\underset{\sim}{x})}{\partial \underset{\sim}{x}})^T (\Delta A\underset{\sim}{x}+\underset{\sim}{b})$$

so

$$f(\underset{\sim}{x};t_2) - f(\underset{\sim}{x};t_1) = (\frac{\partial f(\underset{\sim}{x};t_1)}{\partial \underset{\sim}{x}})^T [\Delta A(\Delta t,t_1)\underset{\sim}{x} + \underset{\sim}{b}(\Delta t,t_1)] \qquad (5.3.2\text{-}4)$$

By definition

$$\frac{df(\underset{\sim}{x};t_1)}{dt} \triangleq \lim_{\Delta t\to 0} \{\frac{f(\underset{\sim}{x};t_2) - f(\underset{\sim}{x};t_1)}{\Delta t}\}$$

$$= \lim_{\Delta t\to 0} \frac{(\frac{\partial f(\underset{\sim}{x};t_1)}{\partial x})^T [A(\Delta t,t_1)-I)\underset{\sim}{x} + \underset{\sim}{b}(\Delta t,t_1)]}{\Delta t} \qquad (5.3.2\text{-}5)$$

Obviously (5.3.2-5) goes to 0/0 in the limit, so we can apply L'Hospital's rule, yielding:

$$\boxed{\frac{df(\underset{\sim}{x};t)}{dt} = (\frac{\partial f(\underset{\sim}{x};t)}{\partial \underset{\sim}{x}})^T [\frac{dA(T)}{dt}\underset{\sim}{x} + \frac{d\underset{\sim}{b}(t)}{dt}]} \qquad (5.3.2\text{-}6)$$

Equation (5.3.2-6) clearly shows that target textural function temporal and spatial evolution are linked, the "link" being the temporal partial derivatives of the affine transform parameters. It is this link or dependence which has been ignored in past efforts, and which, when carefully exploited, leads to the novel tracking algorithms presented in this work. Thus, in the context of the previously developed model, scene temporal evolution is governed by $\frac{\partial A(t)}{\partial t}$, $\frac{\partial \underset{\sim}{b}(t)}{\partial t}$ and $[\frac{\partial s(\underset{\sim}{x};t)}{\partial \underset{\sim}{x}}]^T$.

Furthermore, it is easy to show that (5.3.2-6) also leads to (5.3.1-7), by approximating derivatives in (5.3.2-6) as differentials:

$$\frac{\Delta f(\underset{\sim}{x};t+\Delta T)}{\Delta t} \simeq (\frac{\partial f(\underset{\sim}{x};t)}{\partial x})^T \frac{\Delta A(t)}{\Delta t}\underset{\sim}{x} + \frac{\Delta \underset{\sim}{b}(t)}{\Delta t} \qquad (5.3.2\text{-}7)$$

Multiplying by Δt, recalling

$$\Delta t = t_2 - t_1$$

and

$$\Delta A(\Delta t,t_1) = A(\Delta t,t_1)-A(o,t_1)$$
$$= A(\Delta t,t_1)-I$$

and defining

$$\Delta \underset{\sim}{b}(t_1) = \underset{\sim}{b}(\Delta t,t_1)-\underset{\sim}{b}(o,t_1)$$
$$= \underset{\sim}{b}(\Delta t,t_1)$$

we see that

$$\Delta f(\underset{\sim}{x};t_2) = \left(\frac{\partial f(\underset{\sim}{x};t_1)}{\partial \underset{\sim}{x}}\right)^T [(A(\Delta t,t_1)-I)\ \underset{\sim}{x} + \underset{\sim}{b}(\Delta t,t_1)] \qquad (5.3.2\text{-}8)$$

and from (5.3.1-6), since we are only considering the region where $\lambda = 1$,

$$d(\underset{\sim}{x};t_2) = d(\underset{\sim}{x};t_1+\Delta t)$$
$$= f(\underset{\sim}{x};t_1+\Delta t) - f(\underset{\sim}{x};t_1) \qquad (5.3.2\text{-}9)$$
$$= \Delta f(\underset{\sim}{x};t_1+\Delta t)$$

then, using (5.3.2-8),

$$d(\underset{\sim}{x};t_2) = \left(\frac{\partial f(\underset{\sim}{x};t_1)}{\partial \underset{\sim}{x}}\right)^T [(A(\Delta t,t_1)-I)\underset{\sim}{x} + \underset{\sim}{b}(\Delta t,t_1)] \qquad (5.3.2\text{-}10)$$

which is identical to the previously derived (5.3.1-7) for all points on the target interior.

5.4 Reduction to 1-D Model and Solution Approaches

In this section, the 3-D model of Section 5.3 is reduced to a 1-D model, and it is shown that the video tracking problem may be cast as a time-varying affine transform parameter vector estimation problem. Several possible solution approaches are then presented, in particular, one CCD-implementable solution based upon the formation of a pseudoinverse of an Nx4 partitional matrix is discussed in detail.

5.4.1 Scene Difference Function with Assumed Segmented Scene, Reduction to 1-D Problem and Statistical Sampling

Recall (5.3.1-6), i.e.,

$$d(\underset{\sim}{x};t_2) = p_s(\underset{\sim}{x};t_2) - p_s(\underset{\sim}{x};t_1) \qquad (5.4.1\text{-}1)$$

In the actual discrete time system, $d(\underset{\sim}{x};t_2)$ is not measured, but

rather the sampled version of this function, i.e., $d(\underset{\sim}{i};(k+1)T)$ (Section 4.3.3). This scene difference function may then be "re-sampled" (in a statistical sense); yielding an Nxl 1-D vector, hereafter denoted as $\underset{\sim}{d}((k+1)T)$ or more simply d in the remaining discussion. Thus,

$$\underset{\sim}{d}((k+1)T) = [d(\underset{\sim}{x}_i;\ (k+1)T)] \qquad (5.4.1\text{-}2)$$

where

$$\underset{\sim}{x}_i = \begin{bmatrix} x_{i1} \\ x_{i2} \end{bmatrix} \qquad i=1,\ 2,\ \ldots\ ,N \qquad (5.4.1\text{-}3)$$

is the vector x at the i^{th} sampling point. An important point here is that the x_i's need not be consecutive, i.e., with some limitations, samples in both the horizontal and vertical directions may be skipped since eventually an overdetermined set of linear equations will be formulated and therefore only a set of N "good" statistical samples is necessary.

5.4.2 Matrix Formulation

Defining

$$\Delta A(kT) = \begin{bmatrix} a_{11}(kT) & a_{12}(kT) \\ a_{21}(kT) & a_{22}(kT) \end{bmatrix} \qquad (5.4.2\text{-}1)$$

and

$$\underset{\sim}{b}(kT) = \begin{bmatrix} b_1(kT) \\ b_2(kT) \end{bmatrix} \qquad (5.4.2\text{-}2)$$

and making the following additional definitions

$$\left(\frac{\partial f(\underset{\sim}{x}_i;kT)}{\partial \underset{\sim}{x}}\right)^T = [f'_{i1}\ f'_{i2}] \qquad (5.4.2\text{-}3a)$$

and

$$\underset{\sim}{a}(kT) \triangleq \begin{bmatrix} a_{11}(kT) \\ a_{12}(kT) \\ a_{21}(kT) \\ a_{22}(kT) \\ \text{------} \\ b_1(kT) \\ b_2(kT) \end{bmatrix} = \begin{bmatrix} \\ \underset{\sim}{a}_p(kT) \\ \\ \text{-----} \\ \underset{\sim}{b}(kT) \end{bmatrix} \qquad (5.4.2\text{-}3b)$$

(5.3.1-7), (5.4.1-2) and (5.4.1-3) and the above equations may be used to show (dropping the temporal index for simplicity);

$$d(\underset{\sim}{x}_i) = \left[x_{i1}f'_{i1} \quad x_{i2}f'_{i1} \quad x_{i1}f'_{i2} \quad x_{i2}f'_{i2} \vdots \quad f'_{i1} \quad f'_{i2}\right] \begin{bmatrix} a_{11} \\ a_{12} \\ a_{21} \\ a_{22} \\ \text{---} \\ b_1 \\ b_2 \end{bmatrix} \tag{5.4.2-4}$$

In considering the N samples, the following matrix equation is obtained:

$$\begin{bmatrix} d(\underset{\sim}{x}_1) \\ d(\underset{\sim}{x}_2) \\ \cdot \\ \cdot \\ \cdot \\ \\ \\ d(\underset{\sim}{x}_N) \end{bmatrix} = \left[\begin{array}{cccc:cc} x_{11}f'_{11} & x_{12}f'_{11} & x_{11}f'_{12} & x_{12}f'_{12} & f'_{11} & f'_{12} \\ x_{21}f'_{21} & x_{22}f'_{21} & x_{21}f'_{22} & x_{22}f'_{22} & f'_{21} & f'_{22} \\ & & \cdot & & \cdot & \\ & & \cdot & & \cdot & \\ & & \cdot & & \cdot & \\ \\ \\ x_{N1}f'_{N1} & x_{N2}f'_{N1} & x_{N1}f'_{N2} & x_{N2}f'_{N2} & f'_{N1} & f'_{N2} \end{array}\right] \begin{bmatrix} a_{11} \\ a_{12} \\ a_{21} \\ a_{22} \\ \text{---} \\ b_1 \\ b_2 \\ \end{bmatrix} \tag{5.4.2-5}$$

Written more compactly, (5.4.2-5) yields:

$$\boxed{\underset{\sim}{d}[(k+1)T] = D(kT)\underset{\sim}{a}(kT)} \tag{5.4.2-6}$$

where $\underset{\sim}{d}$ is Nx1, D is Nx6 and $\underset{\sim}{a}$ is 6x1.

For future purposes it is also convenient to partition D, yielding

$$D = [P \vdots G] \tag{5.4.2-7}$$

where, from (5.4.2-5), G is an Nx2 matrix of unweighted spatial derivatives, and P is an Nx4 matrix of weighted spatial derivatives, all evaluated at t = kT. It is assumed that the elements of G and P are known (or calculable) and more will be said about this in Section 5.4.6.1.

5.4.3 The Basic Estimation Problem

Before making a general statement concerning (5.4.2-6), it is desirable to formulate some objectives.

Recall that $\underset{\sim}{a}(kT)$ was defined in (5.4.2-3) as:

$$\underset{\sim}{a}(kT) = \begin{bmatrix} a_{11}(kT) \\ a_{12}(kT) \\ a_{21}(kT) \\ a_{22}(kT) \\ \text{----} \\ b_1(kT) \\ b_2(kT) \end{bmatrix} = \begin{bmatrix} \\ \underset{\sim}{a}_p(kT) \\ \\ \text{----} \\ \\ \underset{\sim}{b}(kT) \\ \\ \end{bmatrix} \qquad (5.4.3\text{-}1)$$

The video image processing system may be interesting in estimating the translational vector, $\underset{\sim}{b}(kT)$, i.e., the last two elements of $\underset{\sim}{a}$, but a knowledge of the entire $\underset{\sim}{a}(kT)$ vector may also be useful, for several reasons:

(1) If the system experiences dilation, it may be highly desirable to adjust the camera zoom lens control and a good control signal could be derived from $\underset{\sim}{a}_p$; and

(2) An estimate of the entire $\underset{\sim}{a}$ vector may also be used to update the system segmentation algorithm to insure only target interior textural points are sampled (Section 5.4.5.1).

Thus there are several different approaches which could be taken at the present time:

(1) Given (5.4.2-6), generate a "good" estimate of $\underset{\sim}{a}(kT)$, with no constraints on $\Delta A(kT)$ other than those necessary to ensure the Taylor series expansion holds. This approach is covered in Section 5.4.5.1;

(2) Given (5.4.2-6), generate a "good" estimate of only $\underset{\sim}{b}(kT)$. This topic is covered in Sections 5.4.4.3 and 5.4.4.4;

(3) Given (5.4.2-6), constrain the $\Delta A(kT)$ parameters in such a way that $\underset{\sim}{a}(kT)$ is reduced to a 4x1 vector, and then generate a good estimate of this vector. This approach is covered in

Sections 5.4.4.2 and 5.4.4.3.

One feature is common to all three of these approaches, namely that each of these solutions may be formulated as a linear least-squares estimation problem leading to the computation of a matrix pseudoinverse. Therefore a summary of least-squares estimation principles is given in Appendix (3), and an overview of matrix pseudoinverse formulation techniques is given in Appendix (4).

One further point should be noted. From (5.3.1-2) and (5.4.2-1) it is obvious that the parameters of $\underset{\sim}{a}_p$ do not correspond directly to the affine transform parameters, but rather must be adjusted due to the fact that

$$\Delta A = A - I \qquad (5.4.3\text{-}2)$$

This presents no problem, however since we may simply adjust the $\underset{\sim}{a}_p$ estimates by a known amount, according to (5.4.3-2).

5.4.4.1 Direct Solution of the Normal Equations

Looking at (5.4.2-6), it is clear that one straightforward approach is to form a Markov estimate of $\underset{\sim}{a}(kT)$, by forming the pseudoinverse of D, based on a Q norm (for minimum variance), as follows:

$$\hat{\underset{\sim}{a}}(kT) = (D^T Q D)^{-1} D^T Q \underset{\sim}{d}[(k+1)T]$$

or (5.4.4.1-1)

$$\hat{\underset{\sim}{a}}(kT) = D_Q^{+} \underset{\sim}{d}[(k+1)T]$$

The most obvious objection to this approach is that the calculation of $(D^T Q D)^{-1}$ requires the inversion of a 6x6 matrix (or equivalently the solution of the six corresponding normal equations). This task is not difficult using a digital minicomputer, but may be formidable if restricted to a dedicated CCD discrete analog processor, as explained in Appendix 4.

5.4.4.2 Modified Direct Approach via Affine Transform Constraints

The previous section indicated that the general solution for $\hat{\underset{\sim}{a}}$, where A was unconstrained (except in the sense that the Taylor Series approximation must hold), required the inversion of a 6x6 matrix (or the

corresponding solution to the six normal equations). However, in the context of the previously developed model, and in light of the computational difficulty encountered otherwise, it appears advantageous to further constrain the parameters of the A matrix.

Referring to Appendix 1, it should be noted that nonzero off-diagonal terms in A, unless they represent rotation, actually correspond to a change of perspective, e.g., a slightly different viewing angle. A little physical reasoning would show that, for a 3-D (in the spatial sense) target, such a change might not only change the textural function, but also introduce new features (e.g., previously unobservable points on the target, particularly around the edges), which would not be adequately represented by the model from Section 4.3.2. This situation obviously would not occur, however, if A were constrained to cause only rotation about the optical axis, dilation, or a combination of both, i.e. if

$$A_d = \begin{bmatrix} \alpha & 0 \\ 0 & \alpha \end{bmatrix} \qquad (5.4.4.2\text{-}1a)$$

$$A_r = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \qquad (5.4.4.2\text{-}1b)$$

or

$$A_c = A_d A_r = A_r A_d = \alpha \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \qquad (5.4.4.2\text{-}1c)$$

Defining

$$\Delta A_c \triangleq \begin{bmatrix} c_1 & c_2 \\ c_2 & c_1 \end{bmatrix} = A_c - I \qquad (5.4.4.2\text{-}2)$$

(5.4.2-4) may be rewritten as:

$$d(\underset{\sim}{x}_i) = \left[(x_{i1}f'_{i1} + x_{i2}f'_{i2}) \; (x_{i1}f'_{i2} - x_{i2}f'_{i1}) \; \vdots \; f'_{i1}f'_{i2} \right] \begin{bmatrix} c_1 \\ c_2 \\ -- \\ b_1 \\ b_2 \end{bmatrix} \qquad (5.4.4.2\text{-}3)$$

It is useful to define a new vector, the constrained homogeneous affine parameter vector, $\underset{\sim}{a}_c$, as

$$\underset{\sim}{a}_c \triangleq \begin{vmatrix} c_1 \\ c_2 \end{vmatrix} \qquad (5.4.4.2\text{-}4)$$

Note that the subscript c is used here and in the following to denote "compacted" or reduced dimension vectors and matrices resulting from the previous restrictions, and that the above vector provides all the information required to compute target rotation and dilation constants.

Letting the newly-formed total affine parameter vector be denoted by $\underset{\sim}{a}$, where

$$\underset{\sim}{a} = \begin{vmatrix} \underset{\sim}{a}_c \\ -- \\ \underset{\sim}{b} \end{vmatrix} \qquad (5.4.4.2\text{-}5)$$

equation (5.4.2-5) may be reduced to

$$\underset{\sim}{d} = [P_c \vdots G]\underset{\sim}{a} = D_c\underset{\sim}{a} \qquad (5.4.4.2\text{-}6)$$

where the i^{th} row of P_c is given by

$$P_c(i^{th}\ \text{row}) = [(x_{i1}f'_{i1}+x_{i2}f'_{i2})\ (x_{i1}f'_{i2}-x_{i2}f'_{i1})] \qquad (5.4.4.2\text{-}7)$$

Thus, P_c is an Nx2 matrix of weighted spatial derivatives, and now D_c is an Nx4 matrix. The form of P_c is considerably simplified if an Nx2 matrix of sampled index values, X, is defined as:

$$X = [\underset{\sim}{x}_1 \vdots \underset{\sim}{x}_2] = [\underset{\sim}{x}_k{}^T] \qquad k = 1,2,\ldots N \qquad (5.4.4.2\text{-}8)$$

Using the definition of the Hadamard product (c.f. Rao [28], p. 11),[2] P_c

---

2) The Hadamard product of two Nxl vectors, e.g. $\underset{\sim}{e} = \underset{\sim}{c}*\underset{\sim}{d}$, where

$\underset{\sim}{c} = [c_j] \qquad j = 1,2,\ldots N$

$\underset{\sim}{d} = [d_j] \qquad j = 1,2,\ldots N$

is also a Nxl vector, and is given by

$\underset{\sim}{e} = [c_j d_j] \qquad j = 1,2,\ldots N$

Therefore the Hadamard product represents a point by point multiplication of the elements of each vector.

may be written as

$$P_c = [\underset{\sim}{x}_1 * \underset{\sim}{f}_1 + \underset{\sim}{x}_2 * \underset{\sim}{f}_2 \quad \underset{\sim}{x}_1 * \underset{\sim}{f}_2 - \underset{\sim}{x}_2 * \underset{\sim}{f}_1] = [\underset{\sim}{p}_1 \quad \underset{\sim}{p}_2] \qquad (5.4.4.2\text{-}9)$$

where * represents the Hadamard product operation.

D_c may also be put into this form:

$$D_c = \left[\begin{array}{cc|cc} \underset{\sim}{x}_1 * \underset{\sim}{f}_1 + \underset{\sim}{x}_1 * \underset{\sim}{f}_2 & \underset{\sim}{x}_1 * \underset{\sim}{f}_2 - \underset{\sim}{x}_2 * \underset{\sim}{f}_1 & \underset{\sim}{f}_1 & \underset{\sim}{f}_2 \end{array}\right] \qquad (5.4.4.2\text{-}10)$$

Thus, the problem is now: given (5.4.4.2-6), generate a "good" estimate of $\underset{\sim}{a}$ (i.e. $\underset{\sim}{a}_c$ and $\underset{\sim}{b}$). Again, the least squares solution appears promising, and as the next section shows, since the problem of forming D_c^+ now involves the inversion of only a 4x4 matrix with particular properties, a CCD implementable solution may be readily obtained.

Having constrained A in the above, it is also useful to consider the calculation of the actual affine parameters α and θ from (5.4.4.2-1c) once $\underset{\sim}{a}_c$ has been estimated. Approximating

$$\sin\theta \approx \theta$$

and

$$\cos\theta \approx 1-\theta^2/2 \qquad (5.4.4.2\text{-}11)$$

and, recalling the definition of a_c from (5.4.4.2-4), it is seen that

$$c_2 = \alpha\theta$$

$$c_1 = \alpha(1-\theta^2/2)-1 \qquad (5.4.4.2\text{-}12)$$

and therefore

$$\hat{c}_1 = c_1+1 = \alpha(1-\theta^2/2) \qquad (5.4.4.2\text{-}13)$$

An estimate of the target rotation about its centroid, θ, may then be obtained, since

$$\frac{c_2}{c_1} = \frac{\alpha\theta}{\alpha(1-\theta^2/2)} \approx \theta(1+\theta^2/2) = \theta+\theta^3/2 \approx \theta \qquad (5.4.4.2\text{-}14a)$$

or more simply

$$\boxed{\theta = c_2/c_1} \qquad (5.4.4.2\text{-}14b)$$

which obviously holds since θ was assumed small. Using (5.4.4.2-13) and

(5.4.4.2-14), the target magnification parameter, α, may then be obtained, from

$$\alpha = \frac{c_1}{(1-\theta^2/2)} \simeq \frac{c_1}{1-\frac{(c_2/c_1)^2}{2}} \simeq c_1\left(1+\frac{(c_2/c_1)^2}{2}\right) \quad (5.4.4.2\text{-}15)$$

or

$$\boxed{\alpha = c_1 + c_2^2/2c_1} \quad (5.4.4.2\text{-}15b)$$

Thus, a knowledge of $\underset{\sim}{a}_c$ uniquely determines α and θ.

5.4.4.3 Implementation of the Modified Direct Approach Using Cline's Theorem

Recall the estimation problem, for suitable constrained A, is, to estimate $\underset{\sim}{a}$, given

$$\underset{\sim}{d} = D_c\underset{\sim}{a} \quad (5.4.4.3\text{-}1)$$

where

$$D_c = [\underset{\sim}{x}_1*\underset{\sim}{f}_1+\underset{\sim}{x}_2*\underset{\sim}{f}_2 \quad \underset{\sim}{x}_1*\underset{\sim}{f}_2-\underset{\sim}{x}_2*\underset{\sim}{f}_1 \quad \underset{\sim}{f}_1 \quad \underset{\sim}{f}_1] \quad (5.4.4.3\text{-}2a)$$

$$\underset{\sim}{a} = \begin{bmatrix}\underset{\sim}{a}_c \\ \underset{\sim}{b}\end{bmatrix}$$

We may form the L.S. estimate of a assuming D_c has full column rank, as:

$$\hat{\underset{\sim}{a}} = D_c^{\dagger}\underset{\sim}{d} \quad (5.4.4.3\text{-}3)$$

(or equivalently solving the four normal equations:

$$D_c^T\underset{\sim}{d} = D_c^TD_c\underset{\sim}{a} \quad (5.4.4.3\text{-}4)$$

for $\underset{\sim}{a}$, however, for reasons cited in Appendix 4, the preferable CCD implementable solution is to form $(D_c)^{\dagger}$). Also note that while the remaining analysis in this section and the following assumes the formulation of a matrix pseudoinverse based upon an identity matrix norm for simplicity, this work is extendable to cases where it is desirable to formulate this pseudoinverse based upon a general matrix norm, Q.

A rigorous mathematical derivation of the conditions necessary for D_c to have full column rank appears quite difficult. Therefore, a simple argument based upon intuitive reasoning is presented, and it is shown

that for the class of textured targets expected, this matrix should always have full column rank.

One way to show that D_c has full column rank, i.e., that the columns of D_c are a set of four linearly independent vectors, is to compute the Gramian of this set of vectors (c.f. 29 , pp. 217-218) and then show that this function is nonzero. This approach has limited analytical utility, however, since exact numerical values of the vectors comprising D_c are not known a priori. (It should be noted, however, that the Gramian calculation was incorporated into the simulation programs described in the following chapter.)

A more reasonable approach is to recall the definition of vector linear independence, i.e. if D_c has full column rank, no column of D_c may be expressed as a linear combination of the remaining three. It is relatively easy to reason that $\underset{\sim}{f}_1$ and $\underset{\sim}{f}_2$ are independent, recalling from (5.4.2-3) that the k^{th}components of each of these vectors are

$$f'_{k1} = \frac{\partial f(x_k)}{\partial x_1}$$

and (5.4.4.3-5)

$$f'_{k2} = \frac{\partial f(x_k)}{\partial x_2}$$

where $\{x_k\}$; k = 1,2. . .N is the set of all sample points. For any reasonably complex textured object (Section 5.2) it is expected that vertical and horizontal target textures are independent, therefore, from (5.4.4.3-5) it is unlikely that $\underset{\sim}{f}_1$ and $\underset{\sim}{f}_2$ are dependent. In fact, in considering targets of practical interest (e.g., aircraft, ships, land-based vehicles, etc.) it is likely that the components of $\underset{\sim}{f}_1$ and $\underset{\sim}{f}_2$ will generally be nonzero, and also "rich" in variation, due to the inherent texture present in these targets.

It is now necessary to show that the remaining two columns of D_c are also linearly independent (of each other and $\underset{\sim}{f}_1$ and $\underset{\sim}{f}_2$). Recall that these columns are obtained by a linear combination of the Hadamard product vectors $\underset{\sim}{x}_1 * \underset{\sim}{f}_1$, $\underset{\sim}{x}_2 * \underset{\sim}{f}_2$, $\underset{\sim}{x}_1 * \underset{\sim}{f}_2$ and $\underset{\sim}{x}_1 * \underset{\sim}{f}_2$, as shown in (5.4.4.3-2a). It is useful to consider the structure of $\underset{\sim}{x}_1$ and $\underset{\sim}{x}_2$, where these vectors are defined in (5.4.4.2-8). Figure 7 illustrates the effect of a raster scan

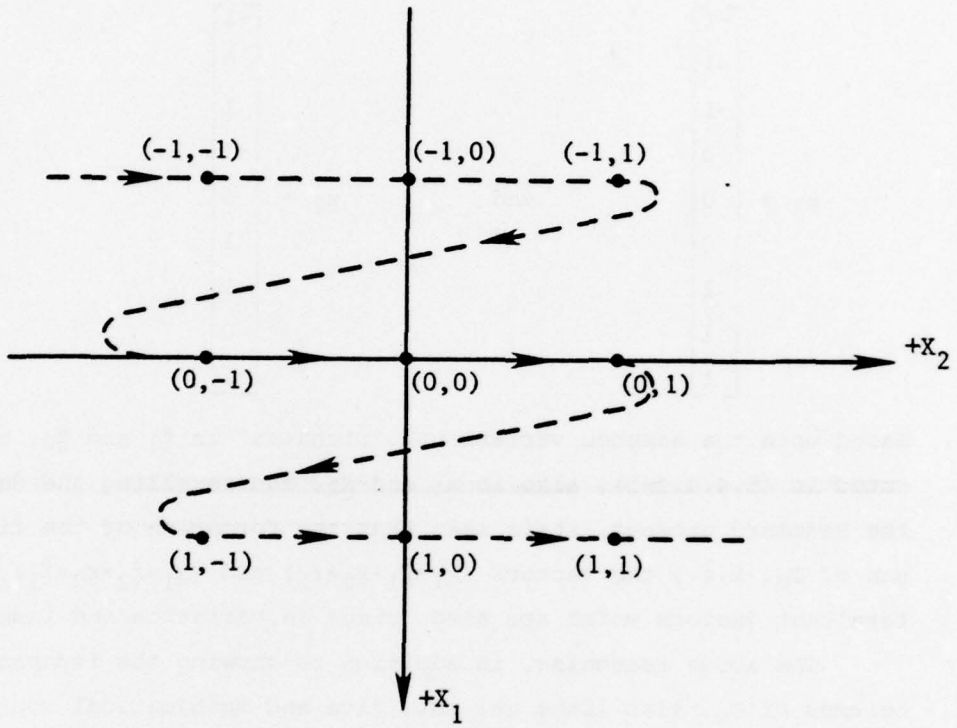


Figure 7 Example of Spatial Sampling and the Formation of $\underset{\sim}{x}_1$ and $\underset{\sim}{x}_2$.

on a 3x3 square sampling lattice with a unity spatial sampling interval and the origin at the center of this lattice. With this strategy $\underset{\sim}{x}_1$ and $\underset{\sim}{x}_2$ would be of the form:

$$\underset{\sim}{x}_1 = \begin{bmatrix} -1 \\ -1 \\ -1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad \text{and} \quad \underset{\sim}{x}_2 = \begin{bmatrix} -1 \\ 0 \\ 1 \\ -1 \\ 0 \\ 1 \\ -1 \\ 0 \\ 1 \end{bmatrix} \qquad (5.4.4.3\text{-}6)$$

Based upon the assumed variational "richness" in $\underset{\sim}{f}_1$ and $\underset{\sim}{f}_2$, and, as indicated in (5.4.4.3-6), also in $\underset{\sim}{x}_1$ and $\underset{\sim}{x}_2$, and recalling the definition of the Hadamard product, it is seen that the formation of the first two column of D_c, i.e., the vectors $(\underset{\sim}{x}_1 * \underset{\sim}{f}_1 + \underset{\sim}{x}_2 * \underset{\sim}{f}_2)$ and $(\underset{\sim}{x}_1 * \underset{\sim}{f}_2 - \underset{\sim}{x}_2 * \underset{\sim}{f}_1)$, should yield resultant vectors which are also "rich" in variation and independent.

The above reasoning, in addition to showing the independence of the columns of D_c, also links the intuitive and mathematical concepts of affine paraparameter "estimatibility" or target perturbation "observability" or "detectability." For example, in many sample cases where D_c has column rank less than 4, a human observer would also be unable to determine certain target perturbations. The most obvious example is the case wherein $\underset{\sim}{f}_1 = \underset{\sim}{f}_2 = \underset{\sim}{0}$, (i.e., the target has no texture, or equivalently, the target interior is constant intensity throughout). In this case $D_c = [0]$, and using only information from the target interior, it would be impossible for a human observer to detect any target perturbation. Another somewhat more realistic example is the case wherein the target is circularly symmetric. In this case the column rank of D_c is less than or equal to 3, and clearly, using only target interior information, a human observer could not detect target rotation, due to the symmetry of the target.

Finally, it should be noted that the simulation described in Chapter VII empirically confirmed the preceding reasoning. Even in cases of very slightly textured targets it was found that the Gramian of the four vectors comprising D_C was nonzero, in fact, it was never close to zero. It is felt that the only practical case in which the question of column rank could become significant is when the target is partitioned into smaller portions consisting of a small number of textural samples, as in the case of Gestalt Segmentation, described in Section 5.4.5.2. In this case it may be possible for the target to regionally lack texture, and therefore TSVIP estimates in these regions may be unreliable.

Thus, if D_C has full column rank, one way to form $D_C^\dagger$ is to take advantage of the preceding partitioning and apply Clines theorem (Appendix 4):

$$D_C^\dagger = [P_C \vdots G]^\dagger = \begin{bmatrix} P_C^\dagger - P_C^\dagger G C^\dagger - P_C^\dagger G (I - C^\dagger C) K G^T (P_C^\dagger)^T P_C^\dagger (I - G C^\dagger) \\ C^\dagger + (I - C^\dagger C) K G^\dagger (P_C^\dagger)^T P_C^\dagger (I - G C^\dagger) \end{bmatrix} \quad (5.4.4.3\text{-}7)$$

where
$$C = (I - P_C P_C^\dagger) G \quad (5.4.4.3\text{-}8)$$

and
$$K = [I + (I - C^\dagger C) G^T (P_C^\dagger)^T P_C G (I - C^\dagger C)]^{-1} \quad (5.4.4.3\text{-}9)$$

Due to their apparent complexity, little (if anything) is to be gained from the implementation of equations 5.4.4.3-7 through 5.4.4.3-9 directly. However, as reasoned in the following, C has full column rank, so that

$$C^\dagger = (C^T C)^{-1} C^T. \quad (5.4.4.3\text{-}10a)$$

$$C^\dagger C = (C^T C)^{-1} C^T C = I \quad (5.4.4.3\text{-}10b)$$

and (5.4.4.3-7) reduces to

$$[P_C \vdots G]^\dagger = \begin{bmatrix} P_C^\dagger - P_C^\dagger G C^\dagger \\ C^\dagger \end{bmatrix} \quad (5.4.4.3\text{-}11)$$

Thus, the task of estimating $\underset{\sim}{a}$ may be accomplished via a CCD-implementable processor, which only uses inner product type operations (and the inversion of 2x2 matrices, which is easily accomplished in closed form). Recall (5.4.4.3-8) and the fact that D_C has full column rank. Then, from (5.4.4.2-9) and (5.4.4.2-10) P_C has full column rank, as does G, and

$$P_c^{\dagger} = (P_c^T P_c)^{-1} P_c^T \qquad (5.4.4.3\text{-}12)$$

so

$$C = (I - P_c(P_c(P_c^T P_c)^{-1} P_c^T)G \qquad (5.4.4.3\text{-}13)$$

Now

$$\begin{aligned} C^T C &= G^T(I - P_c(P_c^T P_c)^{-1} P_c^T)^T (I - P_c(P_c^T P_c)^{-1} P_c^T)G \\ &= G^T(I - P_c(P_c^T P_c)^{-1} P_c^T)(I - P_c(P_c^T P_c)^{-1} P_c^T)G \qquad (5.4.4.3\text{-}14) \\ &= G^T(I - 2P_c(P_c P_c)^{-1} P_c T + P_c(P_c^T P_c)^{-1} P_c^T P_c (P_c^T P_c)^{-1} P_c^T)G \\ &= G^T(I - P_c(P_c^T P_c)^{-1} P_c^T)G \end{aligned}$$

or

$$C^T C = G^T G - G^T P_c (P_c^T P_c)^{-1} P_c^T G \qquad (5.4.4.3\text{-}15)$$

and if C has full column rank, $(C^T C)$ will be an invertible 2x2 matrix of rank 2 (c.f. Cline [30]).

It appears difficult to prove that the matrix described on the left hand side hand side of (5.4.4.3-15) has an inverse. An alternate and somewhat devious approach is to modify the approach of [31] in forming the inverse of partitioned (square) matrices. A brief digression will show this proof. Recall

$$D_c = [P_c \vdots G] \qquad (5.4.4.3\text{-}16)$$

and that D_c (and consequently P_c and G) have full column rank. Therefore $(D_c^T D_c)^{-1}$ exists and is an invertable 4x4 matrix.

Expanding

$$(D_c^T D_c) = \left[\begin{array}{c|c} P_c^T P_c & P_c^T G \\ \hline G^T P_c & G^T G \end{array}\right] \qquad (5.4.3.3\text{-}17)$$

or, for simplicity,

$$(D_c^T D_c) = \left[\begin{array}{c|c} \alpha_{11} & \alpha_{12} \\ \hline \alpha_{21} & \alpha_{22} \end{array}\right] \qquad (5.4.4.3\text{-}18)$$

Clearly α_{11}^{-1} and α_{22}^{-1} exist due to the fact that P_c and G each have full column rank. Thus for any linear set of equations of the form

$$(D_c^T D_c)\,\underset{\sim}{\xi} = \underset{\sim}{y} \qquad (5.4.4.3\text{-}19)$$

there exists a unique solution vector $\underset{\sim}{\xi}$.

Partition $\underset{\sim}{\xi}$ and $\underset{\sim}{x}$ as follows:

$$\underset{\sim}{\xi} = \begin{bmatrix} \underset{\sim}{\xi}_1 \\ -- \\ \underset{\sim}{\xi}_2 \end{bmatrix} \qquad (5.4.4.3\text{-}20)$$

$$\underset{\sim}{y} = \begin{bmatrix} \underset{\sim}{y}_1 \\ -- \\ \underset{\sim}{y}_2 \end{bmatrix} \qquad (5.4.4.3\text{-}21)$$

Clearly $\underset{\sim}{\xi}_1$ and $\underset{\sim}{\xi}_2$ are then also unique. Rewriting (5.4.4.3-19) using (5.4.4.3-18) using (5.4.4.3-20) and (5.4.4.3-21) yields

$$\begin{aligned} \alpha_{11}\underset{\sim}{\xi}_1 + \alpha_{12}\underset{\sim}{\xi}_2 &= \underset{\sim}{y}_1 \\ \alpha_{21}\underset{\sim}{\xi}_1 + \alpha_{22}\underset{\sim}{\xi}_2 &= \underset{\sim}{y}_2 \end{aligned} \qquad (5.4.4.3\text{-}22)$$

Multiplying the first equation by α_{11}^{-1} and simplifying yields

$$\underset{\sim}{\xi}_1 = \alpha_{11}^{-1}\underset{\sim}{y}_1 - \alpha_{11}^{-1}\alpha_{12}\underset{\sim}{\xi}_2 \qquad (5.4.4.3\text{-}23)$$

Substitution into the second equation yields

$$\alpha_{21}\alpha_{11}^{-1}\underset{\sim}{y}_1 - \alpha_{21}\alpha_{11}^{-1}\alpha_{12}\underset{\sim}{\xi}_2 + \alpha_{22}\underset{\sim}{\xi}_2 = \underset{\sim}{y}_2$$

or

$$(\alpha_{22}-\alpha_{21}\alpha_{11}^{-1}\alpha_{12})\,\underset{\sim}{\xi}_2 = \underset{\sim}{y}_2 - \alpha_{21}\alpha_{11}^{-1}\underset{\sim}{y}_1 \qquad (5.4.4.3\text{-}24)$$

Since ξ_2 is the unique solution to this matrix equation, it is apparent that the nullspace of the 2x2 matrix W, where

$$W = (\alpha_{22}-\alpha_{21}\alpha_{11}^{-1}\alpha_{12})$$

is empty, thus, this matrix has full rank, (=2). Recalling the definitions of the α's in (5.4.4.3-18), it therefore is proved that

$$G^T G - G^T P_c (P_c^T P_c)^{-1} P_c^T G = C^T C = W$$

is nonsingular, therefore, from (5.4.4.3-15), C has full column rank.

Since $(C^TC)^{-1}$ exists, (5.4.4.3-11) may be used to form D_C . Thus, to estimate $\underset{\sim}{a}$ by partitioning D_C as in (5.4.2-7), (5.4.4.3-11) may be used, i.e.,

$$\hat{\underset{\sim}{a}} = \begin{bmatrix} P_C^\dagger - P_C^\dagger G C^\dagger \\ \hline C^\dagger \end{bmatrix} \underset{\sim}{d} \qquad (5.4.4.3\text{-}25)$$

or, in fully expanded form

$$\hat{\underset{\sim}{a}} = \begin{bmatrix} (P_C^TP_C)^{-1}P_C^T\left[I - G(G^TG - G^TP_C(P_C^TP_C)^{-1}P_C^TG)^{-1}(I - P_C(P_C^TP_C)^{-1}P_C^TG)\right] \\ (G^TG - G^TP_C(P_C^TP_C)^{-1}P_C^TG)^{-1}(I - P_C(P_C^TP_C)^{-1}P_C^TG) \end{bmatrix} \underset{\sim}{d} \qquad (5.4.4.3\text{-}26)$$

For purposes of clarity, (5.4.4.3-25) will be used to show the simple CCD implementation. It is possible and indeed advantageous to estimate $\underset{\sim}{a}$ using parallel processing since, from (5.4.4.2-5)

$$\hat{\underset{\sim}{a}} = \begin{bmatrix} \hat{\underset{\sim}{a}}_C \\ \hline \underset{\sim}{b} \end{bmatrix}$$

and therefore from (5.4.4.3-25)

$$\hat{\underset{\sim}{a}}_C = (P_C^\dagger - P_C^\dagger G C^\dagger)\underset{\sim}{d} \qquad (5.4.4.3\text{-}27)$$

and

$$\hat{\underset{\sim}{b}} = C^\dagger \underset{\sim}{d}$$

Equations (5.4.4.3-27) and (5.4.4.3-28) are hereafter referred to as the TSVIP algorithm, and comprise the heart of the video tracking process.

If only $\hat{\underset{\sim}{b}}$ is desired, the problem is greatly simplified, since C may be formed using only inner product operations, sums, and the closed form solution for a 2x2 matrix, which only requires the capability to perform analog division. Notice that even if only $\hat{\underset{\sim}{b}}$ is desired, this estimate is obtained in (5.4.4.3-28) by also considering the effects of $\underset{\sim}{a}_C$, even though the latter vector may not also be estimated. It is difficult to overestimate the importance of this result in view of past efforts in the pattern recognition area. This is in direct contrast to the approach taken in the next section, where $\underset{\sim}{a}_C$ effects are modeled as "noise."

At this point a brief digression is in order. The preceding analysis, particularly for the first-time reader, may appear somewhat complicated and difficult to interpret in a physical or intuitive sense. Therefore, the interested reader is referred to Appendix 6, which gives a simple physical interpretation of the TSVIP algorithm approach (for the 1-D case), and presents several detailed numerical examples.

Even though the capability of analog division exists, in order to reduce implementation complexity and achieve high speed processing it may be desirable to eliminate or minimize the need for this operation (c.f. [32], p. 279). As the following analysis shows, there are ways to accomplish this.

While it is not the purpose of this work to consider in detail the various potential types of CCD architectures capable of implementing (5.4.4.3-27) and (5.4.4.3-28), an example is included to establish the feasibility of such a design.

