1

MTI-R89-006-28

Micro-Computer Network Architecture for Range Instrumentation Applications

Volume 1

Mitchell R. Belzer
Yong M. Cho
Jae H. Han
Chun Y. Moon
Jin S. Suh
Sung S. Kwak

Mentor Technologies, Inc.

1992

US Army
White Sands Missile Range

"Views, opinions, and/or findings contained in this report are those of the authors and should not be construed as an official Department of the Army position, policy, or decision unless so designated by other official documentation."

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION
unclassified

1b. RESTRICTIVE MARKINGS

2a. SECURITY CLASSIFICATION AUTHORITY

2b. DECLASSIFICATION / DOWNGPADING SCHEDULE

3. DISTRIBUTION / AVAILABILITY OF REPORT
approved for public release;
distribution is unlimited

4. PERFORMING ORGANIZATION REPORT NUMBER(S)

5. MONITORING ORGANIZATION REPORT NUMBER(S)

6a. NAME OF PERFORMING ORGANIZATION
Mentor Technologies, Inc.

6b. OFFICE SYMBOL
(If applicable)

7a. NAME OF MONITORING ORGANIZATION
U.S. Army White Sands Missile Range

6c. ADDRESS *(City, State, and ZIP Code)*
12750 Twinbrook Parkway, Suite 101
Rockville, Maryland 20852

7b. ADDRESS *(City, State, and ZIP Code)*
Commanding Officer, STEWS-ID-T
U.S. Army White Sands Missile Range
New Mexico 88002-5143

8a. NAME OF FUNDING / SPONSORING ORGANIZATION

8b. OFFICE SYMBOL
(If applicable)

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
DAAD07-89-C-0214

8c. ADDRESS *(City, State, and ZIP Code)*

10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT ACCESSION NO.
665502	1P665502		

11. TITLE *(Include Security Classification)*
Micro-Computer Network Architecture for Range Instrumentation Applications
Volume 1

12. PERSONAL AUTHOR(S)
M. Belzer, Y. Cho, J. Han

13a. TYPE OF REPORT
Final Technical

13b. TIME COVERED
FROM 09Sep89 TO 18Sep91

14. DATE OF REPORT *(Year, Month, Day)*
91 Dec. 18

15. PAGE COUNT

16. SUPPLEMENTARY NOTATION

17. COSATI CODES

FIELD	GROUP	SUB-GROUP

18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
integrated tracking; decentralized estimation; video tracking

19. ABSTRACT *(Continue on reverse if necessary and identify by block number)*

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS

21. ABSTRACT SECURITY CLASSIFICATION
unclassified

22a. NAME OF RESPONSIBLE INDIVIDUAL
Foo Lam

22b. TELEPHONE *(Include Area Code)*
(505) 678-3010

22c. OFFICE SYMBOL
STEWS-ID-T

Previous editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

Abstract

Micro-Computer Network Architecture for
Range Instrumentation Applications

First, the basic Decentralized Square Root Information Filter (DSRIF) theory was extended in 2 ways. An expression for the likelihood function in terms of DSRIF variables, and a method for distributing the prior and process noise statistics over the set of locally optimal filters were derived.

Next, software developed in Phase I research was upgraded to enable multitarget tracking within our distributed filtering environment. Outlier detection/rejection, track initiation, measurement-to-track and track-to-track association routines were encoded and successfully tested using real MLRS data provided by WSMR.

Finally, MTI conducted experimental work wherein 2 video trackers were built and networked with a global processor. Laboratory testing showed that the 2-camera network could track a variety of objects over the entire laboratory space. Moreover, each tracker was able to aid the other in acquiring a common target. Field testing of an individual tracker showed that it could acquire and track objects similar to the SADARM submunition, and against a cloudy background!

Contents

List of Figures

List of Tables

List of Symbols

i	superscripted local system number
j	superscripted vector element number
k	time index, may be subscripted or enclosed in parentheses
x_k	global state vector
w_k	global process noise vector
Φ_k	global state transition matrix
x_k^i	local state vector
${}_o x^i$	origin of a local coordinate system in earth centered earth fixed coordinates
${}_o x$	origin of the global coordinate system in earth centered earth fixed coordinates
${}_o r^i$	radial vector from earth's center to the origin of a local coordinate system
w_k^i	local process noise vector
Φ_k^i	local state transition matrix
y_k^i	local measurement vector
C_k^i	global observation sub-matrix
H_k^i	local observation matrix
v_k^i	local measurement noise vector
$x_k(+)$	measurement updated global state estimate
$x_k(-)$	time updated global state estimate
$x_k^i(+)$	measurement updated local state estimate
$x_k^i(-)$	time updated local state estimate
$P_k(+)$	measurement updated global estimate error covariance matrix

$P_k(-)$	time updated local estimate error covariance matrix
$P_k^i(+)$	measurement updated local estimate error covariance matrix
$P_k^i(-)$	time updated local estimate error covariance matrix
M_k^i	matrix of zeros and ones which partition the global states to the local systems
Q_k	covariance matrix for process noise vector
R_k^i	covariance matrix for local measurement noise vector
$P_0(-)$	initial global estimate error covariance matrix
N	number of time samples
M	number of local systems
M^i	number of local measurement variables
$z_w(k)$	"pseudomeasurement" vector
$R_w(k)$	inverse square root of Q_k
$R_k(+)$	measurement updated global square root information matrix
$R_k(-)$	time updated global square root information matrix
$z_k(+)$	measurement updated global square root information vector
$z_k(-)$	time updated global square root information vector
$R_k^i(+)$	measurement updated local square root information matrix
$R_k^i(-)$	time updated local square root information matrix
$z_k^i(+)$	measurement updated local square root information vector
$z_k^i(-)$	time updated local square root information vector
e_k^i	local "innovations" vector

$R_w^{*(i)}(k)$ $R_{wx}^{*(i)}(k)$ $z_w^{*(i)}(k)$	local smoothing coefficients
$R_w^{*}(k)$ $R_{wx}^{*}(k)$ $z_w^{*}(k)$	global smoothing coefficients
$z^{*}(k)$ $H^{*}(k)$	merge coefficients
t	continuous time variable
d^i	local coordinate system displacement vector
L	latitude of global coordinate system origin
δ	longitude of global coordinate system origin
α^i β^i γ^i	Euler angles defining the orientation of the local coordinate system w.r.t. to the global coordinate system
r_e	average earth radius
${}^1T_k^{\ i}$	orthogonal transformation in local time update
${}^2T_k^{\ i}$	orthogonal transformation in local measurement update
${}^3T_k$	orthogonal transformation in first merge step
${}^4T_k$	orthogonal transformation in second merge step
T^i	orthogonal coordinate transformation between the local and global coordinate system
T^g	orthogonal coordinate transformation between the global coordinate system and the earth centered earth fixed coordinate system
μ	product of G and re
$r_k^{\ i}$	range measurement
$\dot{r}_k^{\ i}$	range rate measurement
$\theta_k^{\ i}$	elevation angle measurement
$\Gamma_k^{\ i}$	azimuth angle measurement

G	gravitational constant
m_e	earth's mass
Ω	earth's angular velocity
$u(t)$	deterministic control vector
$f(x(t),u(t))$	nonlinear dynamics vector
$h^i(x(t))$	nonlinear local observation vector
F_k	linearized dynamics matrix

Summary

This final report describes results obtained over the entire 24 months of the Phase II project. First, the basic Decentralized Square Root Information Filter (DSRIF) theory as described by MTI personnel in a 1985 IEEE Conference Paper was extended in 2 ways. An expression for the likelihood function in terms of DSRIF variables, and a method for distributing the prior and process noise statistics over the set of locally optimal filters were derived.

Next, in support of the Army's efforts to test and evaluate the SADARM submunition, software developed in Phase I research was upgraded to enable multitarget tracking within our distributed filtering environment. Outlier detection/rejection, track initiation, measurement-to-track and track-to-track association routines were encoded and successfully tested using real MLRS data provided by WSMR. A dual network topology in which standoff radar and optical instrumentation provide MLRS tracks to a short range network of video trackers, was formulated. Our thesis is that MLRS tracks will aid the short range network in acquisition and tracking of the submunitions. Also, MTI worked with another contractor to finalize the design of a multiprocessor board set for real-time distributed processing of test range data.

Finally, MTI conducted experimental work wherein 2 video trackers were built and networked with a global processor. Laboratory testing showed that the 2-camera network could track a variety of objects over the entire laboratory space. Moreover, each tracker was able to aid the other in acquiring a common target. Field testing of an individual tracker showed that it could acquire and track objects similar to the SADARM submunition, and against a cloudy background! We conclude that this low cost, dual network approach is highly feasible, and recommend that a follow-on Phase III effort be approved.

1. Introduction

Over the last several years, the U.S. Army White Sands Missile Range, Instrumentation Directorate has been involved in the test and evaluation of the Multiple Launch Rocket System (MLRS) Search and Destroy Armor (SADARM) submunition. Standoff radar and optical trackers, which are well suited for tracking high speed targets of considerable extent, at long range and over large distances, have been used to date but with limited success. Consequently, WSMR has proposed that a new short range instrumentation system, possibly in conjunction with the existing long range one, be used. We believe that internetworking both systems using inexpensive microcomputers is feasible, and this will lead to a unified tracking system with superior performance. Thus, the focus of our Phase II research was the development of this "dual network" approach to SADARM test range tracking.

The outer surface of the submunition is a small cylindrical shell whose diameter, length and mass are 175 mm, 180 to 205 mm, and 12 to 14 kgm respectively. The shell is attached to a small parachute which stabilizes the submunition's vertical motion to an almost constant descent velocity of 70 ft/sec (see figure 1). Downwards looking infrared and millimeter-wave sensors are mounted to the bottom for detection of armored vehicles, such as enemy tanks. Also, the entire submunition precesses at a 30 deg angle with respect to its vertical descent axis. Thus, the sensor's field of view continuously spirals inward as the submunition approaches ground zero. After detection and firing, the submunition will move along an almost straight path towards the target.

Up to 6 submunitions can be delivered to the target area aboard a specially designed rocket, and up to 6 rockets can be launched sequentially in time from a specially designed mobile launcher. Thus, up to 42 targets (36 submunitions and 6 rockets) could be airborne at the same time during the test. Figure 2 portrays a sequence of 9 events which span the interval of time from rocket launch, through ejection and deployment of the submunitions, and ending at target impact. The same sequence of events is followed by the remaining 5 rockets.

To date, WSMR has been successful in tracking the MLRS rockets with their standoff instrumentation however, the same instrumentation has only been able to track submunitions in a few instances. A short range tracking system composed of inexpensive video cameras and radar "hand guns" mounted to 2-axis gimballed platforms under computer control should be capable of tracking many more submunitions and with far greater accuracy. This is true especially when the standoff instrumentation is used to track each rocket and is in communication with the short range network. Our hypothesis is that the standoff network can detect the submunition expulsion event and that this will aid the short range network in target acquisition. Rocket tracks generated by the standoff network can be continuously predicted ahead to expulsion. Then, accurate dynamical models of the submunition during deployment can be numerically integrated to predicted target acquisition points for specific short range sensors. Perturbations of the predicted acquisition points,

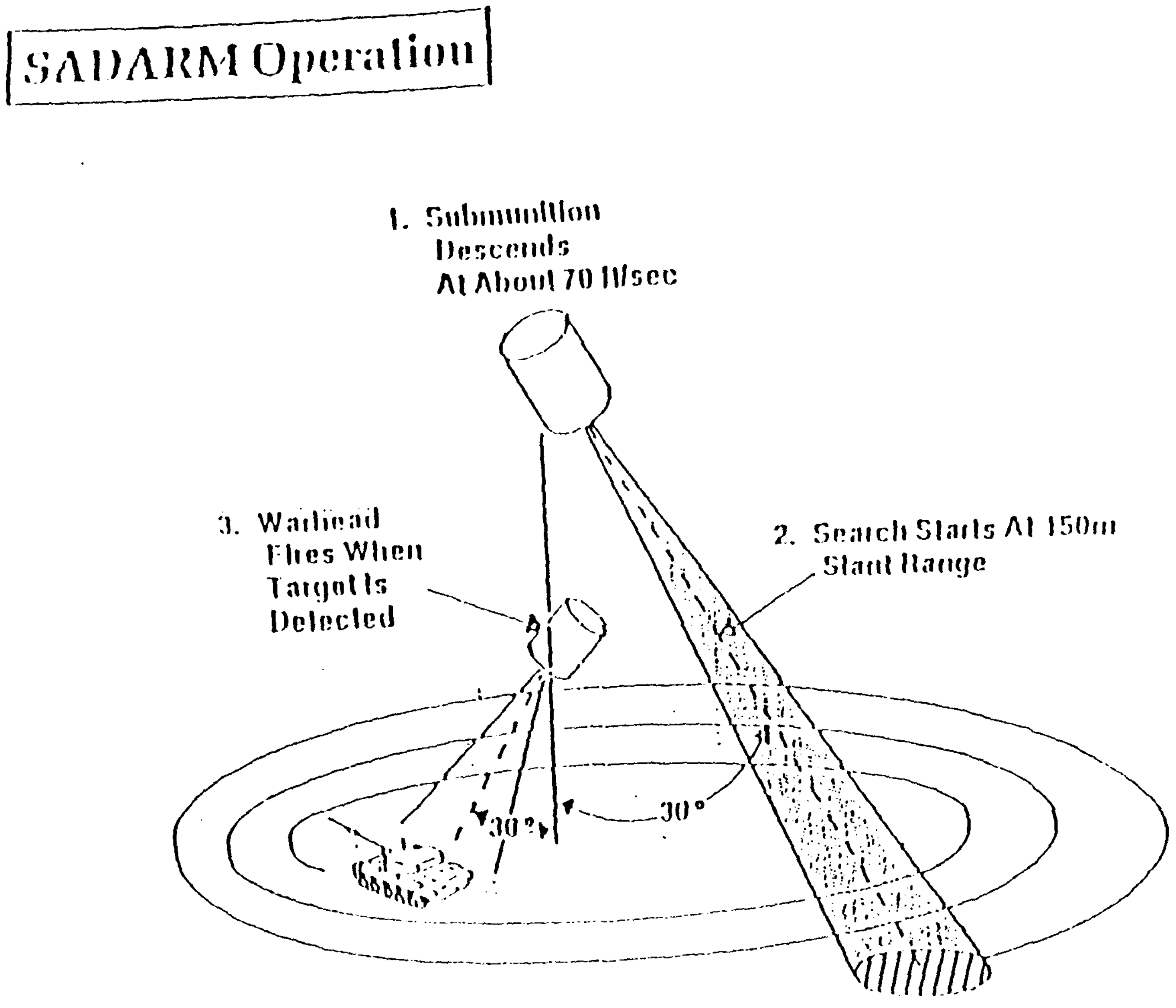
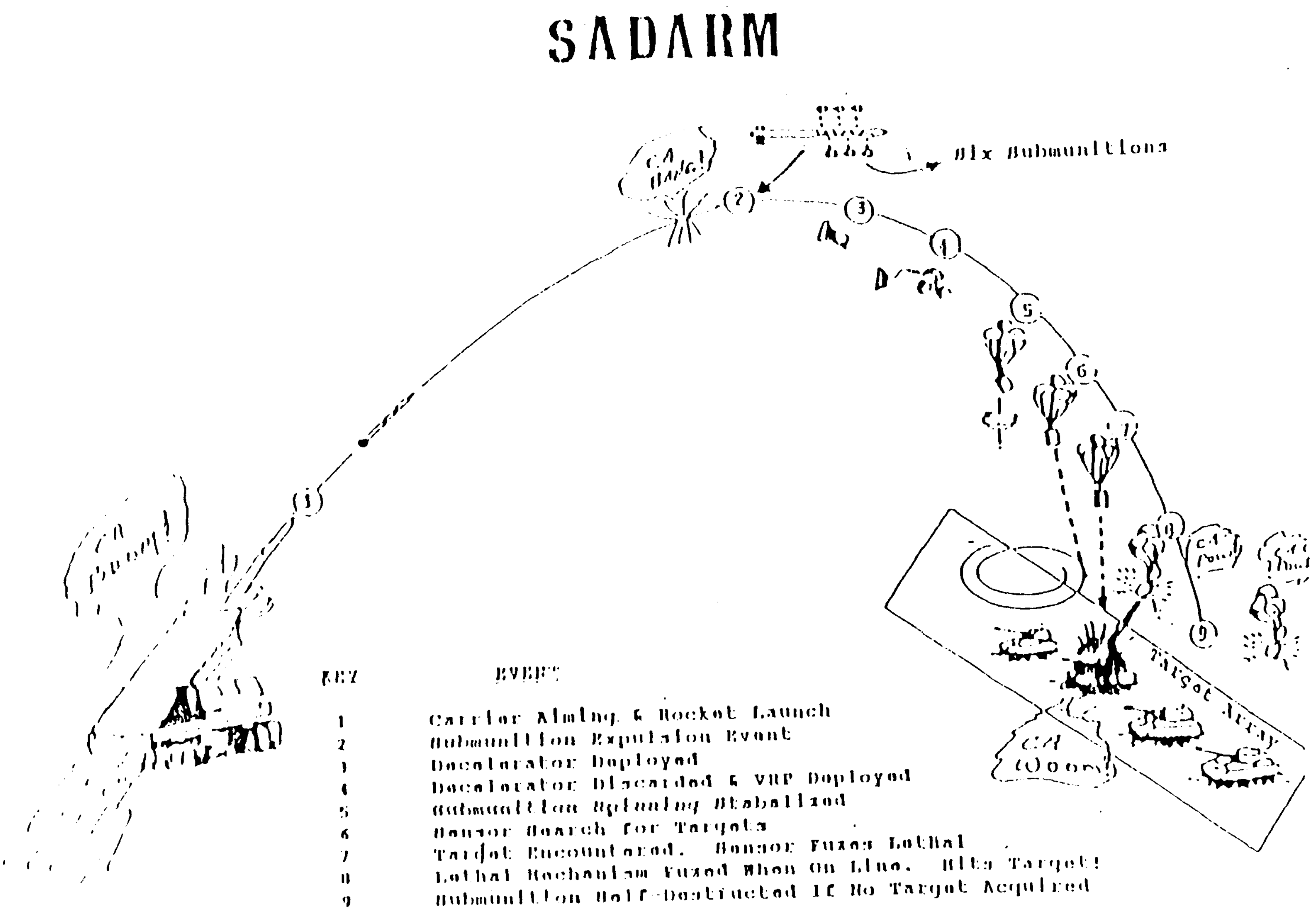

SADARM Operation
1. Submunition Descends At About 70 ft/sec
2. Search Starts At 150m Slant Range
3. Warhead Fires When Target Is Detected
30°
30°

Figure 1: SADARM operation (provided by WSMR).



KEY	EVENT
1	Carrier Aiming & Rocket Launch
2	Submunition Expulsion Event
3	Decelerator Deployed
4	Decelerator Discarded & VRP Deployed
5	Submunition Spinning Stabilized
6	Sensor Search for Targets
7	Target Encountered. Sensor Fuzes Lethal
8	Lethal Mechanism Fuzed When On Line. Hits Target!
9	Submunition Self-Destructed If No Target Acquired

Figure 2: SADARM (provided by WSMR).

caused by wind gusting and other small effects, will determine the minimum field of view and any small search pattern for each sensor, if necessary. After acquisition, the short range sensors will follow the submunition until ignition or impact with the ground.

This report is organized in the following way. In the remainder of this section, project background including a description of the network architecture is provided. Section 2 provides all results of the Phase II research. First, in section 2.1, extensions of our new theory for distributed filtering are given. In Phase I we derived an extended form of our Decentralized Square Root Information Filter (DSRIF) and used it to process real MLRS data provided by WSMR. Phase II extends the theory to allow for evaluation of the likelihood function in terms of DSRIF variables. The association of measurements from multiple sensors with tracks from multiple targets is made using the likelihood function as a metric. That is, correct associations are hypothesized when likelihood function values exceed a predetermined threshold. Thus, we have developed a new theory for data association within a distributed filtering environment. Also, a method for distributing the a priori information over the set of local processors (LPs) is given. This allows each tracker to generate locally optimal tracks so that good tracking performance is maintained even when its link with a global processor (GP) is broken.

Next, in section 2.2, a simulation of almost the entire dual network is presented. The simulation is useful for predicting the actual performance of the combined system, as well as providing a basis for the real-time software in the actual system. Starting with our "CODE 2" package developed under Phase I research, the simulation incorporates the extended theory from section 2.1. The final package contains additional features such as outlier detection/rejection, measurement-to-measurement association via clustering, measurement-to-track association via hypothesis testing using the Method of Maximum Likelihood, and track-to-track association. Thus, the package is based upon traditional approaches to multitarget multisensor tracking, except that we are recasting these methods in terms of our DSRIF.

In section 2.3 we discuss our interaction with another company in attempting to design a specialized processor which will speed up the calculations inherent to the problem at hand. The processor is being built, and we hope to incorporate it into our prototype network under future funding.

Finally, in section 2.4 we describe our experimental work wherein 2 video trackers (without the DSRIF) were built and networked with a global processor. Laboratory testing showed that the 2-camera network could track a variety of objects over the entire laboratory space. Moreover, each tracker was able to aid the other in acquiring a common target. Field testing of an individual tracker showed that it could acquire and track objects similar to the SADARM submunition, against a cloudy background! In section 3.0 we conclude that this low cost, dual network approach is highly feasible, and recommend that a follow-on Phase III effort be approved. Phase III should include incorporation of the DSRIF into the experiment, and upgrade of the tracker hardware.

1.0 Background and Previous Results

From an information perspective, it is reasonable to expect an improvement in tracking performance to the extent that a priori knowledge is included in the track estimation process. A priori knowledge about the target dynamics (including its initial condition) and the amount of uncertainty associated with this knowledge, along with a priori knowledge about the functioning of the sensor and the errors associated with each measurement can be used to predict the range, azimuth and elevation of the target into the future, and fedback to the sensor for much improved tracking performance.

In 1960, R. Kalman first derived a solution to this constrained optimization problem in which "least squares" estimates (and estimate error uncertainties) of target position, velocity and acceleration, subject to this a priori information, may be computed recursively. The solution was appropriately named the (conventional) "Kalman filter", and since then, a multitude of papers have been written on this topic. Most notable are its factorized versions which, unlike Kalman's original formulation, are guaranteed to be numerically stable (positive semidefinite estimate error covariances are guaranteed). One such version is the Square Root Information Filter (SRIF) [1], and more recently its "decentralized" form [2]. Thus, the Decentralized Square Root Information Filter is a computationally distributed form of the conventional Kalman filter but is far superior in many ways.

To summarize its operation, for a particular target, each measurement (such as range, range-rate ...) or group of measurement variables from each tracking sensor may be processed by a local SRIF, which generates a set of smoothing coefficients as well as a square root information matrix and information vector. A centralized merge processor consists of three separate processors which operate in parallel. The first combines the local smoothing coefficients with the effects of process noise and prior information about the initial state. The second merges the local square root information matrices and vectors with output from the first, but only upon demand by the third. The third produces estimates and covariances whenever desired by back-solving an upper triangular system of equations. An important observation is that feedback of information from the merge processor to the local filters is not necessary here. This helps to keep the bandwidth of communication between the local processors and merge processor from exceeding hardware limitations.

Figure 3 is a block diagram of the algorithm. Notice that the local SRIFs can be configured free of prior and/or process noise. This enables fast sequential or parallel testing of different prior and process noise hypotheses in the merge processor, without having to refilter any of the measurement variables over the short time interval in question.

When our extended form of the DSRIF (the E-DSRIF) is being used, local processors must upload their set of smoothing coefficients after each time update step but may upload their square root information matrix and vector after a particular measurement update step upon demand. The central merge processor in turn downloads the time

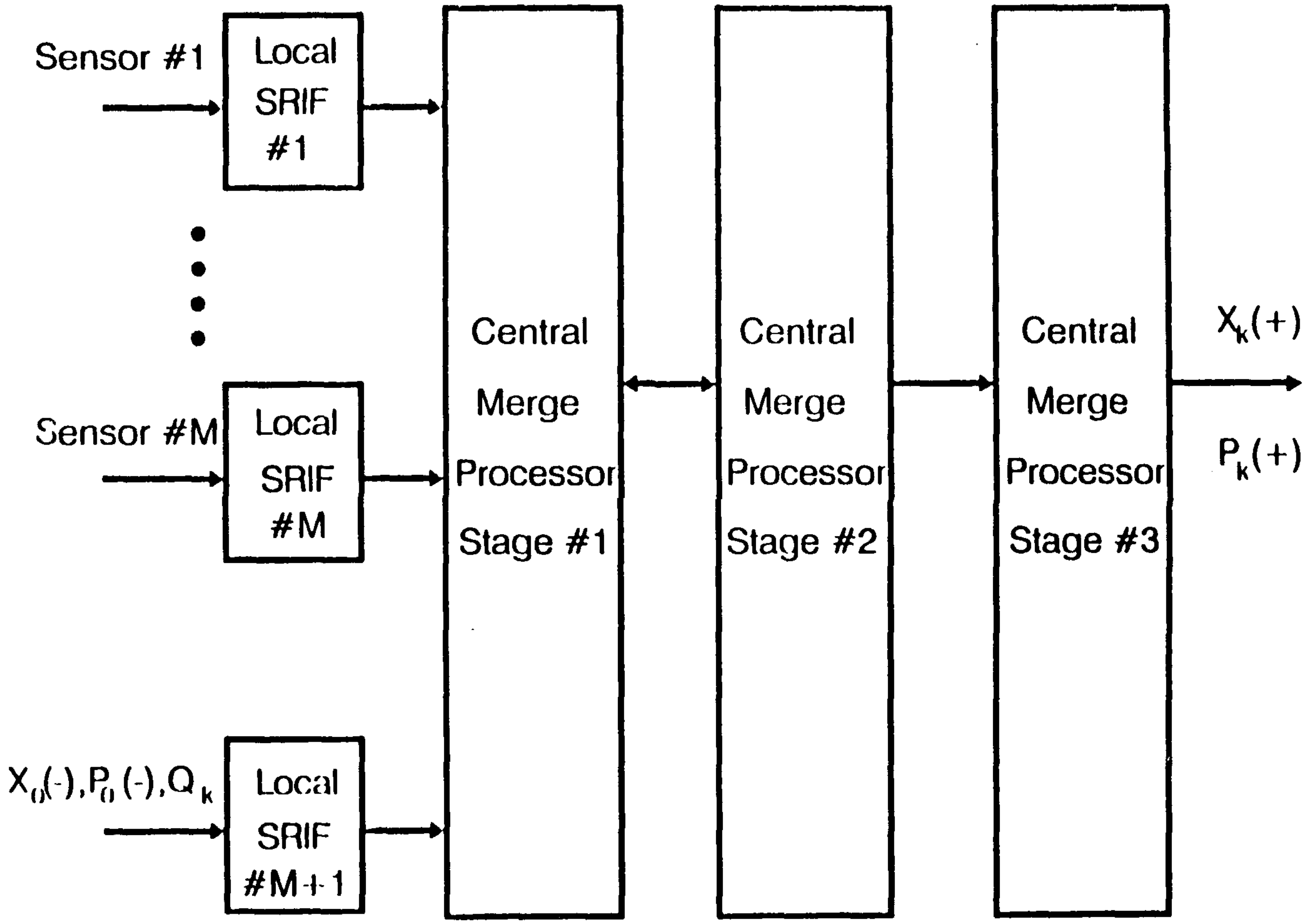

Sensor #1
Local SRIF #1
Sensor #M
Local SRIF #M
$X_0(-), P_0(-), Q_k$
Local SRIF #M+1
Central Merge Processor Stage #1
Central Merge Processor Stage #2
Central Merge Processor Stage #3
$X_k(+)$
$P_k(+)$

Figure 3: Computational architecture of the DSRIF.

updated global state estimate which is needed by the local filter for the relinearization process. The Extended form uses the actual measurement vectors (and not perturbed measurements) to estimate the full (and not perturbed) state vector.

An adaptive form of the E-DSRIF wherein the process noise levels were chosen to be a function of the globally optimal estimate error covariance, was successfully used to track the same set of real data under contract with the Defense Advanced Research Projects Agency (DARPA). This simple method for tuning the filter by using a feedback loop to compute the process noise covariance level, gave exceedingly good results as was shown in figures J through L of our previous report to DARPA [3] (where all measurement errors were chosen to be the nominal values used in WSMR Phase I [14]).

Since 1960, the Kalman filter in discrete form has been programmed and successfully used in numerous off-line situations; for example, Applied Physics Lab/Johns Hopkins uses (a factorized form of) the Kalman filter for post-flight processing of Trident II missile test flight data. On the other hand, real-time applications of the Kalman filter are virtually non-existent because of the relatively large amount of computations that are required for most physical systems. With the arrival of very high speed computing, real-time Kalman filtering is becoming a reality. One iteration of the filter using simple kinematical motion models and geometric measurement models typically requires a few thousand floating point operations. Assuming 10,000 flops as an upper bound, this suggests that data rates on the order of 100 per second are now achievable using compact megaflop processing!

2. Work Carried Out/Results Obtained

2.0 Dual Network Architecture

Figure 3 suggests that a natural network topology for implementing the DSRIF equations is a multi-star configuration. In this structure, each network element (defined as a local processor plus its platform and suite of attached sensors) is connected in parallel to a centrally located global processor. Figure 4 shows the fine structure of this basic star or "cluster", which we formally define to be a group of elements assigned to the same tracking volume. Figure 5 shows the structure of the dual network. Each of the 2 primary networks is a group of clusters connected in parallel to a centrally located "network processor". In turn, both network processors can communicate with one another through a "dual network interface".