In order to implement (5.4.4.3-27) and (5.4.4.3-28) the CCD processor must have some means of manipulating negative numbers, in particular, a four-quadrant multiplication and an addition capability. While presently no such device exists, it is felt that CCD technology has this potential. For example, fixed tap weights already exist (c.f. [6], p. 219), and it appears that by replacing these fixed taps with variable conductance NMOS transistor weighting devices (c.f. [5], p. 37), the capability of both positive and negative variable tap weights could be obtained. In addition, methods are already available for injecting and detecting bipolar signals (c.f. [6], Ch. III, Section C), and the combination of these two technologies could result in CCD devices with the aforementioned desirable capabilities. Therefore, the availability of such devices in the future is assumed.

Assuming the matrix G and the vector $\underset{\sim}{d}$ already exist (the estimation of G is discussed in Section 5.4.6.1), the first problem addressed is the formation of P_C. Recalling the definition of P_C from (5.4.4.2-9), it is easy to see that $\underset{\sim}{p}_1$ and $\underset{\sim}{p}_2$ are easily formed via tapped CCD registers, where the register contents are the vectors $\underset{\sim}{f}_1$ and $\underset{\sim}{f}_2$ and the tap weights

are x_1 and x_2.

Thus, in the implementation of (5.4.4.3-28), the first step could be the formation of $P_c^{\dagger}$. Defining

$$v_1 \triangleq |P_c^T P_c| \tag{5.4.4.3-29}$$

and recalling

$$P_c^{\dagger} = (P_c^T P_c)^{-1} P_c^T \tag{5.4.4.3-30}$$

one possible implementation is as follows:

Step 1: Form $(P_c^T P_c)$ and v_1. Clearly $P_c^T P_c$ is a 2x2 symmetric matrix, which is easily formed using inner product operations.

$$(P_c^T P_c) = \left[\begin{array}{cc} p_1^T p_1 & p_1^T p_2 \\ \hline p_2^T p_1 & p_2^T p_2 \end{array}\right] \tag{5.4.4.3-31}$$

It follows that

$$(P_c^T P_c)^{-1} = \frac{1}{v_1} \left[\begin{array}{cc} p_2^T p_2 & -p_1^T p_2 \\ \hline -p_1^T p_2 & p_1^T p_1 \end{array}\right] \tag{5.4.4.3-32}$$

thus from (5.4.4.3-30)

$$v_1 P_c^{\dagger} = \left[\begin{array}{c} (p_2^T p_2) p_1^T - (p_1^T p_2) p_2^T \\ \hline -(p_1^T p_2) p_1^T + (p_1^T p_1) p_2^T \end{array}\right] \tag{5.4.4.3-33}$$

Equation (5.4.4.3-33) describes the architecture of a CCD based Nx2 "pseudoinverter," which is clearly feasible with the previously assumed CCD building blocks (neglecting temporarily the v term).

Step 2: Form $P_c P_c^{\dagger}$

The formation of $v_1 P_c P_c^{\dagger}$ is as follows:

$$v_1 P_c P_c^{\dagger} = [p_1 \vdots p_2] \left[\begin{array}{c} (p_2^T p_2) p_1^T - (p_1^T p_2) p_2^T \\ \hline -(p_1^T p_2) p_1^T + (p_1^T p_1) p_2^T \end{array}\right] \tag{5.4.4.3-34}$$

It is again obvious that the indicated matrix multiplication is CCD achievable, since each element of the product matrix is the result of an inner product operation.

Step 3: Consider the formation of $(I-P_cP_c^{\dagger})$

Defining

$$M \triangleq -\left[\underset{\sim}{p}_1 \vdots \underset{\sim}{p}_2\right]\begin{bmatrix}(\underset{\sim}{p}_2^T\underset{\sim}{p}_2)\underset{\sim}{p}_1^T-(\underset{\sim}{p}_1^T\underset{\sim}{p}_2)\underset{\sim}{p}_2^T \\ \hline -(\underset{\sim}{p}_1^T\underset{\sim}{p}_2)\underset{\sim}{p}_1^T+(\underset{\sim}{p}_1^T\underset{\sim}{p}_1)\underset{\sim}{p}_2^T\end{bmatrix} \qquad (5.4.4.3\text{-}35)$$

it is easy to see that the formation of M is again a CCD achievable operation, and

$$M = -v_1P_cP_c^{\dagger} \qquad (5.4.4.3\text{-}36)$$

If v_1 is added to each diagonal element of M (e.g. have this "bias" built into the architecture), the result, designated M_{+1}, will be:

$$M_{+1} = V_1(I-P_cP_c^{\dagger}) \qquad (5.4.4.3\text{-}37)$$

Step 4: Form $M_{+1}G$. This step again only requires matrix multiplication. The result is

$$M_{+1}G = V_1(I-P_cP_c^{\dagger})G \qquad (5.4.4.3\text{-}38)$$

$$= v_1C$$

Let $$v_2 = \left|[M_{+1}G]^T[M_{+1}G]\right|. \qquad (5.4.4.3\text{-}39a)$$

Step 5: Using the architecture of (5.4.4.3-33), form

$$\begin{aligned} v_2(M_{+1}G)^{\dagger} &= v_2(v_1C)^{\dagger} \\ &= v_2(v_1C^Tv_1C)^{-1}v_1C^T \qquad (5.4.4.3\text{-}39b) \\ &= \frac{v_2}{v_1}(C^TC)^{-1}C^T \\ &= (\frac{v_2}{v_1})C^{\dagger} \end{aligned}$$

Step 6: Form

$$\begin{aligned} v_2(M_{+1}G)^{\dagger}\underset{\sim}{d} &= \frac{v_2}{v_1}\,C^{\dagger}\underset{\sim}{d} \\ &= (\frac{v_2}{v_1})\hat{\underset{\sim}{b}} \end{aligned} \qquad (5.4.4.3\text{-}40)$$

Thus, using primarily CCD-implementable inner product operations, a scaled estimate of the affine translation vector $\hat{\underset{\sim}{b}}$ (with the scale factor, $(\frac{v_2}{v_1})$

calculable from (5.4.4.3-29) and (5.4.4.3-39a)) is obtained. As shown in Section 5.4.7 this form is acceptable, since $\hat{\underset{\sim}{b}}$ will be again scaled by the overall control algorithm (part 4 of Figure 2) before being used for camera mount controls. Note also that v_1 and v_2 were considered separately throughout. This is advantageous for two reasons:

(1) The problem of analog division is circumvented.

(2) Should P_c not have full column rank, $v_1 = 0$, and a "flag" to signal the impossibility of matrix inversion is available. In addition if v_1 is close to 0, a problem with numerical sensitivity in the algorithm may occur, and v_1 might also be used to identify this situation.

The procedure outlined above for implementation of (5.4.4.3-28) is only provided as an example, and no claims as to uniqueness or optimality are made. The implementation of (5.4.4.3-27) may be shown in a similar manner. For implementation using one analog divider, see Chapter VI.

At this point it is useful to consider the potential processing speed of the aforementioned CCD-based implementation. Under the following set of assumptions, i.e.,

(1) The system is in "steady-state," i.e., initialization of parameters and initial segmentation have been accomplished;

(2) The hardware necessary to enable sampling of all desired scene pixels simultaneously is available;

(3) Information necessary for segmentation and spatial derivative data are available from the previous scene; and

(4) A CCD-implementable vector inner product operation may be accomplished in two or less clock cycles, once the register contents and tap weights are available.

the estimates shown in Table 1 may be obtained. As an example, with a 1 MHz system clock and assuming 36 clock cycles to implement the TSVIP algorithm, the total processing time for this algorithm would be 36 μseconds.

The above estimates are only intended to show the potential speed of the CCD-implementable TSVIP algorithm, and should not be taken as rigid

	Operation	Equation	Estimated Number of Clock Cycles
(1)	Input $p(\underset{\sim}{i}(k+1)T)$	----	2
(2)	Form $d(\underset{\sim}{i},(k+1)T)$	(5.4.1-1)	4
(3)	Form $\underset{\sim}{d}$ $(k+)T$	(5.4.1-2)	2
(4)	Form P_c and D_c, using $\underset{\sim}{x}_1$ and $\underset{\sim}{x}_2$ from (2) and $G(kT)$	(5.4.4.2-9)	4
(5)	Form $(P_c{}^T P_c)^{-1}$ and v_1	(5.4.4.3-31) (5.4.4.3-29) (5.4.4.3-32)	3
(6)	Form $v_1 P_c{}^\dagger$	(5.4.4.3-33)	3
(7)	Form $v_1 P_c P_c{}^\dagger$ and M_{+1}	(5.4.4.3-34) (5.4.4.3-37)	3
(8)	Form $M_{+1}G$	(5.4.4.3-38)	2
(9)	Form $(\frac{1}{v_1})C^\dagger$	(5.4.4.3-39)	3
(10)	Form $(\frac{1}{v_1})\hat{\underset{\sim}{b}}$	(5.4.4.3-40)	2
(11)	Form $\hat{\underset{\sim}{a}}_c$	(5.4.4.3-27)	4
(12)	Scale $\hat{\underset{\sim}{b}}(kT)$ for $\underset{\sim}{\theta}(kT)$	(5.4.7-2)	4
			36 cycles

Table 1

Processing Time Estimate for CCD Implementation of TSVIP Algorithm

design specifications. The ultimate tracking processor speed will be affected by system clock frequency, segmentation procedures, amount of system pipelining, filterin of TSVIP estimates, scene preprocessing, scene sampling, exact algorithm implementation (the preceding implementation is clearly inefficient), camera mount dynamic restrictions, etc. The implementation of the modified direct approach using Cline's theorem by means of CCD devices is discussed in Chapter VI, in which specific configurations of formulas (5.4.4.3-27) and (5.4.4.3-28) are proposed.

Finally, a note should be made concerning scene dimensions, i.e., the total number of pixels processed at each temporal sampling instant in an NXM scene. Since the TSVIP algorithm implementation is temporally efficient, but spatially complex, processing time should be relatively insensitive to scene dimensions, provided assumption (2) above is valid.

5.4.4.4 Incomplete Parameter Vector Estimation Approach

As mentioned in the previous section, the estimation of the homogeneous affine parameters ($\underset{\sim}{a}_p$ or $\underset{\sim}{a}_c$) may be of little interest, especially in cases where dilation may not occur. In this case the method of Section 5.4.4.3 is somewhat simplified, however, an even simpler implementation results if the effects of these parameters are modelled as "noise." As shown below, however, the quality of the estimates obtained via this approach is questionable.

Recall (5.4.2-6), which may be written using (5.4.2-7) as

$$\underset{\sim}{d} = P\underset{\sim}{a}_p + G\underset{\sim}{b} \qquad (5.4.4.4\text{-}1)$$

Clearly, if $\underset{\sim}{a}_p = \underset{\sim}{0}$ and $\underset{\sim}{b} = \underset{\sim}{0}$, then $\underset{\sim}{d} = \underset{\sim}{0}$. Assuming the estimation of $\underset{\sim}{a}_p$ is not required, model the term $P\underset{\sim}{a}_p$ as "noise," i.e.,

$$\underset{\sim}{d} = G\underset{\sim}{b} + \underset{\sim}{\eta} \qquad (5.4.4.4\text{-}2)$$

This approach is similar to the problem of system identification with an incomplete parameter vector (Eykoff [33], p. 199). Thus, the L.S. estimate of $\hat{\underset{\sim}{b}}$, using a Q norm, would be obtained as:

$$\begin{aligned} \underset{\sim}{b} &= G_Q^{+}\underset{\sim}{d} \\ &= (G^T Q G)^{-1} G^T Q \underset{\sim}{d} \end{aligned} \qquad (5.4.4.4\text{-}3)$$

which is much simpler than (5.4.4.3-28), since only a single 2x2 matrix inversion and a few simple inner product operations are required. Using (5.4.4.4-1), however, it is seen that

$$\begin{aligned}\hat{\underset{\sim}{b}} &= (G^T Q G)^{-1} G^T Q G \underset{\sim}{b} + (G^T Q G)^{-1} G^T Q P \underset{\sim}{a}_p \\ &= \underset{\sim}{b} + G_Q^{\dagger} P \underset{\sim}{a}_p\end{aligned} \qquad (5.4.4.4\text{-}4)$$

and therefore the second term in the above equation represents an error in the $\underset{\sim}{b}$ estimate. In fact, unless

$$E\{G_Q^{\dagger} P \underset{\sim}{a}_p\} = \underset{\sim}{0} \qquad (5.4.4.4\text{-}5)$$

this estimate will be biased, and empirical evidence (Chapter VI) suggests there is no reason to assume that (5.4.4.4-5) holds. Furthermore, in order to obtain a "good" estimate according to criteria (1)-(3) in Appendix (3), it would be necessary to compute

$$\text{Cov}\{\underset{\sim}{\eta}\} = E\{(P\underset{\sim}{a}_p)(P\underset{\sim}{a}_p)^T\} \triangleq N \qquad (5.4.4.4\text{-}6)$$

and then choose $Q = N^{-1}$ (minimum variance) in (5.4.4.4-3). This is certainly not easy to do a priori.

5.4.5 Segmentation

As described in Section 2.2, the tracking computer must be able to separate the target from the background, i.e., segment the scene at each temporal sampling instant. For reliable TSVIP estimates, Section 5.3 indicated that only points on the target interior should be used. The discussion of this system requirement has been postponed until after the development of the TSVIP algorithm, since this algorithm may be used in a novel "bootstrap" fashion to aid segmentation.

The automation of the segmentation capability inherent in the human visual system is perhaps the most difficult tracking computer algorithm development task. As noticed by Rosenfeld [1]: "It should be emphasized that there is no single standard approach to segmentation...the perceptual process involved in segmentation of a scene by the human visual system, e.g., the Gestalt laws of organization, are not yet well understood...." Despite this difficulty, numerous segmentation approaches have been devel-

oped. Some of the more popular approaches are discussed in Section 4.4. The varying degrees of success achievable with these algorithms is apparently a function of the assumed picture model, and the complexity of the scenes considered in this work as well as the real-time operational requirement generally rule out the use of these simple approaches.

In developing segmentation algorithms for the present system, a logical guideline is to use a segmentation technique which is suitable for the type or class of scene under consideration. Two general techniques which appear promising are described in the following. The first uses a statistical classifier with temporal updating based upon the TSVIP algorithm. The second makes unprecedented use of unsegmented scene TSVIP estimates and a Gestalt law.

5.4.5.1 Statistical Segmentation with Updating Using TSVIP Estimates

Before discussing these algorithms in detail, it is useful to define the concept of a window, segmentation, or T/B separation function.

As mentioned in Section 5.3.1, it is desired that the sampled picture data points, $\underset{\sim}{x}_i$; i = 1,2. . .N, used by the previously developed algorithm, be chosen strictly from the target interior, i.e., they represent $\underset{\sim}{f}(\underset{\sim}{x}_i;kT)$ since $\lambda(\underset{\sim}{x}_i;kT) = 1$. Since it is only these interior points which are to be considered, some guidance is needed in the sampling process, and this may be achieved using a 3-D window function, $w(\underset{\sim}{i};kT)$, which essentially enables the discrete-analog processor to "see" the scene data through an aperture or window, with this window chosen such that only target interior points are "visible." Thus, we may define the architecture of a 2-D "scene sample enable register," which holds the values of $w(\underset{\sim}{i};kT)$ for all permissible values of $\underset{\sim}{i}$, and together with control and logic circuitry, enable the following decision:

> $\underset{\sim}{i}$ is an acceptable sample point if $w(\underset{\sim}{i};kT) = 1$,
> otherwise, $p(\underset{\sim}{i};kT)$, for this value of $\underset{\sim}{i}$, should
> not be used by the TSVIP algorithm.

Obviously, if $\lambda(\underset{\sim}{i};kT)$ were known, an obvious choice for $w(\underset{\sim}{i};kT)$ to enable estimation of $\underset{\sim}{a}(kT)$ would be to choose

$$w(\underset{\sim}{i};kT) = w(\underset{\sim}{i};(k+1)T) = \begin{cases} 1 \forall \underset{\sim}{i} \text{ where } \lambda(\underset{\sim}{i};kT) = \lambda(\underset{\sim}{i};(k+1)T) = 1 \\ 0 \text{ elsewhere} \end{cases}$$

However, since λ is unknown, $w(i;kT)$ must be estimated.

Note that the problem of estimating $w(i;kT)$ is analogous to the T/B separation problem, and a number of well-known image processing segmentation approaches may be employed. For example, the missile tracking system of Flachs, et al. [18-24] assumes the target and background pixel intensities originate from two different pdf's, utilizes a "window-frame" approach to estimate these pdf's, and then classifies on a pixel-by-pixel basis using a Bayesian test. Other well-known segmentation approaches, for example, texture edge detection, could also be employed.

An important point should be made concerning the linking of the segmentation process and the TSVIP algorithm. It is expected that conventional T/B separation will be a relatively time-consuming process. This is primarily due to the fact that the Flachs approach, as well as many of the standard statistical approaches, must re-segment the entire scene at each sampling interval, thus the processing time allocated for segmentation purposes will remain constant.

The TSVIP algorithm may, however, possess a clear advantage over these conventional approaches. Once the initial window function (i.e., $w(\underset{\sim}{i};0)$) has been determined (perhaps using a conventional segmentation algorithm or a human operation) the TSVIP estimates may be used to update this window, since target translation, rotation and dilation estimates, i.e., $\hat{\underset{\sim}{b}}$, α, and Θ, will be available at $t = kT$. Thus, using the TSVIP algorithm, the segmentation process, in particular, the formation of $w(\underset{\sim}{i};(k+1)T)$ for the scene at $t = (k+1)T$ may be accomplished using the estimate of $\underset{\sim}{a}(kT)$. Therefore, as in the derivative estimation process, the "pipelining" ability of the TSVIP approach is apparent.

The TSVIP algorithm may therefore be combined with the aforementioned segmentation approach to form the overall system shown in Figure 8. Note that this processor contains the necessary algorithms shown in blocks (1) and (2) of Figure 2, and discussed in Section 2.2.

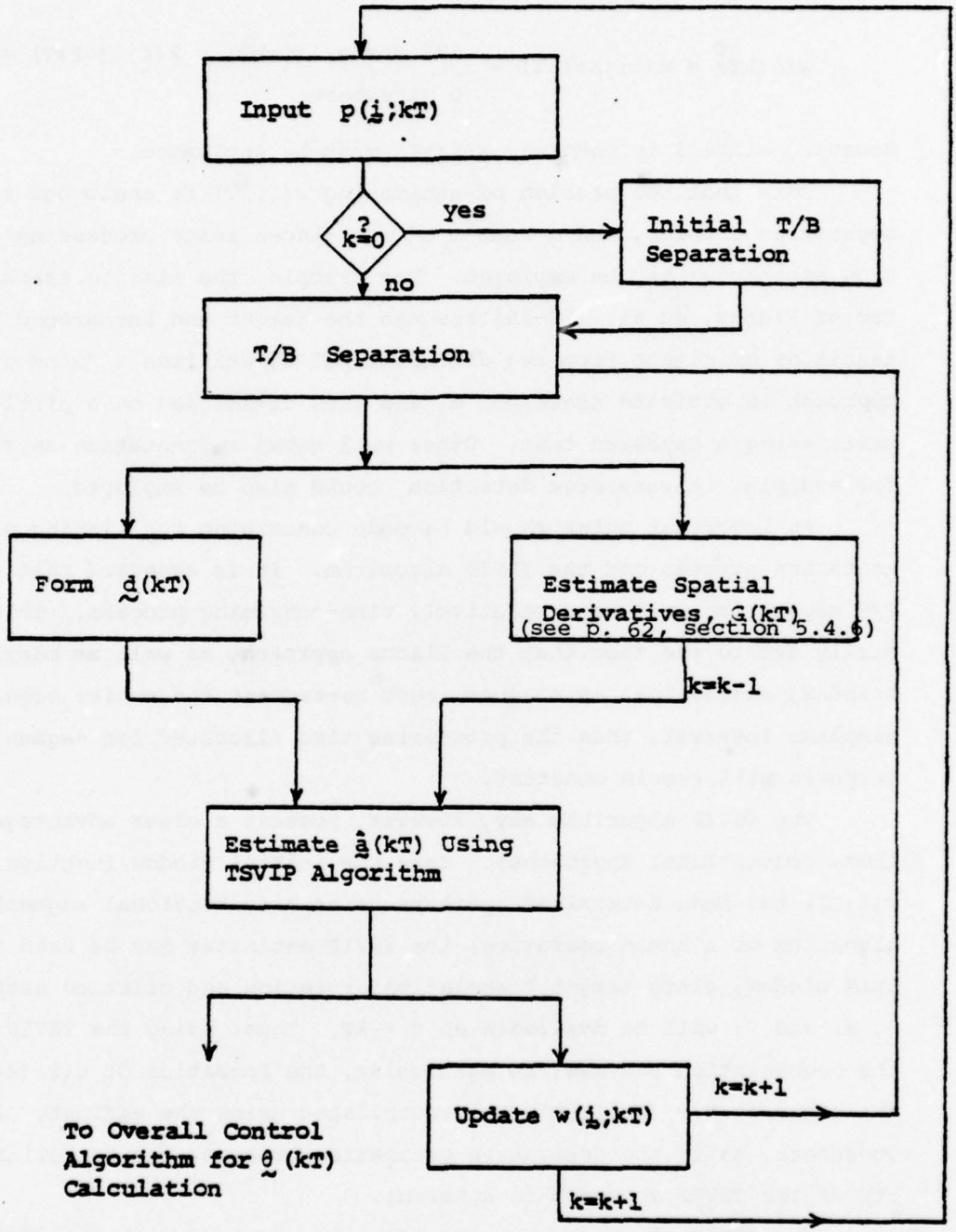


Figure 8

Flow Chart for Video Tracking Processor Using TSVIP Algorithm with Segmentation Window Updating

5.4.5.2 Segmentation Utilizing a Gestalt Law

One of the somewhat sparsely used tools in scene segmentation has been the use of velocity information for segmentation. Obviously, this information is only useful when considering classes of scenes in which certain portions (i.e., the target) are moving, and clearly the image tracking problem is such a case. Potter [34] suggests the use of velocity as a cue to segmentation, an idea also shared by psychologists [1] studying biological systems. Using the previously developed TSVIP algorithms, target perturbational information (which includes velocity information) may be employed in a very straightforward and novel way to enable scene segmentation.

The underlying basis for this segmentation approach is the Gestalt "law of common fate" ([1], p. 61) which states that "...if a collection of parts move in unison, one tends to see them as a single figure...." Recalling that the TSVIP algorithm estimates a vector of target perturbation related parameters, $\hat{\underset{\sim}{a}}$, it is straightforward to modify the algorithm to employ this law. A set of vectors $\{\hat{\underset{\sim}{a}}(q)\}$; q = 1, 2,. . .B corresponding to (parallel) application of this algorithm over a set of B non-overlapping 2-D blocks, is computed and the blocks in which $\hat{\underset{\sim}{a}}(q)$ estimates are identical (or close in some vector norm sense), are classified as beloning to either the target, the background, or both.

The success of this parallel application of the TSVIP algorithm is based upon the following reasoning. Recalling (5.3.1-5), i.e.:

$$p(\underset{\sim}{x};t_2) = f(\underset{\sim}{x};t_1)\lambda(\underset{\sim}{x};t_1) + [(\frac{\partial f(\underset{\sim}{x};t_1)}{\partial \underset{\sim}{x}})^T\lambda(\underset{\sim}{x};t_1)+f(\underset{\sim}{x};t_1)(\frac{\partial \lambda(\underset{\sim}{x};t_1)}{\partial \underset{\sim}{x}})^T](\hat{A}\underset{\sim}{x}+\underset{\sim}{b})$$
$$+ b(\underset{\sim}{x};t_1)[1-\lambda(\underset{\sim}{x};t_1)] - b(\underset{\sim}{x};t_1)(\frac{\partial \lambda(\underset{\sim}{x};t_1)}{\partial \underset{\sim}{x}})^T(\hat{A}\underset{\sim}{x}+\underset{\sim}{b}) \qquad (5.4.5.2\text{-}1$$

it is useful to now examine the remaining two scene regions previously mentioned in Section 5.3.1, but not considered, i.e., where

(1) $\lambda = 0;\ \frac{\partial \lambda}{\partial \underset{\sim}{x}} = 0$ (Background)

and

(2) $\lambda = 0$ or 1; $\frac{\partial \lambda}{\partial \underset{\sim}{x}}$ = "large" (Target contour).

For case (1), (5.4.5.2-1) in this region reduces to:

$$p(\underset{\sim}{x};t_2) = b(\underset{\sim}{x};t_1) \qquad (5.4.5.2\text{-}2)$$

$$= p(\underset{\sim}{x};t_1)$$

therefore it is obvious that for all $\underset{\sim}{x}$ in this region $d(\underset{\sim}{x};t_2) = 0$, and if the TSVIP algorithm were used, with sample points from this region only, the estimates would yield $\hat{\underset{\sim}{a}} = \underset{\sim}{0}$ or $\hat{\underset{\sim}{b}} = \underset{\sim}{0}$. This is intuitively obvious, since the background was assumed to be slowly time varying, and therefore no change is expected over two successive scenes.

In considering case (2), a somewhat different approach must be taken, due to the presence of the $\frac{\partial \lambda}{\partial \underset{\sim}{x}}$ terms in (5.4.5.2-1) and the binary nature of λ. Manipulating (4.3.2-1), (4.3.2-3), (4.3.1-2) and (5.3.1-6), it may be shown

$$\lambda(\underset{\sim}{x};t_2) \neq \lambda(\underset{\sim}{x};t_1)$$

that

$$d(\underset{\sim}{x};t_2) = f(\underset{\sim}{x};t_2)\lambda(\underset{\sim}{x};t_2) - f(\underset{\sim}{x};t_1)\lambda(\underset{\sim}{x};t_1) \qquad (5.4.5.2\text{-}3)$$
$$+ b(\underset{\sim}{x};t_1)[\lambda(\underset{\sim}{x};t_1) - \lambda(\underset{\sim}{x};t_2)]$$

Consideration of the two possible combinations of $(\lambda(\underset{\sim}{x};t_1), \lambda(\underset{\sim}{x};t_2))$ values (i.e., (1,0) or (0,1)) gives an obvious physical interpretation to (5.4.5.2-3): this is the portion of the scene difference function due to either "covering up" or "uncovering" of background points due to target perturbations. Empirical evidence (Chapter VI) suggests that a difference vector formed by using these points will tend to have many components of relatively large magnitudes (since there is little spatial or temporal correlation between the target and background functions), and therefore if the TSVIP algorithm is employed using sample points from these regions, the $\hat{\underset{\sim}{a}}$ or $\hat{\underset{\sim}{b}}$ vectors will tend to be biased towards "large" values. Since these regions occur around the target edges, it may be desirable to detect and eliminate samples originating from these regions, and one possible approach is to form a histogram of the $\underset{\sim}{d}$ vector components and, based upon this data, only accept d_i components below a certain threshold. This approach has been considered in the simulation documented in the following chapter.

Summarizing the above, the following should be noted:

(1) If a majority of sample points are obtained from window regions

interior to both the initial and perturbed targets, the TSVIP algorithm will generate accurate estimates of the $\hat{\underset{\sim}{a}}$ or $\hat{\underset{\sim}{b}}$ vectors, which will be close (in a vector norm sense) for all windows in this region (an exception is noted below).

(2) If a majority of sample points are obtained from window regions in the background regions, the TSVIP estimates of $\hat{\underset{\sim}{a}}$ or $\hat{\underset{\sim}{b}}$ will tend towards $\underset{\sim}{0}$.

(3) If a majority of sample points are obtained from window regions in the "covered" or "uncovered" regions, the TSVIP estimates of $\hat{\underset{\sim}{a}}$ or $\hat{\underset{\sim}{b}}$ will tend towards very large or unreasonable values, which are certainly not consistent with the original assumption of small scene-to-scene target perturbations.

Thus, the above results may be used in a novel way to enable target/background (T/B) separation, by associating those window-related $\hat{\underset{\sim}{a}}$ or $\hat{\underset{\sim}{b}}$ vectors which are similar as representing regions which are also similar, i.e., representing the target interior, "uncovered"/"covered," or background regions. In this way the TSVIP algorithm may be used perhaps as part of a two-pass "bootstrap" procedure, wherein the first set of scene windows is chosen with dimensions small enough to generate a good estimate of the target contour, and then, based upon these regional estimates, all common or similar target region data points are then "pooled" for refined estimates of either $\hat{\underset{\sim}{a}}$ or $\hat{\underset{\sim}{b}}$.

Two final points should be noted. First, in the case where $\underset{\sim}{a} = \underset{\sim}{0}$, obviously $p(\underset{\sim}{i};kT) = p(\underset{\sim}{i};(k+1)T)$. Thus, the preceding procedure would, for every window, generate estimates closely approximating $\hat{\underset{\sim}{a}} = \underset{\sim}{0}$. Therefore, segmentation would not be possible in this case, however, this is of no real concern, since if the target is unchanged, the system does not need to take any corrective action. Second, since the TSVIP algorithm requires that the D_c matrix have full column rank, each window used in the Gestalt segmentation scheme must contain sufficient features such that $(D_c^T D_c)$ is nonsingular. Otherwise, it is not possible to generate a reliable estimate of $\hat{\underset{\sim}{a}}$ or $\hat{\underset{\sim}{b}}$ for this window, and the utility of the

aforementioned segmentation technique becomes questionable.

The TSVIP algorithm may therefore be combined with this type of segmentation approach to yield the overall tracking processor shown in Figure 9. Note that this processor contains the necessary algorithms shown in blocks (1) and (2) of Figure 2, and discussed in Section 2.2.

5.4.6 Consequences of the TSVIP Algorithm Approach

5.4.6.1 Derivative Estimation

In Section 5.4.2 it was indicated that estimates of scene spatial derivatives were required for the TSVIP algorithm. This is not expected to be a critical requirement, provided proper care is exercised in obtaining these derivatives.

Perhaps the most obvious and straightforward approach to spatial derivative estimation is the use of a simple difference approximation. This approach is ideally suited to CCD-implementable calculations. The conditions necessary to insure the validity of such an approximation are well known [35], [36], therefore, for the system considered here this implies one or more of the following:

(1) A (high resolution) sensor is available, with sufficient horizontal and vertical sample resolution, so that $\frac{\partial f(\underline{x})}{\partial x_1}$, and $\frac{\partial f(\underline{x})}{\partial x_2}$ as used in (5.4.2-3a) may be approximated as:

$$\frac{\partial f(\underline{x})}{\partial x_1} \simeq \frac{f\begin{bmatrix} x_1+\Delta \\ x_2 \end{bmatrix} - f\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}}{\Delta}$$

and (5.4.6.1-1)

$$\frac{\partial f(\underline{x})}{\partial x_2} \simeq \frac{f\begin{bmatrix} x_1 \\ x_2+\Delta \end{bmatrix} - f\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}}{\Delta}$$

Note that this does not imply that this sensor must employ a rectangular sampling lattice, but only that at sampling points the adjacent (i.e., i-1 and i+1) samples be sufficiently close.

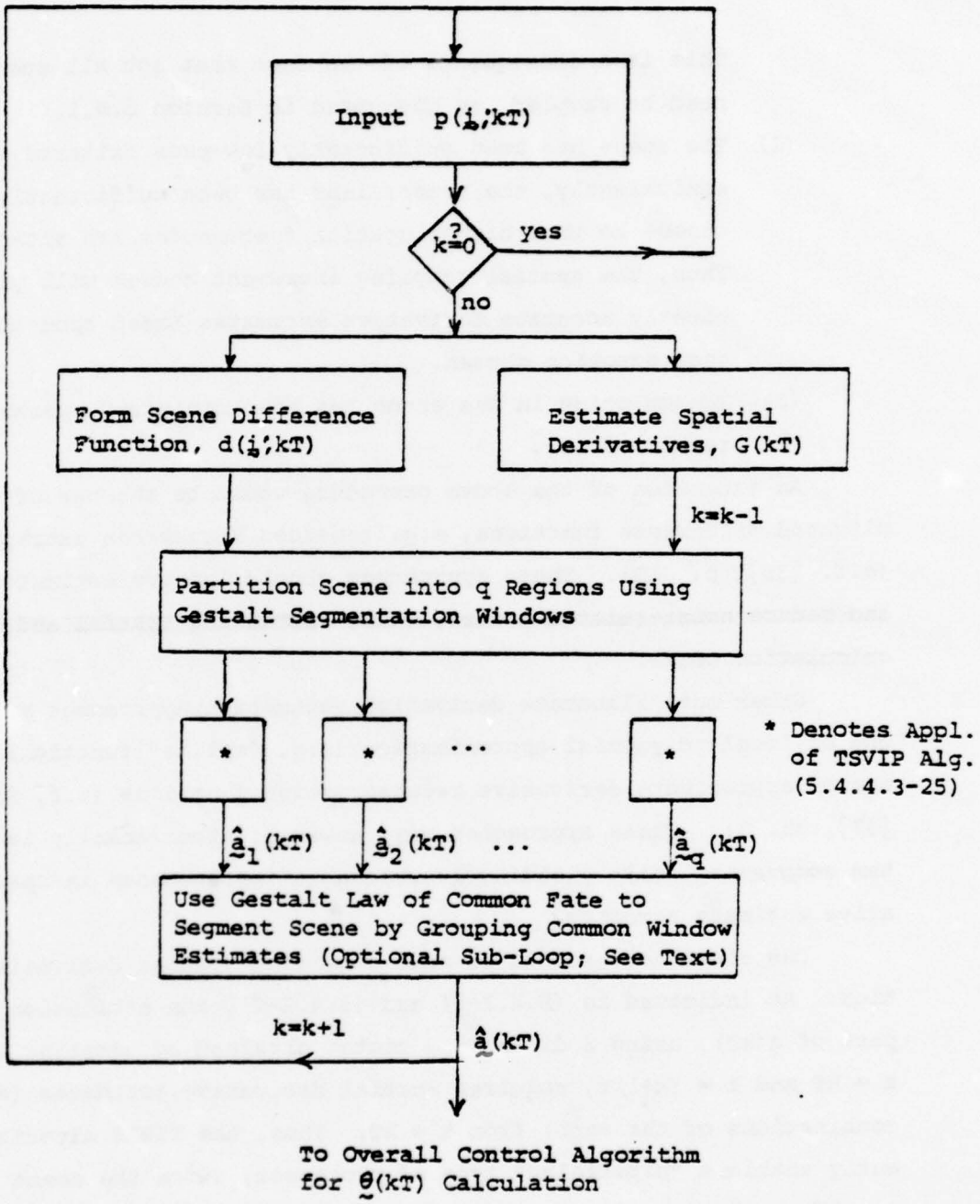


Figure 9

Flow Chart for Video Tracking Processor Using TSVIP Algorithm with Gestalt Segmentation

This is a consequence of the fact that not all scene points need be sampled, as discussed in Section 5.4.1.

(2) The scene has been sufficiently low-pass filtered (or almost equivalently, the sensor lens has been sufficiently defocused) so that higher spatial frequencies are attenuated. Thus, the spatial sampling increment chosen will yield sufficiently accurate derivative estimates based upon the difference equation chosen.

(3) Random noise in the scene has been minimized, perhaps as a result of (2).

An extension of the above procedure would be the use of more complicated difference functions, e.g. repeated Richardson extrapolation (c.f. [35], p. 310). These approaches should improve estimate accuracy and reduce noise-related errors, while increasing spatial and temporal calculation costs.

Other more elaborate derivative estimation approaches might be the use of local polynomial approximation (e.g. "spline" functions) or, the use of approximate derivative reconstruction functions (c.f. Freeman, [37], ch. 5). These approaches may, however, significantly increase system complexity while yielding questionable improvements in spatial derivative estimate accuracy.

One final note should be made regarding spatial derivative estimation. As indicated in (5.4.2-5) and (5.4.2-7), the estimation of all or part of $\underset{\sim}{a}(kT)$, using a difference vector obtained by sampling scenes at $t = kT$ and $t = (k+1)T$, requires spatial derivative estimates (and linear combinations of the same) from $t = kT$. Thus, the TSVIP algorithm inherently enables a "pipelining" type of processor, since the scene data at $t = kT$ may be processed to yield spatial derivatives while simultaneously the scene at $t = (k+1)T$ is being sampled and the difference vector at $t = (k+1)T$ is being formed. Thus, the requirement of spatial derivative estimation should not effect the system temporal sampling rate, but only the processor spatial complexity.

5.4.6.2 Validity of Approximation--Limits on A and $\underset{\sim}{b}$

The major approximation in the TSVIP algorithm development concerned the modelling of target temporal evolution using only the first order terms of the exact 2-D Taylor series expansion. The validity of this approximation is guaranteed only if it can be shown that the effect of second order and higher terms is negligible. As noted in Remark (1) of Section 5.3.1, this implies that $||(A-I)\underset{\sim}{x} + \underset{\sim}{b}||$ should be, in some sense, "small," since the remainder in (5.3.1-7) is known to be (c.f. [38], p. 81):

$$R = \frac{1}{2}[\hat{A}\underset{\sim}{x}+\underset{\sim}{b}]^T \begin{bmatrix} \frac{\partial f(\zeta)}{\partial x_1 \partial x_1} & \frac{\partial f(\zeta)}{\partial x_1 \partial x_2} \\ \frac{\partial f(\zeta)}{\partial x_2 \partial x_1} & \frac{\partial f(\zeta)}{\partial x_2 \partial x_2} \end{bmatrix} [\hat{A}\underset{\sim}{x}+\underset{\sim}{b}] \qquad (5.4.6.2\text{-}1)$$

where

$$\underset{\sim}{\zeta} \varepsilon [\underset{\sim}{x}, \hat{A}\underset{\sim}{x}+\underset{\sim}{b}] \qquad (5.4.6.2\text{-}2)$$

Therefore the error introduced by the Taylor series approximation, for any value of $\underset{\sim}{x}$, is proportional to $||\hat{A}\underset{\sim}{x}+\underset{\sim}{b}||^2$, weighted by a matrix of spatial second derivatives.

One straightforward but computationally lengthy approach might be to assume that the second derivatives of the target textural function are known, and then, for each value of $\underset{\sim}{x}$, use (5.4.6.2-1) and (5.4.6.2-2) to calculate the worst case remainder as a function of $\hat{A}$ and $\underset{\sim}{b}$. Then, by suitably constraining the remainder (e.g., $R \leq .01$), establish limits on $\hat{A}$ and $\underset{\sim}{b}$. With some knowledge of the expected target maneuverability (e.g., $\dot{\theta}_{max}$, $\dot{\underset{\sim}{c}}_{max}$, $\dot{\underset{\sim}{x}}_{max}$), it is then a simple matter to set the system temporal sampling rate such that $\hat{A}$ and $\underset{\sim}{b}$ will be within these limits at all sampling instants.

A more realistic and computationally simpler approach might be to assume that the first derivatives of the target textural function were known a priori, or calculable, and that over some multiple of the spatial sampling interval, i.e., $k\Delta$, these partial derivatives were very slowly varying.

(This approach was used in the simulation described in Chapter VI, where it was found that if Δ was chosen to be approximately 1/6 of the Nyquist spatial sampling increment, this assumption was satisfied for $k \leq 2$.) Then, constraining target perturbations such that

$$||\hat{A}\underset{\sim}{x}+\underset{\sim}{b}|| < k\Delta \qquad (5.4.6.2\text{-}3)$$

it is a straightforward procedure to use maximum target maneuverability data (i.e., $\dot{\theta}_{max}$, $\dot{\underset{\sim}{x}}_{max}$ and $\dot{\underset{\sim}{\sigma}}_{max}$) to set the temporal sampling rate sufficiently high in order that (5.4.6.2-3) is satisfied.

It is important to note, that, in general, the remainder in (5.4.6.2-1) is a function of $\underset{\sim}{x}$. For example, for a simple rotation the magnitude of the term $\hat{A}\underset{\sim}{x}$ increases with increasing $||\underset{\sim}{x}||$. Therefore, if target perturbations were limited by adjustment of the temporal sampling rate as above, it is expected that a very conservative sampling rate would be chosen, due to the effects of large values of $||\underset{\sim}{x}||$ (e.g., at the extremes or "corners" of the target). Furthermore, since this remainder may be thought of as contributing to an additional error term in the formulation of the estimation equations in Sections 5.4.4.1 through 5.4.4.4, the effect of this error on estimates of $\hat{\underset{\sim}{a}}$ or $\hat{\underset{\sim}{b}}$ will be inherently minimized by the least squares estimation algorithms used. Thus, it is more realistic to statistically consider the error term due to (5.4.6.2-1) rather than its maximum value, when choosing the temporal sampling rate.

5.4.6.3 Error Evolution and Minimization

There are a number of ways in which errors arise in the TSVIP algorithm estimates, and no detailed analysis of all possible cases is intended here. Instead, an overview of these errors is given, since many have been mentioned in the previous text.

Generally errors in $\hat{\underset{\sim}{a}}$ or $\hat{\underset{\sim}{b}}$ arise from one or more of the following causes:

(1) Inexact segmentation (i.e., samples are taken from points other than those on the target interior). This topic was

considered in Section 5.4.5.2, where it is shown that this situation may be utilized as a novel aid to the segmentation process.