The 4 types of processors which reside within the dual network are task specific. Local processors perform local measurement and time updating in order to arrive at a locally optimal track for each target within its field of view. They also preprocess the raw data in order to extract target attributes such as attitude and centroid, and they are responsible for providing feedback commands to repoint their respective platforms. Global processors perform global measurement and time updating, as well as data association in order to arrive at a globally optimal track for each target within its tracking volume. They also assign elements to targets within their respective volumes. Network processors are responsible for passing tracks between clusters when targets move from one tracking volume to another. Also, the short range network processor provides submunition acquisition data to cluster elements, based upon MLRS tracks from the long range network processor. The network interface acts as a network hub which will probably reside within the Range Control Center. In principle, other networks could be brought on-line through this interface.

2.1 Extensions of the Phase I DSRIF Theory

2.1.1 The Likelihood Function in Terms of DSRIF Variables

In [1] we showed that

$$||e_k||^2 = || R_k(-)\ x_k(+) - z_k(-)\ ||^2 + ||\ H_k\ x_k(+) - y_k\ ||^2$$

On the other hand, the global measurement update from [2] may be written as

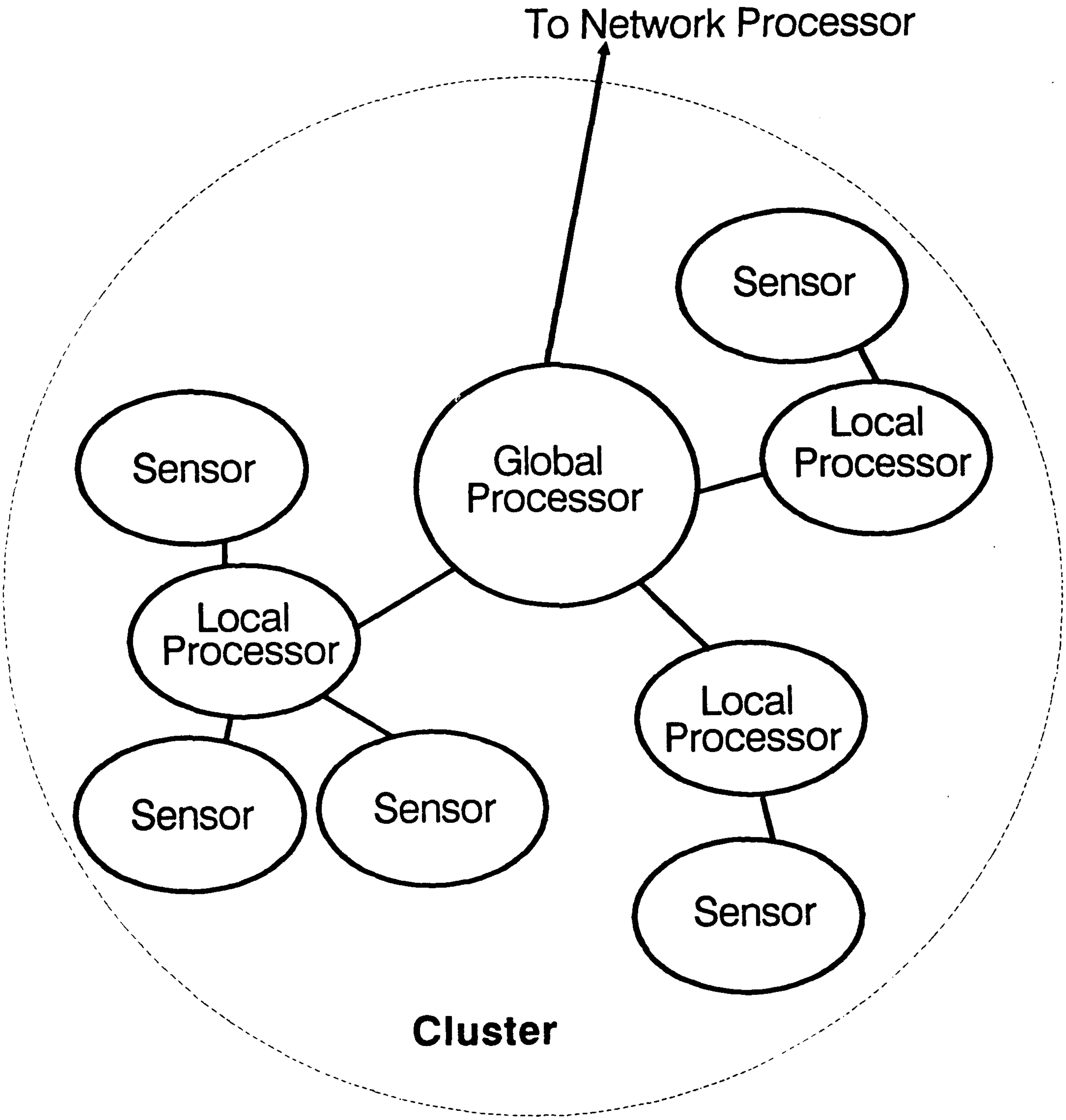


Figure 4: Fine structure of a network cluster.

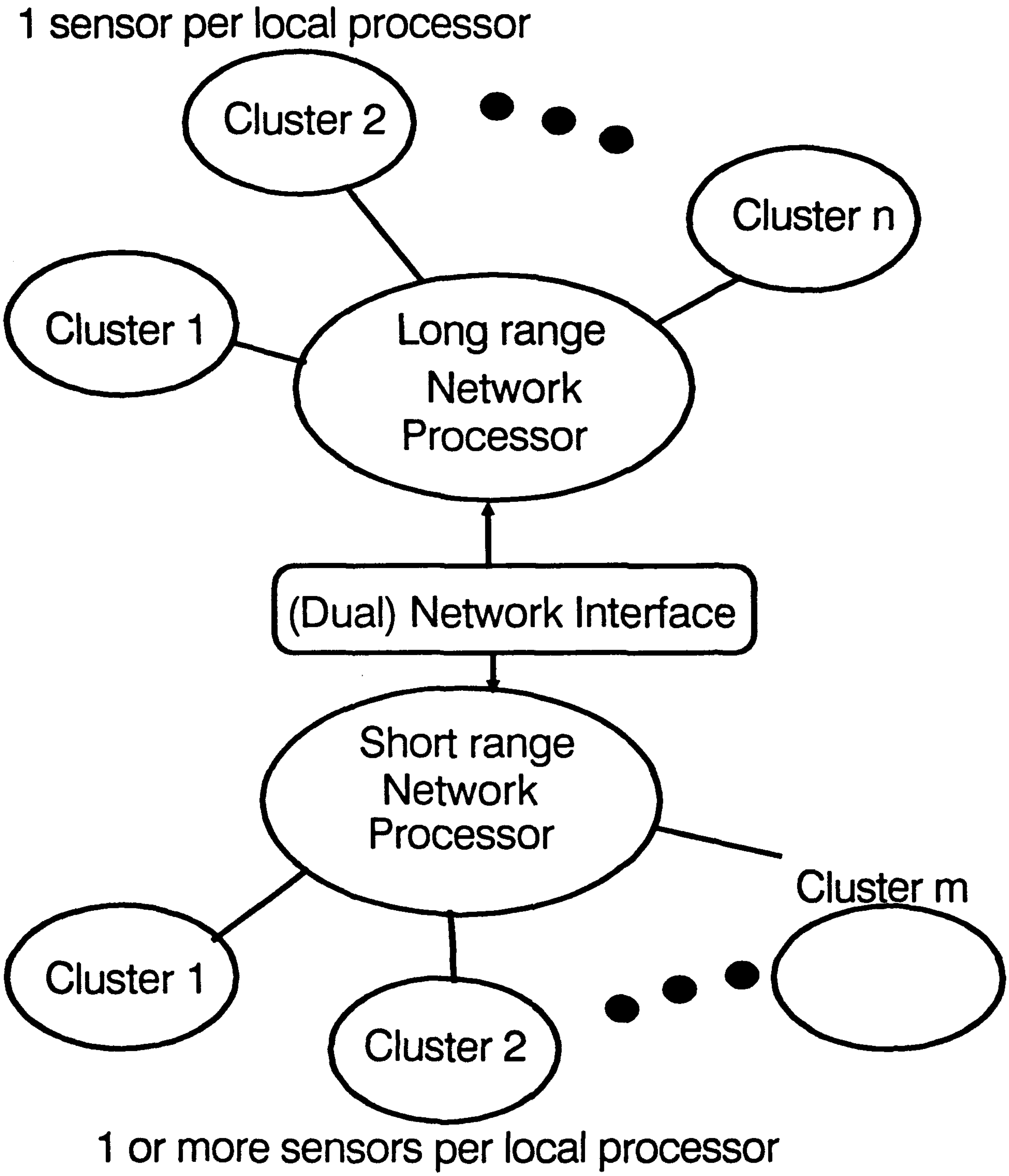

1 sensor per local processor
Cluster 2
Cluster n
Cluster 1
Long range Network Processor
(Dual) Network Interface
Short range Network Processor
Cluster m
Cluster 1
Cluster 2
1 or more sensors per local processor

Figure 5: Dual network topology.

$$
{}^{4}T_k \left[\begin{bmatrix} R_k(+)^{(1)} \\ \\ \\ R_k(+)^{(M)} \\ H^{*}(k-1) \end{bmatrix} x_k(+) \qquad \begin{matrix} z_k(+)^{(1)} \\ \\ \\ z_k(+)^{(M)} \\ z^{*}(k-1) \end{matrix} \right] \begin{bmatrix} 1 \\ \\ -1 \end{bmatrix} = \begin{bmatrix} R_k(+) & z_k(+) \\ \\ 0 & \# \end{bmatrix} \begin{bmatrix} 1 \\ \\ -1 \end{bmatrix}
$$

where

$$
H*(k-1) = R_k(-)
$$

$$
z*(k-1) = z_k(-)
$$

Taking the norm of both sides gives

$$
\sum_{j=1}^{M} || R_k(+)^{(j)} x_k(+) - z_k(+)^{(j)} ||^2 + || R_k(-) x_k(+) - z_k(-) ||^2
$$

$$
= || R_k(+) x_k(+) - z_k(+) ||^2 + || \# ||^2
$$

But

$$
x_k(+) = R_k^{-1} z_k(+)
$$

eliminating the first norm on the r.h.s. of the last equation. Also. from [2] we have that the first term on the l.h.s. of the above equation is just $|| H_k x_k(+) - y_k ||^2$. Thus, comparison of the first and last equations here, shows that

$$
|| \# ||^2 = ||e_k||^2
$$

The Likelihood function from [2] is

$$J = \sum_{k=1}^{N} \tfrac{1}{2}||e_k||^2 + \log \left(\frac{\det R_k(+)}{\det R_k \;\; \det R_k(-)} \right) \qquad (1)$$

where

$$R_k = \mathrm{diag}\{ \; [\; R_k^{(1)} \;], \; \ldots, \; [\; R_k^{(M)} \;] \; \}$$

and all other terms are computed by the Global Processor.

2.1.2 Distributing $P_0(-)$ and Q

The "Data Equations" for the prior and process noise random vectors are

$$z_w(k) = R_w(k) \; w_k + \epsilon_w(k)$$
$$z_0(-) = R_0(-) \; x_k + \epsilon_p(k)$$

where, by construction

$$\epsilon_w(k) \sim N(0,I)$$
$$\epsilon_p(k) \sim N(0,I)$$
$$w_k \sim N(0,R_w^{-1}(k)R_w^{-tr}(k))$$

In the case of process noise, it appears in the Least Squares Criterion as

$$|| \; T \; (\; z_w(k) - R_w(k) \; w_k \;) \; ||^2 = || \; T \; \epsilon_w(k) \; ||^2$$

noting that the statistics of $\{ \; T \; \epsilon_w(k) \; \}$ are the same as $\epsilon_w(k)$, and the orthogonal transformation T puts the data equation in upper triangular form. Also, the statistics of $\epsilon_w(k)$ are invariant w.r.t. the following decomposition

$$\epsilon_w(k) = \epsilon_w^{(1)}(k) + \epsilon_w^{(2)}(k) + \; . \; . \; . \; + \epsilon_w^{(nw)}(k)$$

where by design,

$$\epsilon_w^{(1)}(k) \perp \epsilon_w^{(2)}(k) \perp \ . \ . \ . \ \perp \epsilon_w^{(nw)}(k)$$

Thus

$$||\epsilon_w(k)||^2 = ||\epsilon_w^{(1)}(k)||^2 + ||\epsilon_w^{(2)}(k)||^2 + \ . \ . \ . \ + ||\epsilon_w^{(nw)}(k)||^2$$

and the process noise square root information matrix may be decomposed as

$$R_w(k) = R_w^{(1)}(k) + R_w^{(2)}(k) + \ . \ . \ . \ + R_w^{(nw)}(k)$$

In practice then, up to nw local filters can be driven with process noise, and the DSRIF does not require that all of the process noise be contained in 1 Local Processor (as originally shown in [2]). For example, when $nw = 3$, the matrix $R_w(k)$ may be decomposed as

$$\begin{matrix} X & X & X \\ 0 & X & X \\ 0 & 0 & X \end{matrix} = \begin{matrix} X & X & X \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{matrix} + \begin{matrix} 0 & 0 & 0 \\ 0 & X & X \\ 0 & 0 & 0 \end{matrix} + \begin{matrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & X \end{matrix}$$

and then $\epsilon_w(k)$ may be constructed according to

$$\begin{matrix} X \\ X \\ X \end{matrix} = \begin{matrix} X \\ 0 \\ 0 \end{matrix} + \begin{matrix} 0 \\ X \\ 0 \end{matrix} + \begin{matrix} 0 \\ 0 \\ X \end{matrix}$$

This corresponds to minimizing the norm of **each row** of the data equation for process noise separately, in each of the local processors. Other distributions are possible. The same set of necessary and sufficient conditions holds true for prior information on the initial estimate error.. Also, distributing the prior and process noise amongst local filters will improve the numerical conditioning of the local measurement update step and thus avoid the need for double precision arithmetic there.

2.2 Upgrade of the Phase I Software

On November 11, 1987 six rockets (without deployment of submunitions) were launched sequentially in time over a period of 2 hrs and 30 min at WSMR. Only 1 rocket was airborne at any one time and thus data association for multitarget tracking was not needed. In Phase I, MTI obtained a copy of this MLRS data which contained azimuth and elevation angle measurements (with respect to each local sensor) from 11 optical trackers (OT in figure 6) located at the range coordinates listed in table 3 of the Phase I report. The data set also contained range, azimuth and elevation angle measurements from 3 radars (R1,R2,R3 in figure 6) but with respect to the local coordinate system originating at the launcher (L in figure 6). Their locations and orientations are given in table 4 of the same document. The digitized measurements for all 6 shots were plotted in order to select the best shot (as defined by the least amount of data drop-out and outliers) for processing.