(2) Inexact derivative estimation. This topic was mentioned in Section 5.4.6.1, and it is assumed that this effect is of secondary importance.

(3) Inadequacy of the assumed model. Extensions of the developed model were presented in Section 4.3.4, and it is assumed that this problem is not present.

(4) Failure of the Taylor Series approximation in (5.3.1-7) to hold. This problem can only arise if the system temporal sampling rate is chosen too slow, and therefore, if the guidelines of the previous section are followed, should not occur.

(5) Errors due to CCD implementation of the TSVIP algorithm. This is a very involved topic, highly dependent upon the exact implementation chosen, and is suggested for future study.

(6) Failure of the target to have sufficient texture (i.e., D_c does not have full column rank). Obviously, in this case there is nothing which may be done to improve the TSVIP estimates. The algorithm, however, will output a flag (described at the end of Section 5.4.4.3) indicating that the estimates are unreliable. Another somewhat related case is the situation where $D_c^T D_c$ is nearly singular, in which case the potential for severe algorithm numerical sensitivity is significant. This problem may again be eliminated by a careful a priori investigation of the particular tracking application.

(7) Random picture noise. It is likely that this case will arise in practice, therefore this situation will be discussed briefly. From (5.4.4.3-3) this noise in $p(\underset{\sim}{i};kT)$ will obviously affect $\hat{\underset{\sim}{a}}$ or $\hat{\underset{\sim}{b}}$ in two ways:

(i) In the estimation of D_c; and

(ii) Through errors in $\underset{\sim}{d}[(k+1)T]$.

Clearly, the overall influence of both noise effects mentioned in (7) on the TSVIP estimates will be minimized if each scene is pre-processed by an appropriate noise-reduction filter. If such pre-processing is not used, however, the TSVIP algorithm is still expected to yield good results provided that a reasonably noise-insensitive derivative estimation algorithm is used (as discussed in Section 5.4.6.1) and noise statistics are approximately constant over a period of several scenes. With these assumptions, the primary error effects are due to errors in $\underset{\sim}{d}[(k+1)T]$, since from (5.4.1-1) and (5.4.1-2)

$$\underset{\sim}{d}[(k+1)T] = [p[\underset{\sim}{x}_i;(k+1)T - p[\underset{\sim}{x}_i;kT]] \qquad (5.4.6.3\text{-}1)$$

$$i = 1,2. \; . \; .N$$

and, with additive noise

$$p_N(\underset{\sim}{i};kT) \triangleq p(\underset{\sim}{i};kT) + \eta(\underset{\sim}{i};kT) \qquad (5.4.6.3\text{-}2)$$

where p_N and p represent the noise-corrupted and the exact picture functions respectively, and η is a 3-D noise process, with assumed slowly temporally varying statistics.

Due to the nature of the imaging process (c.f. [6], p. 145), η will not be a zero-mean process. This is a significant point often overlooked in previous efforts. However, from (5.4.6.3-1) and the assumption of slowly time varying noise statistics, it is clear that the noise component of $\underset{\sim}{d}[(k+1)T]$ will be a zero mean vector, and from Appendix 3, the TSVIP estimates of $\hat{\underset{\sim}{a}}$ or $\hat{\underset{\sim}{b}}$ will then be unbiased. Furthermore, if some statistical characterization of η is possible, Appendix 3 also shows how the least squares scheme (and consequently the TSVIP algorithm) may be adapted to obtain minimum variance estimates for $\hat{\underset{\sim}{a}}$ or $\hat{\underset{\sim}{b}}$.

5.4.7 Closed Loop Effects Due to TSVIP

Heretofore, the major emphasis of this work has been on the modelling of scene evolution and the development of the TSVIP algorithm (with some attention also having been devoted to the segmentation problem).

Thus, those components incorporated in the blocks labelled "Video Process I/O Relationship" and "T/B Sep. and Registration" respectively in Figure 3 have been considered. The remaining blocks in Figure 3 will now be addressed, in order to consider the effects of the TSVIP algorithm on the overall closed loop system.

As mentioned in Chapter 2, a necessary system component is a tracking filter, which could improve TSVIP estimates (i.e., $\hat{\underset{\sim}{a}}(kT)$) by minimizing the effect of noisy data, inexact T/B separation, etc. Tracking filters of this type are well known, and an excellent summary of this work is found in [39].

Perhaps a more interesting problem is the processing of the TSVIP estimates, $\hat{\underset{\sim}{a}}(kT)$, to generate the actual camera gimbal control signal vector, $\underset{\sim}{\theta}(kT)$, which from (4.1-1) consists of three components:

$$\underset{\sim}{\theta}(kT) = \begin{bmatrix} \theta_a(kT) \\ \theta_e(kT) \\ z(kT) \end{bmatrix} \qquad (5.4.7\text{-}1)$$

where the first two components of this vector are used to adjust the camera platform azimuth and elevation, and the last component is useful for zoom lens control. Recalling from Section 5.4.4.2 that the TSVIP algorithm enables calculations at times $t = kT$ of α, the target magnification parameter, θ, the target rotation, and $\hat{\underset{\sim}{b}}$, the target horizontal and vertical perturbations in the sensor plane, the following may be reasoned:

(1) Unless it is desirable to update the segmentation window (or perhaps rotate the camera to keep the camera and target aligned in some manner), the utility of the θ estimate is limited;

(2) The estimate of $\alpha(kT)$ may be used directly to form the $z(kT)$ estimate, since, from Appendix 1, α represents the amount of target dilation, i.e., if the target remains the same size $\alpha = 1.0$; and

(3) The estimates of $\hat{\underset{\sim}{b}}(kT)$, together with a knowledge of the

camera lens to sensor plane distance, f, may be used to form the control signals $\Theta_a(kT)$ and $\Theta_e(kT)$. Since these control signals are horizontally and vertically separable, Figure 8 illustrates the 1-D case. For the target perturbation shown in the figure, re-centering the perturbated target at t = (k+1)T requires rotation of the camera mount through an angle $\Theta_p(kT)$ (where $\Theta_p(kT) = \Theta_a(kT)$ if $\hat{b}_i = \hat{b}_1$, and $\Theta_p(kT) = \Theta_e(kT)$ if $\hat{b}_i = \hat{b}_2$ in Figure 8). Simple trigonometry shows that $\Theta_p(kT)$ may be obtained as:

$$\Theta_p(kT) = t_{an}^{-1}\left(\frac{\hat{b}_i}{f}\right) \approx \left(\frac{1}{f}\right)\hat{b}_i \qquad (5.4.7.2)$$

Since the quantity $\frac{b_i}{f}$ is typically very small. Thus, the generation of the actual control signals, in particular, the camera azimuth and elevation controls is straightforward.

Another important closed-loop consideration is the effect of the camera controls. As mentioned in Section 4.3.2, the actual A and $\underset{\sim}{b}$ parameters which control scene temporal evolution are functions of <u>both</u> the camera mount controls and the target trajectory or perturbation. Since the TSVIP algorithm developed herein operates on a scene-to-scene basis, it is necessary to adjust the estimates from this algorithm for known camera movement, and this may be accomplished in a very straightforward manner by solving (5.4.7-2) for $\hat{b}_i$, and using this information to adjust the actual camera mount control signals. Unless this step is taken, it is not difficult to envision a system wherein a simple step input in target position initiates a stable limit cycle in the closed loop system.

Finally it is useful to consider the overall system performance over a large number of successive scenes and various target perturbations, perhaps to establish long-term stability information, or determine the propagation of errors due to initially inexact segmentation,

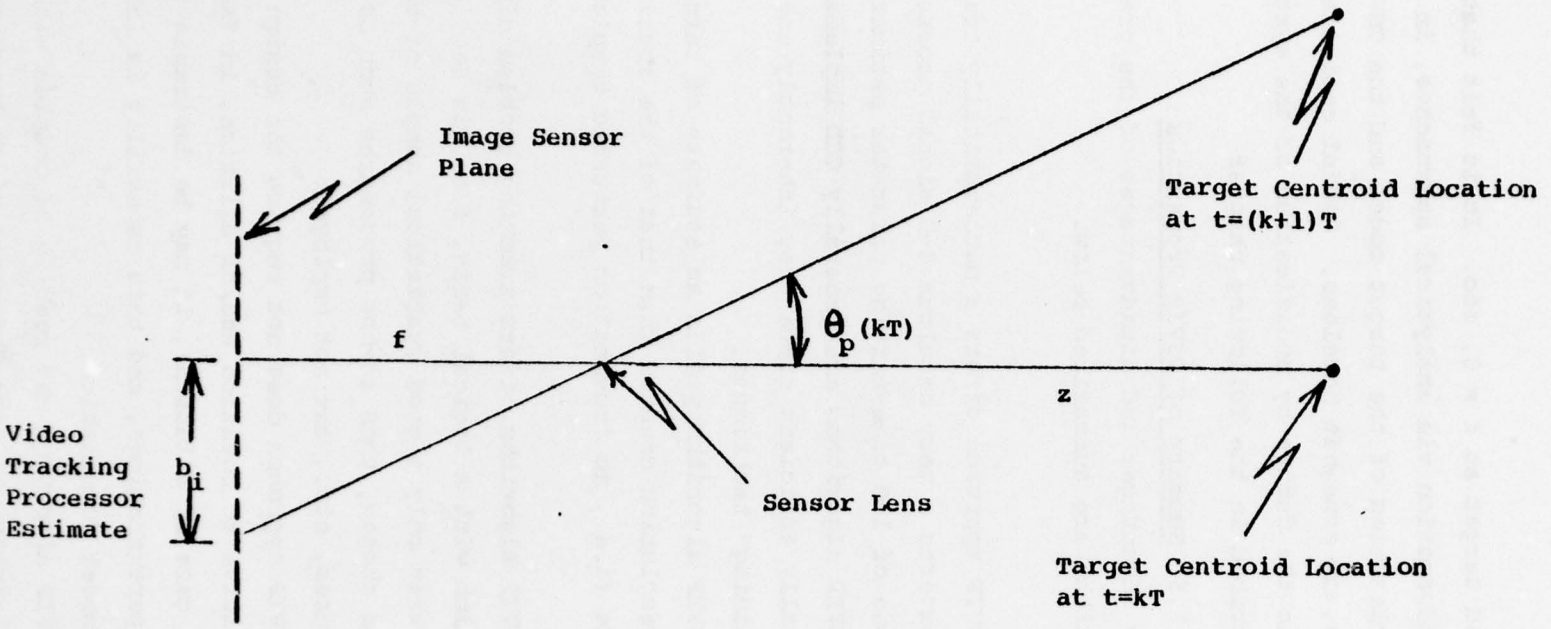


Figure 10

Generation of Camera Mount Control Signals from TSVIP Algorithm Estimates
(1-D Example)

lack of a centered target at $t = 0$, etc. It is felt that attempts at obtaining this information via analytical approaches, in light of the mathematical complexities of the target model and the TSVIP and segmentation algorithms, are somewhat hopeless. Useful empirical information may be obtained in the future by modification of the system simulation capabilities described in the following chapter

5.5 Summary of TSVIP Properties

The primary advantages and disadvantages of the previously proposed TSVIP algorithm are summarized below.

Advantages

(1) The TSVIP approach offers a mathematically tractable model and converts a very nebulous 3-D signal processing problem into one of 1-D time-varying parameter estimation.

(2) The TSVIP algorithms are generally CCD implementable, and temporally efficient since they inherently enable processor "pipelining" techniques.

(3) The TSVIP algorithm yields an estimate of target translation with resolution greater than that of the spatial sampling lattice (i.e., no "moving" or perturbed template is necessary),

(4) The TSVIP algorithm offers tracking problem dimensionality reduction with a logical basis, i.e., in Section 5.4.1 it is shown that only a good statistical sample of target points need be chosen, thus ad-hoc procedures such as averaging, signatures, etc., are not employed.

(5) The TSVIP approach does not require the design of algorithms insensitive to rotation and/or dilation, in fact (especially in the case of dilation), it may be desirable to estimate these perturbations, and this capability is inherent in the TSVIP model formulation.

(6) The TSVIP algorithm may readily be coupled with a segmentation algorithm, to form the basic video tracking processor.

In addition, the TSVIP algorithm enables two novel segmentation approaches, namely: (i) segmentation window updating, and (ii) Gestalt common fate segmentation.

Disadvantages

(1) The TSVIP approach requires the estimation of spatial derivatives. This may lead to spatial oversampling, however, since it is desirable to trade complexity in space for temporal speed, this may be viewed as a system hardware concern.

(2) The object to be tracked must have adequate textural features, otherwise the TSVIP algorithm estimates are not reliable. Non-textured targets are, however, a trivial case and would logically be handled using simpler algorithms.

(3) Target perturbations must be restricted temporally in order to insure the validity of the first order Taylor Series approximation. However, this is merely dependent upon adequate a priori choice of the system temporal sampling rate.

CHAPTER VI

IMPLEMENTATION OF THE MODIFIED DIRECT APPROACH USING CLINE'S METHOD

In this chapter specific architecture is suggested for implementation of the constrained TSVIP algorithm using CCD devices. The method developed in Section 5.4.4.3 will be followed. Recent developments of CCD programmable transversal filters look promising for the calculation of the inner product of two variable vectors. This is an essential function in the proposed system.

6.1 Summary of Necessary Formulas

Recall (formula (5.4.4.3-1) that

$$\underset{\sim}{d} = D_C \underset{\sim}{a} \tag{6.1.1}$$

where $\underset{\sim}{d}$ is the Nx1 scene difference vector, which corresponds to the charge (proportional to light intensity) in the sensor sampling pixels; D_C is the constrained Nx4 matrix formed by the Nx2 P_C matrix and the Nx2 G matrix. It is written

$$D_C = [P_C \vdots G] \tag{6.1.2}$$

Vector $\underset{\sim}{a}$ is the 4x1 total affine parameter vector. The estimate of $\underset{\sim}{a}$ is

$$\hat{\underset{\sim}{a}} = D_C^{\dagger} \underset{\sim}{d} \tag{6.1.3}$$

where

$$\begin{aligned} D_C^{\dagger} &= [P_C \vdots G]^{\dagger} \\ &= \begin{bmatrix} P_C^{\dagger} - P_C^{\dagger} G C^{\dagger} \\ C^{\dagger} \end{bmatrix} \end{aligned} \tag{6.1.4}$$

where "†" denotes pseudo inverse, and

$$\begin{aligned} C &\triangleq [I - P_C (P_C{}^T P_C)^{-1} P_C{}^T] G \\ &= [I - P_C P_C{}^{\dagger}] G \end{aligned} \tag{6.1.5}$$

Recall, also, that,

$$C^{\dagger} \triangleq (C^T C)^{-1} C^T \tag{6.1.6}$$

and $P_C{}^{\dagger}$ is defined in a similar manner. According to (6.1.3) and (6.1.4), the TSVIP algorithm can be expressed as

$$\hat{\underset{\sim}{a}}_c = (P_C{}^{\dagger} - P_C{}^{\dagger} G C^{\dagger}) \underset{\sim}{d} \tag{6.1.7}$$

$$\underset{\sim}{b} = C^{\dagger} \underset{\sim}{d} \tag{6.1.8}$$

Notice that once $\underset{\sim}{b}$ is estimated, (6.1.7) can be written as

$$\hat{\underset{\sim}{a}}_c = P_C{}^{\dagger} \underset{\sim}{d} - P_C{}^{\dagger} G \hat{\underset{\sim}{b}}$$

providing an alternate implementation for $\hat{\underset{\sim}{a}}$.

6.2 Estimation of the Total Affine Parameter Vector $\underset{\sim}{a}$

6.2.1 Estimation of the Translational Vector $\underset{\sim}{b}$

Assuming that all the sampling points are in the target, $p = s = f$ is measured. The location of the sampling points with respect to the sampling lattice origin is known, and we have

x_{i1} = value of x_1 at sampling point i

x_{i2} = value of x_2 at sampling point i

f'_{i1} = spatial derivative with respect to x_1 at sampling point i

f'_{i2} = spatial derivative with respect to x_2 at sampling point i.

We can then write expressions for P_C and G as follows:

$$P_C = \begin{bmatrix} p_{11} & p_{12} \\ p_{21} & p_{22} \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ p_{N1} & p_{N2} \end{bmatrix} \triangleq [\underset{\sim}{p}_1 \vdots \underset{\sim}{p}_2] \tag{6.2.1.1}$$

where $p_{i1} = x_{i1} f'_{i1} + x_{i2} f'_{i2}$; $p_{i2} = x_{i1} f'_{i2} = x_{i2} f'_{i1}$

(recall (5.4.4.2-3)), i = 1,2,. . ., N and

$$G = \begin{bmatrix} f'_{11} & f'_{12} \\ f'_{21} & f'_{22} \\ \vdots & \\ \cdot & \\ f'_{N1} & f'_{N2} \end{bmatrix} \triangleq [\underset{\sim}{f}'_{i1} \vdots \underset{\sim}{f}'_{i2}]$$

It is necessary to have an estimate of the spatial derivatives f'_{i1} and f'_{i2}. The simplest, although perhaps not the most efficient way of obtaining these estimates is to measure the texture function (proportional to sampling cell charge) at the sampling points and at two very close points in the x_1 and x_2 directions, separated from each sampling point by a known distance Δ. This is illustrated for a 3x3 (nine point) sampling lattice in Figure 11. Notice that this sampling lattice corresponds to the vectors $\underset{\sim}{x}_1$ and $\underset{\sim}{x}_2$ given by (5.4.4.3-6) and Figure 7. Recall also that the vectors $\underset{\sim}{x}_1$ and $\underset{\sim}{x}_2$ are defined in (5.4.4.2-8) as $X \triangleq [\underset{\sim}{x}_1 \vdots \underset{\sim}{x}_2] = \underset{\sim}{x}_k^T$, K= 1,2,. . .,N, and must not be confused with the vector

$$\underset{\sim}{x}_i \triangleq \begin{bmatrix} x_{i1} \\ x_{i2} \end{bmatrix}$$

as defined in (4.3.3-2).

For Δ sufficiently small,

$$\left[\frac{\partial f(\underset{\sim}{x})}{\partial x_1}\right]_i \triangleq f'_{i1} \simeq \frac{f_i\begin{bmatrix} x_{i1}+\Delta \\ x_{i2} \end{bmatrix} - f_i\begin{bmatrix} x_{i1} \\ x_{i2} \end{bmatrix}}{\Delta} \qquad (6.2.1.2)$$

$$\left[\frac{\partial f(\underset{\sim}{x})}{\partial x_2}\right]_i \triangleq f_{i2} \simeq \frac{f_i\begin{bmatrix} x_{i1} \\ x_{i2}+\Delta \end{bmatrix} - f\begin{bmatrix} x_{i1} \\ x_{i2} \end{bmatrix}}{\Delta}$$

for i = 1,2,. . .,N.

The sampling points vector x_i and Δ are kept constant. Hence $\frac{1}{\Delta}$ = constant and

$$f'_{i1} \simeq \frac{f_{i1+\Delta} - f_i}{\Delta}$$
$$f'_{i2} \simeq \frac{f_{i2+\Delta} - f_i}{\Delta} \qquad (6.2.1.3)$$

where $f_{i1+\Delta}$ and $f_{i2+\Delta}$ indicate displacement in the x_1 and x_2 directions, respectively. N measurements of f_i, $f_{i1+\Delta}$ and $f_{i2+\Delta}$ plus the value of Δ will allow us to determine, approximately, the matrix G as indicated in figure 12. Notice that the f's represent charges, and that the $\frac{1}{\Delta}$'s represent fixed tap weights. The implementation of negative coefficients

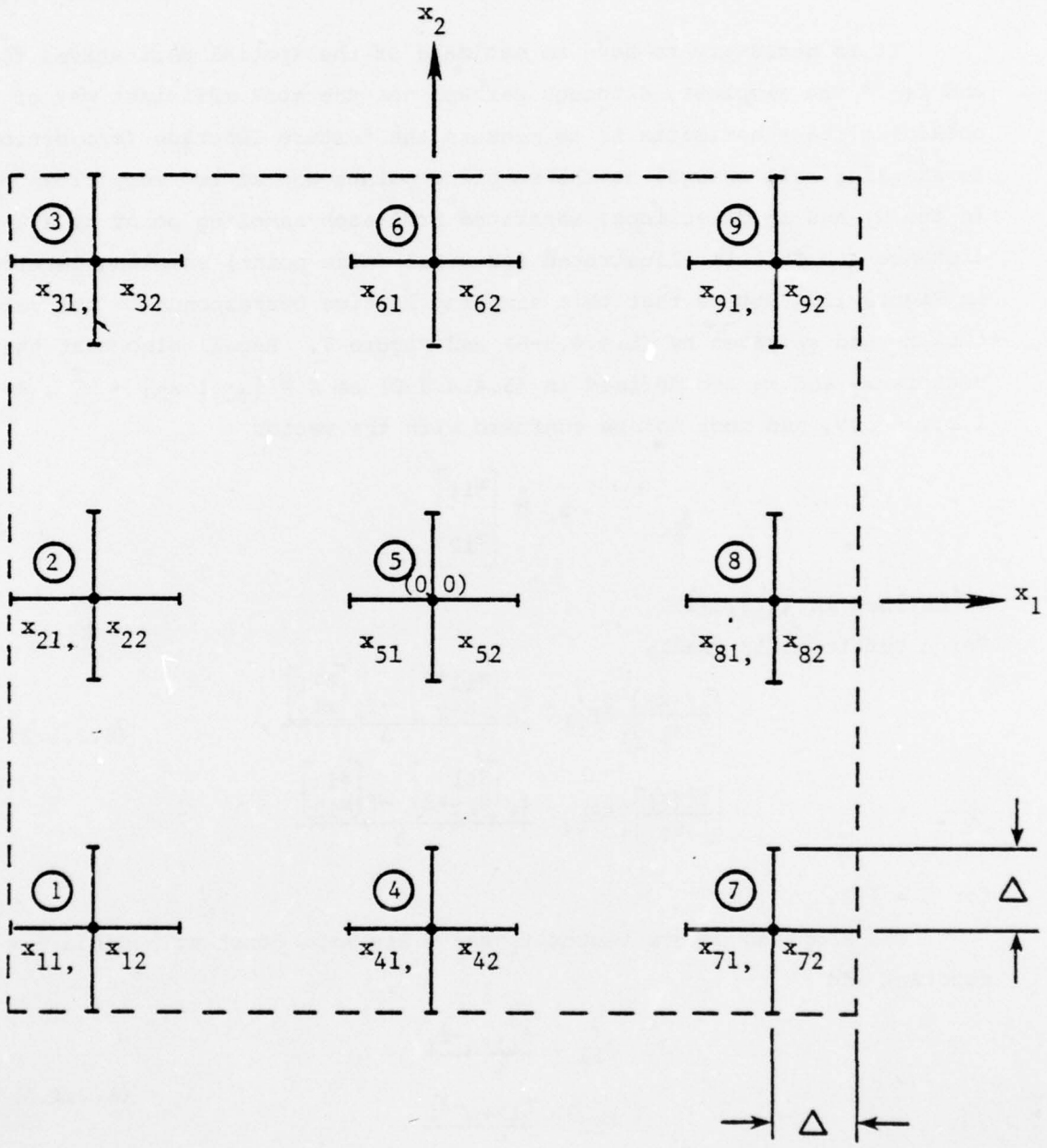


Figure 11 Example of Nine Point Sampling Lattice

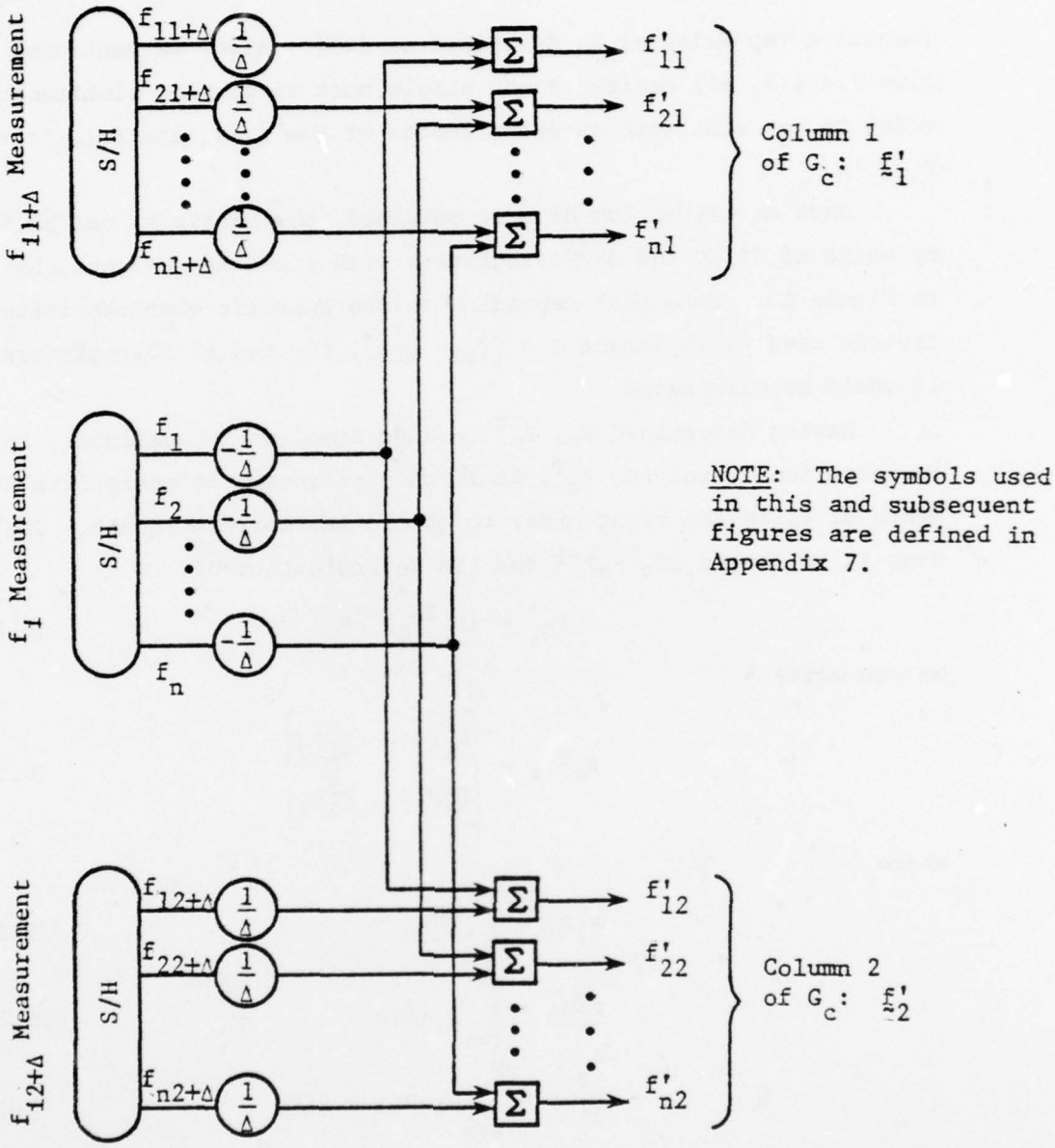


Figure 12 Implementation of the Spatial Derivative Matrix G

(negative tap weights) is described in [47]. Also, as mentioned in Section 5.4.4.3, all desired scene pixels must be sampled simultaneously in order to get simultaneous measurements of the f_i's, the $f_{i1+\Delta}$'s and the $f_{i2+\Delta}$'s.

Once an estimation of G is obtained, the matrix P_c can be implemented by means of PI/SO and SI/PO registers with fixed tap weights, as indicated in Figure 13. Note that depending on the specific characteristics of the devices used to implement $G = [f'_{i1} \quad f'_{i2}]$, the two PI/SO registers of Fig. 13 could be eliminated.

Having determined P_c, $P_c{}^T$ is also immediately available, because in operations involving $P_c{}^T$, it is only necessary to manipulate the elements of P_c in the right order to obtain the correct result. The next step is to obtain $(P_c{}^T P_c)^{-1}$ for the determination of

$$P_c{}^{\dagger} = (P_c{}^T P_c)^{-1} P_c{}^T \tag{6.2.1.4}$$

We can write

$$P_c{}^T P_c = \begin{bmatrix} \underset{\sim}{P}_1^T \underset{\sim}{P}_1 & \underset{\sim}{P}_1^T \underset{\sim}{P}_2 \\ \underset{\sim}{P}_2^T \underset{\sim}{P}_1 & \underset{\sim}{P}_2^T \underset{\sim}{P}_2 \end{bmatrix} \tag{6.2.1.5}$$

where

$$\underset{\sim}{P}_1^T \underset{\sim}{P}_1 = \sum_{i=1}^{N} p_{i1}^2 \tag{6.2.1.5a}$$

$$\underset{\sim}{P}_1^T \underset{\sim}{P}_2 = \sum_{i=1}^{N} p_{i1} p_{i2} \tag{6.2.1.5b}$$

$$\underset{\sim}{P}_2^T \underset{\sim}{P}_1 = \sum_{i=1}^{N} p_{i2} p_{i1} = \underset{\sim}{P}_1^T \underset{\sim}{P}_2 \tag{6.2.1.5c}$$

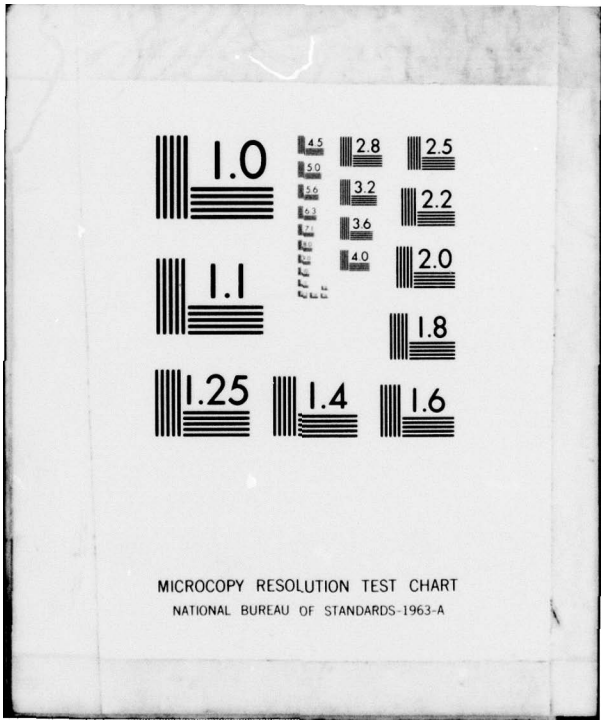
$$\underset{\sim}{P}_2^T \underset{\sim}{P}_2 = \sum_{i=1}^{N} p_{i2}^2 \tag{6.2.1.5d}$$

It is evident that the implementation of the formulas above require the multiplication of two variable vectors. This will also be true for other parts of the system. The problem can be solved in a variety of ways, which can be grouped into three categories:

AD-A078 809 VIRGINIA UNIV CHARLOTTESVILLE RESEARCH LABS FOR THE--ETC F/G 17/7
DISCRETE ANALOG PROCESSING FOR TRACKING AND GUIDANCE CONTROL.(U)
AUG 79 E S MCVEY , E A PARRISH , R M INIGO N00014-78-C-0695
UNCLASSIFIED UVA/525350/EE79/102 ONR-CR233-092-1 NL

2 OF 2
AD
A078809

END
DATE
FILMED
1-80
DDC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
6.3
7.1
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

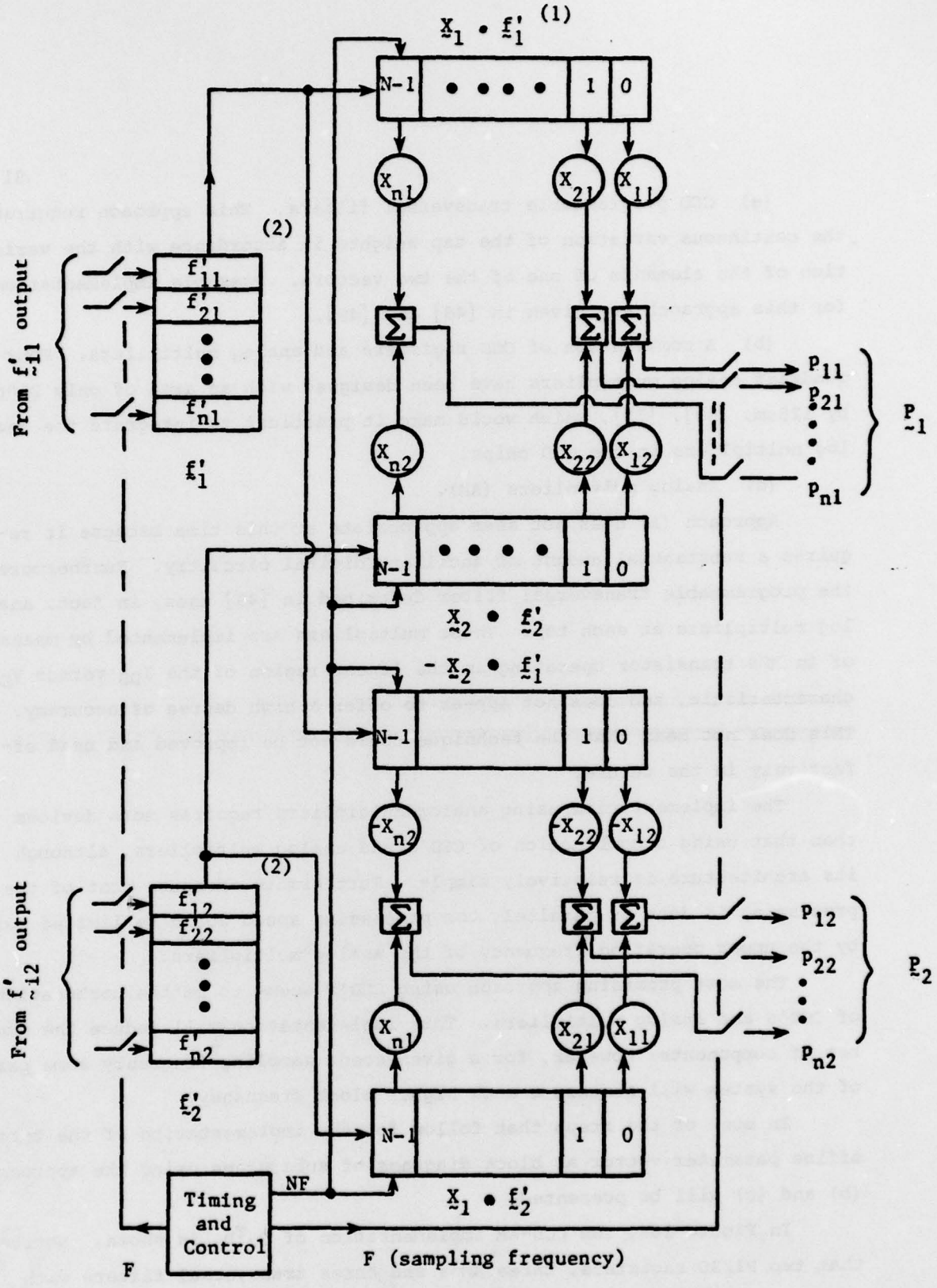


Figure 13 Implementation of Matrix P_c

NOTES: (1) $\underset{\sim}{X}_1 \bullet \underset{\sim}{f}'_j$ denotes the Hadamard products of the vectors $\underset{\sim}{X}_i$ and $\underset{\sim}{f}'_j$.

(2) These PI/SO registers are necessary for the implementation of C, Fig. 17, and of $\hat{a}_c$, Fig. 19.

(a) CCD programmable transversal filters. This approach requires the continuous variation of the tap weights in accordance with the variation of the elements of one of the two vectors. Possible implementations for this appraoch are given in [48] and [49].

(b) A combination of CCD registers and analog multipliers. Four quadrant analog multipliers have been designed with an area of only 250μm by 125μm, [50], [51], which would make it practical to integrate the analog multipliers in the CCD chips.

(c) Analog multipliers (AM).

Approach (a) does not seem appropriate at this time because it requires a substantial amount of ancillary digital circuitry. Furthermore, the programmable transversal filter described in [49] uses, in fact, analog multipliers at each tap. These multipliers are implemented by means of an MOS transistor operating in the linear region of the I_{DS} versus V_{DS} characteristic, and does not appear to offer a high degree of accuracy. This does not mean that the technique could not be improved and used effectively in the future.

The implementation using analog multipliers requires more devices than that using a combination of CCD's and analog multipliers, although its architecture is relatively simple. Furthermore, because most of the processing is done in parallel, the processing speed would be limited mainly by the upper operating frequency of the analog multipliers.

The most promising approach using CCD's seems to be the combination of CCD's and analog multipliers. This implementation will reduce the number of components; however, for a given scene sampling frequency some parts of the system will require a much higher clock frequency.

In most of the steps that follow for the implementation of the total affine parameter vector $\underline{a}$, block diagrams of subsystems using the approaches (b) and (c) will be presented.

In Figure 14a, the CCD-AM implementation of $P_c^T P_c$ is shown. Notice that two PI/SO registers, three AM's and three transversal filters with unity tap weights are necessary. For the AM implementation, Figure 14b, 3N AMs plus 3 summers are required. Figures 14a and 14b also include the necessary blocks for the inversion of $P_c^T P_c$.

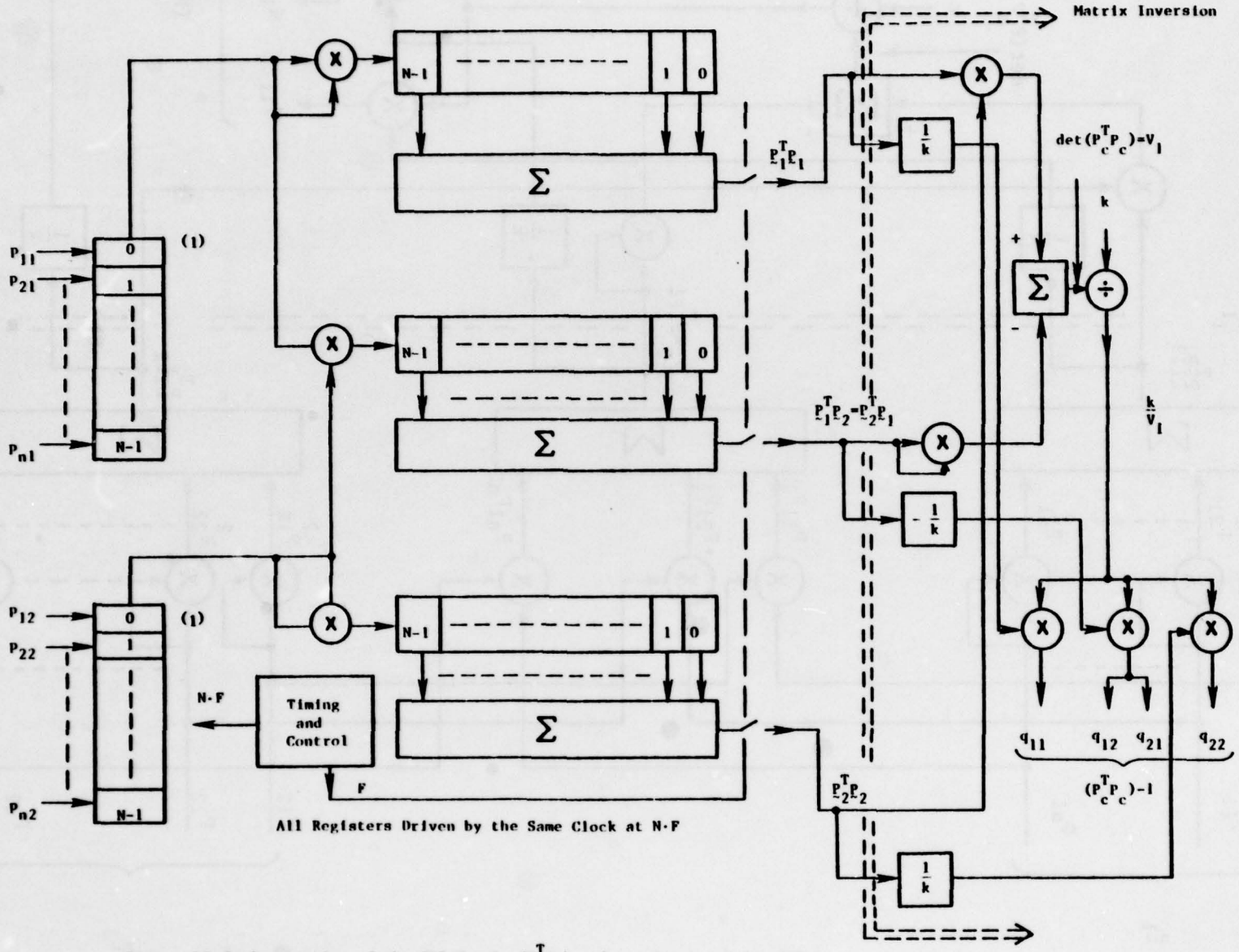


Figure 14a Implementation of the 2X2 Matrix $(P_c^T P_c)$ and Its Inverse Using CCD Registers and Analog Multipliers

(1) These registers are also used in other parts of the system.