Figure 6 shows the configuration of the long range network as well as the outer boundary of the short range one, a 1.1 km by 1.1 km area. Reducing the plotting scale, figure 7 shows the outer boundary of the short range network plus the 6 rocket tracks (generated using the DSRIF in section 2.2.2) projected onto the X,Y plane.

The Global Coordinate System (GCS) is a right handed coordinate system with one axis pointing east along latitude 32.380 deg and another north along longitude 106.481 deg. The third axis is collinear with the radial vector which points outward from the earth's center and passes through the origin of the GCS.

The Local Coordinate System (LCS) is a right handed coordinate system with one axis pointing east along the local latitude, and another north along the local meridian. The third axis is collinear with the radial vector which points outward from the earth's center and passes through the origin of the LCS. Each sensor records measurements with respect to its own LCS but are expressed in terms of global states.

In order to evaluate the feasibility of this dual-network approach, as well as make progress in the development of the actual real-time software, much new software was developed and tested in Phase II. First, a perturbation study of the input parameters for shot #2 was performed. Then, MTI processed all of the data from each of the 14 sensors, and for all 6 rockets, 1 rocket at a time. To do this, the E-DSRIF software from Phase I was upgraded to detect and reject outliers and missing data. Next, the tracking software was upgraded again to handle multiple rockets being within the field of view of each sensor. A new data simulator which created measurement files corresponding to shots fired sequentially with a user determined delay, was written. Thus, the upgrade provided a new capability for data association and fusion. Finally, a data simulator for computing submunition (and MLRS) trajectories is presented. All of the simulation is encoded in Fortran '77 and executes on an IBM (clone) Model "386" desktop computer (640K ram, 60 Mbyte hard disk, Intel 80387 math coprocessor).

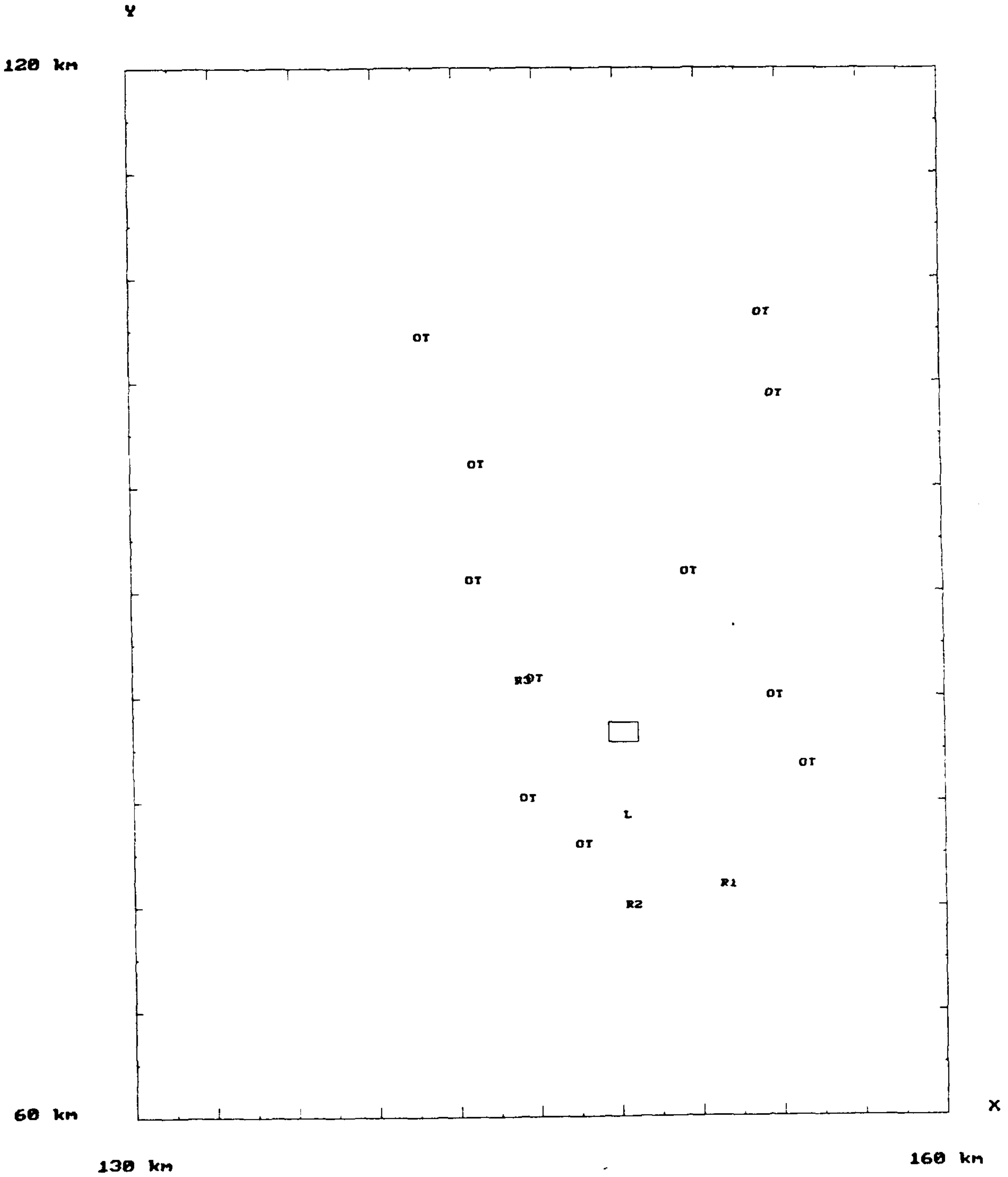


Figure 6: Location of the Long Range and Perimeter of the Short Range Sensors.

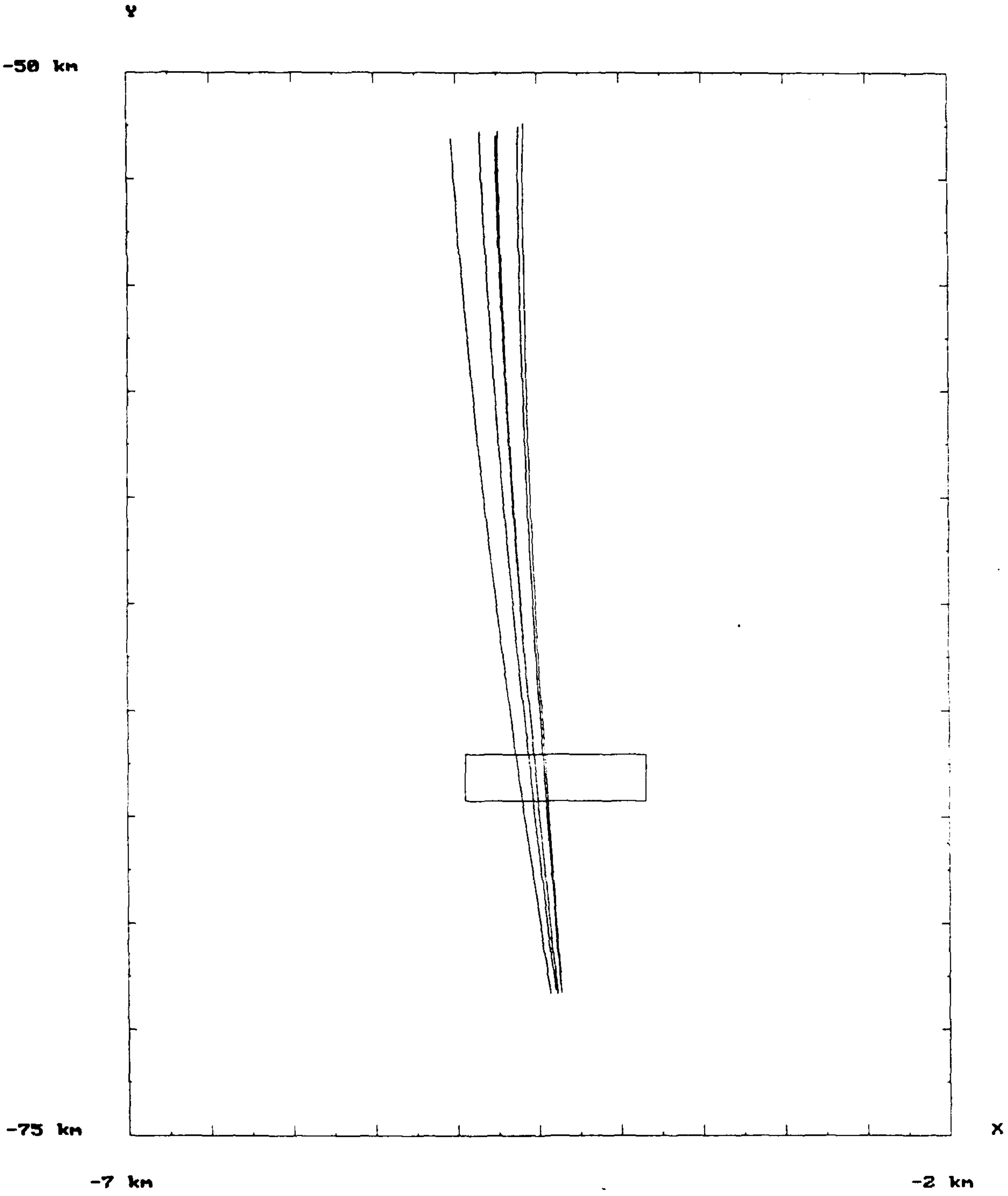


Figure 7: Perimeter and Location of the Short Range Sensors.

2.2.1 Tuning the MLRS Filter for Shot #2

For initialization of the DSRIF (or E-DSRIF), the input parameters, namely, the initial state vector and its estimate error covariance, process noise mean vector and its covariance matrix, measurement noise mean vector (usually assumed to be 0) and its covariance matrix, are necessary. Their values depend upon the suite of sensors as well as the target dynamics. They should be adjusted in order to achieve, at the very least numerical stability. In Phase I, MTI successfully used the E-DSRIF to track the second MLRS shot using measurement data from 5 sensors (radars 350,394, and optical trackers G30,G80,G110). The same simulation was repeated many times in order to obtain the following parameter values for all 6 shots (where "D" denotes a double precision number):

diag {0.01D+14, 0.01D+14, 0.01D+14, 0.05D+14, 0.05D+14, 0.05D+14, 0.01D+14, 0.01D+14, 0.01D+14} for an initial state estimate error covariance matrix,

diag {-10.0, -0.5, -0.48} for a process noise mean vector,

diag {0.17D+18, 0.74D+18, 0.93D+18} for a process noise covariance matrix,

diag {0.01D+12, 0.01D+12} for a measurement noise covariance matrix (optical tracker), and

diag {0.01D+3, 0.01D+2, 0.01D+2} for a measurement noise covariance matrix (range radar).

In each of the simulation studies, only one type of parameter was changed with all other types fixed to the above values. Changes in the x component of the global measurement updated state were noted. In general, the study shows that the E-DSRIF is robust with respect to wide variations in input parameter values.

Study 1: In this case, diagonal elements of the initial state estimate error covariance matrix were changed. Very little change in the "x component" was observed for increases of the diagonal elements up to 10^{28} (see figure 8), and decreases down to 10^{-06} (see figure 9). However when they are increased to 10^{29}, unstable estimates are obtained (see figure 10).

Study 2: In this case, diagonal elements of the process noise covariance matrix were changed. Very little change in the "x component" was observed for increases of the diagonal elements up to 10^{26}, (see figure 11), and decreases down to 10^{08} (see figure 12). However when they are increased to 10^{27}, unstable estimates are obtained (see figure 13).

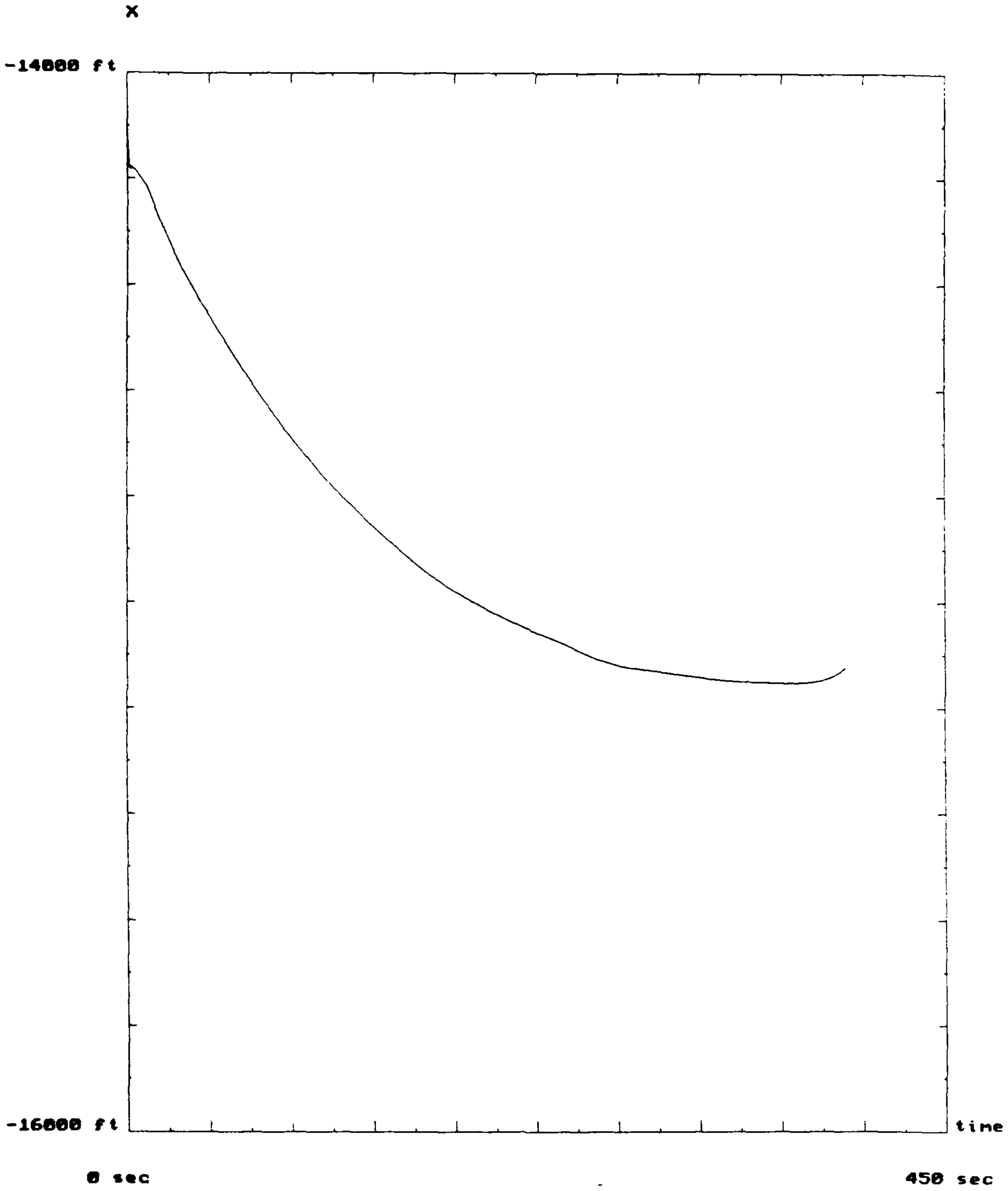


Figure 8: Stability of the DSRIF with respect to increases in the initial estimate error covariance.