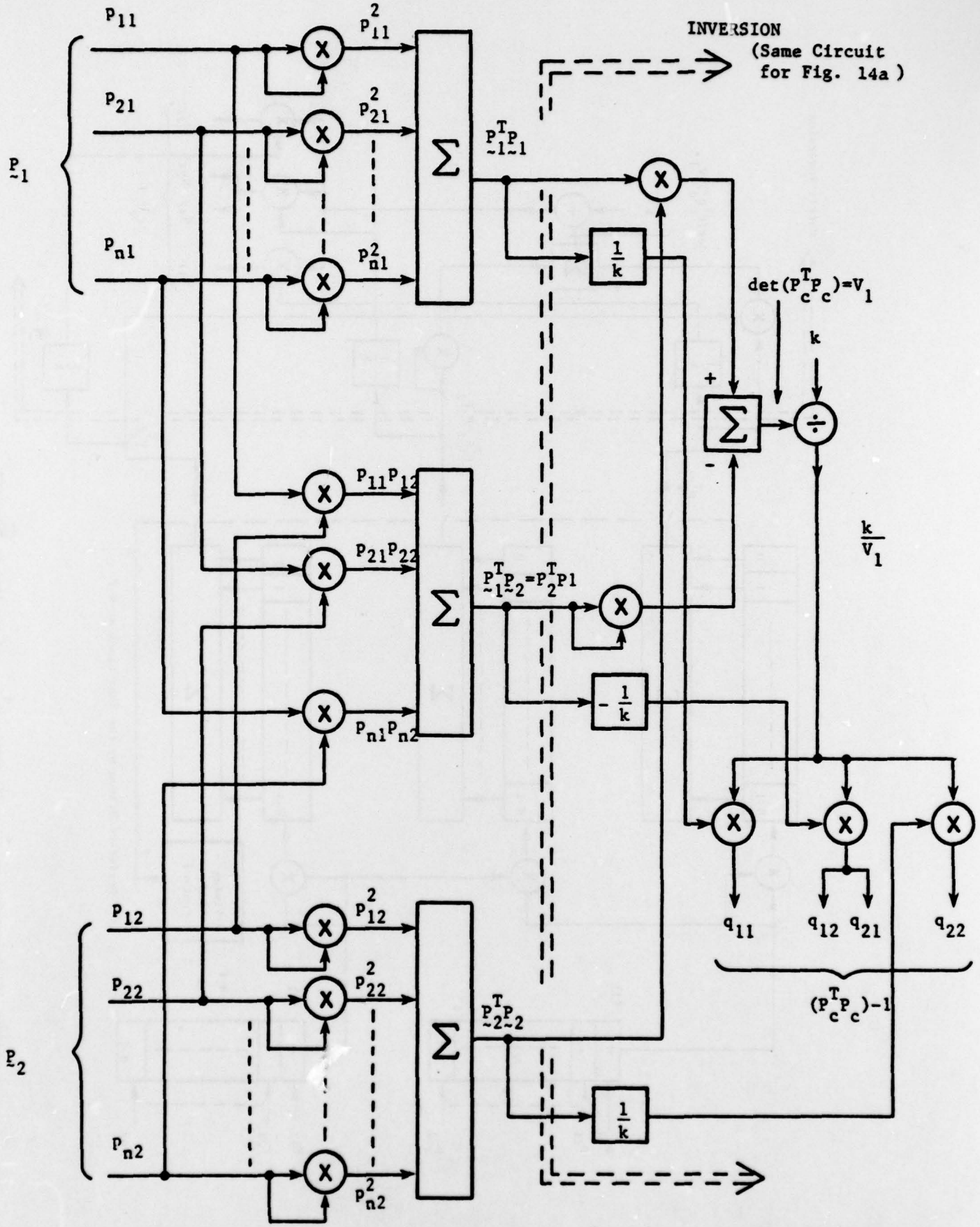


Figure 14b Implementation of the 2X2 Matrix $(P_c^TP_c)^{-1}$ Using Analog Multipliers.

In Figure 14 we have defined

$$(P_c{}^T P_c)^{-1} \triangleq \begin{bmatrix} q_{11} & q_{12} \\ q_{21} & q_{22} \end{bmatrix} = \frac{1}{v_1} \begin{bmatrix} \sum_{i=1}^{N} p_{i2}^2 & -\sum_{i=1}^{N} p_{i1}p_{i2} \\ -\sum_{i=1}^{N} p_{i1}p_{i2} & \sum_{i=1}^{N} p_{i1}^2 \end{bmatrix} \quad (6.2.1.6)$$

where v_1 is the determinant of $P_c{}^T P_c$.

The four elements of this matrix and the 2N elements of the $P_c{}^T$ matrix are now available in parallel form (Figs. 14 and 13, respectively). The pseudo inverse of P_c, $P_c{}^\dagger = (P_c{}^T P_c)^{-1} P_c{}^T$ can now be computed. It is

$$P_c^\dagger = (P_c{}^T P_c)^{-1} P_c{}^T \triangleq \begin{bmatrix} Q_{11} & Q_{12} & \cdots & Q_{1N} \\ Q_{21} & Q_{22} & \cdots & Q_{2N} \end{bmatrix} \quad (6.2.1.7)$$

where, for the first row

$$Q_{1j} = \frac{1}{v_1}\left(p_{j1}\sum_{i=1}^{N} p_{i2}^2 - p_{j2}\sum_{i=1}^{N} p_{i1}p_{i2}\right), \qquad j=1,2,\ldots,N \quad (6.2.1.8)$$

and for the second row

$$Q_{2j} = \frac{1}{v_1}\left(p_{j2}\sum_{i=1}^{N} p_{i1}^2 - p_{j1}\sum_{i=1}^{N} p_{i1}p_{i2}\right), \qquad j=1,2,\ldots,N \quad (6.2.1.9)$$

or,

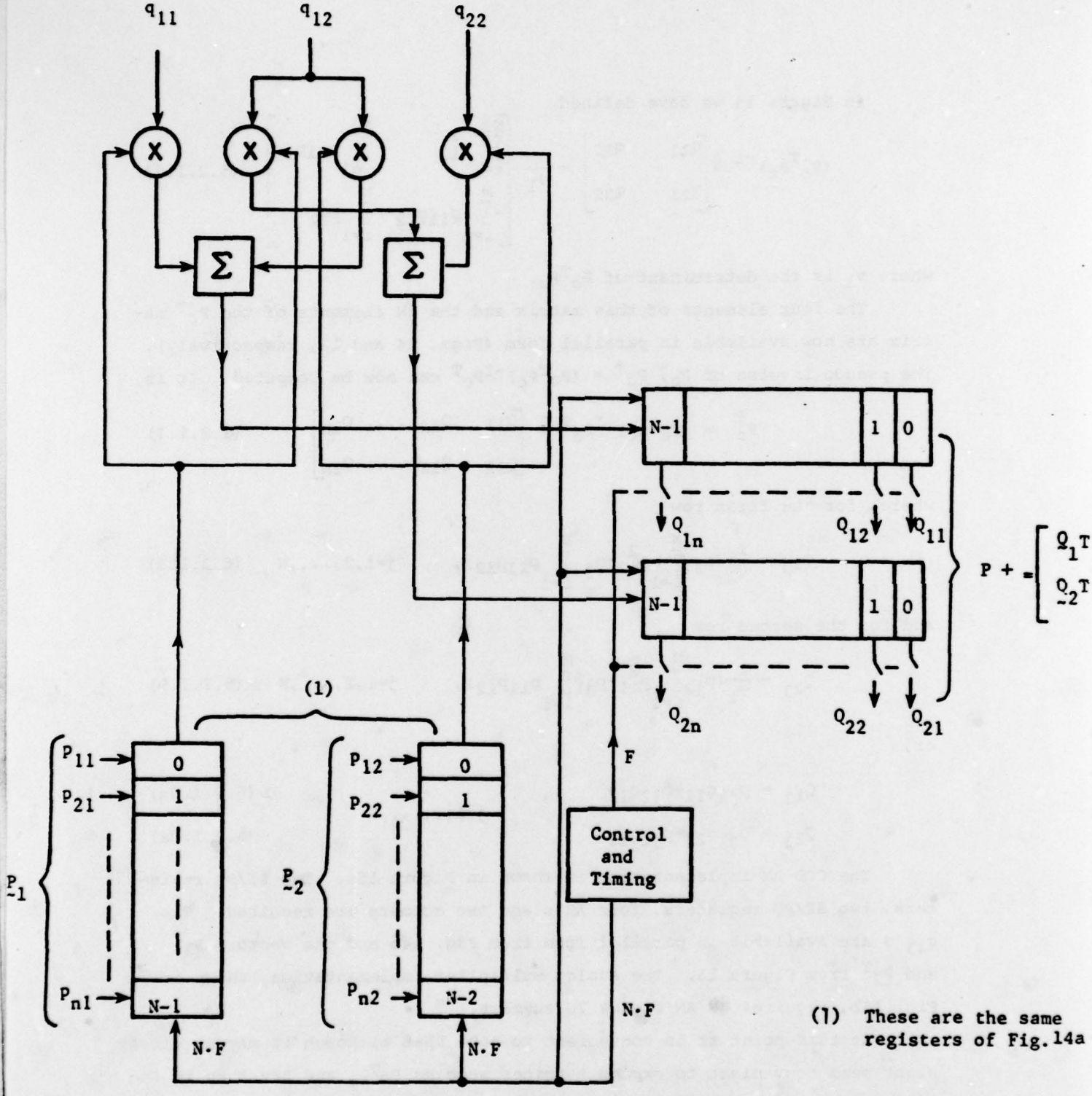
$$Q_{ij} = p_{j1}q_{11} + p_{j2}q_{12} \qquad (6.2.1.8a)$$

$$Q_{2j} = p_{j2}q_{22} + p_{j1}q_{12} \qquad (6.2.1.9a)$$

$$j=1,2,\ldots,N$$

The CCD-AM implementation is shown in Figure 15a. Two PI/SO registers, two SI/PO registers, four AM's and two summers are required. The q_{ij}'s are available in parallel form from Fig. 14b and the vectors $\underset{\sim}{P}_1{}^T$ and $\underset{\sim}{P}_2{}^T$ from Figure 13. The analog multiplier implementation, shown in Fig. 15b, requires 4N AM's plus 2N summers.

At this point it is convenient to note that although it may at first sight seem convenient to expand matrices such as $P_c{}^T P_c$ and try then to implement the TSVIP in terms of the elements of the expanded matrices, this is not definite. A little matrix algebra shows that for the first element



(1) These are the same registers of Fig. 14a

Figure 15a CCD-AM Implementation of P^+

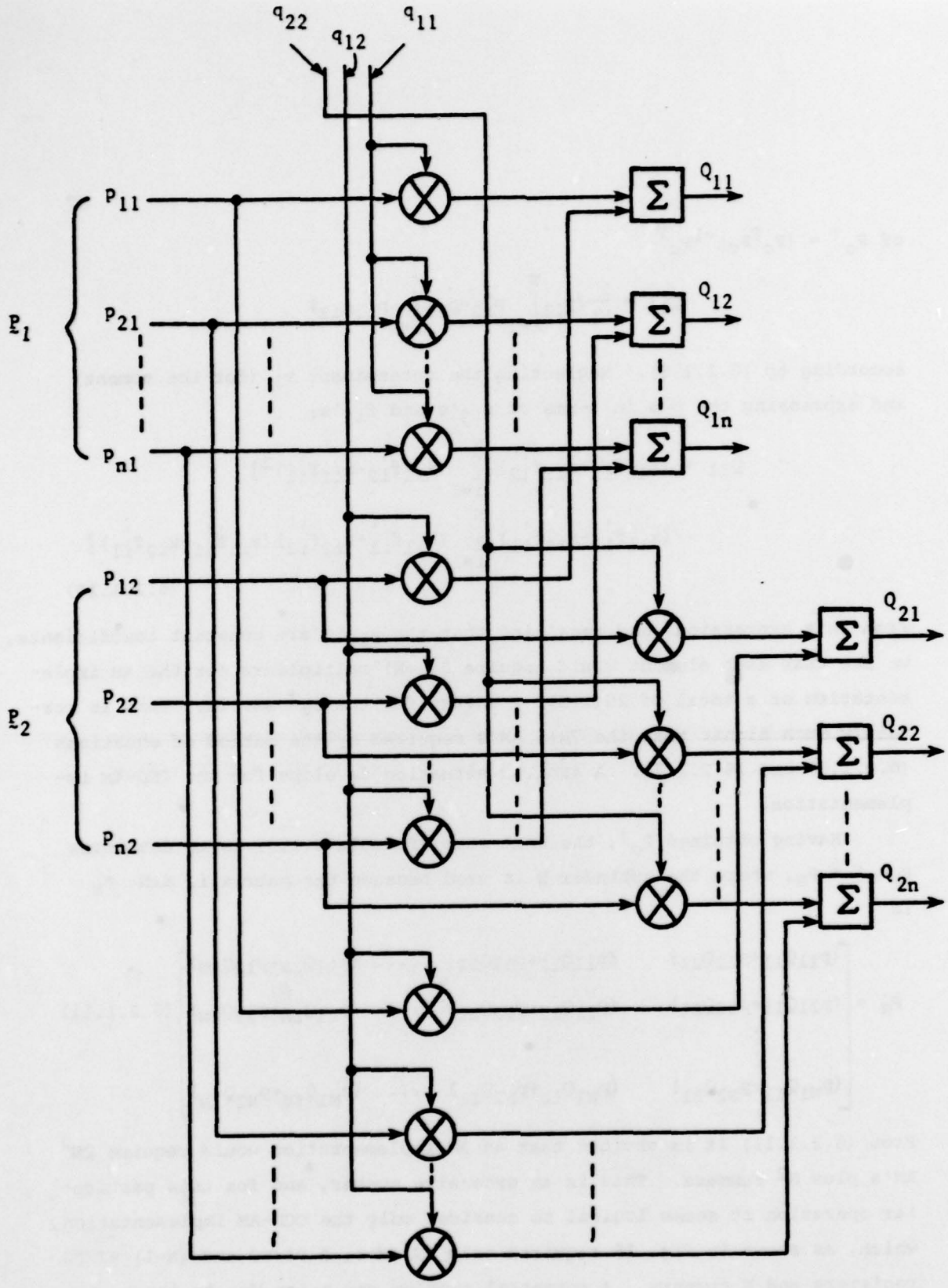


Figure 15b AM Implementation of P_c^+

of $P_c^{\dagger} = (P_c^T P_c)^{-1} P_c^T$

$$Q_{11} = \frac{1}{v_1}\left(p_{11}\sum_{i=1}^{N} p_{i2}^2 - p_{i2}\sum_{i=1}^{N} p_{i1}p_{i2}\right)$$

according to (6.2.1.8). Neglecting the determinant v_1 (for the moment) and expressing the p's in terms of x_{ij}'s and f_{ij}'s:

$$Q_{11} = (x_{11}f'_{11}+x_{12}f'_{12})\left[\sum_{i=1}^{N}(x_{i1}f'_{i2}-x_{i2}f'_{i1})^2\right] - (x_{11}f'_{12}-x_{12}f'_{11})\left[\sum_{i=1}^{N}(x_{i1}f_{i1}'+x_{i2}f'_{i2})(x_{i1}f'_{i1}-x_{i2}f'_{i1})\right] \quad (6.2.1.10)$$

From this expression, and recalling that the x_{ij}'s are constant coefficients, we see that each element would require 2(1+N) multipliers for the AM implementation or a total of $2N(1+N)2 = 4N+4N^2$ for the $P_c^{\dagger}$ matrix. This is certainly much higher than the 7N+5 AM's required by the method of equations (6.2.1.6) and (6.2.1.7). A similar situation develops for the CCD-AM implementation.

Having obtained $P_c^{\dagger}$, the next step (formula 6.1.5) is to determine $P_c P_c^{\dagger} \triangleq P_N$, where the subindex N is used because the matrix is NxN, P_N is

$$P_N = \begin{bmatrix} (p_{11}Q_{11}+p_{12}Q_{21}) & (p_{11}Q_{12}+p_{12}Q_{22}) & \cdots\cdots & (p_{11}Q_{1N}+p_{12}Q_{2N}) \\ (p_{21}Q_{11}+p_{22}Q_{21}) & (p_{21}Q_{12}+p_{22}Q_{22}) & \cdots\cdots & (p_{21}Q_{1N}+p_{22}Q_{2N}) \\ \vdots & & & \\ (p_{N1}Q_{11}+p_{N2}Q_{21}) & (p_{N1}Q_{12}+p_{N2}Q_{22}) & \cdots\cdots & (p_{N1}Q_{1N}+p_{N2}Q_{2N}) \end{bmatrix} \quad (6.2.1.11)$$

From (6.2.1.11) it is obvious that an AM implementation would require $2N^2$ AM's plus N^2 summers. This is an excessive number, and for this particular operation it seems logical to consider only the CCD-AM implementation, which, as shown in Fig. 16 requires only 2N AM's, 3 PI/SO and (N-1) SI/PO registers and N summers. A potential problem can arise due to the high frequency required for the SI/SO registers (see Section 6.3).

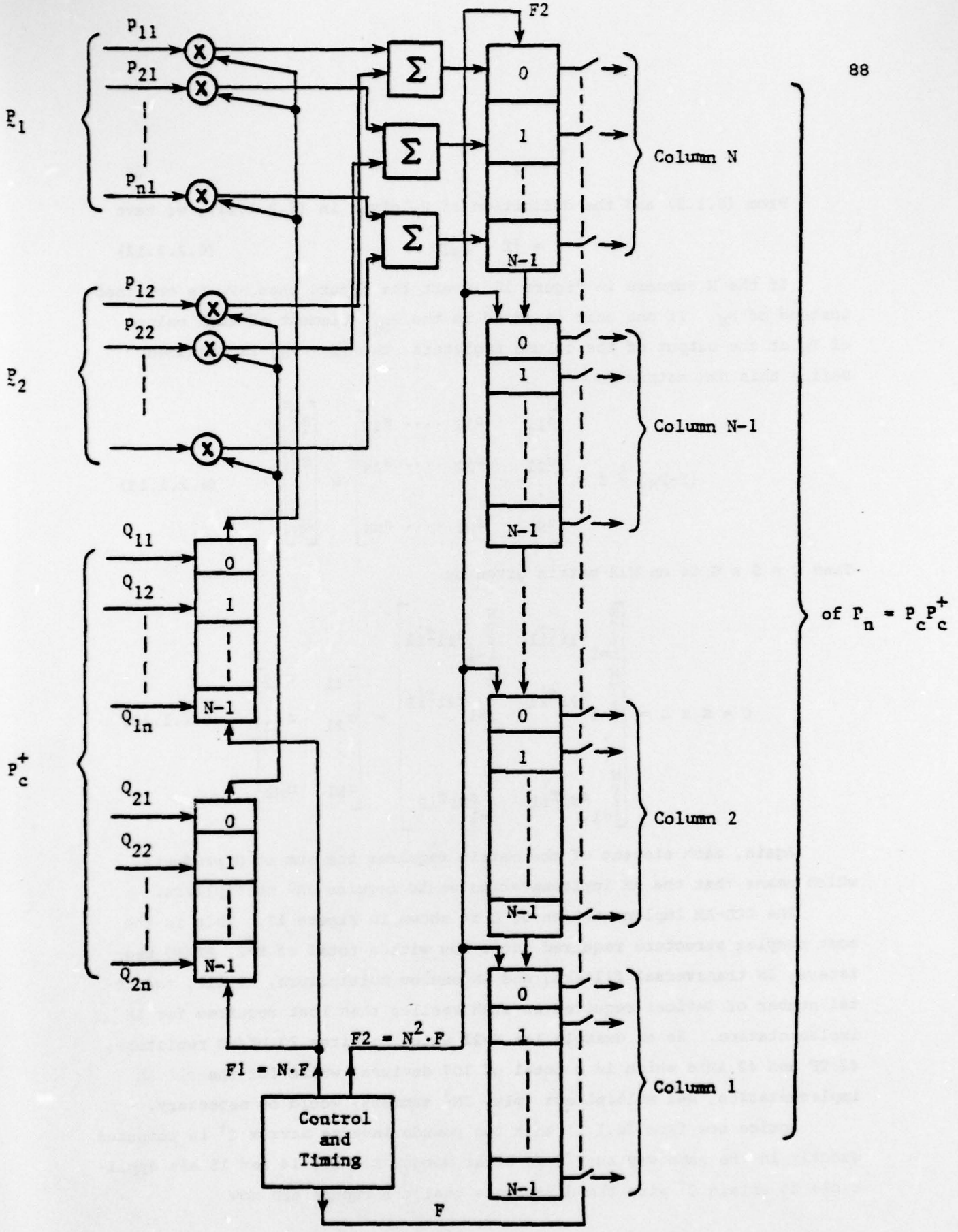


Figure 16 CCD-AM Implementation of the NxN Matrix $P_c P_c^+$

From (6.1.5) and the definition of P_N given in (6.2.1.11), we have

$$C = [I - P_N]G \tag{6.2.1.12}$$

If the N summers in Figure 16 invert the input, then $-P_N$ is obtained, instead of P_N. If one unit is added to the $P_{N_{ii}}$ element of each column of P_N at the output of the column registers, then $[I - P_N]$ is obtained. Define this NxN matrix as

$$[I-P_N] \triangleq S = \begin{bmatrix} s_{11} & s_{12} \cdots s_{1N} \\ s_{21} & s_{22} \cdots s_{2N} \\ \vdots & \\ s_{N1} & s_{N2} \cdots s_{NN} \end{bmatrix} \triangleq \begin{bmatrix} \underset{\sim}{S}_1^T \\ \underset{\sim}{S}_2^T \\ \vdots \\ \underset{\sim}{S}_N^T \end{bmatrix} \tag{6.2.1.13}$$

Then C = S x G is an Nx2 matrix given by

$$C = S \times G = \begin{bmatrix} \sum_{i=1}^{N} s_{1i}f'_{i1} & \sum_{i=1}^{N} s_{1i}f'_{i2} \\ \sum_{i=1}^{N} s_{2i}f'_{i1} & \sum_{i=1}^{N} s_{2i}f'_{i2} \\ \vdots & \vdots \\ \sum_{i=1}^{N} s_{Ni}f'_{i1} & \sum_{i=1}^{N} s_{Ni}f'_{i2} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \\ \vdots & \\ c_{N1} & c_{N2} \end{bmatrix} \tag{6.2.1.13}$$

Again, each element of the matrix requires the sum of N products, which means that the AM implementation would require $2N^2$ multipliers.

The CCD-AM implementation of C is shown in Figure 17. This is the most complex structure required until now with a total of N+2 PI/SO registers, 2N transversal filters, and 2N analog multipliers. Still, the total number of devices required is much smaller than that required for AM implementation. As an example let N=21 which requires 23 PI/SO registers, 42 TF and 42 AM's which is a total of 107 devices, while for the all AM implementation, 441 multipliers (plus $2N^2$ summers) would be necessary.

Notice now from (6.1.5) that the pseudo inverse matrix C^+ is computed exactly in the same way as P^+. In other words, Figures 14 and 15 are applicable to obtain C^+ with the difference that the inputs are now

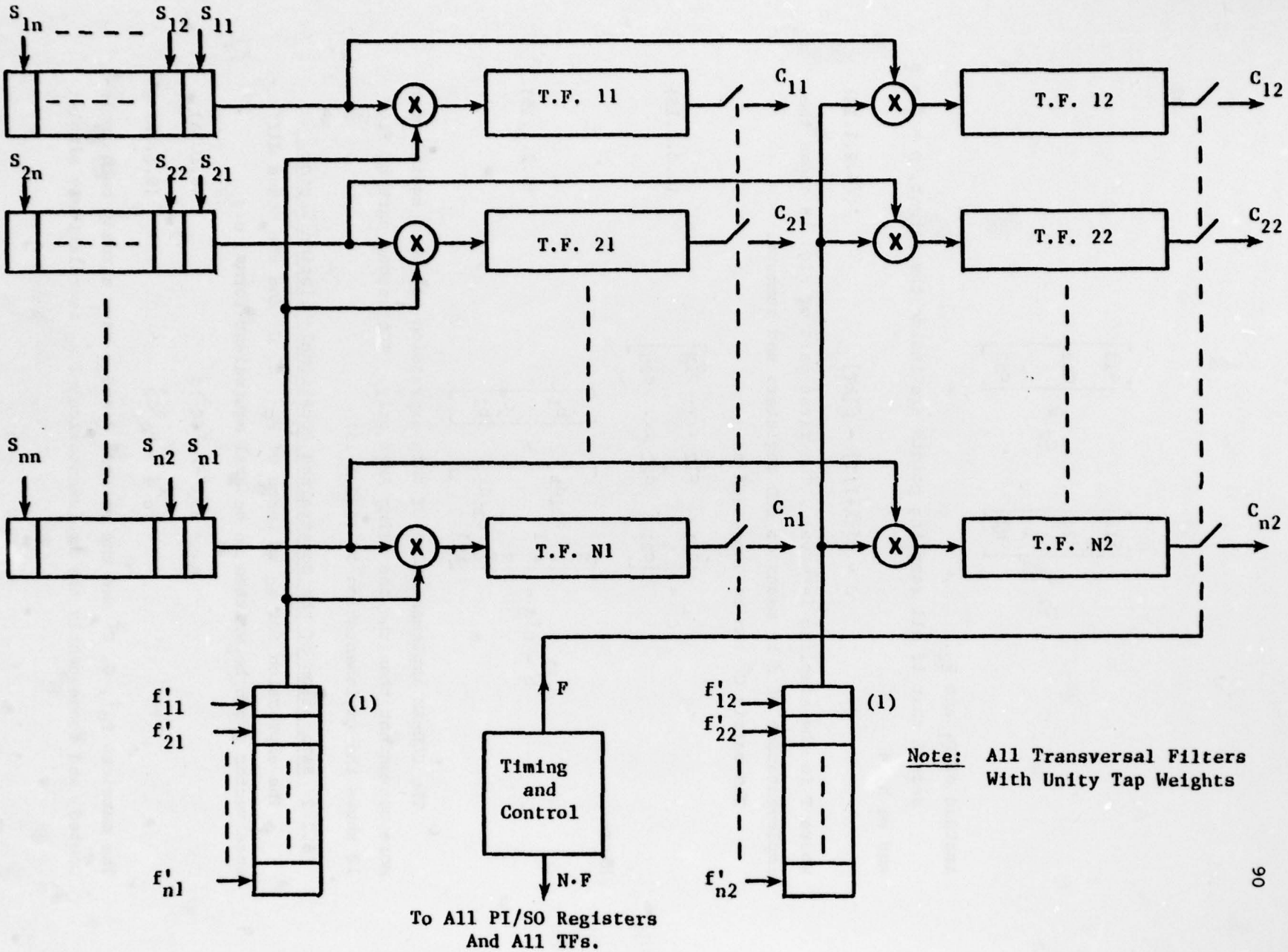


Figure 17 Implementation of the C Matrix

(1) These are the same registers of figure 13

$$\underset{\sim}{C}_1 \triangleq \begin{bmatrix} c_{11} \\ c_{21} \\ \vdots \\ c_{N1} \end{bmatrix} \qquad \underset{\sim}{C}_2 \triangleq \begin{bmatrix} c_{12} \\ c_{22} \\ \vdots \\ c_{N2} \end{bmatrix}$$

instead of $\underset{\sim}{P}_1$ and $\underset{\sim}{P}_2$.

Recall that if all sampling points are inside the target, $\underset{\sim}{p} = \underset{\sim}{f} = \underset{\sim}{s}$ and we have

$$\underset{\sim}{d} = \underset{\sim}{p}[(k+1)T] - \underset{\sim}{p}[kT] \qquad (6.2.1.14)$$

where T is the sampling interval. The first part of Fig. 18 shows the implementation of $\underset{\sim}{d}$ by means of CCD registers and summers.

The matrix $C^\dagger$ can be expressed as

$$C^\dagger = \begin{bmatrix} c_{11}^\dagger & c_{12}^\dagger \ \ldots\ . \ c_{1N}^\dagger \\ c_{21}^\dagger & c_{22}^\dagger \ \ldots\ . \ c_{2N}^\dagger \end{bmatrix} \qquad (6.2.1.15)$$

Hence

$$\underset{\sim}{\hat{b}} = C^\dagger \underset{\sim}{d} = \begin{bmatrix} \sum_{i=1}^{N} c_{1i}^\dagger d_i \\ \sum_{i=1}^{N} c_{2i}^\dagger d_i \end{bmatrix} = \begin{bmatrix} \hat{b}_1 \\ \hat{b}_2 \end{bmatrix} \qquad (6.2.1.16)$$

The CCD-AM implementation of this expression is, once again, much more convenient than the one using AM's only. The second part of Fig. 18 shows the implementation of (6.2.1.16)

6.2.2 Estimation of the Constrained Rotational-Dilation Vector, $\hat{\underset{\sim}{a}}_c$

The expression for $\underset{\sim}{a}_c$ in terms of $P_c^\dagger$, G, $C^\dagger$ and the scene difference vector $\underset{\sim}{d}$ can be written in several equivalent forms, e.g.,

$$\hat{\underset{\sim}{a}}_c = (P_c^\dagger - P_c^\dagger G C^\dagger)\underset{\sim}{d} \qquad (6.1.7a)$$

$$= P_c^\dagger \underset{\sim}{d} - P_c^\dagger G \hat{\underset{\sim}{b}} \qquad (6.1.7b)$$

The matrices $P_c^\dagger$, G, $C^\dagger$ and the vectors $\underset{\sim}{d}$ and $\hat{\underset{\sim}{b}}$ have already been implemented, and consequently the implementation of $\hat{\underset{\sim}{a}}_c$ is relatively simple.

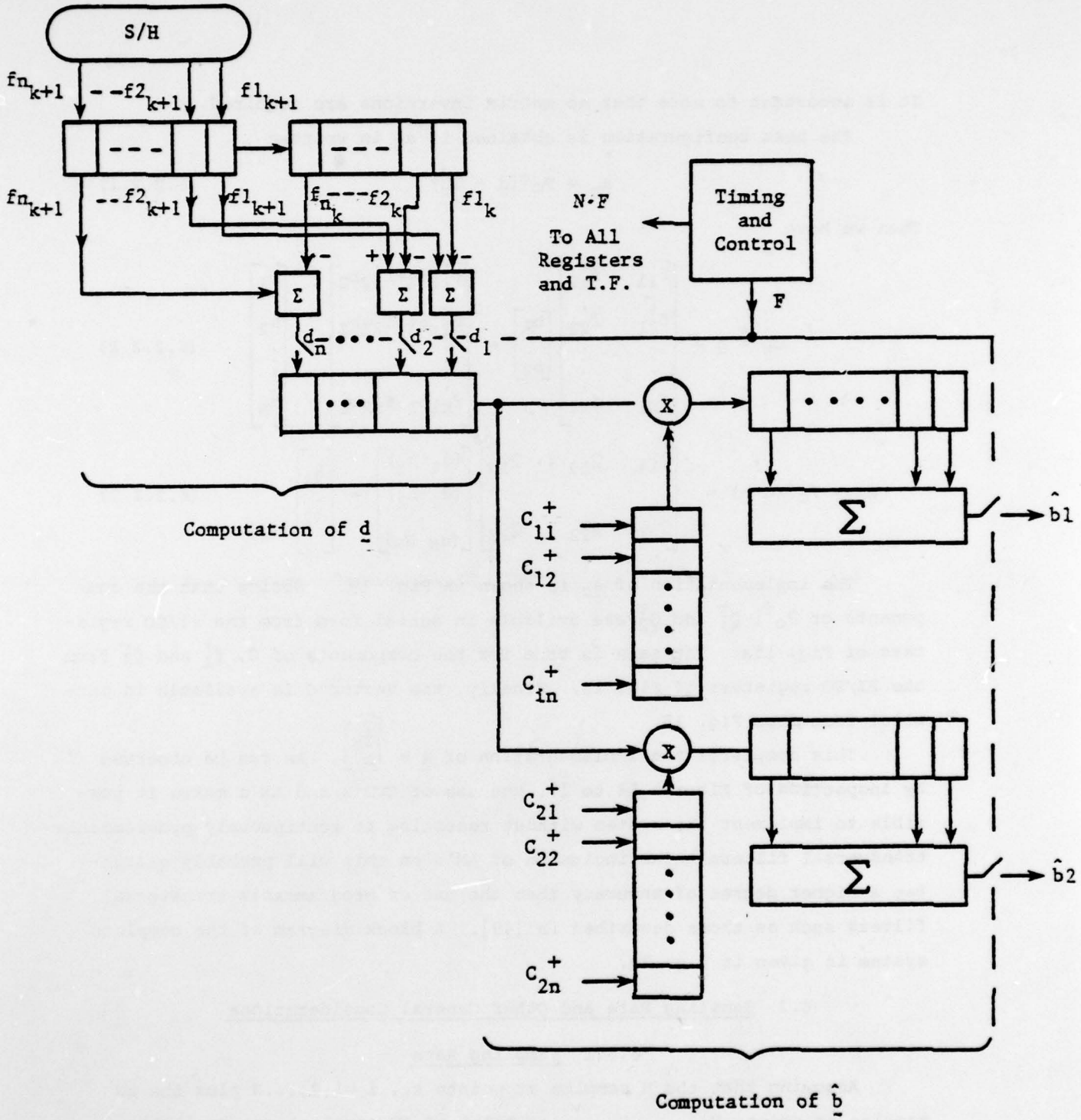


Figure 18 Implementation of the Scene Difference Vector, $\underline{d}$, and the estimate of the Translational Vector, $\hat{\underline{b}}$.

It is important to note that no matrix inversions are required.

The best configuration is obtained if $\hat{\underset{\sim}{a}}_c$ is written

$$\underset{\sim}{a}_c = P_c^{\dagger}(\underset{\sim}{d} - G\hat{\underset{\sim}{b}}) \qquad (6.2.2.1)$$

Then we have

$$G\hat{\underset{\sim}{b}} \triangleq \underset{\sim}{h} = \begin{bmatrix} f'_{11} & f'_{12} \\ f'_{21} & f'_{22} \\ \vdots & \vdots \\ f'_{N1} & f'_{N2} \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} f'_{11}b_1 + f'_{12}b_2 \\ f'_{21}b_1 \quad f'_{22}b_2 \\ \vdots \\ f'_{N1}b_1 \quad f'_{N2}b_2 \end{bmatrix} = \begin{bmatrix} h_1 \\ h_2 \\ \vdots \\ h_N \end{bmatrix} \qquad (6.2.2.2)$$

$$\hat{\underset{\sim}{a}}_c = P_c^{\dagger}(\underset{\sim}{d}-\underset{\sim}{h}) = \begin{bmatrix} Q_{11} & Q_{12} \cdots & Q_{1N} \\ & & \\ Q_{21} & Q_{22} & Q_{2N} \end{bmatrix} \begin{bmatrix} (d_1-h_1) \\ (d_2-h_2) \\ \vdots \\ (d_N-h_N) \end{bmatrix} = \begin{bmatrix} \hat{a}_1 \\ \hat{a}_2 \end{bmatrix} \qquad (6.2.2.3)$$

The implementation of $\hat{\underset{\sim}{a}}_c$ is shown in Fig. 19. Notice that the components of $P_c^{\dagger}$, $\underset{\sim}{Q}_1^T$ and $\underset{\sim}{Q}_2^T$ are avilable in serial form from the PI/SO registers of Fig. 15a. The same is true for the components of G, $\underset{\sim}{f}'_1$ and $\underset{\sim}{f}'_2$ from the PI/SO registers of Fig. 13. Finally, the vector $\underset{\sim}{d}$ is available in parallel form from Fig. 18.

This completes the implementation of $\underset{\sim}{a} = \begin{bmatrix} \hat{\underset{\sim}{a}}_c \\ \hat{\underset{\sim}{b}} \end{bmatrix}$. As can be observed by inspection of Figures 12 to 19, the use of CCD's and AM's makes it possible to implement the system without resorting to continuously programmable transversal filters. The inclusion of AM's on chip will probably guarantee a higher degree of accuracy than the use of programmable transversal filters such as those described in [49]. A block diagram of the complete system is given in Fig. 20.