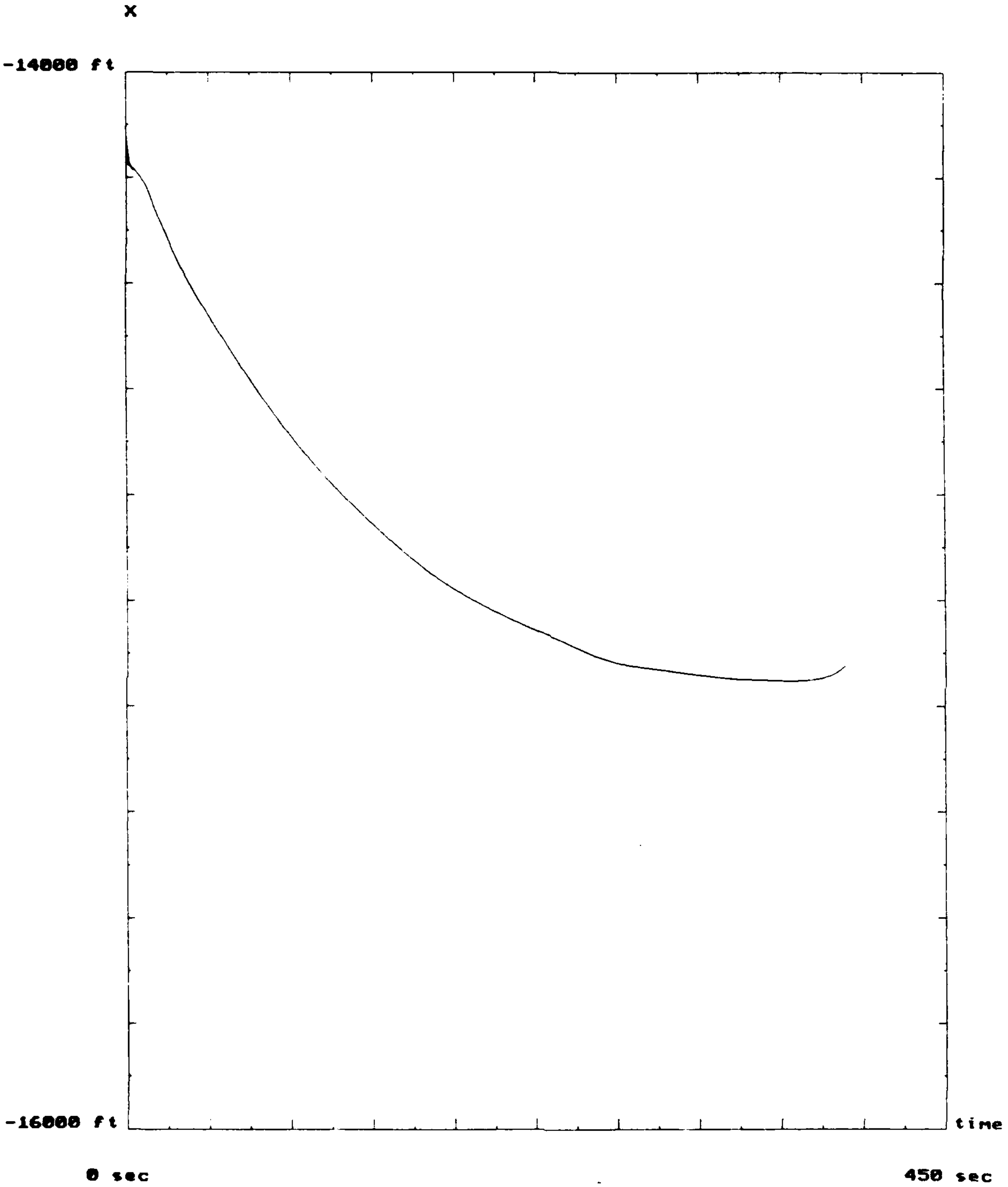


Figure 9: Stability of the DSRIF with respect to decreases in the initial estimate error covariance.

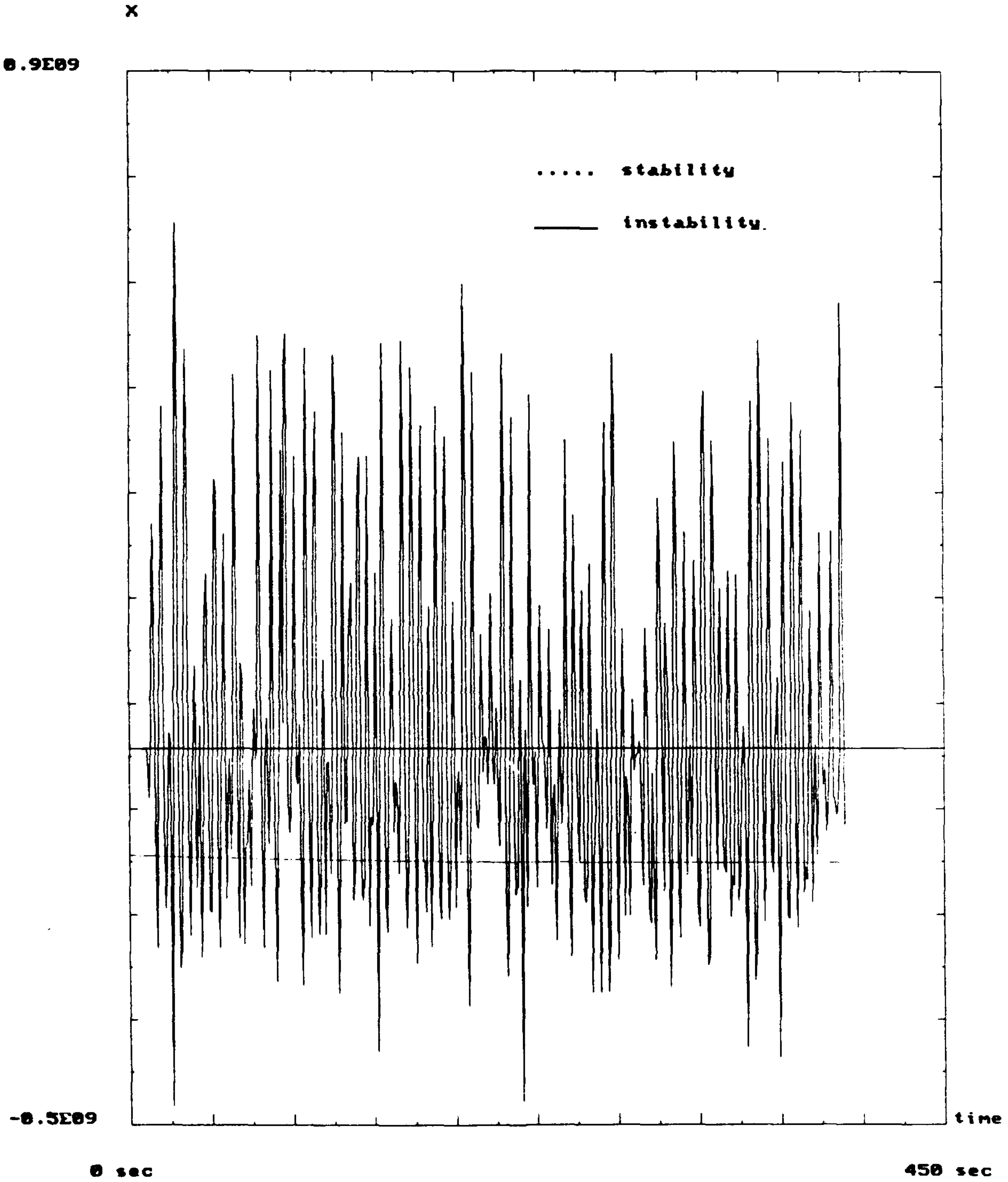


Figure 10: Instability of the DSRIF with respect to changes in the initial estimate error covariance.

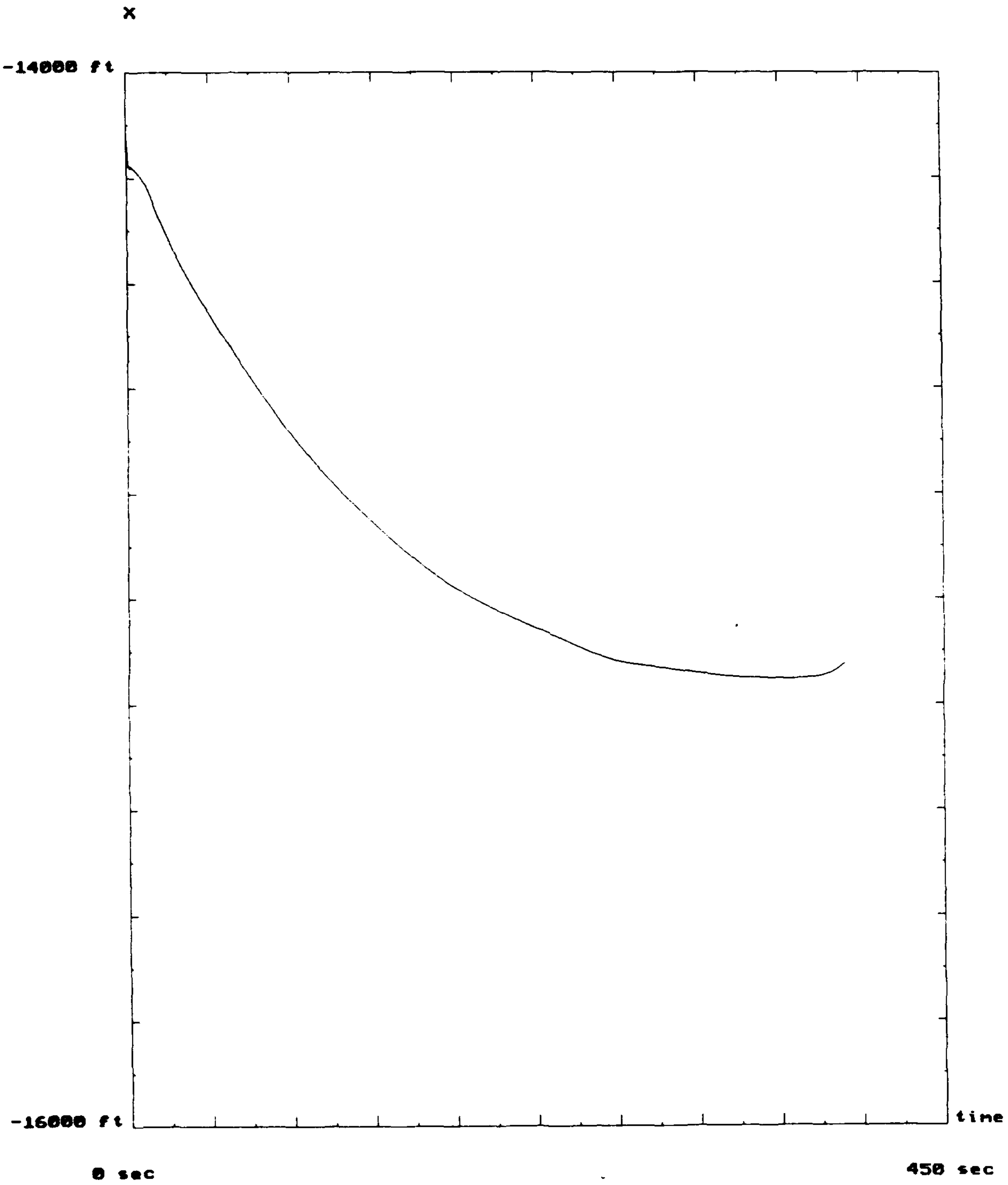


Figure 11: Stability of the DSRIF with respect to increase in the process noise covariance

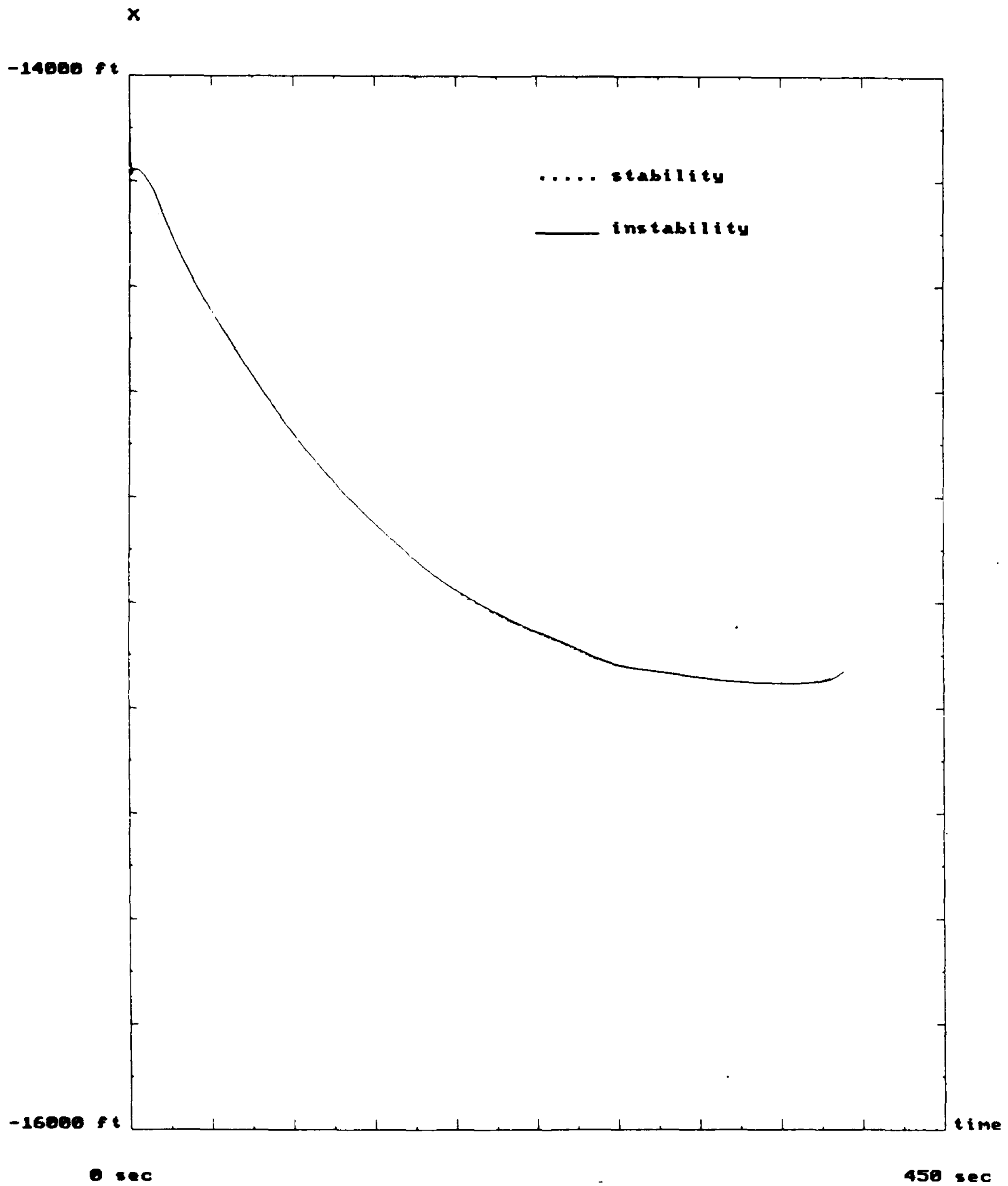


Figure 12: Stability of the DSRIF with respect to decrease in the process noise covariance.