6.3 Sampling Rate and Other General Considerations

6.3.1 Sampling Rate

Assuming that the N samples at points x_i, i =1,2,...N plus the 2N samples at points $X_{i+\Delta x1}$, $X_{i+\Delta x2}$ (a total of 3N samples) are available in parallel form, a knowledge of the focal length, f; the focal point to target plane perpendicular distance, Z, and the velocity of the target, v_T,

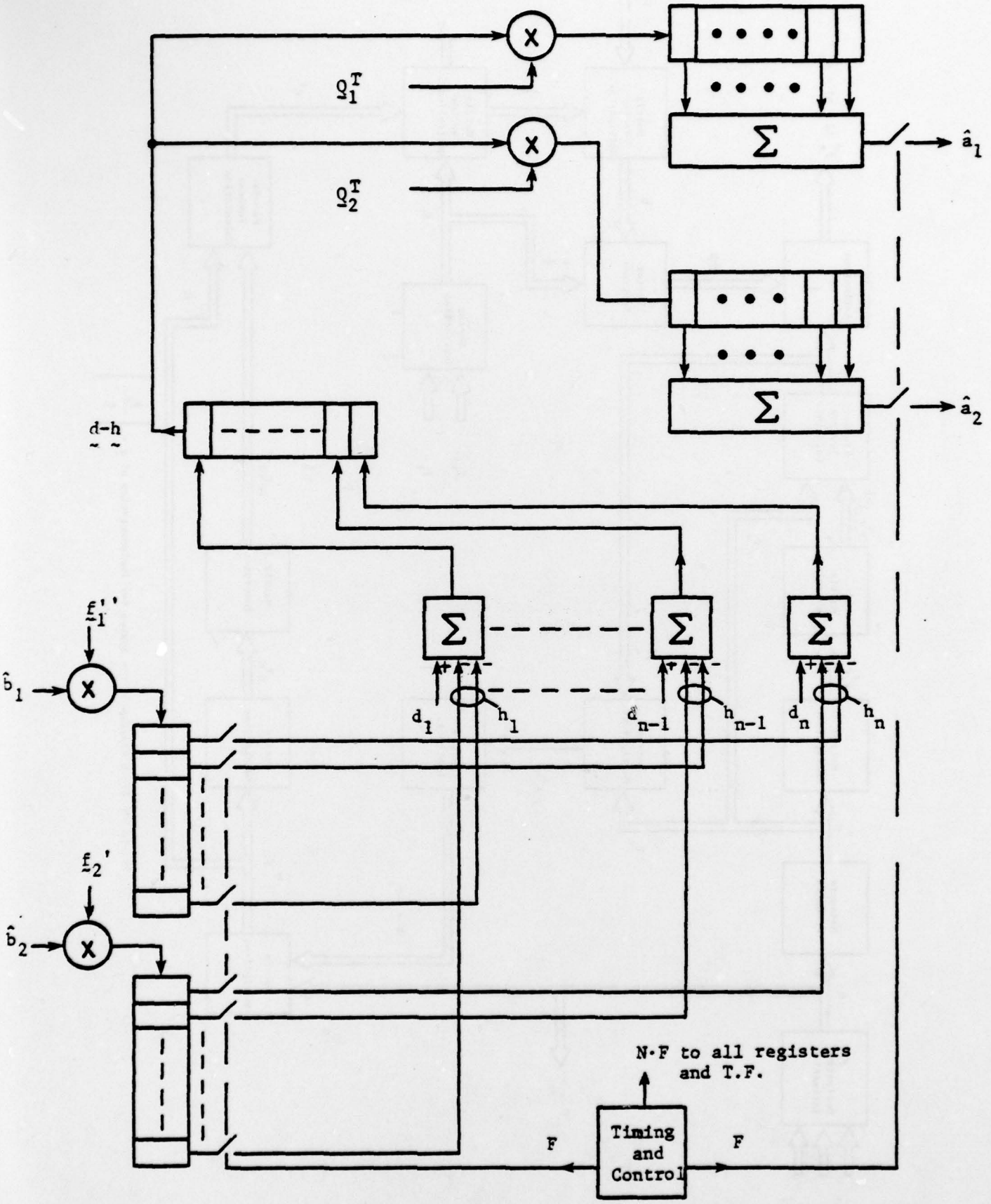


Figure 19 Implementation of the Rotation-Dilation Vector $\hat{\underset{\sim}{a}}_c$

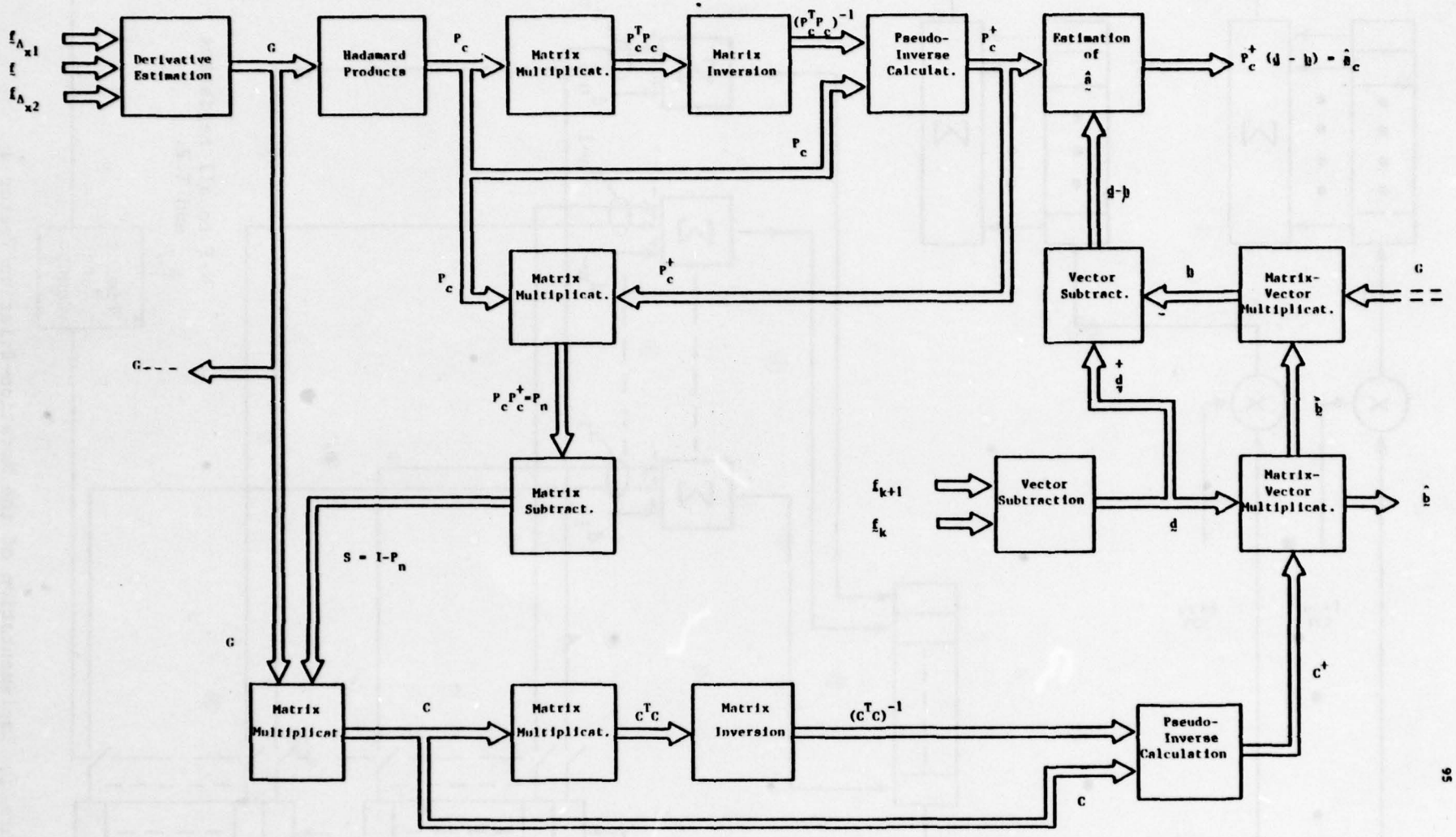


Figure 20 Block Diagram of the System for Implementation of $\hat{a} = \begin{bmatrix} \hat{a}_c \\ \hline b \end{bmatrix}$

in the target plane, allows calculation of the translational vector $\underset{\sim}{b}$ (assuming that dilation is not involved) for the sampling interval T. This assumes a sampling rate, f_s samples/second. With reference to Figure 21 (for simplicity we have considered movement in one dimension), assume

$$f = 50 \text{ mm}, \; Z = 100 \text{ m}, \; v_T = 9000 \frac{\text{Km}}{\text{hr}} \text{ (mach 7.35)}$$

Then

$$\Delta\hat{\underset{\sim}{b}}_i = \frac{f}{Z} \Delta\underset{\sim}{b}_{i_T} = 5\text{x}10^{-4}\Delta\underset{\sim}{b}_{i_T} \qquad (6.3.1.1)$$

The basic assumption for the main body of this work has been that the background varies (evolves) slowly. If for the chosen sampling rate $\Delta\hat{\underset{\sim}{b}}_i$ is very small, then we can certainly assume that

$$b(\underset{\sim}{x},t_1) \simeq b(x,t_2)$$

Here, once again, recall that $\underset{\sim}{b}$ is the target translational vector, and b(x,t) is the background function.

With the figures given above, the speed in the image sensor plane is

$$v_{is} = 5\text{x}10^{-4}\text{x}9\text{x}10^{3} = 4.5 \text{ Km/hr}$$

or

$$v_{is} = 1250 \text{ mm/sec}$$

This appears to indicate that a sampling rate (for all pixels in parallel) of F=1 MHZ is adequate. Then

$$\hat{b}_{i_{k+1}} = \hat{b}_{i_k} + \frac{1250}{10^6} \text{ mm}$$
$$= \hat{b}_{i_k} + 1.25 \text{ } \mu\text{m}$$

The target (in the target plane) will move by a distance

$$\Delta\hat{b}_{i_T} = \frac{9\text{x}10^9}{3600} \cdot \frac{1}{10^6} = 2.5 \text{ mm}$$

The elements of an image sensor device are extremely small. For example the Reticon MC520 100x100 matrix camera has a distance of 60μm between center of elements. The Fairchild CCD211 area image sensor has 244x190 elements

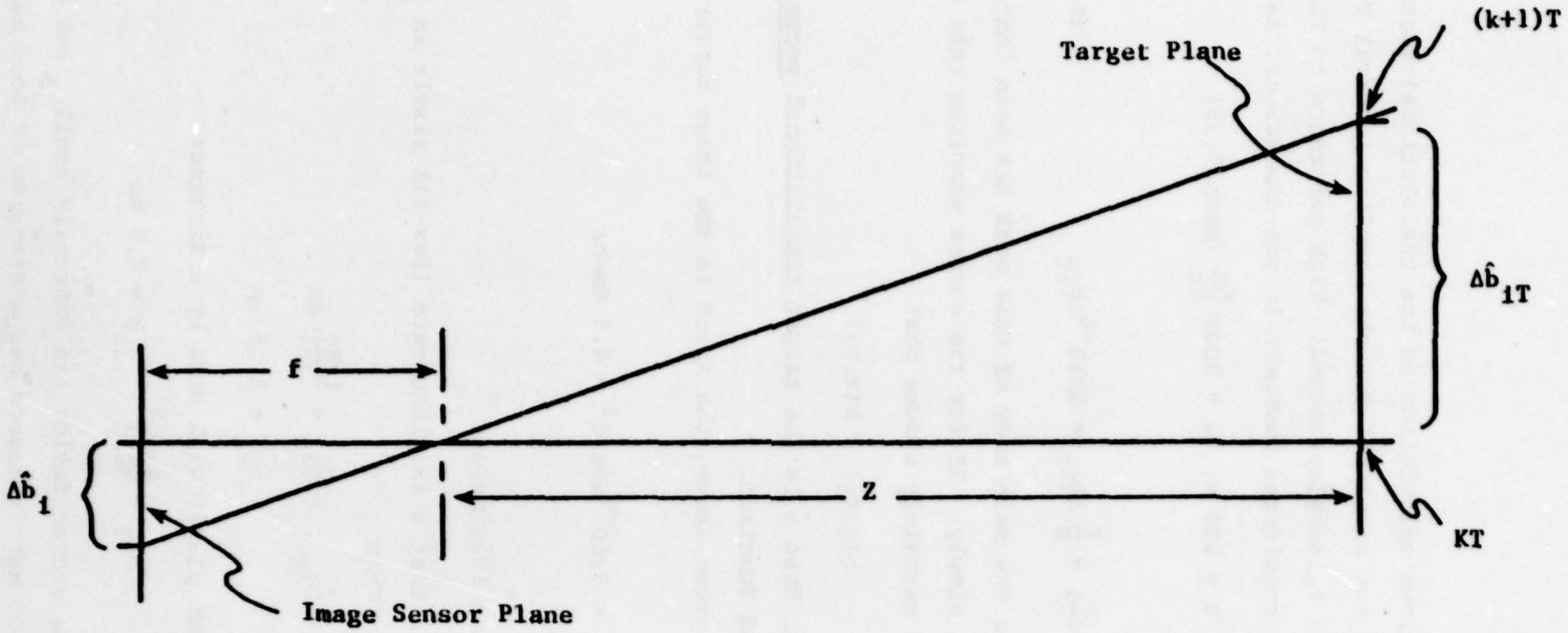


Figure 21 Distances From Target Plane to Lense and From Lense to Image Sensor Plane

with dimensions of 14μm horizontally and 18 m vertically. For both cameras, the change in $\underline{b}$ in the image sensor plane, $\Delta\hat{b}_i$ = 1.25μm, is less than one tenth the smallest dimension of the element, which means that all the assumptions and approximations will be amply satisfied. Obviously, if Z>100 m and/or v_T<9000 Km/hr, a better situation will exist. Taking now the opposite extreme of a very short distance, Z = 1 m, and a very low speed, v_T = 5 m/sec, v_{is} = 0.25 m/sec, $\Delta\hat{b}_i$ = 0.25μm which is even a better estimation than the other case.

6.3.2 Compatibility of CCD's, AM's and Assumed Sampling Rate

For the assumed sampling rate of F=1 MHZ some of the CCD devices and AM's will have to work at a frequency of N·F. Assuming a 10x10 sampling lattice (which once segmentation is achieved will be more than enough), N = 10 and these devices will have to operate at 10 MHZ. With presently available technology this operating frequency will represent no problem for the CCD devices, [52]. Integrated analog multipliers are presently available with bandwidths of only 1 MHZ, although the availability of 10 MHZ hybrid multipliers, [53], seems to indicate that IC AM's with a bandwidth in this range will be available in the near future. Finally, some SI/PO registers require a clock frequency of N^2F, which for N=10, F=1 MHZ will be too high for presently available CCD devices (see Fig. 16). However, peristaltic CCD's with operating frequency of more than 135 MHZ have been reported in the literature [54].

In conclusion, a CCD implementation of the TSVIP appears to be both possible and practical. For a sampling rate of 1 MHZ and lattice size N = 10, some presently available AM's and CCD's may not provide the necessary performance. However, a prototype can be constructed with a lower sampling rate F = 0.5 MHZ and n = 5, which will result in a highest operating frequency of 12.5 MHZ for the CCD's, and of 2.5 MHZ for the AM's. Alternately, the tracking velocity can be reduced to make operation easier to achieve.

CHAPTER VII

SIMULATION

The computer simulation described in this chapter was initiated in order to verify empirically the preceding theoretical results, in particular the validity of the TSVIP algorithm. The interactive video tracking simulation program developed is also useful as an empirical research tool. Using simple analytic functions to represent target texture, simulated TSVIP algorithm performance is shown to follow theoretical predictions very closely.

7.1 Program Description

A detailed flow chart and listing of interactive video tracking program TSVPD is included in Appendix 5. Smooth analytic functions are used to simulate the target textural function, thus this function may be perturbed exactly via the affine transform. Smoothness was necessary in order to permit estimation of spatial derivatives using a simple difference approximation. A 21x21 scene is used, and background and scene noise values are taken from normal distributions. The program computes the original and perturbed scenes in accordance with the model of Section 4.3.3, and then uses scene differences and the TSVIP algorithm (emulating the CCD implementation of Section 5.4.4.3) to estimate target perturbations. The program itself is well-documented through the use of extensive comment statements and the flow chart of App. 5. However, for illustration purposes, a simplified flow chart of program operation is shown in Figure 22. There are a total of 25 parameters or options which must be input, and these are described in the program listing.

7.2 Simulation Summary

As mentioned, simulation was undertaken to verify theoretical derivations, particularly the validity of the Taylor series approximation and estimation algorithm implementation using Cline's theorem. Therefore, no extensive parameter or sensitivity analysis is presented. A

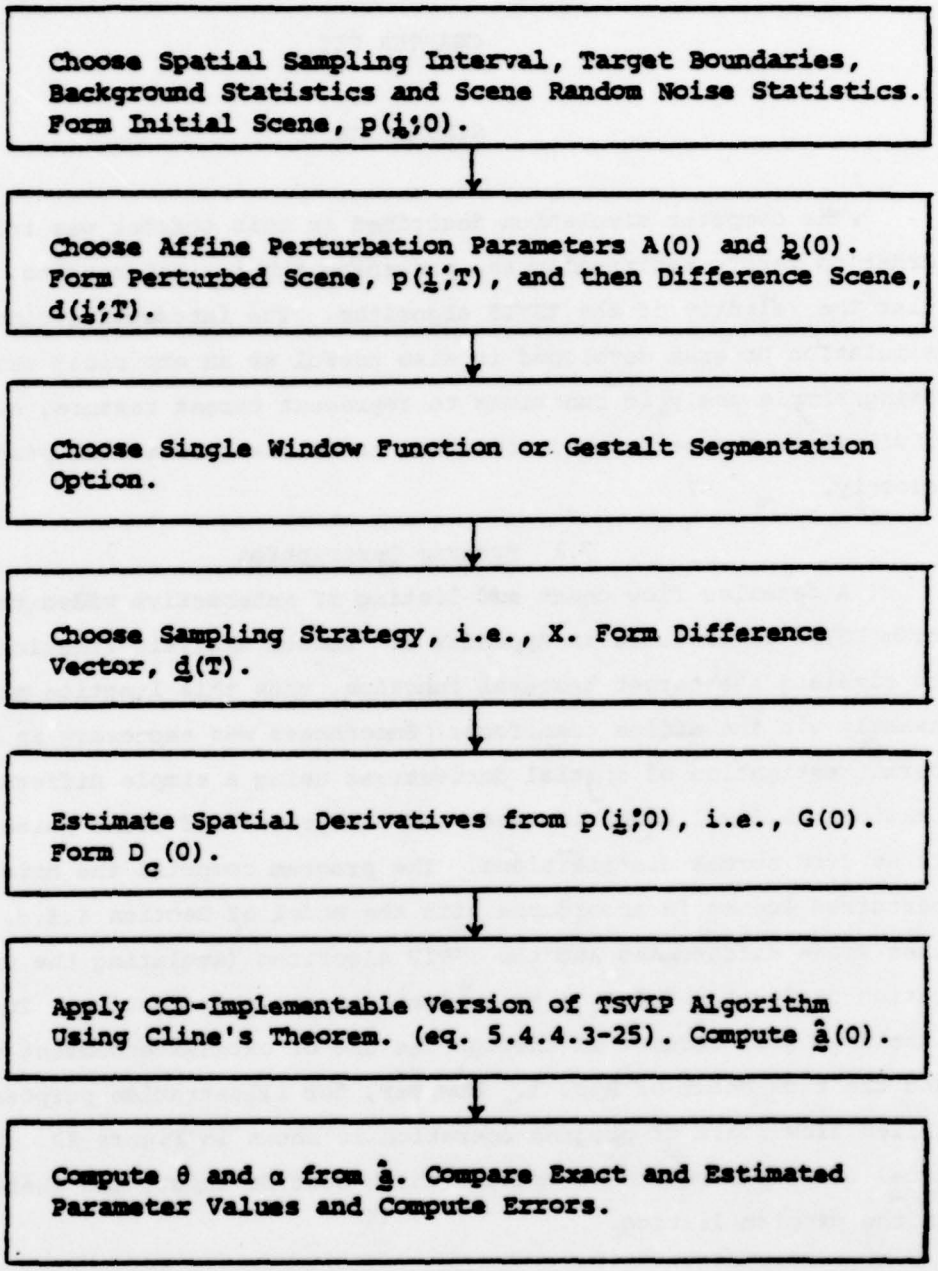


Figure 22

Flow Chart of Interactive Video Tracking Simulation Program TSVPD

number of typical sample cases may be used, however, to confirm the theory.

Unless otherwise noted, the following data was used to obtain the graphical data shown:

(1) Target Textural Function--

$$f(\underset{\sim}{i};0) = \left[\frac{\sin i\Delta}{i\Delta}\right]\left[\frac{\sin(j\Delta/2.5)}{(j\Delta/2.5)} + 0.3j\Delta\right] \qquad (7.2\text{-}1)$$

(2) Spatial Sampling Interval--

$\Delta = 0.2$ "sensor units" (Approximately 10X the Nyquist sampling rate)

(3) Background Statistics--

$b(\underset{\sim}{i};0) \sim N(4,1)$

(4) Target Outline--

Square; $16\Delta x16\Delta$ (Horizontal X vertical dimensions)

An important simulation result is a comparison of exact and estimated target translational parameters, i.e., $\hat{\underset{\sim}{b}}$ and $\underset{\sim}{b}$. Figure 10 compares actual translation parameters and TSVIP algorithm estimates for the case of $\alpha = 1.0$ and $\theta = 0.0$ (i.e., the target does not experience rotation or magnification). Two points should be noted:

(1) The segmentation window was manually chosen such that only target interior points were sampled; and

(2) For simplicity only the first components of the exact and estimated $\underset{\sim}{b}$ vectors are shown, however, the actual exact perturbations were equal in the horizontal and vertical directions, i.e., $b_1 = b_2$, and the results for b_1 and b_2 are similar.

As shown in Figure 23, the TSVIP algorithm performance is quite good for $|b_1| < 0.5$. Note that as the target perturbations increase in magnitude the estimates worsen, as predicted in Section 5.4.6.2. It should also be noted that this empirically-obtained I/O function is approximately

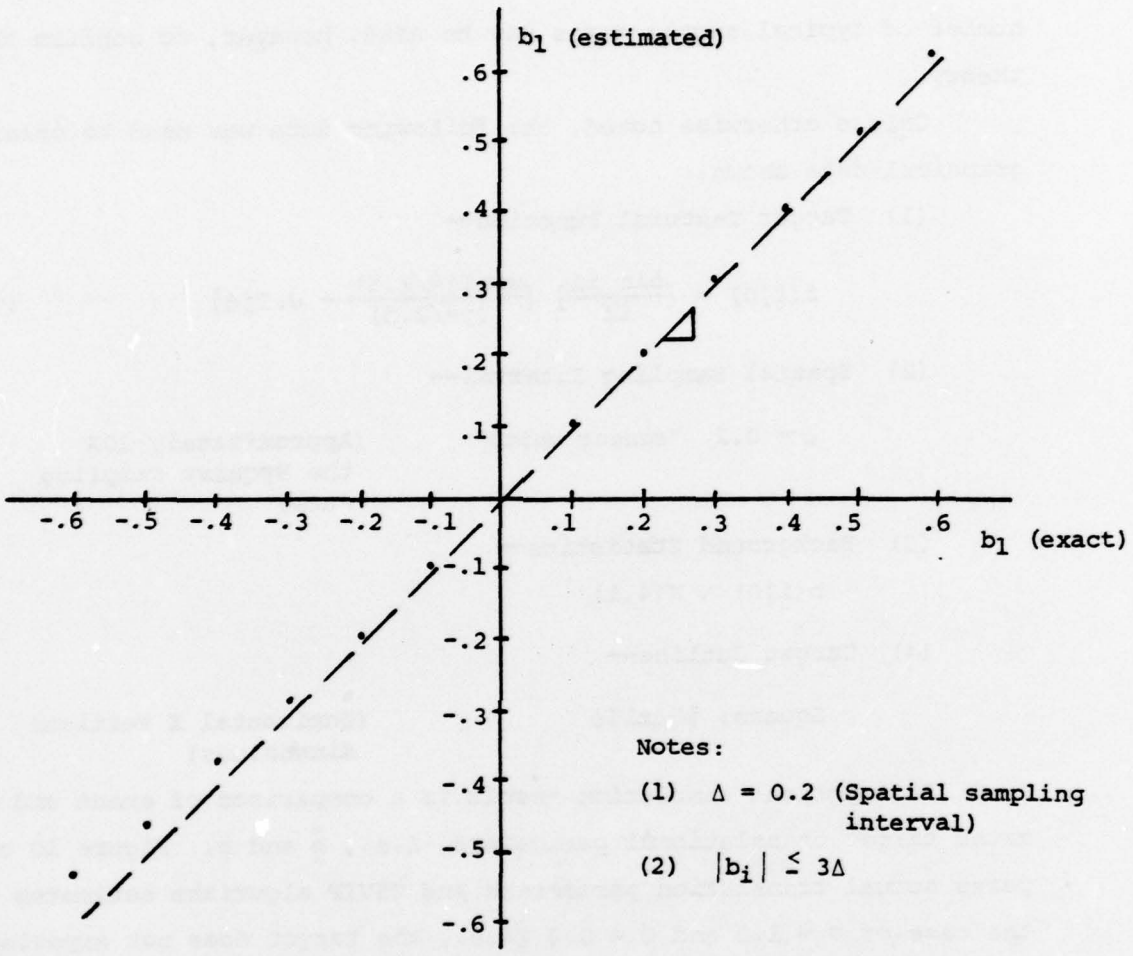


Figure 23

TSVIP Algorithm Translational I/O Function (θ=0.0; α=1.0)
(Other Parameters in Text)

linear, a very useful fact for closed-loop stability analysis. Obviously (in this figure and the following), the dotted line shown with a slope of +1 represents the ideal or error-free tracking processor I/O function.

Perhaps an even more interesting case is shown in Figure 24, where $\alpha = .97$ and $\theta = 3^0$ (i.e., in addition to translation, the target experiences rotation and dilation). The TSVIP translational estimate accuracy in this case is virtually unchanged from that of Figure 10, in other words, the TSVIP algorithm is empirically shown to be insensitive to rotation and dilation. In fact, this empirical result has been verified (for this sample function as well as several others) for cases wherein $|\alpha-1| = .1$ and $|\theta|<10^0$. These empirical results therefore confirm perhaps the most important tracking algorithm design goal mentioned in Section 2.1, i.e., insensitivity to rotation and dilation.

Clearly, the TSVIP algorithm based system is a MIMO system, since there are four scalar I/O functions, i.e.,

(1) $\hat{b}_1$ vs. b_1;

(2) $\hat{b}_2$ vs. b_2;

(3) $\hat{\theta}$ vs. θ; and

(4) $\hat{\alpha}$ vs. α.

The empirical results for all four I/O functions were found to be similar, e.g., Figure 25 shows $\hat{\theta}$ vs. θ for the case of $b_1 = b_2 = 0.1$ and $\alpha = 1.0$.

An obvious practical consideration is the effect of random scene noise (as distinguished from inexact segmentation which causes background data to be processed and results in a different type of process "noise") on the TSVIP algorithm estimates. A sample of this type of empirical result is shown in Figure 24. In order to simulate the type of noise process described in Section 5.4.6.3, a zero mean, normally distributed, independent random process was generated and variates with negative values were discarded. Thus, additive scene noise was therefore always positive, originating from a truncated normal distribution.

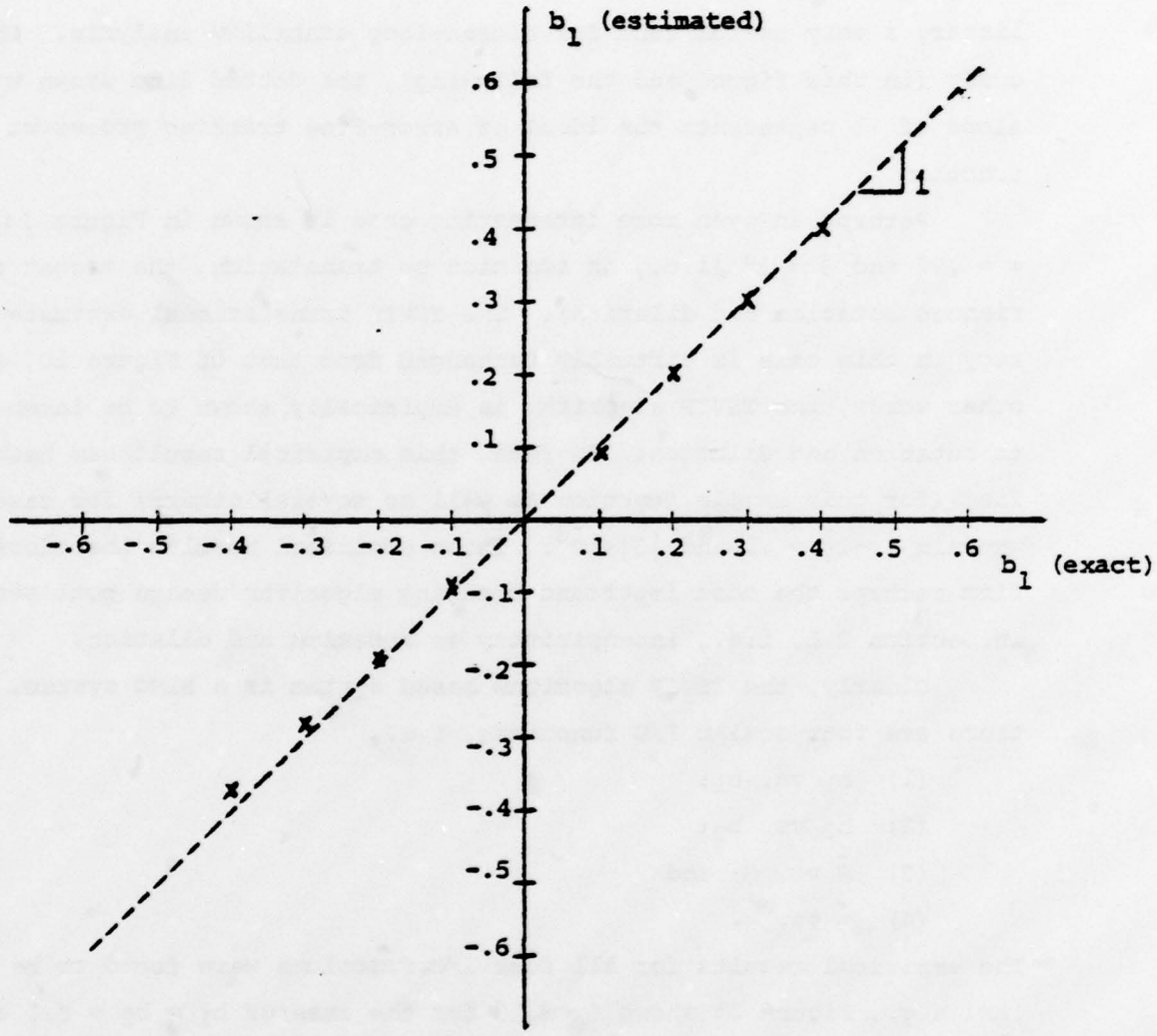


Figure 24

TSVIP Algorithm Translational I/O Function (θ=3.0°;α=0.97)

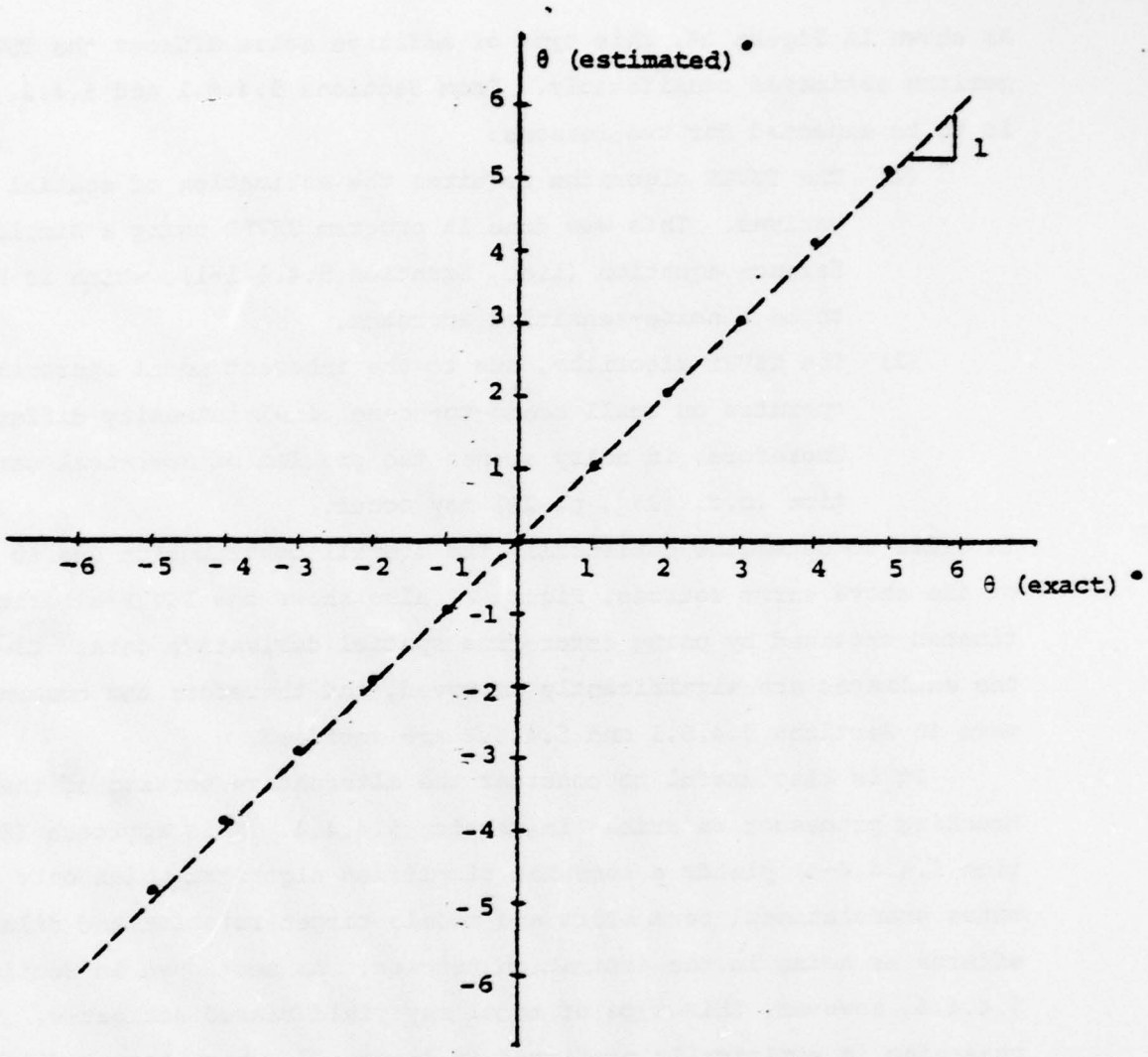


Figure 25

TSVIP Algorithm Rotational I/O Function ($b_1=b_2=.1$; $\alpha=1.0$)

As shown in Figure 26, this type of additive noise affects the TSVIP algorithm estimates considerably. From Sections 5.4.6.1 and 5.4.6.3 this is to be expected for two reasons:

(1) The TSVIP algorithm requires the estimation of spatial derivatives. This was done in program TSVPD using a simple difference equation (i.e., Equation 5.4.6.1-1), which is known to be a noise-sensitive approach.

(2) The TSVIP algorithm, due to the inherent model approximation, operates on small scene-to-scene pixel intensity differences. Therefore, in noisy scenes the problem of numerical cancellation (c.f. [35], p. 28) may occur.

In order to determine empirically the overall contribution due to each of the above error sources, Figure 24 also shows the TSVIP algorithm estimates obtained by using error-free spatial derivative data. Clearly, the estimates are significantly improved, and therefore the comments made in Sections 5.4.6.1 and 5.4.6.3 are verified.

It is also useful to consider the alternative version of the tracking processor described in Section 5.4.4.4. This approach (Equation 5.4.4.4-3) yields a somewhat simplified algorithm which only estimates translational parameters and models target rotation and dilation effects as noise in the estimation process. As mentioned in Section 5.4.4.4, however, this type of model may yield biased estimates. This reasoning is empirically confirmed in Figure 27, where target dilation and positive or negative rotation are shown to cause a bias in the algorithm translational estimates. As predicted by (5.4.4.4-4), for constant α and θ this bias is also constant over the range of $\underline{b}$ considered.

As mentioned in Section 5.4.5.2, an additional error arises in the TSVIP algorithm when the segmentation window is too large and pixels from other than the target interior are sampled. This effect on TSVIP algorithm translational estimates is shown in Figure 28, for two cases with differing background noise statistics. Here the segmentation window was chosen 25% larger than the maximum permissible value, therefore

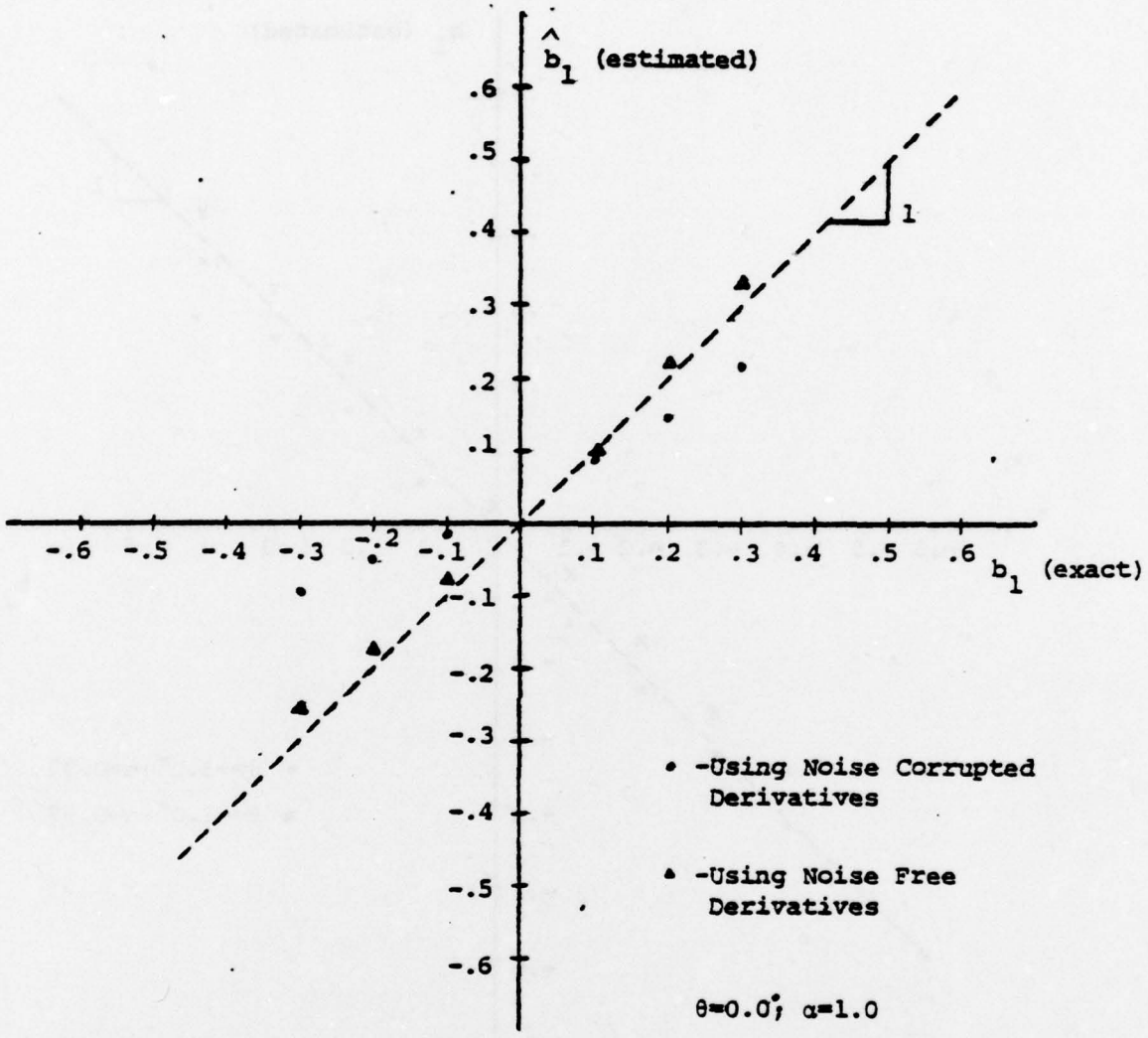


Figure 26

Effect of Random Additive Picture Noise on TSVIP Algorithm (Noise Process Described in Text)

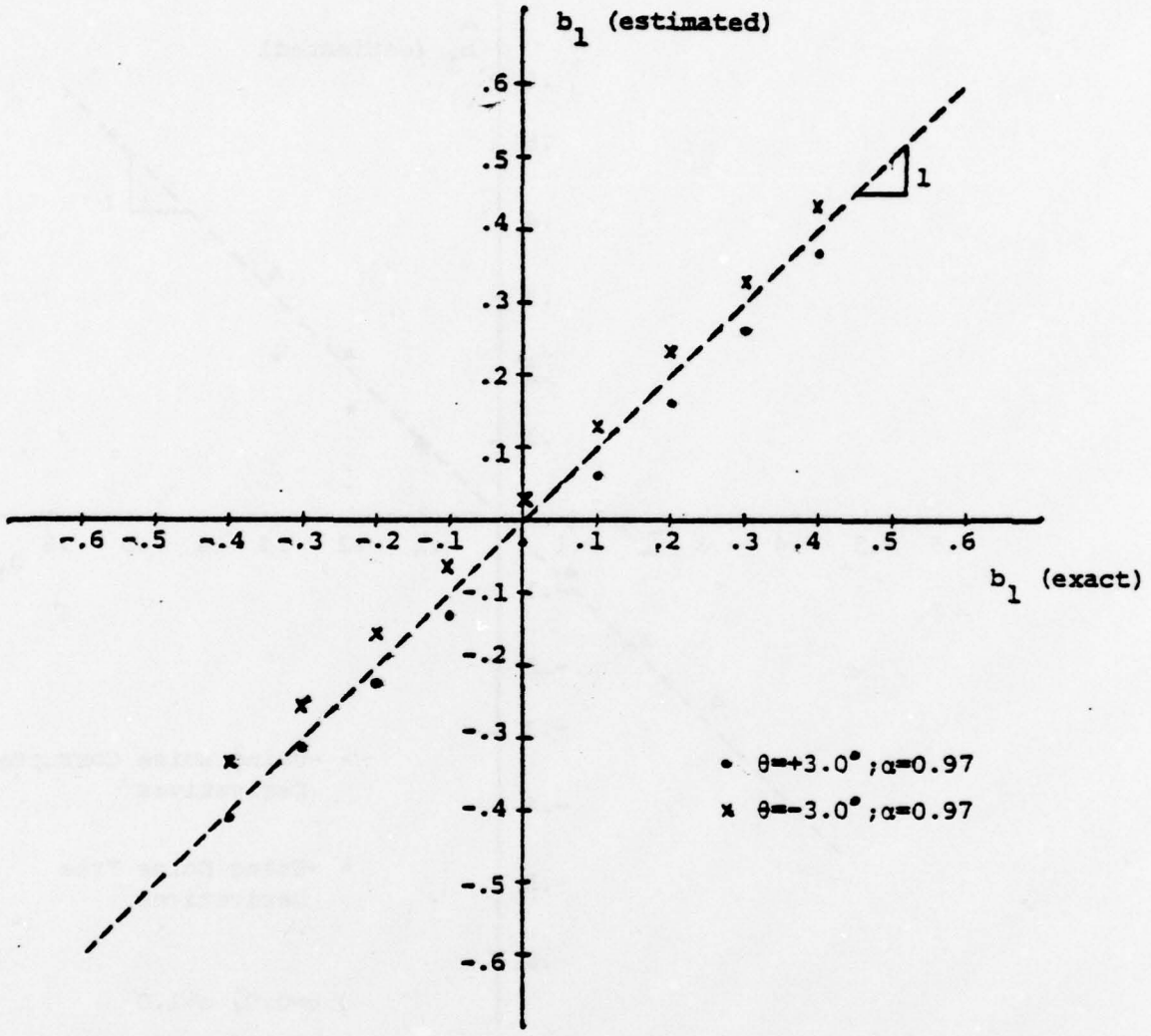


Figure 27

Translational I/O Function of Simplified Estimation Algorithm Which Models Rotation and Dilation as Noise