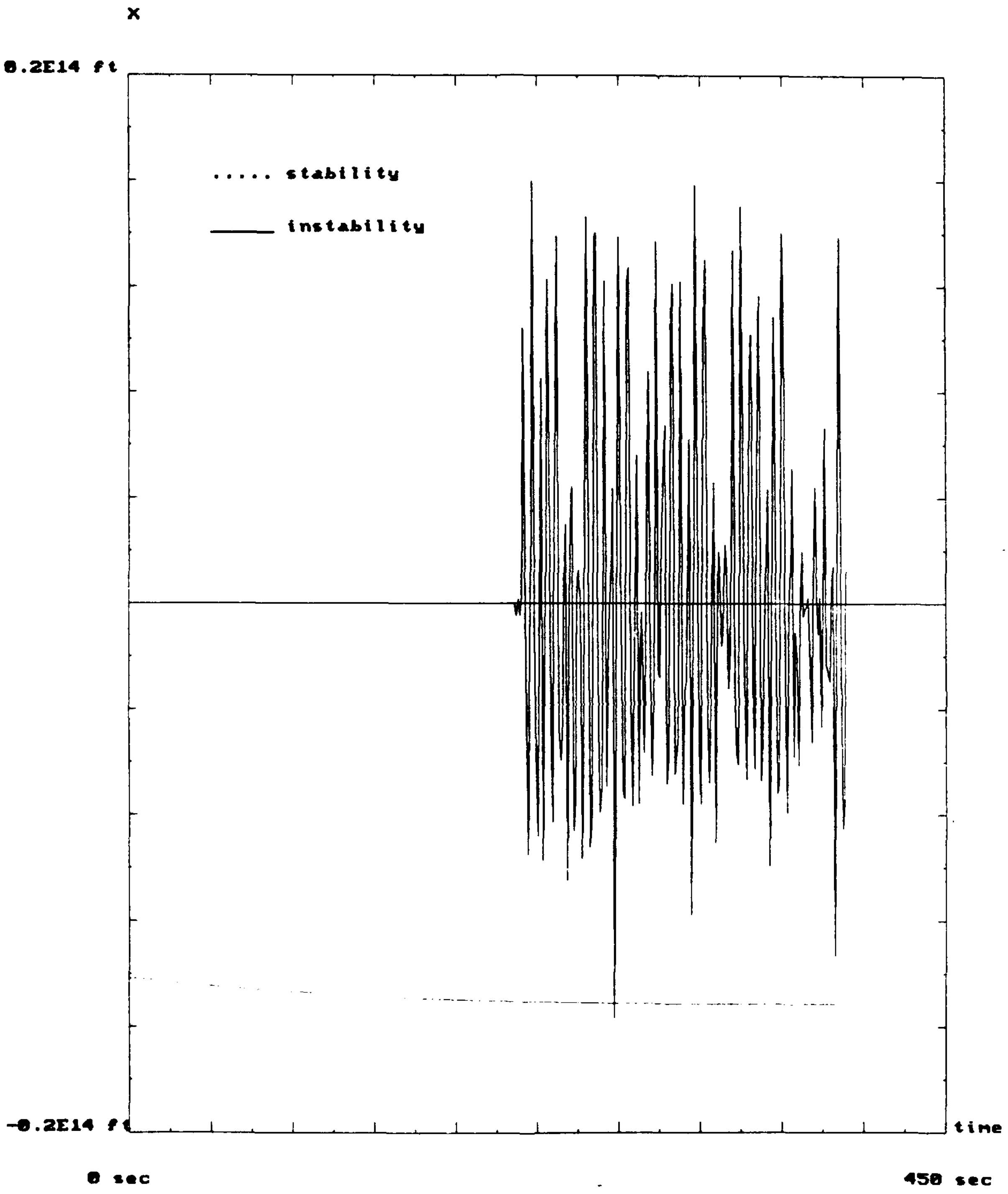


Figure 13: Instability of the DSRIF with respect to changes in the process noise covariance.

Study 3: In this case, diagonal elements of the measurement noise covariance matrices for range radars and optical trackers were changed, simultaneously. Very little change in the "x component" was observed for increases of the range radar diagonal elements up to 10^{22}, and optical tracker diagonal elements up to 10^{13} (see figure 14). However, when they are decreased to 10^{05} and 10, respectively, unstable estimates are obtained (see figure 15).

Study 4: In this case, the values of the initial state were changed by a multiplying factor. Stable estimates were observed for factors up to 1.008 (see figure 16). Multiplication by 1.009 leads to unstable estimates (see figure 17).

Automated tuning of the input parameters for the E-DSRIF is a possibility for future research. A maximum likelihood technique for estimating these parameters using the SRIF was recently developed [4]. The theory can be easily extended to incorporate the DSRIF (or E-DSRIF). The feasibility of this approach for retuning the input parameters in real-time will depend upon the amount of data required for convergence as well as the computational horsepower available.

2.2.2 Outlier Detection and Rejection

"Reasonable" values of the target's maximum speed and acceleration in each direction were chosen from the MLRS data set. At each iteration of the filter, a gate centered at the global measurement updated state, and with a radius equal to the maximum possible flight distance in each direction, is drawn. The maximum possible distances are given by

$$
\begin{aligned}
d_{max,x} &= v_{max,x}\ (\Delta T) + \tfrac{1}{2}\ a_{max,x}\ (\Delta T)^2 \\
d_{max,y} &= v_{max,y}\ (\Delta T) + \tfrac{1}{2}\ a_{max,y}\ (\Delta T)^2 \\
d_{max,z} &= v_{max,z}\ (\Delta T) + \tfrac{1}{2}\ a_{max,z}\ (\Delta T)^2
\end{aligned}
$$

where ΔT is the time update interval. If a measurement is outside of the gate as determined by the above condition, then it is regarded as an outlier and rejected from the filter. One cycle of detection and rejection was programmed as follows:

step 1: Let $x_{k-1}(+)$ be the global measurement updated state at time $k-1$. The GP broadcasts $x_{k-1}(+)$ (as well as $x_k(-)$) to all LPs within the cluster. Then, each LP sets up a gate centered at $h^i(x_{k-1}(+))$.

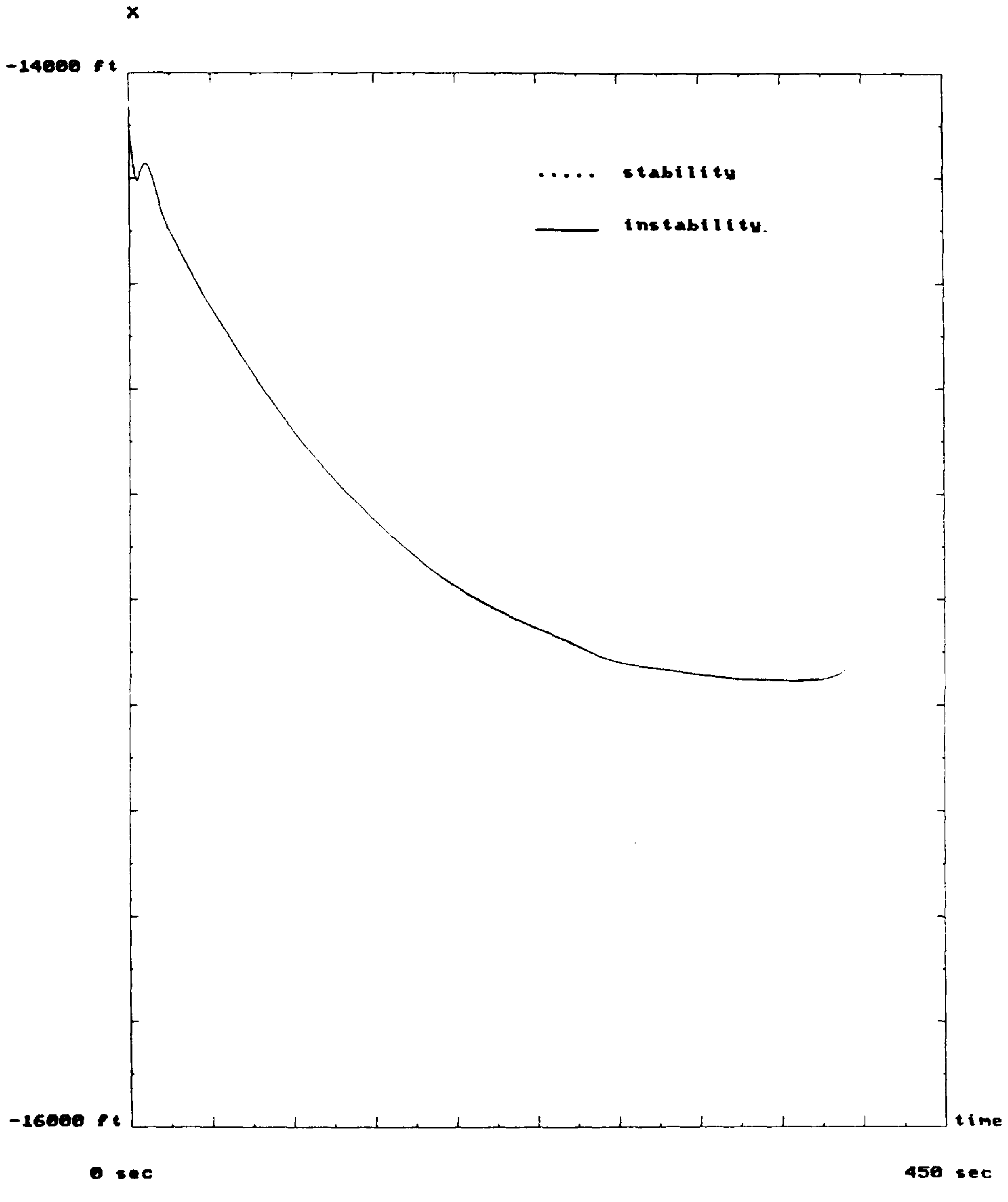


Figure 14: Stability of the DSRIF with respect to increases in the measurement noise covariance.

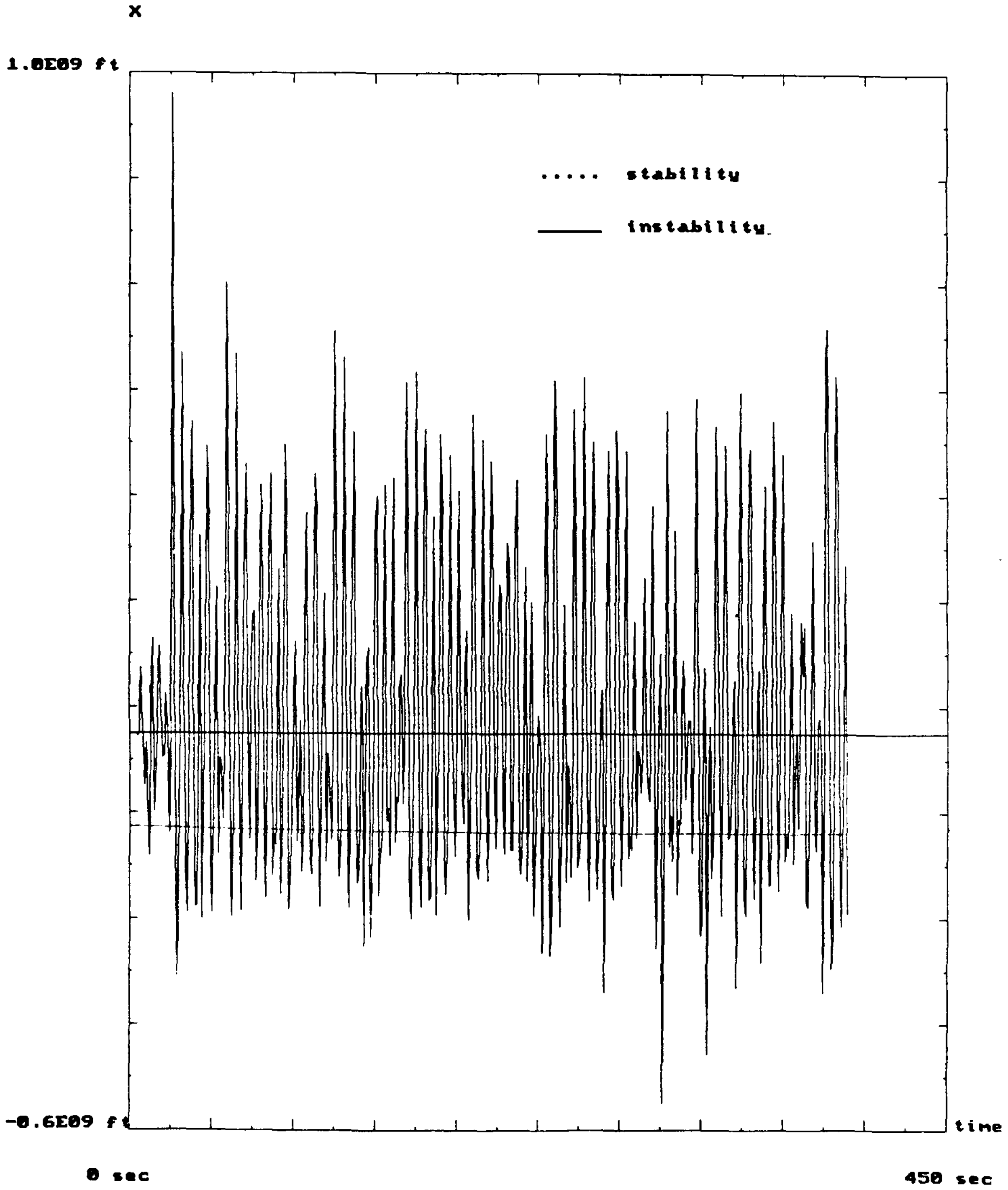


Figure 17: Instability of the DSRI^F with respect to changes in the initial state.

step 2: Outlier detection at each LP is performed by checking the gate condition on the k^{th} measurement. If the condition is satisfied, then standard local measurement updating proceeds as usual, starting from $x_k(-)$. If the condition is not satisfied, then the measurement is rejected by performing a local measurement update with the observation matrix and measurement vector set to zero.

step 3: In either case, each LP then sends its measurement updated square root information matrix and vector to the GP.

step 4: Global measurement updating is performed using the square root information matrices and vectors from all of the LPs. The new global measurement updated state $x_k(+)$, is obtained.