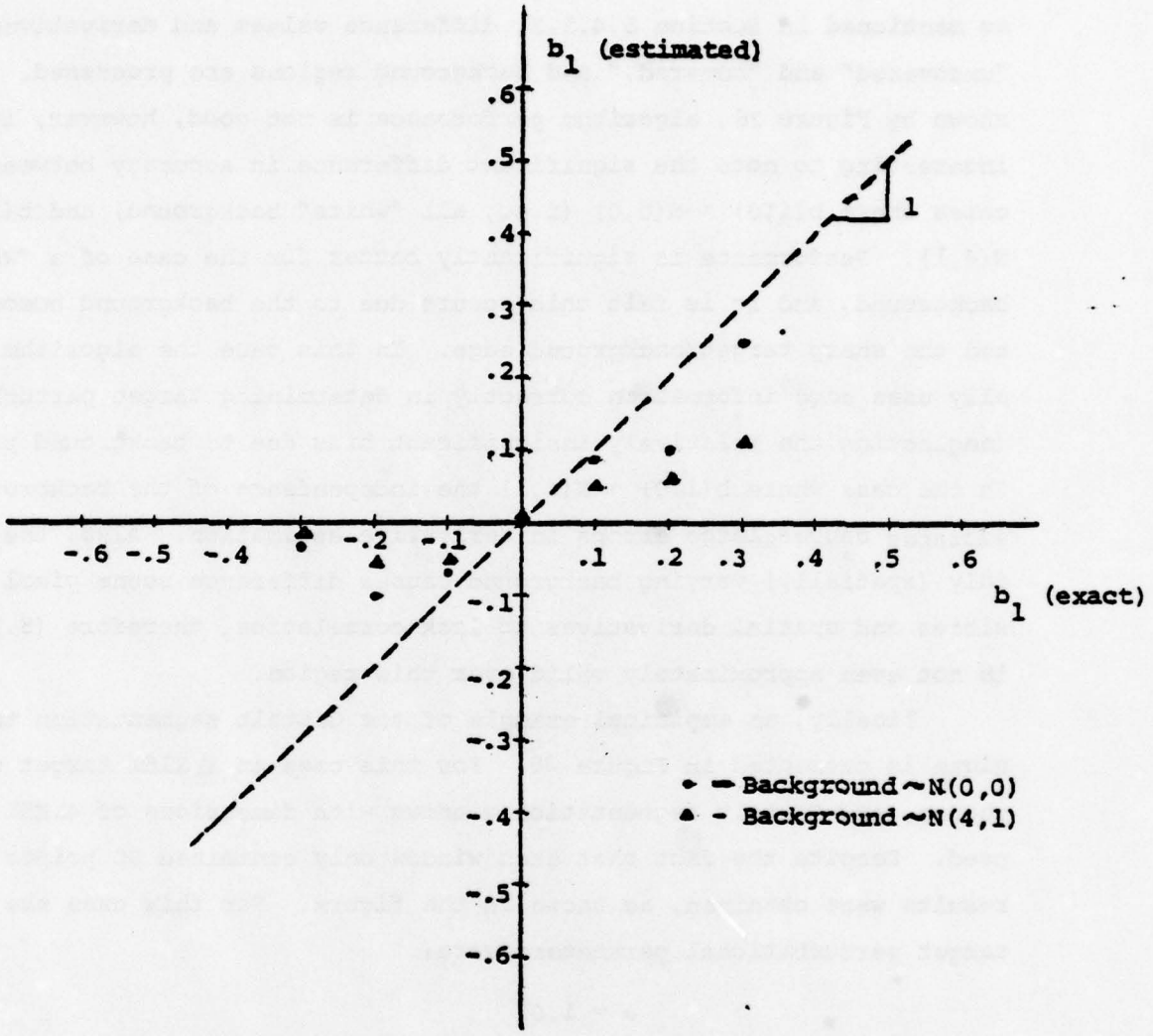


Figure 28

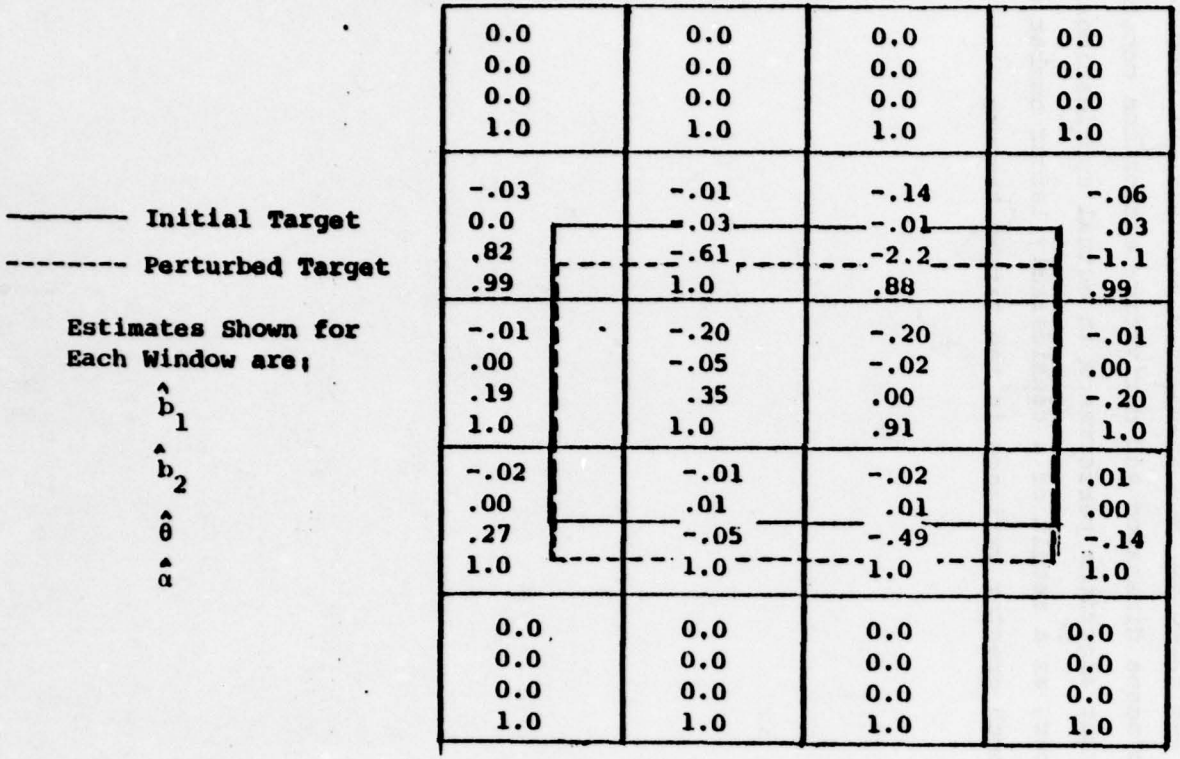
TSVIP Algorithm Segmentation Window Effect (θ=0.0°, α=1.0)

as mentioned in Section 5.4.5.2, difference values and derivatives from "uncovered" and "covered," and background regions are processed. As shown by Figure 28, algorithm performance is not good, however, it is interesting to note the significant difference in accuracy between the cases where $b(\underset{\sim}{i};0) \sim N(0,0)$ (i.e., all "white" background) and $b(\underset{\sim}{i};0) \sim N(4,1)$. Performance is significantly better for the case of a "white" background, and it is felt this occurs due to the background homogeneity and the sharp target/background edge. In this case the algorithm actually uses edge information correctly in determining target perturbations (neglecting the relatively insignificant bias due to background pixels). In the case where $b(\underset{\sim}{i};0) \sim N(4,1)$ the independence of the background variates causes large errors in derivative estimation. Also, the rapidly (spatially) varying background causes difference scene pixel intensities and spatial derivatives to lack correlation, therefore (5.3.1-7) is not even approximately valid over this region.

Finally, an empirical example of the Gestalt segmentation technique is presented in Figure 29. For this case an $8\Delta \times 16\Delta$ target was chosen, and Gestalt segmentation windows with dimensions of $4\Delta \times 5\Delta$ were used. Despite the fact that each window only contained 30 points, good results were obtained, as shown in the figure. For this case the exact target perturbational parameters were:

$$\alpha = 1.0$$
$$\theta = 0.0$$
$$b_1 = -0.2$$
$$b_2 = 0.0$$

As shown in the figure, the three types of estimates predicted in Section 5.4.5.2 are obtained. Obviously, for windows wholly containing background points the estimates indicate no measurable perturbation. Furthermore, for windows encompassing target edges, the estimates indicate either very slight perturbations (due to a majority of background pixels and lack of a sharp target edge) or large perturbations (due to large difference scene pixel intensities and erroneous derivative



——— Initial Target

-------- Perturbed Target

Estimates Shown for Each Window are;

$\hat{b}_1$

$\hat{b}_2$

$\hat{\theta}$

$\hat{\alpha}$

Exact Parameters:

$b_1 = -.20$

$b_2 = .00$

$\theta = 0.0$

$\alpha = 1.0$

Background ~ N(4,1)

30 Pixels per Gestalt Window

Figure 29

Gestalt Segmentation Using TSVIP Algorithm

estimates resulting from the "covering" or "uncovering" of the background). These estimates are easily distinguishable from those obtained using windows wholly containing target interior points. These latter estimates, as shown in Figure 29, are in excellent agreement with the exact target perturbationed parameters.

Finally, since the aforementioned simulation utilized a scene of relatively small dimensions (in real applications 256x256 to 512x512 pixel scenes are expected) and relatively slightly textured target functions (due to scene dimension and derivative estimation constraints), it is expected that algorithm performance in actual applications may be somewhat better, as a result of a considerably larger number of pixel samples and much greater variance in the target texture.

CHAPTER VIII

CONCLUSIONS AND SUGGESTIONS FOR FUTURE STUDY

A suitable algorithm for tracking in realistic situations consistent with using small portable hardware has not been found, so it has been necessary to do research on algorithm development. The TSVIP algorithm presented herein is a novel, CCD-implementable (with the possible exception of some analog multiplication) and temporally efficient approach to the realistic real-time video tracking problem. By capitalizing on successive scene temporal dependence and target texture, the algorithm estimates target rotation, dilation and translation parameters. High order matrix inversion problems are circumvented and a simple CCD implementation is achieved in the estimation process by employing a matrix pseudoinverse formulation theorem [30]. The TSVIP approach is potentially advantageous in complex scenes where the target/background separation problem is complicated or time consuming, since the algorithm may be used to quickly update the segmentation window or (if applied in parallel over a number of sub-scenes) to classify groups of pixels as members of the target or background sets. Computer simulation using smooth analytic functions to represent target texture confirms the validity of the CCD implementable version and these empirical results suggest an approximately linear input-output function for the resulting tracking processor.

Some obvious and desirable extensions of the research reported herein are as follows:

(1) Since the TSVIP algorithm essentially ignores target edge information, further study should be undertaken using the model of Section 4.3.3 to determine if this information is estimable (with a reasonable processor complexity) and useful for tracking purposes. If so, it may be possible to process both texture and edge information in parallel to generate more reliable tracking signals. Note that this might also permit easing of target textural constraints.

(2) Derivative estimation techniques, especially those which are CCD-implementable and reasonably noise insensitive, should be investigated.

(3) Simulation using actual scene data from maneuvering targets should be undertaken to further confirm the validity of the TSVIP approach.

(4) TSVIP algorithm sensitivity to CCD implementation, particularly with respect to dark current and transfer efficiency effects, should be studied.

(5) In the TSVIP algorithm, only the pixel intensity differences from two successive scenes are utilized to estimate $\hat{\underset{\sim}{a}}$. The possibility of using more than two scenes should be investigated.

(6) In situations where a high reliability system is desired (e.g., military applications) the possibility of using several different algorithms in parallel together with an estimate weighting scheme should be investigated.

There is a very large amount of research that can now be defined for tracking based on image analysis, but the most important step is to construct a simple experimental system and that is what we proposed to do during the next year. Such a system should emulate as closely as practical (or simulate) the technology to be used in actual systems. The system will allow tracking algorithm development and testing and provide a means to start tying together the hierarchy of decision and control algorithm that will be necessary for a practical system.

The development of the TSVIP tracking algorithm has made it practical to consider the use of a microcomputer as an aid to CCD implementation, or even as competition to CCD implementation. Work is in progress to obtain experimental results based on the implementation configurations discussed in Chapter 6, while alternate possibilities are also being pursued. The sequential (iterative) solutions being considered are especially interesting.

Two papers have been written and another is being planned. The paper, "Algorithm Development for Real-Time Automatic Video Tracking Systems"

will be given at COMPSAC in November as an invited paper and "A Model and Tracking Algorithm for a Class of Video Targets" has been submitted to the IEEE Transactions on Pattern Analysis and Machine Intelligence for publication.

BIBLIOGRAPHY

[1] Rosenfeld, A., and Kak, A. C., Digital Picture Processing, Academic Press, N.Y., 1976.

[2] Tanimoto, S., "A Comparison of Some Image Searching Methods," Proc. 1978 Conf. on Pattern Recognition and Image Processing, Chicago, May 1978, pp. 280-286.

[3] McVey, E. S., and Parrish, E. A., "The Application of Charge-Coupled Device Processors in Automatic Control Systems," IEEE Trans. on Automatic Control, Vol. AC-22, No. 4, Aug. 1977, pp. 680-682.

[4] Parrish, E. A., and McVey, E. S., "Implications of Charge-Coupled Devices for Pattern Recognition," IEEE Trans. on Computers, Nov. 1976.

[5] Barbe, D. F., Baker, W. D. and Davis, K. L., "Signal Processing with Charge-Coupled Devices," IEEE Journal of Solid State Circuits, Vol. SC-13, No. 1, Feb. 1978.

[6] Sequin, C. H., and Thomsett, M. F., Charge Transfer Devices, Academic Press, N.Y., 1970.

[7] Sequin, C. H., "Two Dimensional Charge Transfer Arrays," IEEE Journal of Solid State Circuits, Vol. SC-9, No. 3, June, 1974.

[8] Gersho, A., "Charge Transfer Filtering," Proc. IEEE, Vol. 67, No. 2, Feb. 1979, pp. 196-218.

[9] Armstrong, R. W., "Charge-Coupled Device Architectures for Signal Processing," Ph.D. Dissertation, University of Virginia, Charlottesville, Va., May 1978.

[10] Moskowitz, S., "Terminal Guidance by Pattern Recognition: A New Approach," IEEE Trans. on Aerospace and Navigational Electronics, Dec. 1964, pp. 254-265.

[11] Gromes, R. J. and Yi, C. J., "Analysis and Simulation of a Video Tracking System," Proc. '74 IEEE Systems, Man and Cybernetics Conf. Dallas, Texas, 1974, pp. 93-98.

BIBLIOGRAPHY

(Continued)

[12] Milstein, L. B. and Lazicky, T., "Algorithms to Track a Moving Target," Proc. '77 Conference on Pattern Recognition and Image Processing, pp. 148-152.

[13] McVey, E. S. and Baker, D. E., "Control Considerations for an Image Tracking System," Proc. 1978 Joint Automatic Control Conference, Philadelphia, Pa., Oct. 1978, pp. 277-286.

[14] McVey, E. S. and Woolard, W. B., "A Perturbation Method for Obtaining Control Signals in an Image Tracking System," accepted for publication in Proc. 1979 Joint Automatic Control Conference.

[15] Lubinski, K., Dickson, K., and Cairns, J., "Microprocessor-Based Interface Converts Video Signals for Object Tracking," Computer Design, Dec. 1977, pp. 81-87.

[16] Chow, W. K. and Aggarval, J. K., "Computer Analysis of Planar Curvilinear Moving Images," IEEE Trans. on Computers, Feb. 1977, pp. 187-197.

[17] Uno, T., Ejinu, M., and Tokunaga, T., "A Method of Real-Time Recognition of Moving Objects and Its Application," Pattern Recognition, Vol. 8, 1976, pp. 201-208.

[18] Black, R. J., Whitney, J. T., and Flacks, G. M., "A Pre-Prototype Real Time Video Tracking System," NAECON '76 Record, Dayton, Ohio May 1976, pp. 161-168.

[19] Flachs, G. M., Thompson, W. E., U., Y. H., Gilbert, A. L., "A Real-Time Structural Tracking Algorithm," NAECON '76 Record, Dayton, Ohio, May 1976, pp. 161-168.

[20] Thompson, W. E., and Flachs, G. M., "A Structure and Dynamic Mathematical Model of a Real-Time Video Tracking System," NAECON '76 Record, Dayton, Ohio, May 1976, pp. 169-172.

[21] Vilela, J. A., and Flachs, G. M., "A Structural Model for Polygonal Patterns," NAECON '76 Record, Dayton, Ohio, May 1976, pp. 173-175.

(Continued)

[22] Flachs, G. M., Perez, P. I., Rogers, R. B., Szymanski, J. M., Taylor, J. M., Thompson, W. E., and U, Y. H., "Real-Time Video Tracking Concepts," U.S. Army Research Office Report, No. NMSU-TR-78-1, May 1978.

[23] Flachs, G. M., Thompson, W. E., Black, R. B., Taylor, J. M., Canna, W., Rogers, R. B., U, Y. H., "An Automated Video Tracking System," NAECON '77 Record, pp. 361-368.

[24] Thompson, W. E., Flachs, G. M., and Kiang, T. R., "Evaluation of Filtering and Prediction Techniques for Real-Time Video Tracking of High Performance Missiles," NAECON '78 Record, May 1978.

[25] Duda, R. O., and Hart, P. E., Pattern Classification and Scene Analysis, Wiley, N.Y., 1973.

[26] Blaauw, G., Digital System Implementation, Prentice-Hall, N.J., 1976.

[27] Nahi, N., and Lopez-Mora, S., "Estimation-Detection of Object Boundaries in Noisy Images," IEEE Trans. Automatic Control, Vol. AC-13, No. 5, Oct. 1978, pp. 834-845.

[28] Rao, C., and Mitra, S., Generalized Inverse of Matrices and Its Applications, Wiley, N.Y., 1971

[29] DeRusso, P. M., Roy, R. J., and Close, C. M., State Variables for Engineers, Wiley, N.Y., 1965.

[30] Cline, R. E., "Representations for the Generalized Inverse of a Partitioned Matrix," SIAM Journal, Vol. 12, No. 3, Sept. 1964, pp. 589-600.

[31] Guillemin, E. A., The Mathematics of Circuit Analysis, MIT Press, 1949.

[32] Tobey, et al., Operational Amplifiers, Design and Applications, McGraw-Hill, N.Y., 1971.

[33] Eykhoff, P., System Identification, Wiley, N.Y., 1974.

[34] Potter, J. L., "Velocity as a Cue to Segmentation," IEEE Trans.

BIBLIOGRAPHY

(Continued)

[47] Goldberg, H. S., et al, "A Mask Programmable Charge Transfer Analog Multiplier," ISSCC Digest of Technical Papers, Vol. XX, 1977, pp. 26-27.

[48] Hague, Y. A. and Copeland, M. A., "Design and characterization of a real-time-correlator," IEEE J. Solid-State Circuits, Vol. SC-12, Dec. 1977, pp. 642-649.

[49] Denyer, P. B., Mabor, J. and Arthur, J. W., "Miniature Programmable Transversal Filter," Proceedings of the IEEE, Vol. 67, No. 1, Jan. 1978, pp. 42-50.

[50] Gilbert, B., "A New Technique for Analog Multiplication," IEEE J. of Solid-State Circuits, Vol. SC-10, No. 6, Dec. 1975, pp. 437-447.

[51] Smith, J., "A Second-Generation Carrier Domain Four Quadrant Multiplier," IEEE J. Solid-State Circuits, Vol. SC-10, No. 6, Dec. 1975, pp. 448-457.

[52] Reticon Product Summary: Discrete Time Analog Signal Processing Devices, 1978.

[53] Analog Devices Data Acquisition Products Catalog, 1978.

[54] Esser, L. J. M., "The Peristaltic Charge-Coupled Device for High-Speed Charge Transfer," IEEE, ISSCC Dig. Tech. Papers, 1974, pp. 23-29.

BIBLIOGRAPHY

(Continued)

on Systems, Man, and Cybernetics, May 1975, pp. 390-394.

[35] Dahlquist, G., and Bjork, A., Numerical Methods, Prentice-Hall, N.J., 1974.

[36] Oppenheim, A. V., and Schafer, R. W., Digital Signal Processing, Prentice-Hall, N.J., 1975.

[37] Freeman, H., Discrete-Time Systems, Wiley, N.Y., 1960.

[38] Courant, R., Differential and Integral Calculus, Vol. II, Nordeman, N.Y., 1945.

[39] Singer, R. A., and Behnke, K. W., "Real-Time Tracking Filter Evaluation and Selection for Tactical Applications," IEEE Trans. on Aerospace and Electronic Systems, Vol. AES-7, No. Jan. 1971.

[40] Mostafavi, H., and Smith, F., "Image Correlation with Geometric Distortion, Part I: Acquisition Performance," IEEE Trans. on Aerospace and Electronic Systems, Vol. AES-14, No. 3, May 1978.

[41] Mostafavi, H., and Smith, F., "Image Correlation with Geometric Distortion Part II: Effect on Local Accuracy," IEEE Trans. on Aerospace and Electronic Systems, Vol. AES-14, No. 3, May 1978, pp. 494-500.

[42] Gans, D., Transformations and Geometries, Appleton-Century-Crafts, N.Y., 1969, Ch. IV.

[43] Peterson, D., and Middleton, D., "Sampling and Reconstruction of Wave-Number Limited Functions in N-Dimensional Euclidean Spaces," Information and Control, No. 5, 1962, pp. 279-323.

[44] Nathan, A., "Plain and Covariant Multivariate Fourier Transforms," Information and Control, 39, 1978, pp. 73-81.

[45] Deutsch, R., Estimation Theory, Prentice-Hall, N.Y., 1965.

[46] Penrose, R., "A Generalized Inverse for Matrices," Proc. Cambridge Philos. Soc., 51 (1955), pp. 406-413.

Appendix 1: Affine Transform Summary

(a) Mathematical Background

The 2-D affine transform has been used previously to model small target function perturbations (e.g., see Mostafavi and Smith [40-41].) While a detailed description of this transform may be found in Gans [42], a brief summary is presented here. Basically, the 2-D affine transform represents a linear operation on the arguments of a 2-D function as follows:

For a picture function of the form

$$p(\underset{\sim}{x}); \quad \underset{\sim}{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{A.1-2}$$

the general affine transformed version of this function is denoted by

$$p'(\underset{\sim}{x}) = p(\underset{\sim}{x}') \tag{A.1-2}$$

where

$$\underset{\sim}{x}' = A\underset{\sim}{x} + \underset{\sim}{b} \tag{A.1-3}$$

It should be noted that both A and $\underset{\sim}{b}$ may be functions of time. Thus, the affine transform, with respect to the indices of $p(\underset{\sim}{x})$ represents a linear transformation of the plane onto itself.

Expanding (A.1-3):

$$\begin{bmatrix} x_1' \\ x_2' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \tag{A.1-4}$$

A is denoted the homogeneous affine transform matrix, and $\underset{\sim}{b}$ is the translation vector. Two well known versions of the homogeneous affine transform are:

$$A = \begin{bmatrix} \alpha & 0 \\ 0 & \alpha \end{bmatrix} \qquad \text{(dilation)} \tag{A.1-5}$$

$$A = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \qquad \text{(rotation about the origin through an angle, } \theta\text{)} \tag{A.1-6}$$

Another important case is where A = I, and

$$\underset{\sim}{b} = \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \end{bmatrix} \qquad \text{(simple translation)} \quad \text{(A.1-7)}$$

One important result, which is quite useful for image tracking purposes, is the transform representation of a rotation through an angle θ about a point, (h,k) where $(h,k) \neq (0,0)$. This type of affine transform obviously will not leave the target centered in the FOV of the camera, therefore it is an important point. Gans [42] has shown that this rotation may be represented as:

$$\begin{bmatrix} x_1' - h \\ x_2' - k \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x_1 - h \\ x_2 - k \end{bmatrix} \qquad \text{(A.1-8)}$$

which may easily be simplified to:

$$\begin{bmatrix} x_1' \\ x_2' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} - \begin{bmatrix} (\cos\theta - 1) & -\sin\theta \\ \sin\theta & (\cos\theta - 1) \end{bmatrix} \begin{bmatrix} h \\ k \end{bmatrix} \qquad \text{(A.1-9)}$$

This expression represents an equivalent rotation about the origin (first term) and translation (second term). The translational term is obviously of importance to the tracking system. Two additional points should be noted:

(1) Combined affine transforms, e.g., a rotation followed by translation, then another rotation, etc., may also be represented by a resultant affine transform of the form in (A.1-2). <u>However, the order in which these transforms occur is significant</u> since the result of a rotation followed by a translation is not necessarily the same as if the transform were instead the translation first, then the rotation. This does not affect the model developed in Section 4.3.2, or the ultimate processor, however, since the system will only need to estimate the resultant rotational and translational parameters (A and $\underset{\sim}{b}$), not the individual rotations.

(2) Since the affine transform operates on the target function arguments by essentially changing the coordinate axes, A and

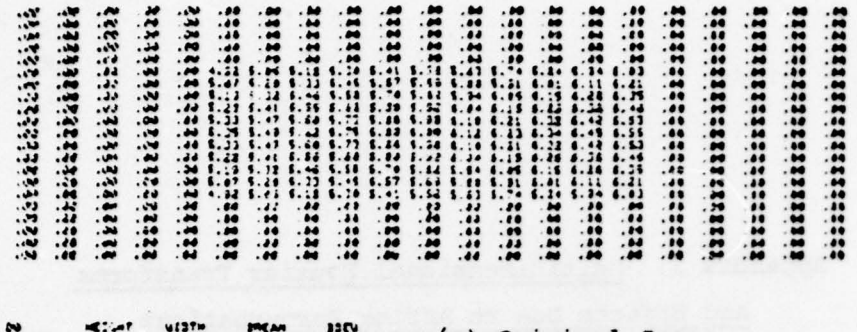
b actually affect the target function in the inverse way as the coordinates, e.g. if

$$A = \begin{bmatrix} .9 & 0 \\ 0 & .9 \end{bmatrix}; \qquad \underset{\sim}{b} = \underset{\sim}{0}$$

we would actually see the picture function magnified by a factor of $1/0.9$ in both directions. Likewise, if A represents a rotation of θ on the coordinate system, the picture function is actually rotated by $-\theta$. This fact is important in the generation of proper polarity of control signals.

(b) Affine Transform Examples Using Sampled Scene Data

Using the computer program described in Chapter VI of Appendix 5, numerical examples of affine perturbations were generated. An $||x||$ target was chosen, and background pixel intensities are set equal to zero, for clarity. Figure 30 shows examples of dilation, rotation and translation.



(a) Original Scene

(b) Dilation (α= 0.5)

(c) Rotation (θ= 45°)

(d) Translation (4*Δ Right & Down)

Figure 30 Affine Transform Numerical Examples

Appendix 2: Multidimensional Fourier Transforms and Effects Due to Affine Perturbations

The Fourier Transform of a 3-D function may be written as

$$\mathcal{F}[p(\underset{\sim}{x};t)] \triangleq F(\underset{\sim}{u},w) = \int_{X,T} p(\underset{\sim}{x};t)e^{-j2\pi[\underset{\sim}{u}^T;w]}\begin{bmatrix}\underset{\sim}{x}\\ t\end{bmatrix} d\underset{\sim}{x}dt \qquad (A.2\text{-}1)$$

where X and T are the regions in 2-D space and 1-D time over which $p(\underset{\sim}{x},t) \neq 0$. Note that the integral sign represents a 3-fold integral.

Letting

$$p'(\underset{\sim}{x};t) = p(\underset{\sim}{x}';t) \qquad (A.2\text{-}2)$$

where

$$\underset{\sim}{x}' = A(t)\underset{\sim}{x}+\underset{\sim}{b}(t) \qquad (A.2\text{-}3)$$

it is useful to consider the effects of the affine transform on

$$F'(\underset{\sim}{u},w) = \mathcal{F}[p(\underset{\sim}{x};t)] \qquad (A.2\text{-}4)$$

especially if a functional relationship between $F'(\underset{\sim}{u},w)$ and $F(\underset{\sim}{u},w)$ may be established. By definition

$$F'(\underset{\sim}{u},w) = \int_{X',T'} p'(\underset{\sim}{x};t)e^{-j2\pi[\underset{\sim}{u}^T;w]}\begin{bmatrix}\underset{\sim}{x}\\ t\end{bmatrix} d\underset{\sim}{x}dt \qquad (A.2\text{-}5)$$

A change of variables is made as follows:

$$\begin{bmatrix}\underset{\sim}{x}'\\ t\end{bmatrix} = \begin{bmatrix}A(t) & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}\underset{\sim}{x}\\ t\end{bmatrix} + \begin{bmatrix}\underset{\sim}{b}(t)\\ 0\end{bmatrix} \qquad (A.2\text{-}6)$$

and

$$d\underset{\sim}{x}' = D\cdot d\underset{\sim}{x} \qquad (A.2\text{-}7)$$

where D is the Jacobian of the transformation, i.e.,

$$D = |\det A(t)| \qquad (A.2\text{-}8)$$

It is a simple matter to show the inverse transform of (A.2-6), i.e.,

$$\begin{bmatrix}\underset{\sim}{x}\\ t\end{bmatrix} = \begin{bmatrix}A^{-1}(t) & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}\underset{\sim}{x}'\\ t\end{bmatrix} - \begin{bmatrix}\underset{\sim}{b}(t)\\ 0\end{bmatrix} . \qquad (A.2\text{-}9)$$

Substitution of (A.2-7) and (A.2-9) into (A.2-5) yields

$$F'(\underset{\sim}{u},w) = \int_{X',T} \frac{e^{j2\pi \underset{\sim}{u}^T A^{-1}(t)\underset{\sim}{b}(t)}}{|\det A(t)|} \left[p(\underset{\sim}{x}';t)e^{-j2\pi \underset{\sim}{u}^T A^{-1}(t)\underset{\sim}{x}} d\underset{\sim}{x}\right] e^{-j2\pi wt} dt \qquad \text{(A.2-10)}$$

Making the substitution

$$\underset{\sim}{u}_o^T = \underset{\sim}{u}^T A^{-1}(t) \qquad \text{(A.2-11)}$$

it is seen that (A.2-10) may be reduced to

$$F'(u,w) = \int_T \frac{e^{j2\pi \underset{\sim}{u}^T A^{-1}(t)\underset{\sim}{b}(t)}}{|\det A(t)|} F[(A^T(t))^{-1}\underset{\sim}{u}] e^{-j2\pi wt} dt \qquad \text{(A.2-12)}$$

(A.2-12) may now be used to show a variety of effects of the affine transform on the Fourier transform of the picture function. Some special cases are:

(1) $A(t) = I$; $\quad \underset{\sim}{b}(t) = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$

in which case

$F'(\underset{\sim}{u},w)$ only differs from $F(u,w)$ in phase;

and

(2) $\underset{\sim}{b}(t) = \underset{\sim}{0}$ and

$$A(t) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

in which case

$[A^T(t)]^{-1} = A(t)$

since A is orthogonal. Thus, we rotate the coordinate axis in Fourier space through the same angle as in the signal or picture space.

Equation (A.2-12) is also extremely useful in that it easily enables calculation of affined-perturbed function spectra when the unperturbed spectra are known. Recently Nathan [44] has derived a special case of (A.2-12) for 2-D functions where $\underset{\sim}{b} = \underset{\sim}{0}$.

Appendix 3: General Least Squares Estimators

A particularly important parameter dependent (linear in the parameters) process, corrupted by noise, is of the following form:

$$\underset{\sim}{y} = U\underset{\sim}{b} + \underset{\sim}{\eta} \tag{A.3-1}$$

where

$\underset{\sim}{y}$ is the Nxl process "output" vector;

U is an NxM known, measurable or calculable matrix;

$\underset{\sim}{b}$ is the Mxl parameter vector to be estimated (denote this estimate by $\hat{\underset{\sim}{b}}$); and

$\underset{\sim}{\eta}$ is an Nxl "noise" vector.

It is desired to estimate $\underset{\sim}{b}$, with the following constraints on the estimate:

(1)	$\hat{\underset{\sim}{b}} = M\underset{\sim}{y}$	linearity	
(2)	$E\{\hat{\underset{\sim}{b}}\} = \underset{\sim}{b}$	unbiasedness	(A.3-2)
(3)	$Cov\{\hat{\underset{\sim}{b}}\} =$	"small" or perhaps minimal in some sense.	

Note that (1) is intended to enable CCD implementation, provided that M may be also formed via CCD type operations.

It may be shown (Eykhoff [33], Deutsch [45]) that the estimate which minimizes a loss function of the type:

$$||\underset{\sim}{y} - U\hat{\underset{\sim}{b}}||_R$$

in accordance with (1) is of the form:

$$\hat{\underset{\sim}{b}} = M\underset{\sim}{y} \tag{A.3-3}$$

where

$$M = U_R^+ \tag{A.3-4}$$

$$= (U^T R U)^{-1} U^T R$$

i.e., M equals the pseudoinverse of U, based on an R norm, denoted by U_R^+. Assuming that U and $\underset{\sim}{\eta}$ are statistically independent, (2) may be

satisfied, since

$$E\{\hat{\underline{b}}\} = E\{(U^TRU)^{-1}U^TRU\underline{b}\} + E\{(U^TRU)^{-1}U^TR\underline{\eta}\}$$

$$= \underline{b} + E\{(U^TRU)^{-1}U^TR\}E\{\underline{\eta}\}$$

if either

$$E(\underline{\eta}) = 0 \quad \text{or} \tag{A.3-5}$$

$$E\{(U^TRU)^{-1}U^TR\} = [0] \tag{A.3-6}$$

Since (c.f. [33], p. 187)

$$\text{cov}[\hat{\underline{b}}] = E\{[MU\underline{b}+M\underline{\eta}-\underline{b}][MU\underline{b}-M\underline{\eta}-\underline{b}]^T\}$$

$$= (U^TRU)^{-1}U^TRNRU(U^TRU)^{-1}$$

where

$$\text{cov}[\underline{\eta}] \triangleq N$$

(3) may be satisfied by choosing $R = N^{-1}$ and therefore

$$\text{cov}\{\hat{\underline{b}}_M\} = (U^TN^{-1}U)^{-1} \tag{A.3-7}$$

which can be shown [45] to be minimum variance, hence efficient.

Appendix 4: Cline's Theorem for the Formulation of a Pseudoinverse of a Partitioned Matrix

The problem of forming an inverse of a rectangular matrix, A, with specified properties has been studied by Penrose [46], Cline [30] and a number of other researchers. The pseudoinverse of this NXM real matrix A, is a MXN matrix denoted by $A^{\dagger}$. Examples of desirable properties are:

$$\begin{aligned} AA^{\dagger}A &= A \\ A^{\dagger}AA^{\dagger} &= A^{\dagger} \\ (AA^{\dagger})^{T} &= AA^{\dagger} \end{aligned} \qquad \text{(A.4-1)}$$

It is well known that if A has full column rank, one inverse of considerable interest is the so-called least squares inverse denoted by

$$A^{\dagger} = (A^{T}A)^{-1}A^{T} \qquad \text{(A.4-2)}$$

The properties of this solution are considered extensively in references on least squares estimation, interpolation, etc. One important note of particular importance to the present work is that the formulation of this pseudoinverse requires the inversion of an MXM non-singular matrix (or conversely, the solution of the so-called normal equations, which are also of order M). While numerous successful algorithms for the solution of this problem on the digital computer have been known for a long time, the adaptation of these algorithms for the CCD or discrete analog computer is very difficult. This is at least partially due to two factors:

(1) In solving the normal equations, Gauss Elimination requires extensive division and may require row exchanges, which are difficult operations in the CCD implementation; and

(2) If the inverse of $(A^{T}A)$ is computed directly, e.g., via Cramers rule, the memory requirements and calculations become excessive.

Cline [30] studied the problem of computing the pseudoinverse of

partitioned matrices of the form:

$$A = [U \vdots V] \tag{A.4-2}$$

Basically this solution is as follows:

For a partitioned Nx(R+(M-R)) matrix

$$A = [U \vdots V]$$

$$[U \vdots V]^{\dagger} = \begin{bmatrix} U^{\dagger} - U^{\dagger}VC^{\dagger} - U^{\dagger}V(I-C^{\dagger}C)KV^{T}(U^{\dagger})^{T}U^{\dagger}(I-VC^{\dagger}) \\ C^{\dagger} + (I-C^{\dagger}C)KV^{\dagger}(U^{\dagger})^{T}U^{\dagger}(I-VC^{\dagger}) \end{bmatrix} \tag{A.4-3a}$$

where

$$C = (I-UU^{\dagger})V \tag{A.4-3b}$$

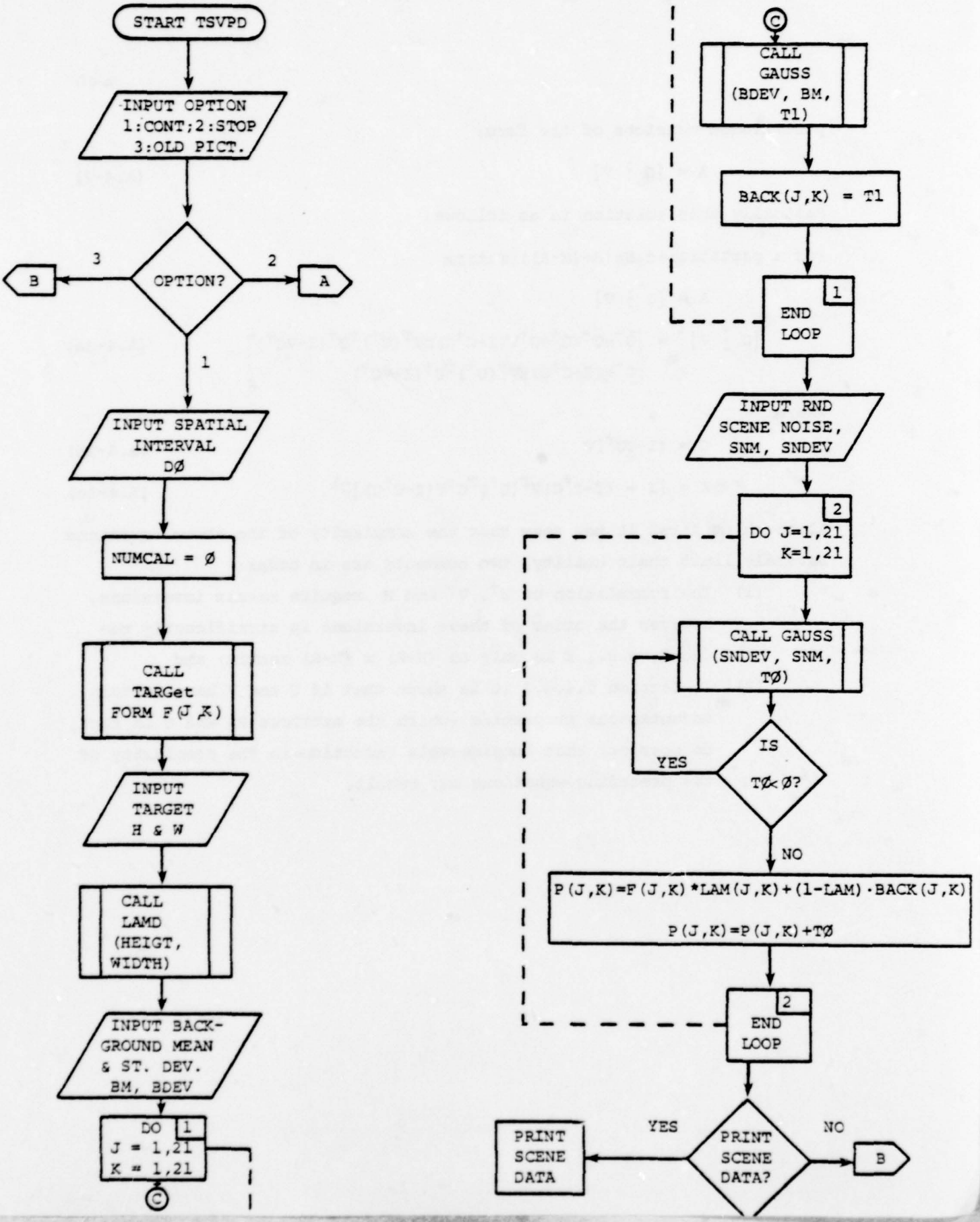
$$K = [I + (I-C^{\dagger}C)V^{T}(U^{\dagger})^{T}U^{\dagger}V(I-C^{\dagger}C)]^{-1} \tag{A.4-3c}$$

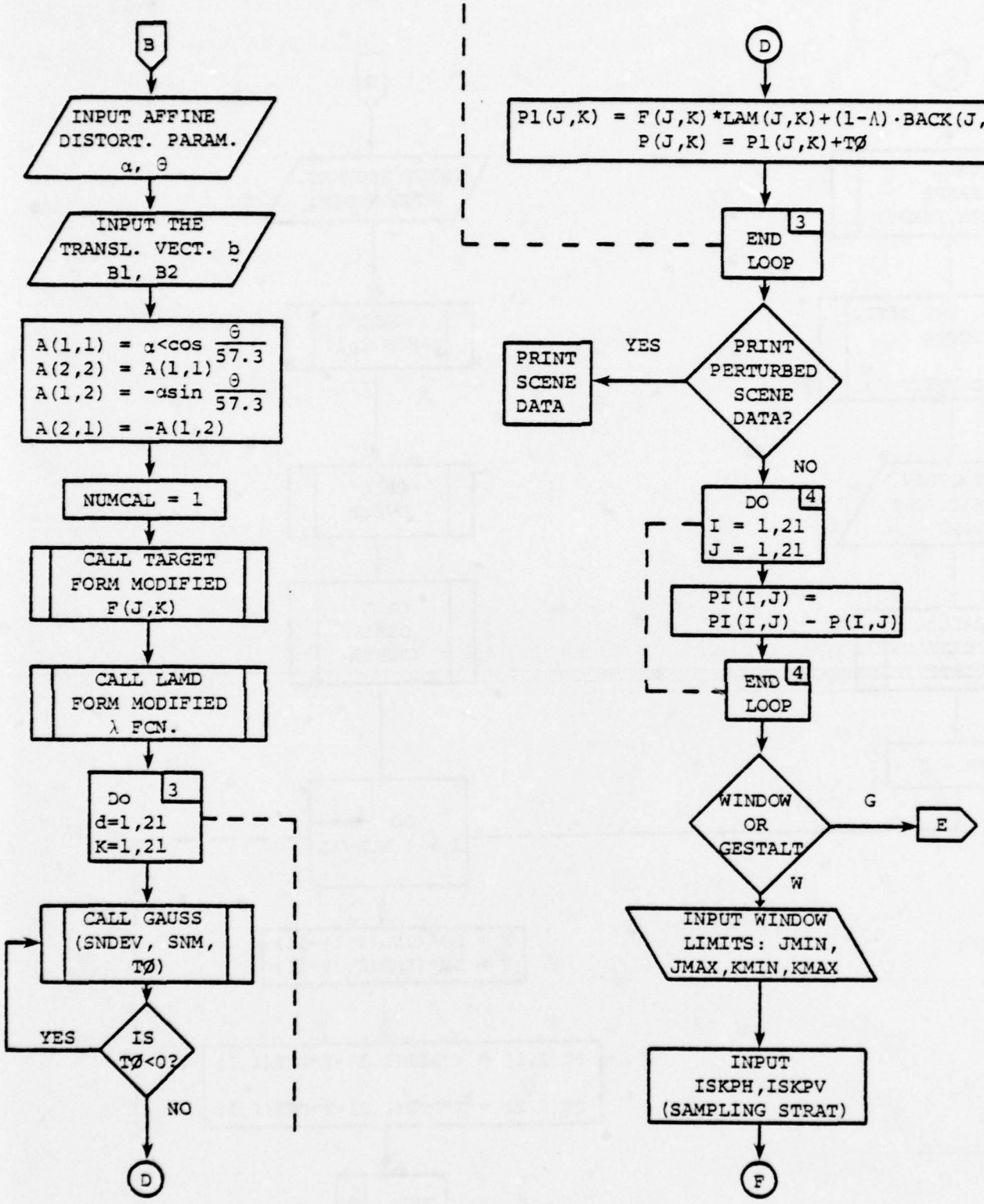
Although at first it may seem that the complexity of the above equations severely limit their utility, two comments are in order:

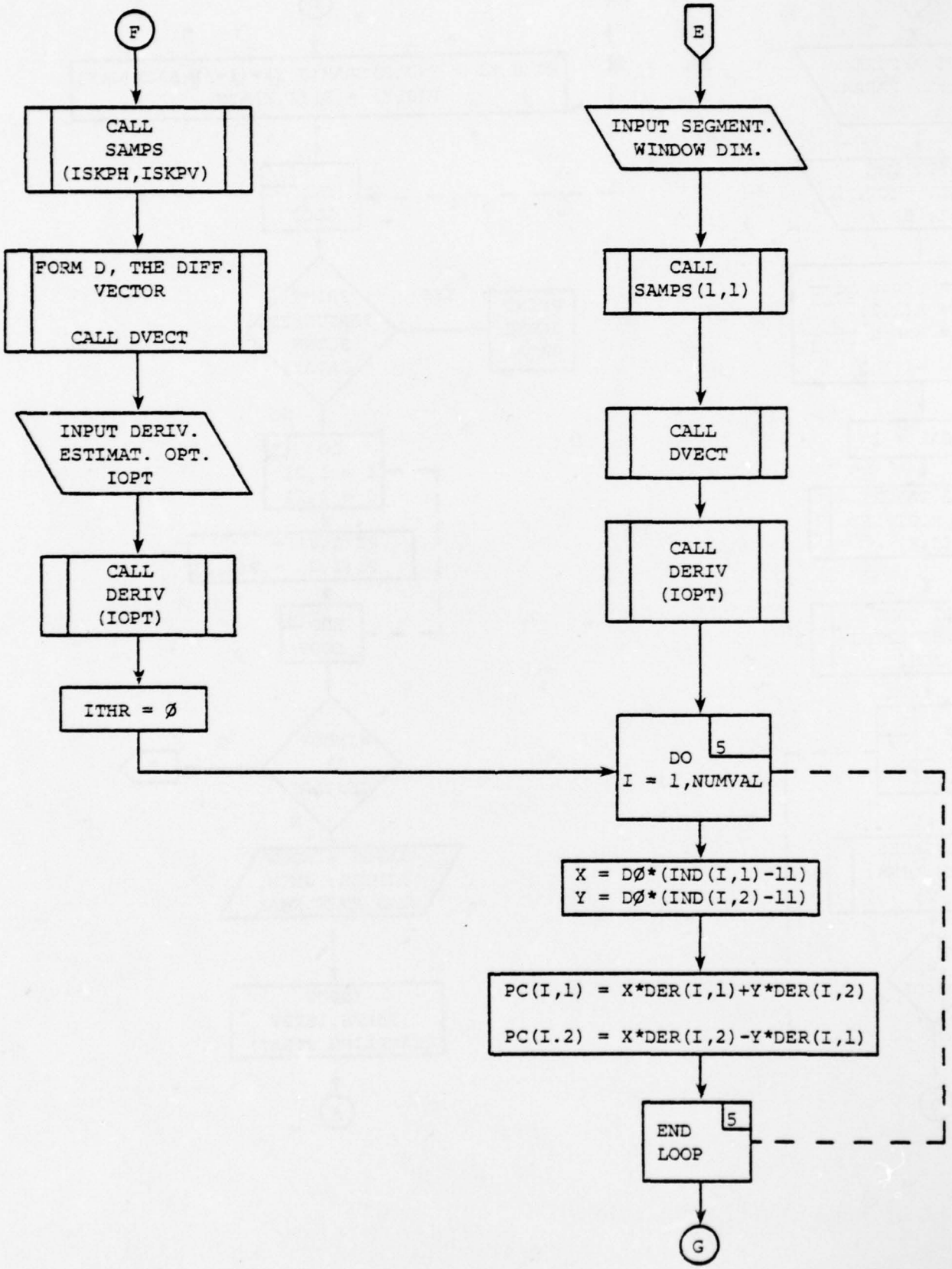
(1) The formulation of $U^{\dagger}$, $V^{\dagger}$ and K require matrix inversions, however the order of these inversions is significantly reduced, e.g., K is only an (M-R) x (M-R) matrix; and

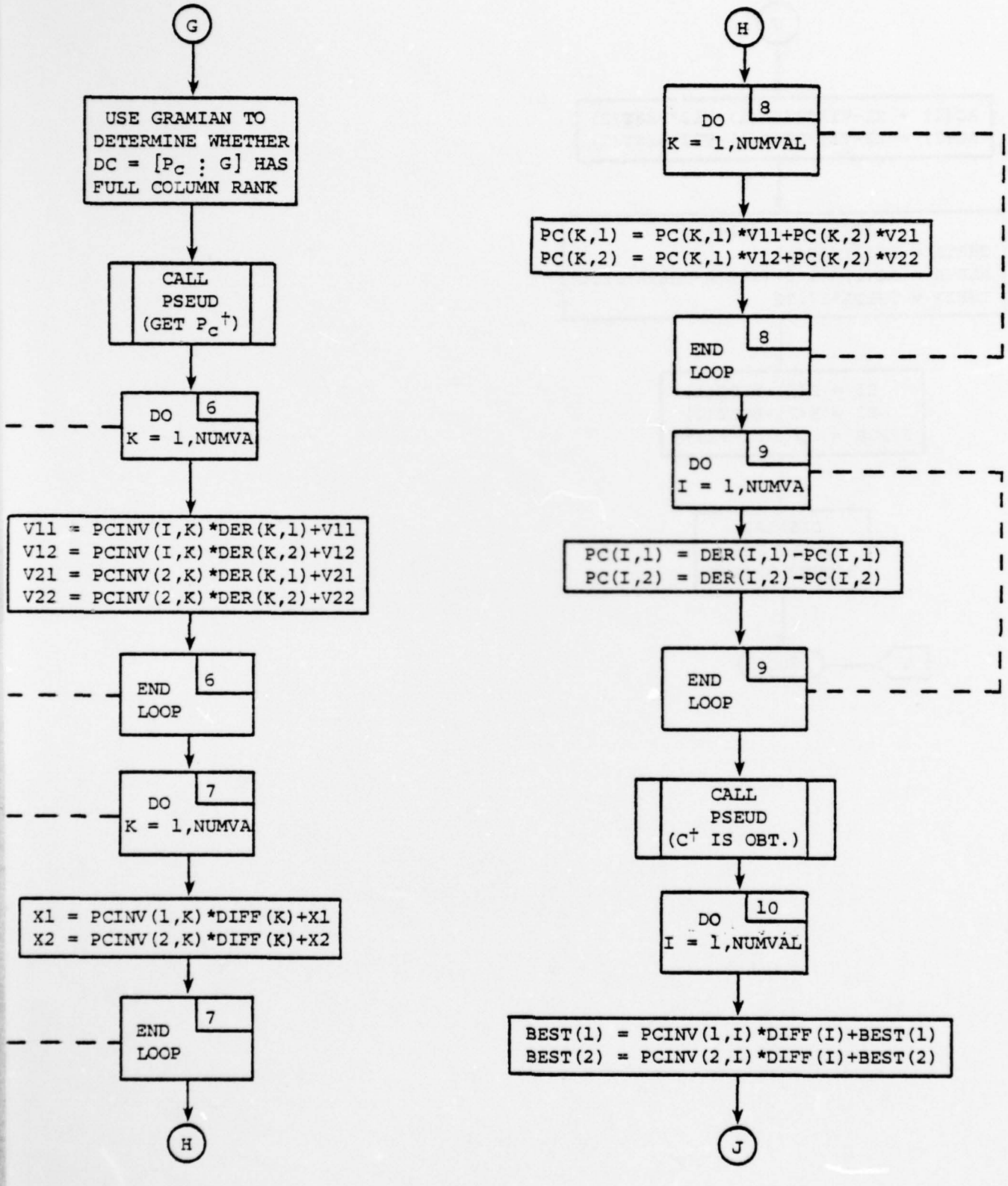
(2) In Section 5.4.4.3 it is shown that if U and V have certain advantageous properties (which the matrices P_c and G in fact, do possess) that considerable reduction in the complexity of the preceding equations may result.

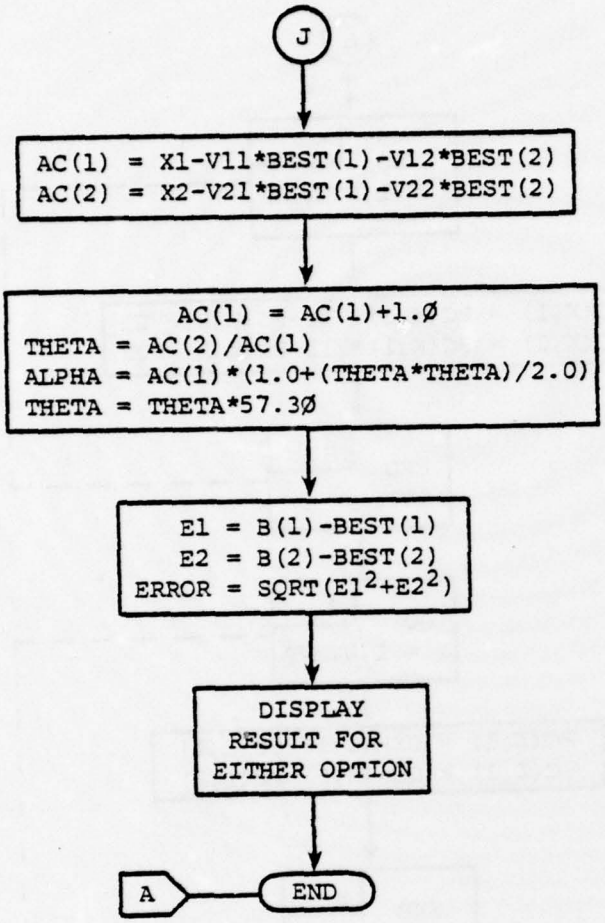
Appendix 5.1 - Flow Chart for Program TSVPD and Subroutines




B
INPUT AFFINE DISTORT. PARAM. α, θ
INPUT THE TRANSL. VECT. b̃ B1, B2
A(1,1) = α<cos θ/57.3
A(2,2) = A(1,1)
A(1,2) = -αsin θ/57.3
A(2,1) = -A(1,2)
NUMCAL = 1
CALL TARGET FORM MODIFIED F(J,K)
CALL LAMD FORM MODIFIED λ FCN.
Do 3 d=1,21 K=1,21
CALL GAUSS (SNDEV, SNM, TØ)
IS TØ<0?
YES
NO
D
D
P1(J,K) = F(J,K)*LAM(J,K)+(1-Λ)·BACK(J,K)
P(J,K) = P1(J,K)+TØ
END 3 LOOP
PRINT PERTURBED SCENE DATA?
YES
PRINT SCENE DATA
NO
DO 4 I = 1,21 J = 1,21
PI(I,J) = PI(I,J) - P(I,J)
END 4 LOOP
WINDOW OR GESTALT
G
E
W
INPUT WINDOW LIMITS: JMIN, JMAX,KMIN,KMAX
INPUT ISKPH,ISKPV (SAMPLING STRAT)
F

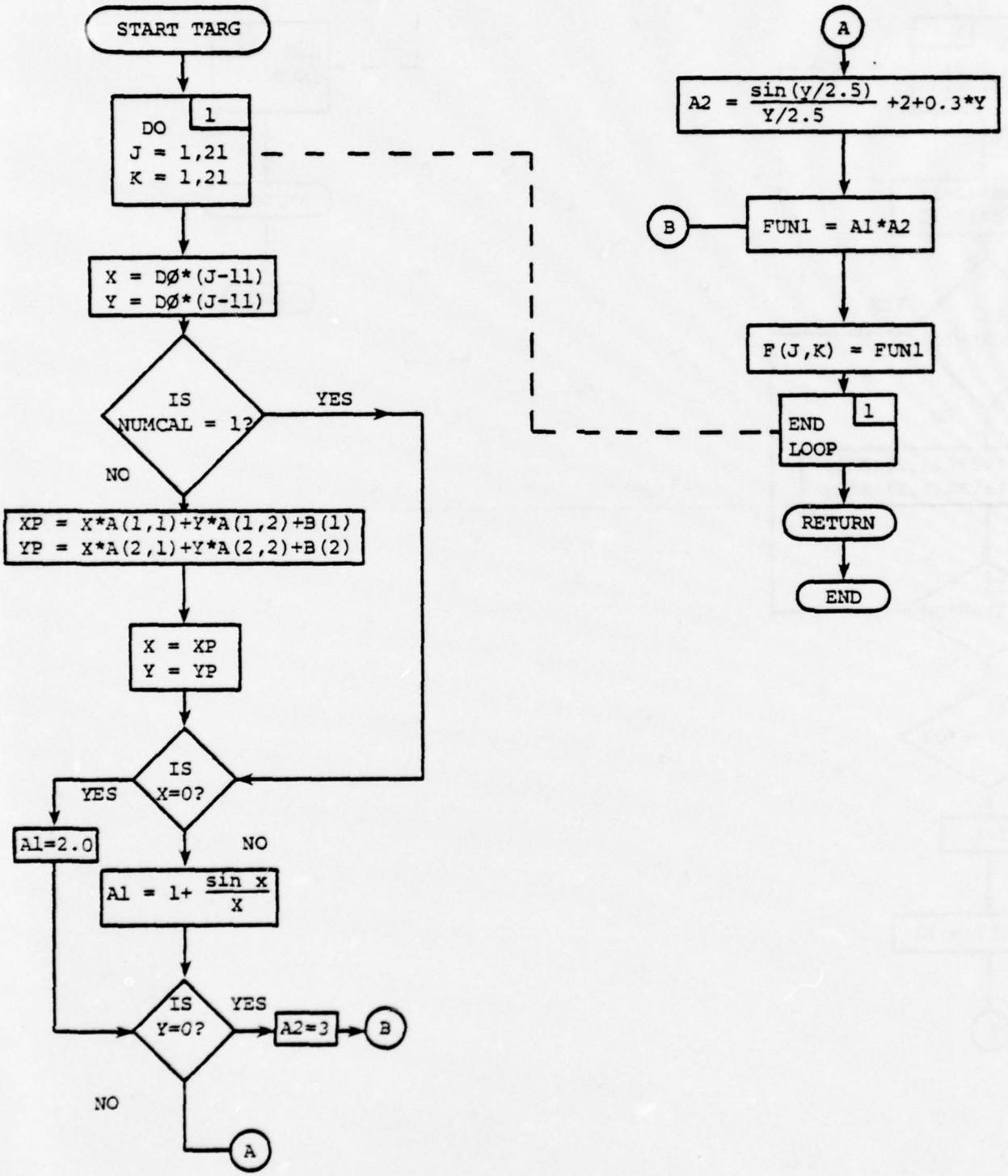

F
CALL SAMPS (ISKPH,ISKPV)
FORM D, THE DIFF. VECTOR
CALL DVECT
INPUT DERIV. ESTIMAT. OPT. IOPT
CALL DERIV (IOPT)
ITHR = Ø
E
INPUT SEGMENT. WINDOW DIM.
CALL SAMPS(1,1)
CALL DVECT
CALL DERIV (IOPT)
DO 5 I = 1,NUMVAL
X = DØ*(IND(I,1)-11)
Y = DØ*(IND(I,2)-11)
PC(I,1) = X*DER(I,1)+Y*DER(I,2)
PC(I.2) = X*DER(I,2)-Y*DER(I,1)
END 5 LOOP
G


G
USE GRAMIAN TO DETERMINE WHETHER DC = [P_C : G] HAS FULL COLUMN RANK
CALL PSEUD (GET $P_C^\dagger$)
DO 6 K = 1,NUMVA
V11 = PCINV(I,K)*DER(K,1)+V11
V12 = PCINV(I,K)*DER(K,2)+V12
V21 = PCINV(2,K)*DER(K,1)+V21
V22 = PCINV(2,K)*DER(K,2)+V22
END LOOP 6
DO 7 K = 1,NUMVA
X1 = PCINV(1,K)*DIFF(K)+X1
X2 = PCINV(2,K)*DIFF(K)+X2
END LOOP 7
H
H
DO 8 K = 1,NUMVAL
PC(K,1) = PC(K,1)*V11+PC(K,2)*V21
PC(K,2) = PC(K,1)*V12+PC(K,2)*V22
END LOOP 8
DO 9 I = 1,NUMVA
PC(I,1) = DER(I,1)-PC(I,1)
PC(I,2) = DER(I,2)-PC(I,2)
END LOOP 9
CALL PSEUD ($C^\dagger$ IS OBT.)
DO 10 I = 1,NUMVAL
BEST(1) = PCINV(1,I)*DIFF(I)+BEST(1)
BEST(2) = PCINV(2,I)*DIFF(I)+BEST(2)
J

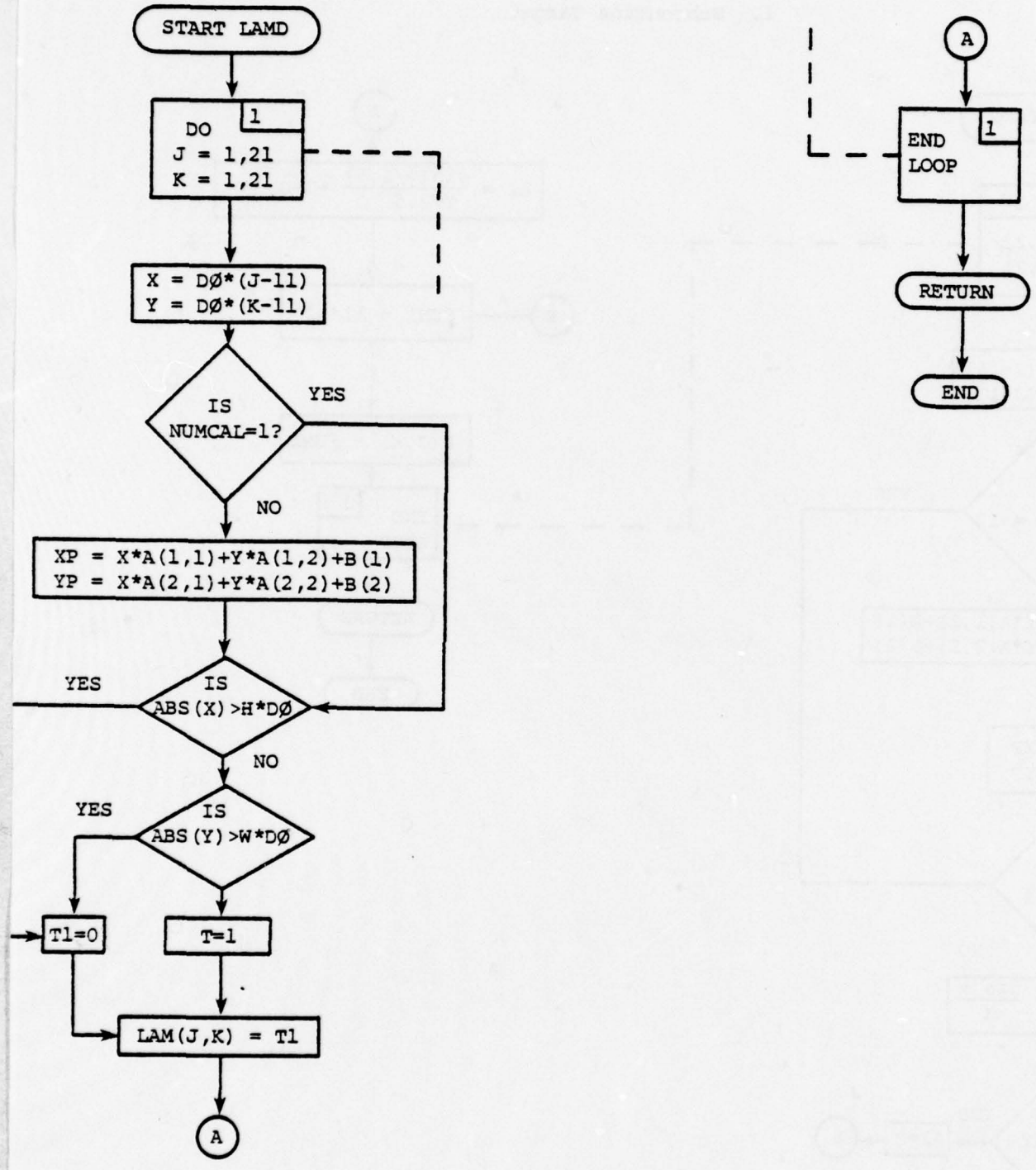

J
AC(1) = X1-V11*BEST(1)-V12*BEST(2)
AC(2) = X2-V21*BEST(1)-V22*BEST(2)
AC(1) = AC(1)+1.Ø
THETA = AC(2)/AC(1)
ALPHA = AC(1)*(1.0+(THETA*THETA)/2.0)
THETA = THETA*57.3Ø
E1 = B(1)-BEST(1)
E2 = B(2)-BEST(2)
ERROR = SQRT(E1**2+E2**2)
DISPLAY RESULT FOR EITHER OPTION
A
END