Figure 18 shows the X,Y components of the global measurement updated state for shot #1. In this figure, "13-sensor" and "14-sensor" represent the cases of "without using the G393 data", and "with using the G393 date" but with invoking the outlier detection procedure, respectively. Figures 19 and 20 contain the same information for the Y,Z and Z,X components.

This method is simple, but does not take into account the statistical nature of the process and measurement noise. Also, more adaptive methods of determining the radius of the gate should be considered in future work.

2.2.3 Data Association and Fusion

Associating measurements with measurements, measurements with tracks, and tracks with tracks are essential parts of multisensor multitarget tracking [5]. In a multisensor environment, measurements from all sensors within a cluster must be grouped in order to decide how many targets are present and which measurements originate from the same target. This is termed measurement-to-measurement association. Thereafter, each group of measurements may be compressed or fused to produce one representative measurement for the group. This process is called compression.

Compressed data from a grouping represents the position of a single provisional target at some instant in time. The tentative track of a provisional target can be generated by associating compressed data from one iteration to the next. The measurement-to-track association process, determines which current compressed measurements belong to which of the previous tracks.

If, on the other hand, each sensor is equipped with its own LP, and produces its own tracks, it is necessary to decide whether two tracks from different sensors represent the

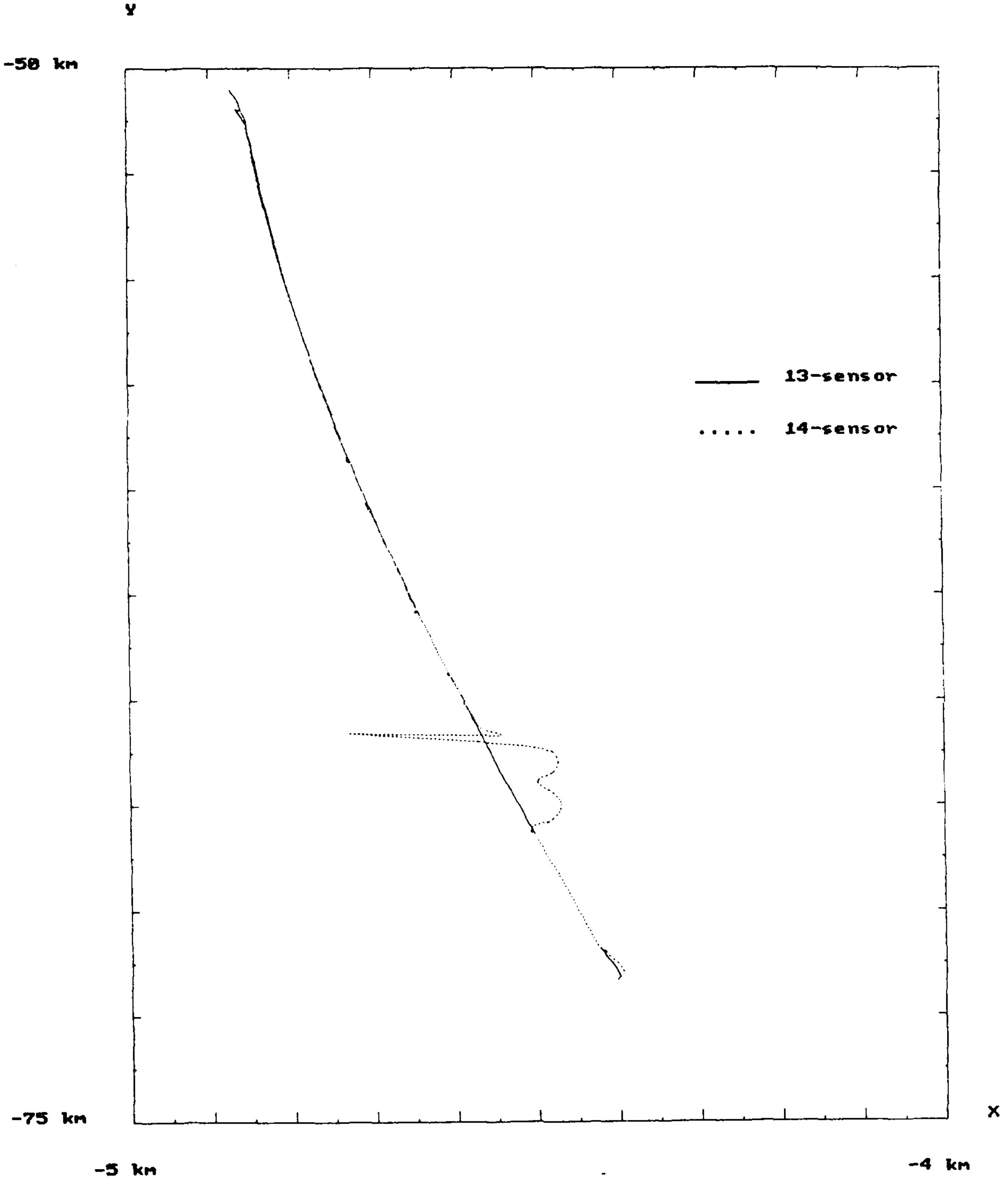


Figure 18: X,Y components of global measurement updated states with and without outlier detection.

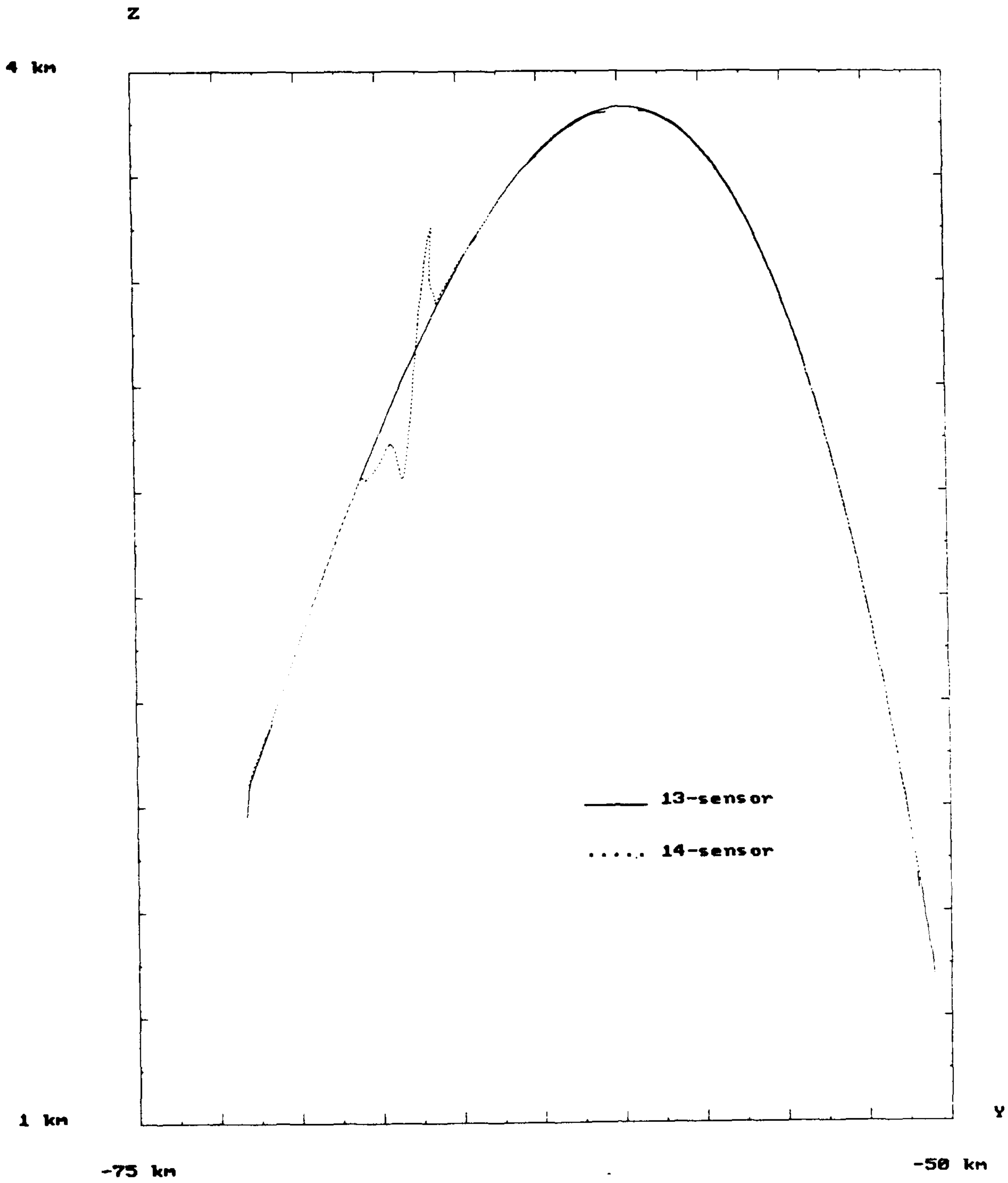


Figure 19: Y,Z components of global measurement updated states with and without outlier detection.

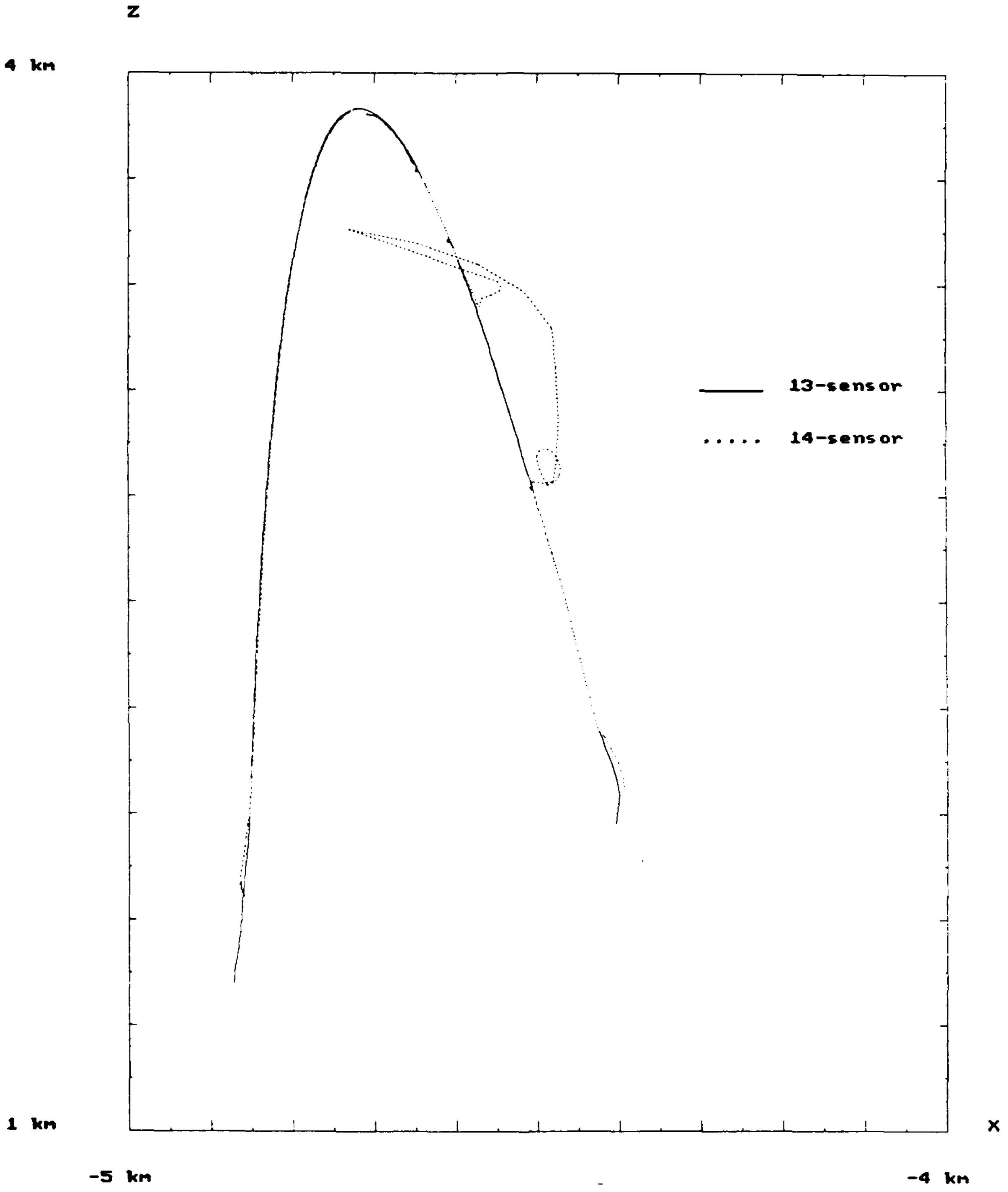


Figure 20: Z,X components of global measurement updated states with and without outlier detection.

same target at the GP. Hence, grouping of targets, or track-to-track association, is required. Once tracks which represent the same target have been grouped together, the next problem is how to combine the corresponding state estimates to get a globally optimal one. This process is called global fusion (or merging).

In this report, we are mainly interested in measurement-to-measurement association, compression, and measurement-to-track association. Track-to-track association and global fusion (merging) are solved in the application of the E-DSRIF (or DSRIF) method to the problem, since the measurement-to-track association and global merging processes (i.e., global time updating and measurement updating processes) generate globally optimal estimates.

For each type of association, whether measurement-to-measurement, measurement-to-track, or track-to-track, one may adopt either a hard or soft decision approach. The hard decision approach makes associations after each cycle of the filter. It makes efficient use of computational power and memory, but precludes rectification of errors in association. In the soft decision approach, association can be postponed until additional information has been accumulated to corroborate proper association. Hence, the risk of misassociation is reduced, but the approach demands considerably more computational power and memory (and is, therefore, less suitable for real-time applications).

Our approach is a hybrid of the hard and soft decision approaches. Measurement-to-measurement association is considered related with track initiation and addition of new tracks. *In initiating tracks and detecting new targets*, a hypothesis tree is generated, analogous to the soft decision approach. On the other hand, a hard decision approach is adopted for global measurement-to-track association. The measurement-to-track association problem is divided into local and global processes to utilize the distributed data association concept. Our main tools for association are association matrices and the likelihood function. Association matrices are very similar to the assignment matrices in [6]. A very simple rule is applied to resolve conflicts in the association matrix. When conflicts are not fully resolved, the likelihood function or other methods are employed to make a final decision.