SUBROUTINES

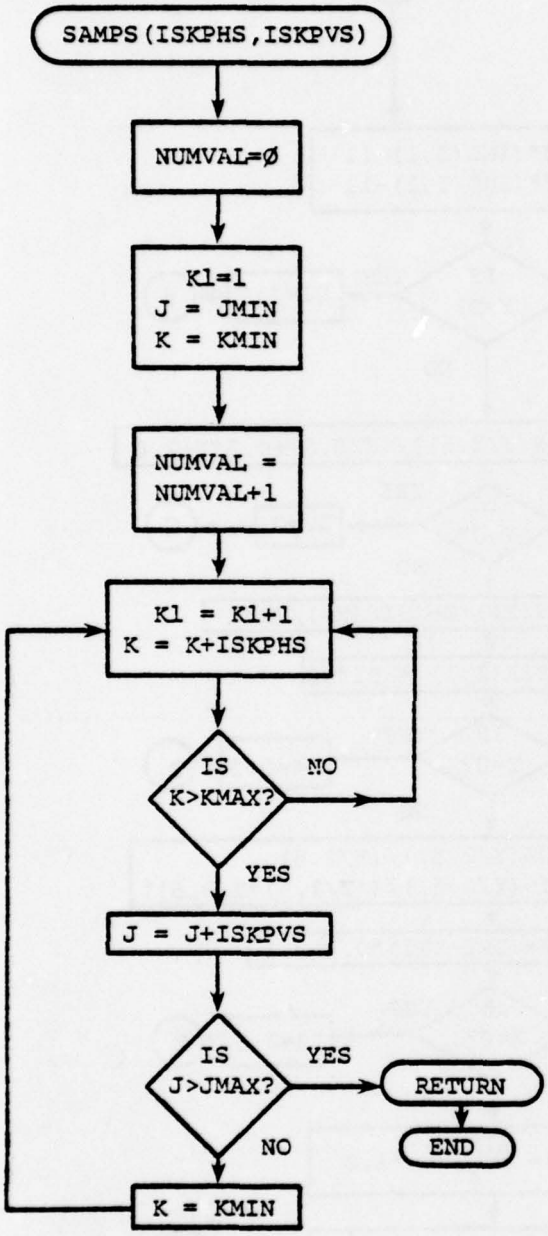
1. Subroutine Target



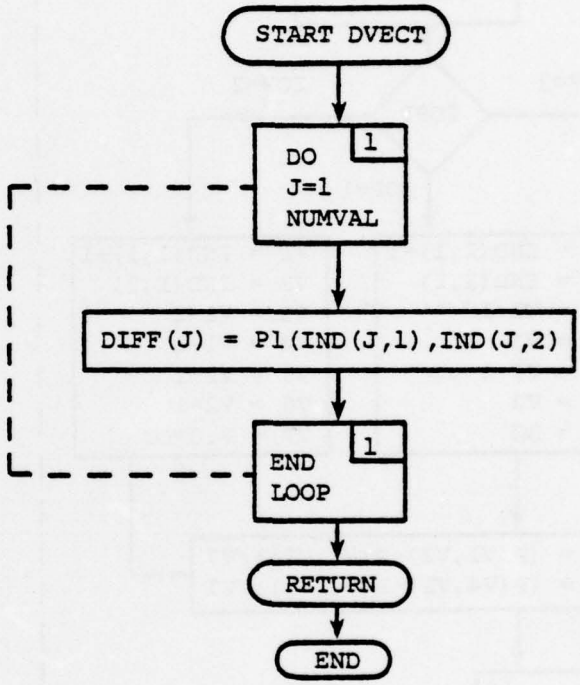
2, Subroutine Lambda



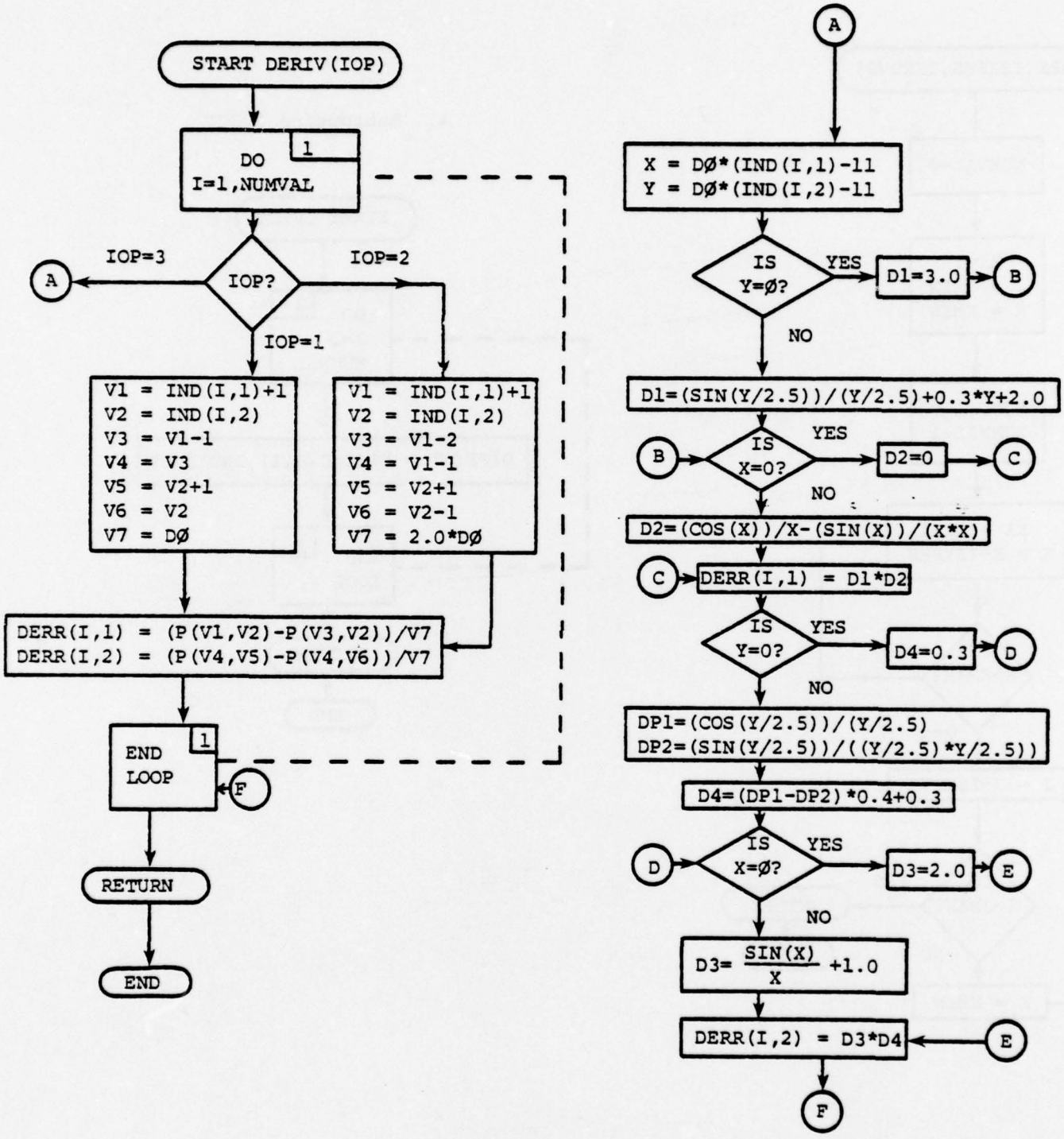
3. Subroutine Samps



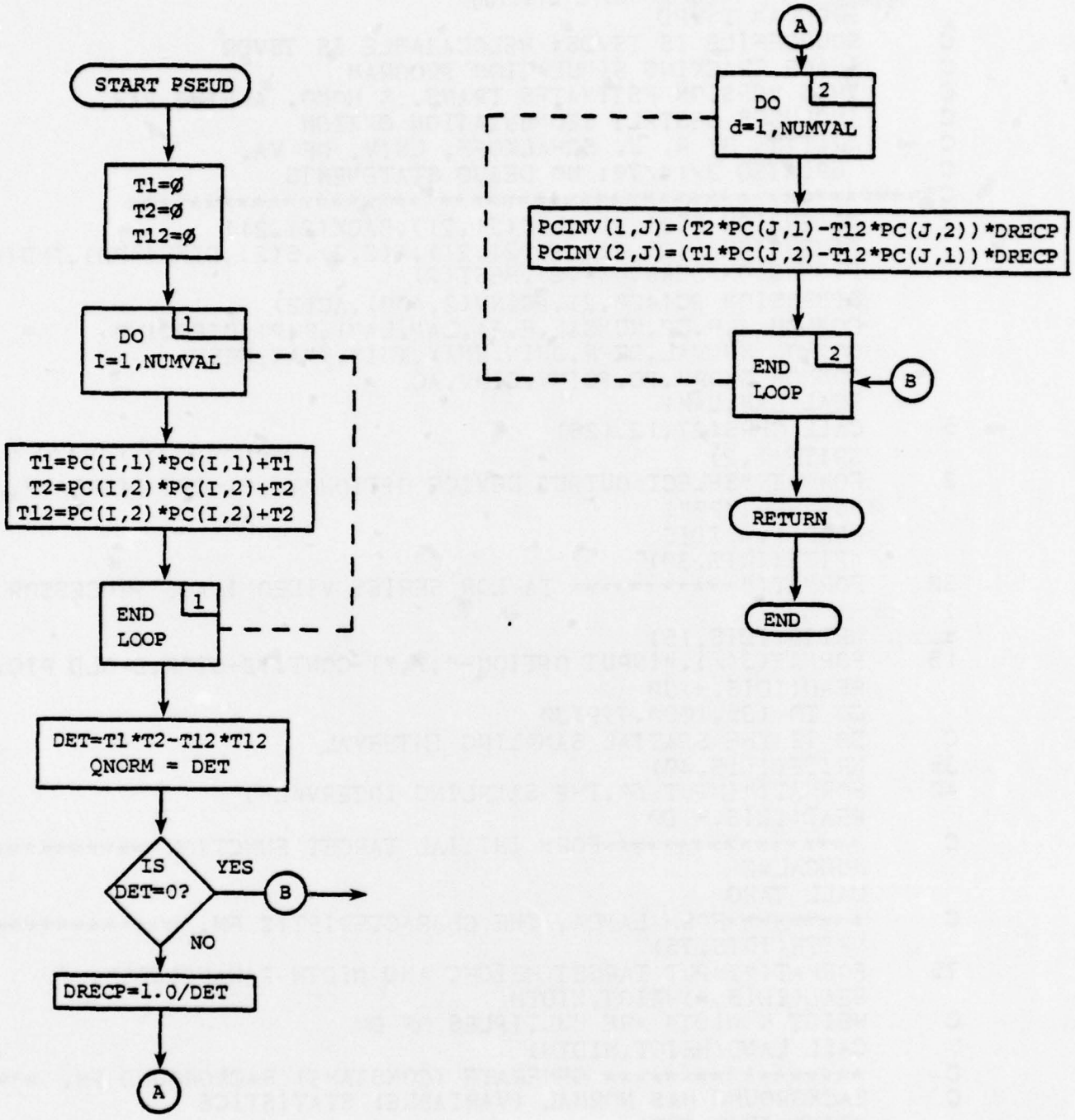
4. Subroutine DVECT



5. Subroutine DERIV(IOP)



6. Subroutine Pseud



Appendix 5.2 Program TSVPD Listing

```
      PROGRAM TSVPD
C     SOURCEFILE IS TSVDS; RELOCATABLE IS TSVDR
C     IMAGE TRACKING SIMULATION PROGRAM
C     THIS VERSION ESTIMATES TRANS. & HOMO. AFFINE PAR.
C     INCLUDES GESTALT SEGMENTATION OPTION
C     WRITTEN BY R. J. SCHALKOFF, UNIV. OF VA.
C      UPDATED 3/14/79; NO DEBUG STATEMENTS
C*************************************************
      DIMENSION F(21,21),LAM(21,21),BACK(21,21)
      DIMENSION P(21,21),PI(21,21),A(2,2),B(2),DIFF(400),IND(400,2)
      DIMENSION DERR(400,2),BEST(2)
      DIMENSION PC(400,2),PCINV(2,400),AC(2)
      COMMON A,B,D0,NUMCAL,F,FI,LAM,LAMI,P,PI,DIFF,IND
      COMMON NUMVAL,DERR,JMIN,JMAX,KMIN,KMAX,BEST
      COMMON ONORM,PC,PCINV,CINV,AC
      REAL LAM,LAMI
5     CALL CHRS(27,12,128)
      WRITE(1,2)
2     FORMAT("SELECT OUTPUT DEVICE OPTION",/,"1-CRT DISPLAY",/,
     &"6-TELETYPE")
      READ(1,*)IDIS
      WRITE(IDIS,30)
30    FORMAT("********** TAYLOR SERIES VIDEO IMAGE PROCESSOR ****
)
8     WRITE(IDIS,15)
15    FORMAT(3(/),"INPUT OPTION-",/,"1-CONT.;2-STOP,3-OLD PIC.")
      READ(IDIS,*)J0
      GO TO (35,1000,799)J0
C     D0 IS THE SPATIAL SAMPLING INTERVAL
35    WRITE(IDIS,40)
40    FORMAT("INPUT D0,THE SAMPLING INTERVAL")
      READ(IDIS,*)D0
C     ******************FORM INITIAL TARGET FUNCTION ***************
      NUMCAL=0
      CALL TARG
C     *********FORM LAMDA, THE CHARACTERISTIC FN. ************
      WRITE(IDIS,75)
75    FORMAT("INPUT TARGET HEIGHT AND WIDTH PARAMETERS")
      READ(IDIS,*)HEIGT,WIDTH
C     HEIGT & WIDTH ARE MULTIPLES OF D0
      CALL LAMD(HEIGT,WIDTH)
C     ****************** GENERATE (CONSTANT) BACKGROUND FN. *******
C     BACKGROUND HAS NORMAL (VARIABLE) STATISTICS
      WRITE(IDIS,350)
350   FORMAT("INPUT BACKGROUND MEAN, STD. DEV.")
      READ(IDIS,*)BM,BDEV
      DO 400 J=1,21
      DO 400 K=1,21
      CALL GAUSS(BDEV,BM,TI)
      BACK(J,K)=TI
400   CONTINUE
```

```
C     ********** ADD RANDOM (GAUSSIAN) NOISE TO SCENE ********
C     NOTE THAT NOISE IS ALWAYS >=0 IN MAGNITUDE
      WRITE(IDIS,505)
505   FORMAT("INPUT RANDOM SCENE NOISE MEAN, STD. DEV.")
      READ(IDIS,*)SNM,SNDEV
C     GENERATE COMPLETE INITIAL SCENE
      DO 500 J=1,21
      DO 500 K=1,21
402   CALL GAUSS(SNDEV,SNM,T0)
      IF(T0.LT.0.0)GO TO 402
      P(J,K)=F(J,K)*LAM(J,K)+(1.0-LAM(J,K))*BACK(J,K)
      P(J,K)=P(J,K)+T0
500   CONTINUE
C     ********* PROVIDE SCENE DISPLAY OPTION **************
      WRITE(IDIS,510)
510   FORMAT("PRINT SCENE DATA?")
      READ(IDIS,*)J1
      IF(J1.EQ.0)GO TO 799
      CALL CHRS(27,12,128)
      WRITE(IDIS,530)((P(J,K),K=1,21),J=1,21)
530   FORMAT(10(/),21(/,21(F4.2,1X)))
      WRITE(IDIS,905)D0,HEIGT,WIDTH,BM,BDEV
      WRITE(IDIS,532)
532   FORMAT(5(/),40X,"SCENE DATA: INITIAL SCENE")
C     *************** FORM PETURBED SCENE ****************
C     INPUT AFFINE DISTORTION PARAMETERS
799   WRITE(IDIS,800)
800   FORMAT(10(/),"INPUT ALPHA & THETA (DEGREES) ")
      READ(IDIS,*)ALPEX,THAEX
      WRITE(IDIS,810)
810   FORMAT("NOW THE TRANSLATION VECTOR,B")
      READ(IDIS,*)B(1),B(2)
C     FORM HOMOGENEOUS AFFINE MATRIX
      A(1,1)=ALPEX*COS(THAEX/57.2958)
      A(2,2)=A(1,1)
      A(1,2)=-1.0*ALPEX*SIN(THAEX/57.2958)
      A(2,1)=-1.0*A(1,2)
      NUMCAL=1
      CALL TARG
      CALL LAMD
      DO 880 J=1,21
      DO 880 K=1,21
812   CALL GAUSS(SNDEV,SNM,T0)
      IF(T0.LT.0.0)GO TO 812
      PI(J,K)=F(J,K)*LAM(J,K)+(1.0-LAM(J,K))*BACK(J,K)
      PI(J,K)=PI(J,K)+T0
880   CONTINUE
C     PETURBED SCENE DISPLAY OPTION
      WRITE(IDIS,521)
521   FORMAT("DISPLAY PERTURBED SCENE?")
      READ(IDIS,*)J2
      IF(J2.EQ.0)GO TO 450
```

```
      CALL CHRS(27,12,128)
      WRITE(IDIS,530)((P1(J,K),K=1,21),J=1,21)
      WRITE(IDIS,522)
522   FORMAT(5(/),40X,"PERTURBED SCENE DISPLAY")
      WRITE(IDIS,920)A(1,1),A(1,2),B(1),A(2,1),A(2,2),B(2)
      WRITE(IDIS,200)
200   FORMAT(3(/),"ACCEPTABLE PERTURBATION?")
      READ(IDIS,*)J5
      IF(J5.EQ.0)GO TO 799
C     HERE FILTER SCENE(S) IF DESIRED
C     LOOK AT SCENE DIFFERENCES
C     USE P1 TO HOLD DIFFRENCE SCENE
450   DO 300 I=1,21
      DO 300 J=1,21
      P1(I,J)=P1(I,J)-P(I,J)
300   CONTINUE
      WRITE(IDIS,302)
302   FORMAT(5(/),"DISPLAY SCENE DIFFERENCES?")
      READ(IDIS,*)IDIFF
      IF(IDIFF.EQ.0)GO TO 309
      CALL CHRS(27,12,128)
      WRITE(IDIS,530)((P1(I,J),J=1,21),I=1,21)
      WRITE(IDIS,308)
308   FORMAT(5(/),40X,"DIFFERENCE BETW. ORIG. & PERT. SCENES")
309   WRITE(IDIS,310)
310   FORMAT(5(/),"AUTOMATIC GESTALT SEGMENTATION OPTION?")
      READ(IDIS,*)IGES
      IF(IGES.EQ.1)GO TO 360
C     ********** WINDOW FUNCTION FORMULATION ******************
550   WRITE(IDIS,570)
570   FORMAT("INPUT WINDOW FN. LIMITS-JMIN,JMAX,KMIN,KMAX",/,
     &"RECALL JMIN,KMIN>=2: JMAX,KMAX<=20")
      READ(IDIS,*)JMIN,JMAX,KMIN,KMAX
C     ************** SAMPLING STRATEGY OPTIONS ***********
C     RECALL MAX. SCENE DIM. IS (N-1)X(N-1)
C     SAMPS IS SKIP SAMPLED
620   WRITE(IDIS,622)
622   FORMAT("INPUT SKIPHORIZ.;SKIPVERT")
      READ(IDIS,*)ISKPH,ISKPV
      CALL SAMPS(ISKPH,ISKPV)
C     ***************** FORM D, THE DIFFERENCE VECTOR ************
675   CALL DVECT
C     ********** FORM DERRIVATIVE ESTIMATES ***********
      WRITE(IDIS,690)
690   FORMAT("INPUT DERRIVATIVE ESTIMATION OPTION")
      READ(IDIS,*)IOPT2
C
      CALL DERIV(IOPT2)
      GO TO 355
C     ********AUTO. GESTALT SEGMENTATION OPTION ***********
360   WRITE(IDIS,362)
362   FORMAT("INPUT WIND U.L.H. CORNER BOUNDS,X,Y,DIM. &DER OPT ")
```

```
      READ(IDIS,*)ICSX,ICSY,ICFX,ICFY,IWX,IWY,IOPT2
      JMIN=ICSX
      JMAX=JMIN+IWX
      KMIN=ICSY
      KMAX=KMIN+IWY
941   CALL SAMPS(1,1)
      CALL DVECT
      CALL DERIV(IOPT2)
      GO TO 940
392   KMIN=KMAX
      KMAX=KMIN+IWY
      IF(KMIN.LT.ICFY)GO TO 941
      KMIN=ICSY
      KMAX=KMIN+IWY
      JMIN=JMAX
      JMAX=JMIN+IWX
      IF(JMIN.LT.ICFX)GO TO 941
      GO TO 8
C     ********* THRESHOLDING OPTION **********************
355   ITHR=0
C     ***************** TAYLOR SERIES & L.S. PARAMETER VECTOR EST.
**********
C     ONLY PERMITS TRANSLATION, ROTATION & MAGNIFICATION
C     FORM PC
940   DO 945 I=1,NUMVAL
      X=D0*(IND(I,1)-11)
      Y=D0*(IND(I,2)-11)
      PC(I,1)=X*DERR(I,1)+Y*DERR(I,2)
      PC(I,2)=X*DERR(I,2)-Y*DERR(I,1)
945   CONTINUE
C     FORM GRAMIAN TO SEE IF DC HAS FULL COLUMN RANK
      D11=0.
      D12=0.
      D13=0.
      D14=0.
      D22=0.
      D23=0.
      D24=0.
      D33=0.
      D44=0.
      DO 947 I=1,NUMVAL
      D11=PC(I,1)*PC(I,1)+D11
      D12=PC(I,1)*PC(I,2)+D12
      D13=PC(I,1)*DERR(I,1)+D13
      D14=PC(I,1)*DERR(I,2)+D14
      D22=PC(I,2)*PC(I,2)+D22
      D23=PC(I,2)*DERR(I,1)+D23
      D24=PC(I,2)*DERR(I,2)+D24
      D33=DERR(I,1)*DERR(I,1)+D33
      D34=DERR(I,1)*DERR(I,2)+D34
      D44=DERR(I,2)*DERR(I,2)+D44
947   CONTINUE
```

```
C       NOW FORM DETERMINANT
        D1=D33*D44-D34*D34
        D2=D13*D24-D14*D23
        D3=D13*D44-D14*D34
        D4=D23*D44-D24*D34
        D5=D23*D34-D24*D33
        D6=D13*D34-D14*D33
        P1=D11*(D22*D1-D23*D4+D24*D5)
        P2=D12*(D12*D1-D23*D3+D24*D6)
        P3=D13*(D12*D4-D22*D3+D24*D2)
        P4=D14*(D12*D5-D22*D6+D23*D2)
        D=P1-P2+P3-P4
        WRITE(IDIS,910)D
910     FORMAT(3(/),"GRAMIAN DETERMINANT =  ",E10.2)
C        FORM PCJNV
        CALL PSEUD
C       FORM PCINV*G
        V11=0.
        V12=0.
        V21=0.
        V22=0.
        DO 950 K=1,NUMVAL
        V11=PCINV(1,K)*DERR(K,1)+V11
        V12=PCINV(1,K)*DERR(K,2)+V12
        V21=PCINV(2,K)*DERR(K,1)+V21
        V22=PCINV(2,K)*DERR(K,2)+V22
950     CONTINUE
C        FORM PCINV*D
        X1=0.
        X2=0.
        DO 970 K=1,NUMVAL
        X1=PCINV(1,K)*DIFF(K)+X1
        X2=PCINV(2,K)*DIFF(K)+X2
970     CONTINUE
C       FORM -PC*PCINV*G; RESULT IN PC
        DO 955 K=1,NUMVAL
        PC(K,1)=PC(K,1)*V11+PC(K,2)*V21
        PC(K,2)=PC(K,1)*V12+PC(K,2)*V22
955     CONTINUE
C       FORM C
        DO 960 I=1,NUMVAL
        PC(I,1)=DERR(I,1)-PC(I,1)
        PC(I,2)=DERR(I,2)-PC(I,2)
960     CONTINUE
C       FORM BEST VECTOR
        CALL PSEUD
C        RETURNS CINV IN PCINV
        BEST(1)=0.
        BEST(2)=0.
        DO 963 I=1,NUMVAL
        BEST(1)=PCINV(1,I)*DIFF(I)+BEST(1)
        BEST(2)=PCINV(2,I)*DIFF(I)+BEST(2)
```

```
963   CONTINUE
C     FORM AC
      AC(1)=X1-V11*BEST(1)-V12*BEST(2)
      AC(2)=X2-V21*BEST(1)-V22*BEST(2)
C     FORM ESTIMATES OF ALPHA & THETA
      AC(1)=AC(1)+1.0
      THETA=AC(2)/AC(1)
      ALPHA=AC(1)*(1.0+(THETA*THETA)/2.0)
      THETA=THETA*57.30
C     COMPUTE B VECTOR EST. ERROR BASED ON L2 VECTOR NORM
      E1=B(1)-BEST(1)
      E2=B(2)-BEST(2)
      ERROR=SQRT(E1*E1+E2*E2)
C     ***************DISPLAY OF ALGORITHM RESULTS **********
***
      IF(IGES.EQ.1)GO TO 370
979   IF(ITHR.EQ.1)GO TO 980
      THRESH=0.0
      ISAVE=NUMVAL
980   CALL CHRS(27,12,128)
      WRITE(IDIS,900)
900   FORMAT(5(/),"---- FOR THIS CASE THE FOLLOWING WAS INPUT----")
      WRITE(IDIS,905)D0,HEIGT,WIDTH,BM,BDEV
905   FORMAT(5(/),"D0",8X,"HEIGHT",4X,"WIDTH",5X,"BMEAN",5X,"BDEV",/
(F
     &6.3,4X))
      WRITE(IDIS,914)ALPEX,THAEX
914   FORMAT(3(/),"ALPHA= ",3X,F8.5,"THETA= ",3X,F8.5)
      WRITE(IDIS,915)
915   FORMAT(3(/),"AFFINE TRANSFORM PARAMETERS==========")
      WRITE(IDIS,920)A(1,1),A(1,2),B(1),A(2,1),A(2,2),B(2)
920   FORMAT(2(/),10X,"A=",3X,2(F6.4,2X),"B=",F6.4,/,15X,2(F6.4,2X),
,F
     &6.4)
      WRITE(IDIS,925)JMIN,JMAX,KMIN,KMAX
925   FORMAT(3(/),"WITH WINDOW LIMITS---",/,"JMIN=",I2,5X,"JMAX=",I2
,
     &"KMIN=",I2,5X,"KMAX=",I2)
      WRITE(IDIS,926)ISAMP,ISKPH,ISKPV,ISAVE
926   FORMAT(3(/),"SAMPLING STRAT.",3X,"HORIZ. INCR.",3X,"VERT. INCR
,3
     &X,"RAW NO. SAMPLES",/,5X,I1,17X,I2,12X,I2,14X,I3)
      WRITE(IDIS,927)IOPT2,WOPT
927   FORMAT(3(/),"DERR. EST. OPTION",5X,"L.S. WEIGHTING OPTION",/,9
I1
     &,22X,I2)
      WRITE(IDIS,928)ITHR,THRESH,ICOUNT
928   FORMAT(3(/),"THRESHOLD OPTION",5X,"THRESHOLD",5X,"ADJ. NO. SAM
ES
     &",/,9X,I1,16X,F4.2,15X,I3)
      WRITE(IDIS,930)B(1),BEST(1),B(2),BEST(2)
930   FORMAT(5(/),5X,"B (EXACT)=",F6.4,10X,"B (EST)=",F6.4,/,
```

```
     &15X,F6.4,19X,F6.4)
      WRITE(IDIS,932)AC(1),AC(2)
932   FORMAT(5(/),5X,"AC(1)=",3X,F6.4,10X,"AC(2)=",F6.4)
      WRITE(IDIS,933)ALPHA,THETA
933   FORMAT(2(/),"ALPHA (EST)=  ",F8.5,5X,"THETA (EST)=   ",F8.5)
      WRITE(IDIS,931)ERROR
931   FORMAT(3(/),"B VECTOR ERROR,L2 NORM,=   ",F15.12)
      GO TO 8
C     *********GESTALT SEG. OPTION DISPLAY*******
370   WRITE(IDIS,925)JMIN,JMAX,KMIN,KMAX
      WRITE(IDIS,930)B(1),BEST(1),B(2),BEST(2)
      WRITE(IDIS,933)ALPHA,THETA
      GO TO 392
1000  END
C
C     ***************** SUBROUTINES ********************
      SUBROUTINE TARG
      DIMENSION F(21,21),LAM(21,21),BACK(21,21)
      DIMENSION P(21,21),P1(21,21),A(2,2),B(2),DIFF(400),IND(400,2)
      DIMENSION DERR(400,2),BEST(2)
      DIMENSION PC(400,2),PCINV(2,400),AC(2)
      COMMON A,B,D0,NUMCAL,F,F1,LAM,LAM1,P,P1,DIFF,IND
      COMMON NUMVAL,DERR,JMIN,JMAX,KMIN,KMAX,BEST
      COMMON ONORM,PC,PCINV,CINV,AC
      REAL LAM,LAM1
C     THIS ROUTINE GENERATES THE INITIAL AND PERTURBED TARGET FNS
      DO 200 J=1,21
      DO 200 K=1,21
C     THE POINT (J,K)=(11,11) CORRESP. TO (X,Y)=(0,0)
      X=D0*(J-11)
      Y=D0*(K-11)
      IF(NUMCAL.EQ.1)120,125
120   XP=X*A(1,1)+Y*A(1,2)+B(1)
      YP=X*A(2,1)+Y*A(2,2)+B(2)
      X=XP
      Y=YP
125   IF(X.EQ.0.0)GO TO 154
      A1=(SIN(X))/X+1.0
      GO TO 155
154   A1=2.0
155   IF(Y.EQ.0.0)GO TO 158
      A2=(SIN(Y/2.5))/(Y/2.5)+2.0+0.3*Y
      GO TO 159
158   A2=3.0
159   FUN1=A1*A2
185   F(J,K)=FUN1
200   CONTINUE
      RETURN
      END
      SUBROUTINE LAND(H,W)
      DIMENSION F(21,21),LAM(21,21),BACK(21,21)
      DIMENSION P(21,21),P1(21,21),A(2,2),B(2),DIFF(400),IND(400,2)
```

```
      DIMENSION DERR(400,2),BEST(2)
      DIMENSION PC(400,2),PCINV(2,400),AC(2)
      COMMON A,B,D0,NUMCAL,F,F1,LAM,LAM1,P,P1,DIFF,IND
      COMMON NUMVAL,DERR,JMIN,JMAX,KMIN,KMAX,BEST
      COMMON QNORM,PC,PCINV,CINV,AC
      REAL LAM,LAM1
C     FORMS THE INITIAL AND PERTURBED CHAR. FUNCTIONS
C     OBJECT INITIAL BOUNDARY IS FN. OF HEIGT & WIDTH
      DO 300 J=1,21
      DO 300 K=1,21
      X=D0*(J-11)
      Y=D0*(K-11)
      IF(NUMCAL.EQ.1)320,325
320   XP=X*A(1,1)+Y*A(1,2)+B(1)
      YP=X*A(2,1)+Y*A(2,2)+B(2)
      X=XP
      Y=YP
C     LATER MAKE CHARACTERISTIC FUNCTION USER ADJUSTABLE
325   IF(ABS(X).GT.(H*D0))GO TO 265
      IF(ABS(Y).GT.(W*D0))GO TO 265
      T1=1
      GO TO 285
265   T1=0.0
285   LAM(J,K)=T1
300   CONTINUE
      RETURN
      END
      SUBROUTINE SAMPS(ISKPHS,ISKPVS)
      DIMENSION F(21,21),LAM(21,21),BACK(21,21)
      DIMENSION P(21,21),P1(21,21),A(2,2),B(2),DIFF(400),IND(400,2)
      DIMENSION DERR(400,2),BEST(2)
      DIMENSION PC(400,2),PCINV(2,400),AC(2)
      COMMON A,B,D0,NUMCAL,F,F1,LAM,LAM1,P,P1,DIFF,IND
      COMMON NUMVAL,DERR,JMIN,JMAX,KMIN,KMAX,BEST
      COMMON QNORM,PC,PCINV,CINV,AC
      REAL LAM,LAM1
C     RECTANGULAR SKIP-SAMPLED STRATEGY
C     NUMVAL IS NO. OF VALID ROWS IN IND( , )
      NUMVAL=0
      K1=1
      J=JMIN
      K=KMIN
20    IND(K1,1)=J
      IND(K1,2)=K
      NUMVAL=NUMVAL+1
      K1=K1+1
      K=K+ISKPHS
      IF(K.GT.KMAX)GO TO 105
      GO TO 20
105   J=J+ISKPVS
      IF(J.GT.JMAX)GO TO 106
      K=KMIN
```

```
      GO TO 20
106   RETURN
      END
      SUBROUTINE DVECT
      DIMENSION F(21,21),LAM(21,21),BACK(21,21)
      DIMENSION P(21,21),P1(21,21),A(2,2),B(2),DIFF(400),IND(400,2)
      DIMENSION DERR(400,2),BEST(2)
      DIMENSION PC(400,2),PCINV(2,400),AC(2)
      COMMON A,B,D0,NUMCAL,F,F1,LAM,LAM1,P,P1,DIFF,IND
      COMMON NUMVAL,DERR,JMIN,JMAX,KMIN,KMAX,BEST
      COMMON QNORM,PC,PCINV,CINV,AC
      REAL LAM,LAM1
C     FORMS D VECTOR, BASED UPON SAMPLING STRATEGY
      DO 100 J=1,NUMVAL
      DIFF(J)=P1(IND(J,1),IND(J,2))
100   CONTINUE
      RETURN
      END
      SUBROUTINE DERIV(IOP)
      DIMENSION F(21,21),LAM(21,21),BACK(21,21)
      DIMENSION P(21,21),P1(21,21),A(2,2),B(2),DIFF(400),IND(400,2)
      DIMENSION DERR(400,2),BEST(2)
      DIMENSION PC(400,2),PCINV(2,400),AC(2)
      COMMON A,B,D0,NUMCAL,F,F1,LAM,LAM1,P,P1,DIFF,IND
      COMMON NUMVAL,DERR,JMIN,JMAX,KMIN,KMAX,BEST
      COMMON QNORM,PC,PCINV,CINV,AC
      REAL LAM,LAM1
C     FORMS SPATIAL DERRIVATIVE ESTIMATES FROM P( , )

C     IOP DETERMINES DERRIVATIVE ESTIMATION ALGORITHM
C     IOP=1: USES I &I+1, IOP=2: USES I-1 &I+1
C     IOP=3: COMPUTES EXACT DERIVATIVES
      DO 750 I=1,NUMVAL
      GO TO (100,300,400)IOP
100   V1=IND(I,1)+1
      V2=IND(I,2)
      V3=V1-1
      V4=V3
      V5=V2+1
      V6=V2
      V7=D0
      GO TO 500
300   V1=IND(I,1)+1
      V2=IND(I,2)
      V3=V1-2
      V4=V1-1
      V5=V2+1
      V6=V2-1
      V7=2.0*D0
      GO TO 500
C      EXACT DERIVATIVE COMPUTATION
400   X=D0*(IND(I,1)-11)
```

```
      Y=D0*(IND(I,2)-11)
C     FORM DF/DX
      IF(Y.EQ.0.0)GO TO 10
      D1=(SIN(Y/2.5))/(Y/2.5)+0.3*Y+2.0
      GO TO 12
10    D1=3.0
12    IF(X.EQ.0.0)GO TO 20
      D2=(COS(X))/X-(SIN(X))/(X*X)
      GO TO 22
20    D2=0.0
22    DERR(I,1)=D1*D2
C     FORM DF/DY
      IF(Y.EQ.0.0)GO TO 30
      DP1=(COS(Y/2.5))/(Y/2.5)
      DP2=(SIN(Y/2.5))/((Y/2.5)*(Y/2.5))
      D4=0.4*(DP1-DP2)+0.3
      GO TO 32
30    D4=0.3
32    IF(X.EQ.0.0)GO TO 40
      D3=(SIN(X))/X+1.0
      GO TO 42
40    D3=2.0
42    DERR(I,2)=D3*D4
      GO TO 750
500   DERR(I,1)=(P(V1,V2)-P(V3,V2))/V7
      DERR(I,2)=(P(V4,V5)-P(V4,V6))/V7
750   CONTINUE
      RETURN
      END
      SUBROUTINE PSEUD
C     COMPUTES THE PSEUDOINVERSE OF AN NSIZEX2 MATRIX
      DIMENSION F(21,21),LAM(21,21),BACK(21,21)
      DIMENSION P(21,21),P1(21,21),A(2,2),B(2),DIFF(400),IND(400,2)
      DIMENSION DERR(400,2),BEST(2)
      DIMENSION PC(400,2),PCINV(2,400),AC(2)
      COMMON A,B,D0,NUMCAL,F,F1,LAM,LAM1,P,P1,DIFF,IND
      COMMON NUMVAL,DERR,JMIN,JMAX,KMIN,KMAX,BEST
      COMMON QNORM,PC,PCINV,CINV,AC
      REAL LAM,LAM1
      T1=0.
      T2=0.
      T12=0.
      DO 100 I=1,NUMVAL
      T1=PC(I,1)*PC(I,1)+T1
      T2=PC(I,2)*PC(I,2)+T2
      T12=PC(I,1)*PC(I,2)+T12
100   CONTINUE
      DET=T1*T2-T12*T12
      QNORM=DET
      IF(DET.EQ.0.)GO TO 200
      DRECP=1.0/DET
      DO 200 J=1,NUMVAL
```

```
      PCINV(1,J)=(T2*PC(J,1)-T12*PC(J,2))*DRECP
      PCINV(2,J)=(T1*PC(J,2)-T12*PC(J,1))*DRECP
200   CONTINUE
      RETURN
      END
      END$
```

Appendix 6 - Simple Physical Interpretation of the TSVIP Approach and Some Numerical Examples

(a) Physical Interpretation in 1-D

Some insight into the analysis of Section 5.4 may be obtained by considering a simple 1-D example. Consider a 1-D function $f(x;t)$, which is subjected to a time-varying translation, i.e.,

$$f(x;t_2) = f[x + \delta(t_2-t_1);t_1] \qquad \text{(A.6-1)}$$

A first order Taylor series approximation to this function would be:

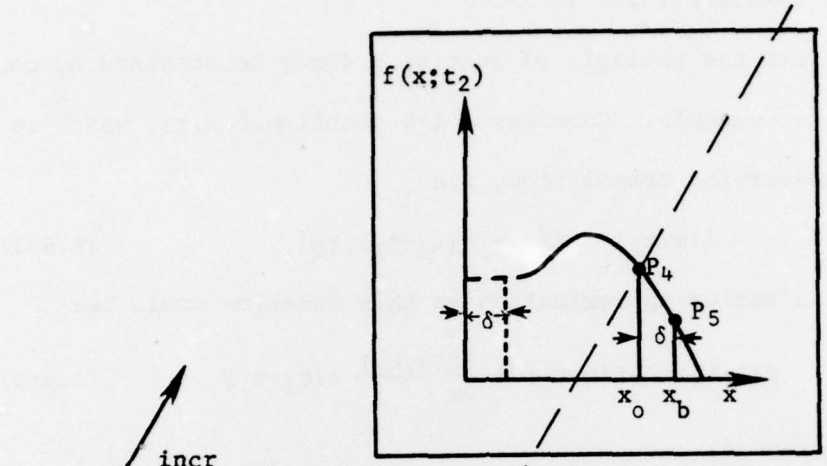
$$f(x;t_2) \simeq f(x;t_1) + \frac{\partial f(x;t_1)}{\partial x}\,\delta(t_2-t_1) \qquad \text{(A.6-2)}$$

which may be rewritten as:

$$\delta(t_2-t_1) \simeq \frac{f[x;t_2] - f[x;t_1]}{\frac{\partial f[x;t_1]}{\partial x}} \qquad \text{(A.6-3)}$$

Therefore, for any value of x_o in the sensor range, $f(x_o;t)$ at times t_1 and t_2 and an estimate of $\frac{df[(x_o;t_1)]}{dx}$ may be used to estimate $\delta(t_2-t_1)$ using (A.6-3).

A physical interpretation of this approach is shown in Figure 31. Two successive samples (i.e. "scenes") of a 1-D function $f(x;t)$ are shown. For simplicity, in the time interval between t_1 and t_2, $f(x;t)$ is translated by an amount, δ , as shown in Figure 31. Due to this translation, the functional intensity at point x_o and time t_1, i.e. $f(x_o;t_1)$, shown as P_1 in the figure is no longer the same at this same point at time t_2, i.e., at P_4. It is clear, however, that at time t_2, the value of the function would be equal to the value at P_5, i.e. $f(x_b;t_2) = f(x_o;t_1)$. (Also note that this translation would be accomplished, with the x axis positive direction as shown, by making $f(x;t_2) = f(x-\delta;t_1)$ since the function at $t=t_2$ is actually a spatially delayed version of $f(x;t_1)$.) The sensor "line of sight" is chosen such that the function intensity is sampled at point x_o in both scenes.

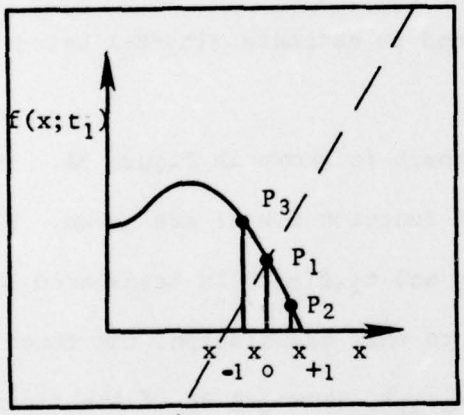


Present Field
$t = t_2 = (k+1)T$

$$\delta(t_2 - t_1) \approx \frac{f(x_o;t_2) - f(x_o;t_1)}{\left(\frac{\partial f(x_o;t_1)}{\partial x}\right)}$$

F2

incr time



Past Field
$t = t_1 = kT$

$$\frac{\partial f(x_o;t_1)}{\partial x} \approx \frac{f(x_{+1};t_1) - f(x_{-1};t_1)}{x_{+1} - x_{-1}}$$

F1

Sensor "Line of Sight", i.e.,
Sensor Coordinate X_o

Figure 31 Simple Physical Interpretation of TSVIP Approach in 1-D

Thus, to employ the TSVIP approach, we proceed as follows:

(1) At time $t = t_1$ and $x = x_o$, the spatial derivative of $f(x;t_1)$, i.e., $\frac{\partial f(x_o;t_1)}{\partial x}$ is estimated using the value of the function at points x_{-1} and x_{+1} as

$$\frac{\partial f(x_o;t_1)}{\partial x} \simeq \frac{f(x_{+1};t) - f(x_{-1};t)}{x_{+1}-x_{-1}} \qquad \text{(A.6-7)}$$

(2) Using the values of the function at x_o and times t_1 and t_2 (i.e., these intensities shown as P_1 and P_4 in Figure 20) and the above derivative estimate, $\delta(t_2-t_1)$ is calculated using (A.6-3).

It is also interesting to note if $\frac{\partial f(x_o;t_1)}{\partial x} = 0$, the algorithm obviously fails. This is due to the fact that $f(x;t_1)$ has no 1-D "texture," or equivalently, the "matrix" D_C (actually a 1x1 matrix in 1-D) in (5.4.4.2-10) is "singular." This is analogous to the 2-D case examined in Sections 5.4.4.2 and 5.4.4.3.

(b) 1-D Numerical Examples

(Example 1)

In this simple example, the functions

$$f(x;t_1) = \cos x \qquad \text{and}$$

$$f(x;t_2) = \cos(x+\delta)$$

are considered. For $\Delta = \pi/6 = .52360$ (six times the Nyquist rate) and $\delta = -2\pi/20 = -.15708$ (30% of Δ), three consecutive sample function values would be:

x	$x+\delta$	$\cos x$	$\cos x+\delta$
.52360	.36652	.86603	.93358
1.04720	.89012	.50000	.62932
1.5708	1.4137	.00000	.15645

For simplicty, x_o is chosen as the second sample, i.e. x_o=1.04720. The calculations, as outlined in part (a) would proceed as follows:

(1) at $t = t_1$

$$\frac{\partial f(x_o;t_1)}{\partial x} \simeq \frac{0 - .86603}{1.5708-.52360} = -.8270$$

(Note: the exact $\frac{\partial f}{\partial x}(x_o;t_1) = -\sin(1.04720) = -.866$)

(2) Using (A.6-3) and the above data,

$$\delta(t_2-t_1) = \frac{.62932-.50000}{-.8270} = -.1564$$

$$\% \text{ error} = \frac{|-.1564 -(-.15708)|}{+.15708} = 0.43\%$$

(Example 2)

Using the same data as in Example 1, but with a larger translation, i.e., let $\delta(t_2-t_1) = -2\pi/10 = -.62832$ (120% of Δ)

at x_o; $\delta(t_2-t_1)+x = 1.047720 - .62832 = .41888$

$$\cos(x_o+\delta) = \cos(.41888) = .91355$$

Therefore, from (A.6-3), the translational estimate is

$$\delta(t_2-t_1) = \frac{.91355-.50000}{-.8270} = -.5006$$

$$\% \text{ error} = \frac{|-.5006-(-.62832)|}{+.62832} = 20.3\%$$

Clearly, the effect of the assumption as to a "small" perturbation on the algorithm estimates is evidenced in the above two examples.

(c) 2-D Numerical Examples Using the TSVIP Algorithm

A simple numerical example of the TSVIP algorithm procedures is given here. The raw data used for this example is taken from the sample scene described in Chapter VI. The initial, perturbed and difference scenes are shown in Figure 32. For simplicity, N = 4 was chosen, and the four points resulting from a raster scan are shown circled in the figure. Derivatives

```
0.0033  0.0518  0.1783  0.2956  0.4180  0.5844  0.6389  0.7375  0.8383  0.9361  0.0001
0.0700  0.2027  0.3305  0.4644  0.5743  0.6001  0.0010  0.0006  0.0133  0.1131  0.2009
0.1995  0.3244  0.4052  0.5020  0.7047  0.0230  0.0370  0.0470  0.1540  0.2561  0.3542
0.2704  0.4179  0.5465  0.6754  0.0001  0.0006  0.0300  0.1401  0.2570  0.3000  0.4000
0.3003  0.4670  0.6002  0.7302  1.0000  0.0000  0.0070  0.2107  0.3190  0.4246  0.5253
0.3471  0.4640  0.6209  0.7010  0.0770  0.0000  0.1170  0.2310  0.3400  0.4460  0.5471
0.3003  0.4670  0.6002  0.7302  1.0000  0.0000  0.0070  0.2107  0.3190  0.4246  0.5253
0.2704  0.4179  0.5465  0.6754  0.0001  0.0006  0.0300  0.1401  0.2570  0.3000  0.4000
0.1995  0.3244  0.4052  0.5020  0.7047  0.0230  0.0370  0.0470  0.1540  0.2561  0.3542
0.0700  0.2027  0.3305  0.4644  0.5743  0.6001  0.0010  0.0006  0.0133  0.1131  0.2009
0.0033  0.0518  0.1783  0.2956  0.4180  0.5844  0.6389  0.7375  0.8383  0.9361  0.0001

SD        WEIGHT    WIDTH     MEAN      DISC
.200      0.000     0.000      .000      .000

                    SCENE DATA; INITIAL SCENE

0.0000  0.1240  0.3181  0.4383  0.5546  0.6667  0.7740  0.8750  0.9705  0.0700   .0000
0.0012  0.3327  0.4601  0.5835  0.7000  0.0100  0.0200  0.0300  0.1300  0.2300   .0000
0.1956  0.4327  0.5607  0.6056  0.0173  0.0340  0.0400  0.1573  0.2624  0.3634   .0000
0.2704  0.0125  0.0442  0.7710  0.0001  0.0140  0.1291  0.2300  0.3402  0.4400   .0000
0.4186  0.5493  0.0020  0.0104  1.0000  0.0544  0.1701  0.0010  0.3006  0.4017   .0000
0.4186  0.5493  0.0020  0.0104  0.9340  0.0044  0.1701  0.0010  0.3006  0.4017   .0000
0.2704  0.0125  0.0442  0.7710  0.0001  0.0140  0.1291  0.2300  0.3402  0.4400   .0000
0.1956  0.4327  0.5607  0.6056  0.0173  0.0340  0.0400  0.1573  0.2624  0.3634   .0000
0.0012  0.3327  0.4601  0.5835  0.7000  0.0100  0.0200  0.0300  0.1300  0.2300   .0000
0.0000  0.1240  0.3181  0.4383  0.5546  0.6667  0.7740  0.8750  0.9705  0.0700   .0000
 .0000   .0000   .0000   .0000   .0000   .0000   .0000   .0000   .0000   .0000   .0000

                    PERTURBED SCENE DISPLAY

                A=  1.0000   .0000  B= .1000
                     .0000  1.0000     .1000

 .1407   .1427   .1402   .1407   .1425   .1423   .1400   .1417   .1413   .1400 -0.0001
 .1303   .1300   .1296   .1291   .1005   .1070   .1073   .1000   .1000   .1001 -0.0000
 .1101   .1103   .1140   .1130   .1105   .1110   .1100   .1000   .1004   .1073 -0.3542
 .1002   .0920   .0977   .0004   .0000   .0030   .0000   .0007   .0000   .0070 -0.4000
 .0033   .0010   .0700   .0700   .0740   .0744   .0700   .0707   .0000   .0071 -0.5253
 .0000   .0033   .0011   .0000   .0007   .0044   .0000   .0000   .0470   .0450 -0.0471
 .0473   .0447   .0401   .0304   .0300   .0340   .0310   .0000   .0005   .0040 -0.5253
 .0003   .0000   .0032   .0000   .0171   .0140   .0110   .0003   .0004   .0000 -0.4000
 .0117   .0003   .0040   .0010  -.0019  -.0000  -.0000  -.0117  -.0140  -.0170 -0.3542
-.0040  -.0007  -.0104  -.0101  -.0100  -.0034  -.0000  -.0304  -.0330  -.0371 -0.0000
-0.0033 -0.0518 -0.1703 -0.2000 -0.4100 -0.0044 -0.0300 -0.7370 -0.0303 -0.0301 -0.0001

                    DIFFERENCE BETW. ORIG. & PERT. SCENES
```


Figure 32 Initial, Perturbed and Difference Scenes with Four Points Used for the Numerical Example

were estimated using (A.6-4). It should also be noted that the scene pixel location (6,6) corresponds to the (arbitrarily chosen) coordinate system origin, with positive x_1 and x_2 axes downward and to the right respectively. In this example, only translation was considered, and the following exact parameters were used:

$$\alpha = 1.0 \text{ (no dilation)} \tag{A.6-5a}$$

$$\Theta = 0.0 \text{ (no rotation)} \tag{A.6-5b}$$

$$\underset{\sim}{b} = \begin{bmatrix} .1 \\ .1 \end{bmatrix} \text{ (translation of } 0.5\Delta \text{ up and to the left)} \tag{A.6-5c}$$

Using these points from Figure 32 , the following sample data would result:

$$d = \begin{bmatrix} .03160 \\ .0726 \\ .0762 \\ .0368 \end{bmatrix} \tag{A.6-6}$$

$$G = \begin{bmatrix} -0.2025 & 0.47675 \\ 0.2025 & 0.57675 \\ 0.19450 & 0.61925 \\ -0.19450 & 0.61925 \end{bmatrix} \tag{A.6-7}$$

$$P_C = \begin{bmatrix} 0.07485 & 0.15585 \\ 0.07485 & -0.15585 \\ -0.16275 & -0.08495 \\ -0.16275 & 0.08495 \end{bmatrix} \tag{A.6-8}$$

Using the above data and (5.4.4.3-31) yields

$$P_C{}^T P_C = \begin{bmatrix} -.06718 & 0 \\ 0 & .06301 \end{bmatrix} \tag{A.6-9}$$

and from (5.4.4.3-29)

$$v_1 = 4.05 \times 10^{-3} \qquad \text{(A.6-10)}$$

Recall (5.4.4.3-8), i.e.,

$$C = (I - P_c P_c^{\dagger})G \qquad \text{(A.6-11)}$$

Therefore from (5.4.4.3-13), (A.6-8) and (A.6-9) we get

$$C = \begin{bmatrix} .03535 & .71113 \\ -.03535 & .71113 \\ .06485 & .32705 \\ -.06485 & .32705 \end{bmatrix} \qquad \text{(A.6-12)}$$

Then, using (5.4.4.3-12),

$$C^{\dagger} = \begin{bmatrix} 3.2398 & -3.2398 & 5.9437 & -5.9437 \\ .58035 & .58035 & .26691 & .26691 \end{bmatrix} \qquad \text{(A.6-13)}$$

So the estimated translational vector, b, is obtained using (A.6-13), (A.6-6) and (5.4.4.3-28), i.e.,

$$\hat{\underset{\sim}{b}} = C^{\dagger}\underset{\sim}{d} \qquad \text{(A.6-14)}$$

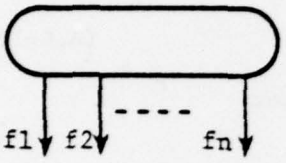
or, numerically,

$$\hat{\underset{\sim}{b}} = \begin{bmatrix} .10135 \\ .09063 \end{bmatrix} \qquad \text{(A.6-15)}$$

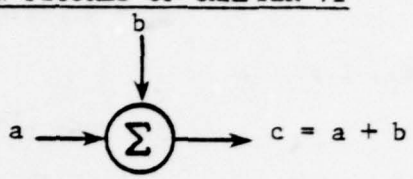
It should be noted that this estimate is in excellent agreement with the exact parameter values given in (A.6-5c).

APPENDIX 7.

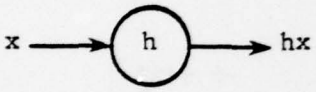
SYMBOLS USED IN FIGURES OF CHAPTER VI



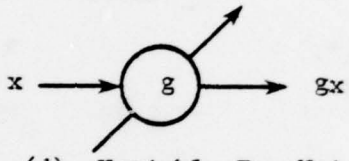
(a) Sensor



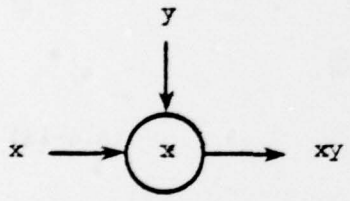
(b) Summer



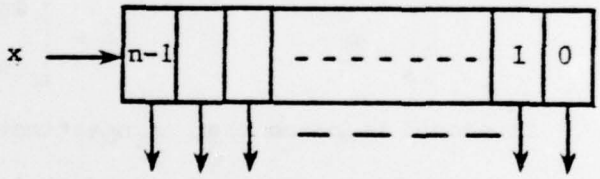
(c) Fixed Weight



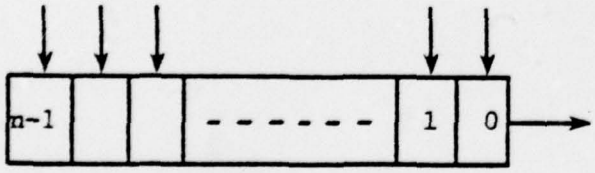
(d) Variable Tap Weight



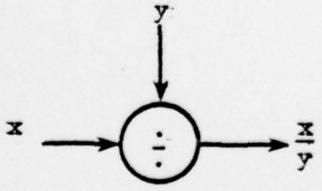
(e) Analog Multiplier



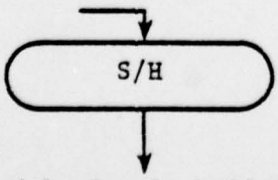
(f) SI/PO Shift Register



(g) PI/SO Shift Register



(h) Analog Divider



(i) Sample-hold Device

DISTRIBUTION LIST

Copy No.	
1	Dr. T. F. Tao, Code 62TV Naval Post Graduate School Monterey, CA 93940
2	Library Naval Post Graduate School Monterey, CA 93940
3 - 6	Office of Naval Research Code 212 800 N. Quincy St. Arlington, VA 22217
7	Office of Naval Research Code 221 800 N. Quincy St. Arlington, VA 22217
8	Office of Naval Research Code 432 800 N. Quincy St. Arlington, VA 22217
9	Naval Sea Systems Command Code 0341 Washington, D. C. 20362
10	Naval Sea Systems Command Code 03132 Washington, D. C. 20362
11	Naval Research Laboratory Code 5550 Washington, D. C. 20375
12	Naval Research Laboratory Code 1409 Washington, D. C. 20375
13	Naval Surface Weapons Center Dahlgren Laboratory Dahlgren, VA 22448 Code N-54

DISTRIBUTION LIST (continued)

Copy No.	
14	Naval Surface Weapons Center White Oak Laboratory Silver Spring, MD 20910 Code G-42
15	Naval Weapons Center Code 3945 China Lake, CA 93555
16	Naval Weapons Center Code 3133 China Lake, CA 93555
17	Naval Air Systems Command Code 360 Washington, D. C. 20361
18	U. S. Army Electronics Command Code NL-BP Fort Monmouth, NJ 07703
19 - 30	Defense Documentation Center Bldg. 5, Cameron Station Alexandria, VA 22314
31	Defense Advanced Research Project Agency 1400 Wilson Boulevard Arlington, VA 22209 CDR T. Wiener
32	Army Research Office P. O. Box 12211 Research Triangle Park, NC 27709
33	Brooks O. Bartholow Office of University Research DOT Headquarters Building 499 7th St., S. W. Washington, D. C.
34	ONRRR Attention: C. R. Main 2110 G. St., N. W. Washington, D. C. 20037

DISTRIBUTION LIST (continued)

Copy No.	
35 - 36	E. S. McVey
37 - 38	E. A. Parrish
39 - 40	R. M. Inigo
41 - 42	R. J. Schalkoff
43	I. A. Fisher
44	E. H. Pancake Science/Technology Information Center Clark Hall
45	RLES Files

IBUTION LIST (continued)

S. McVey

A. Parrish

M. Inigo

J. Schalkoff

A. Fisher

H. Pancake
ience/Technology Information Center
ark Hall

ES Files