2.2.3.1 Measurement-to-Measurement Association

Without any attributes except kinematic data, the grouping method is mainly based on gating the distance between any two measurements. The distance metric may be determined by a simple calculation of the spatial distance measure, for example, the conventional Euclidean norm. Alternatively, a more complex method may be chosen which takes the statistical nature of measurement noise into account. For example, a statistical distance measure defined by

$$d^2 = (y_1 - y_2)^{tr} (R_1 + R_2)^{-1} (y_1 - y_2)$$

also can be used. Here R_1 and R_2 are measurement covariance matrices of sensor 1 and sensor 2, respectively, and y_1 is from sensor 1 and y_2 from sensor 2. Once a distance metric is chosen, the basic idea of grouping is that measurements which are closer than a specified threshold are regarded as originating from the same target.

For each measurement vector y_1 in one data set, let $d(y_1,y_2)$ denote the distance between y_1 and y_2 in another data set. Then, for each y_1, the y from the other data set, which is chosen in such a way that

$$\arg\min \{ d(y_1,y) < thd \}$$

is assumed to be a measurement from the same target. Here, thd is a threshold whose initial value can be determined by statistical requirement analysis (for example, a Chi-square test), or by a tuning approach using repeated simulation.

2.2.3.2 Compression

We have chosen to use the following well known compression method [7]. Let y_1, y_2, ... , y_n be the measurements in one group, and each y_i is from sensor i whose measurement noise covariance is given by R_1, R_2, ... , R_n. Then a composite measurement covariance R is defined by

$$R^{-1} = \sum_{i=1}^{n} R_i^{-1} \qquad (2)$$

and the compressed measurement y is

$$y = R \left[\sum_{i=1}^{n} R_i^{-1} z_i \right] \qquad (3)$$

2.2.3.3 Measurement-to-Track Association

In our program, the measurement-to-track association process is part of the local filtering process, more specifically, the local measurement updating process. Each LP is informed as to which new measurements are correlated with existing tracks in the track file. The local measurement-to-track association process is comprised of two subprocesses in our system. The first one is based on the global predicted state estimate, and the second is based on the global filtered state estimate. This is described in greater detail in section 2.2.4.

2.2.3.4 Track Initiation and New Track Addition

The problems of how to initiate tracks and when to add new tracks can be solved by using a distance metric procedure similar to the grouping process. We assume that all sensors are time synchronized. The track initiation method followed is a modification of "A Logic-Based Multitarget Track Initiator" described by Bar-Shalom and Fortmann [5].

Let $z_i^l(k)$ be the l^{th} component of measurement i at time k, where $l = 1, \ldots, n_z$ and $i = 1, \ldots, m$. Then, for example, for $k = 1,2$ the distance vector between measurements $z_i(1)$ and $z_i(2)$ is defined as having components

$$d_{ij}^l = \max[z_j^l(2) - z_i^l(1) - v_{max}^l \blacktriangle, 0] + \max[z_i^l(1) - z_j^l(2) - v_{min}^l \blacktriangle, 0]$$

where $\blacktriangle$ is the time interval between scans. The above expression consists of the observed position displacement beyond the maximum (minimum) possible distance traveled, i.e., due to the noise. Then, assuming the measurement errors to be independent, normal, and zero-mean with covariances $R_i(k)$, the normalized distance squared

$$D_{ij} = d_{ij}^{tr} [R_i(1) + R_j(2)]^{-1} d_{ij}$$

will be the test statistic. The test for associating $z_i(1)$ with $z_j(2)$ is

$$D_{ij} \leq \tau$$

where τ is a threshold obtained from the Chi-square tables with n_z degrees of freedom such that

$$P[X_{nz} \leq \tau] = 1 - \alpha$$

and α is the probability of miss.

In this approach, a distance metric criterion, or gate, is used to associate measurements from the previous scan with each candidate from the current scan. This method generates time sequences of associated measurements, which represent the tentative tracks of a set of provisional targets. The acceptance region of the gate accounts for measurement noise variance and motion of the target, characterized by a maximum and minimum velocity: v_{max}^{l}, v_{min}^{l} respectively, for coordinate l.

2.2.3.5 Software Configuration

The new "Code 3" consists of 2 software subsystems, 5 modules, and 28 subroutines in addition to the Estimation Subroutine Library developed by G. Bierman. The 2 subsystems are the Local Processing System (resides within each LP) and the Global Processing System (GP resident). The local data association module and the local filtering module are part of the Local Processing System. The global data association module, global filtering module, and the track file management module are part of the Global Processing System. Figure 21 shows this breakdown. Also, each module consists of several processes.

Local Processing System

A) Local Data Association Module

1) Coordinate Transformation Process

This process is performed only for the range radar measurements. Range, azimuth, and elevation measurements are transformed from the launch site coordinate system to the GCS.

- Subroutine called

COORTR

2) Local Measurement ID Setting Process

This process is performed for all range radars and optical trackers. Based on the list of global measurement identification numbers (IDs) which is broadcast by the GP, each LP adjusts its local measurement IDs accordingly.

- Subroutines called

GTOL, SETID

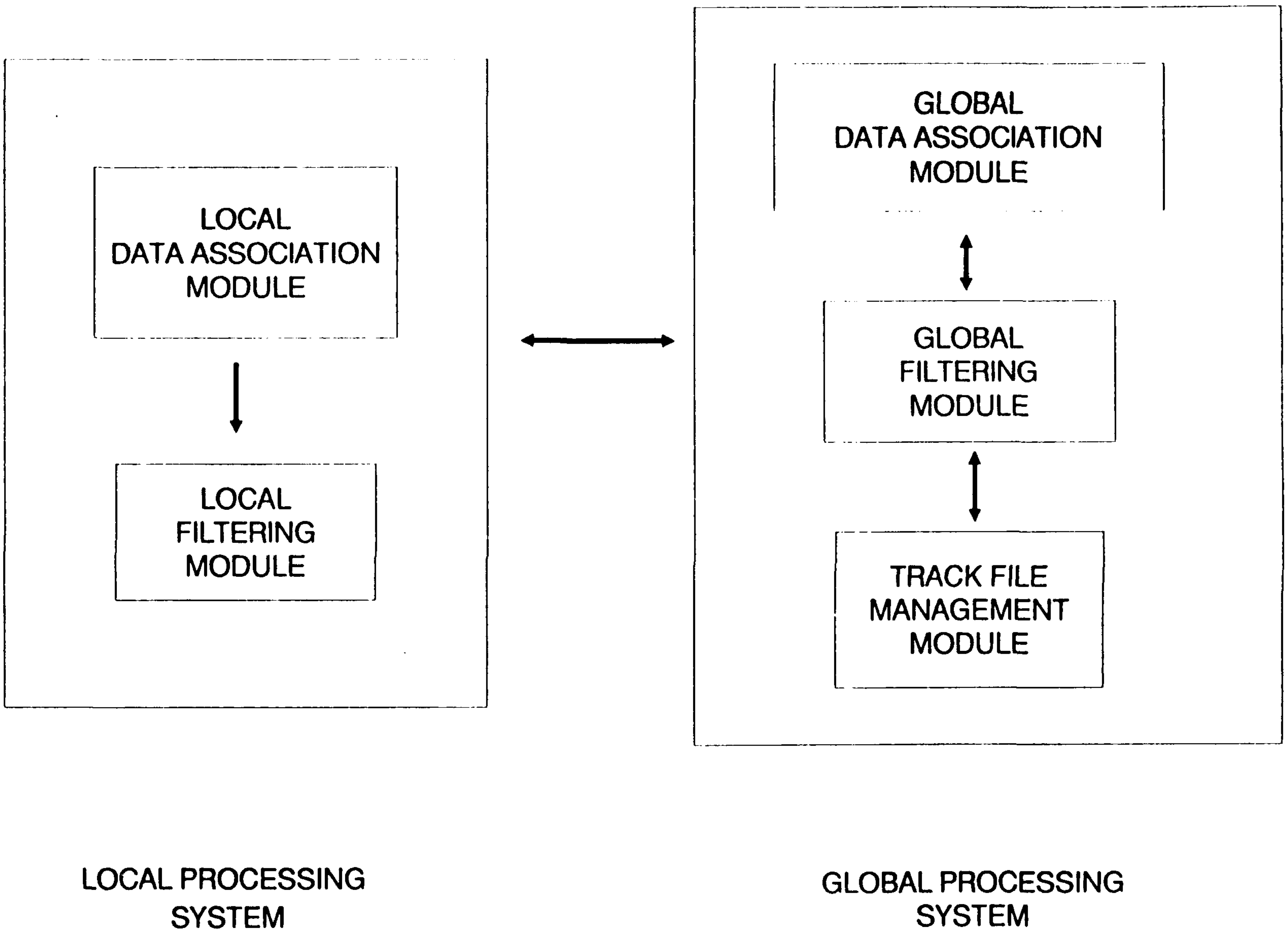


Figure 21: Configuration of the multisensor multitarget tracking software

3) Local Measurement-to-Track Association Process

Each LP performs a local measurement-to-track association and generates a local association matrix. The size of the matrix is determined by the number of targets in the current track file, as well as the number of measurements detected. A gate is formed centered on each measurement. Each entry of the local association matrix is "I", "O", or "M" depending upon whether the measurement is In the gate, Out of the gate, or Missing entirely (a drop-out), respectively.

- Subroutines called

MEASN_M, MEASN, CMAT_M, MTOTA, CR_XGP,

B) Local Filtering Module

1) Local Measurement Update Process

Once the local association matrix is generated, local measurement updating proceeds track by track, using all of the measurements marked "I" for each track. Then each LP sends its local association matrix with the results of local measurement updating to the GP. Outlier rejection also begins.

- Subroutine called

LMUP

2) Local Time Update Process

Usually, local time updating is performed once data association has been accomplished through the global measurement updating process.

- Subroutine called

LTUP

Global Processing

A) Global Data Association Module

1) Measurement Compression Process

After receipt of coordinate transformed measurements from each of the 3 range radars, this process decides which radar measurements originate from the same target.

- Subroutines called

RECIV, COMPR

2) Global Measurement ID Setting Process

In this process, a measurement ID is assigned to each of the compressed measurements. For local measurement ID setting, compressed measurements with ID's are then broadcast to optical tracker LPs. The original measurements with new IDs are broadcast to the range radar LPs.

- Subroutines called

GTOL, CRD

3) Global Measurement-to-Measurement Association Process

This process finds correlations between external nodes (leaves) of the hypothesis tree and newly obtained compressed measurements. The resulting correlation is sent to the hypothesis management process.

Hypothesis Tree Generation For Tentative Tracks: First, all measurements with IDs from the previous scan at time k, are assigned as root nodes of the hypothesis tree. Then, tentative tracks are formed from each root to the measurements of the current scan at time k+1, only if they are not correlated with any existing tracks and are inside an acceptance region (of a leaf). If more than 2 measurements are inside the gate (leaf), the track is split to form several branches, each of which represents a tentative track. Measurements which do not belong to any gates will become root nodes.

- Subroutine called

MTOMS

4) Initial State Determination Process

This process computes the initial state of targets which are confirmed by the hypothesis management process. At any time step, if any leaf with level 3 contains new measurements within the gate, a new track is confirmed. To choose one measurement from the gate, first assume that the target trajectory is approximated the second order polynomial,

$$x(t) = a_0 + a_1x + a_2x^2$$

Then, least square estimates of the coefficients a_0, a_1, a_2 are available based on 3 observations (i.e., three measurements at three different times) given by

$$z = Hx_i + w, \quad i = 1, 2, 3$$

The predicted value of the next position is computed using these estimated coefficients. Then, new measurements inside a gate are compared with the estimated position, and only the track which gives the smallest difference between the predicted value and a measurement survives. Level 2 and level 3 measurements are finite differenced with the newly connected measurements, in order to determine an initial state of the newly confirmed track. The initial state is then used to initialize the filter.

- Subroutines called

MKINT, DEL, FINC

5) Global Measurement-to-Track Association Process

After the GP collects all of the local association matrices, global measurement-to-track association is performed by fusing them. Three screening methods are used to ensure correct associations and resolve conflicts. The Majority Voting Method and the Rule are our basic tools. A likelihood function evaluation process is invoked when conflicts still remain.

- Subroutines called

FUSION, RULE

B) Global Filtering Module

1) Global Measurement Update Process

Global measurement updating is performed based on the result of the global measurement-to-track association process, especially after Majority Voting and the Rule have been applied. When conflicts exist, global measurement updating is performed for all conflicting cases. Filtering results are then sent to the likelihood function evaluation process for a final decision.

- Subroutines called

GMUP, LKHD

2) Global Time Update Process

Performs a global time update step, noting that in future work, data association using local smoothing coefficients could be done here.

- Subroutine called

GTUP

3) Likelihood Function Evaluation Process

Likelihood function evaluation process is performed using its equation from section 2.1.1.

- Subroutines called

LKHD, DET

C) Track File Management Module

1) Hypothesis Tree Management Process

When track lengths are less than 5, correlation results from the global measurement-to-measurement process are received here, and the hypothesis tree management process is invoked. Hypothesis trees continue to branch out until their length reaches 4. Then, a polynomial extrapolation method is used to make a final decision, and all unnecessary branches are pruned. Confirmed tracks are sent to the initial state determination process.

Tree Pruning For Tentative Track Deletion: It is necessary to delete unrealistic tentative tracks in order to reduce computational burden. At any time step, if any leaf with level less than or equal to 3 does not have any new measurements in its gate, it is deleted assuming that it resulted from clutter or noise. If any leaf with level 3 contains several new measurements in its gate, polynomial extrapolation is used to choose the best of these measurements. Then, using the branch which leads to the measurement chosen, the new track is confirmed, and all other branches which share nodes on the confirmed track are deleted from the hypothesis tree.

- Subroutines called

DELET, MKNEW, STORE

2) Track File Updating Process

In our system, the GP updates the track file and broadcasts it to all of the LPs. The track file contains the track ID, as well as the global measurement and time updated states. This process also includes track merging and deleting subprocesses.

- Subroutine called

UPTRA

2.2.3.6 Functional Block Diagram and Test Results

Figure 22 is a functional block diagram of the upgraded software. It represents one cycle of information processing. The field of view for each sensor is assumed to be sufficiently wide so that each sensor is able to detect all of the airborne targets, at each instant of time. All of the LPs and the GP have the same track file. Only measurement data from 3 range radars and 2 optical trackers (G30, G80) for shot 1, shot 2, and shot 3, was processed. However, the program is capable of processing data from all 14 sensors for all of the 6 shots, on a computer with sufficient memory. Also, we have not included the first 50 measurements from each sensor in our simulation, i.e., processing begins after 5 seconds of flight.

(1) Firing Period

The time interval between shots was set to 4.5 sec, as per information provided by WSMR.

(2) Initialization

First, radar measurements are coordinate transformed and sent to the GP for compression and measurement ID setting. When the second set of measurements is received, the later is repeated, and then a global measurement-to-measurement association is invoked with generation of a corresponding hypothesis tree. As new sets are received, global measurement-to-measurement associations are made, and the hypothesis tree is expanded. Global measurement-to-measurement association and tree expansion is repeated until tracks are confirmed. Confirmed tracks are then stored in a track file.

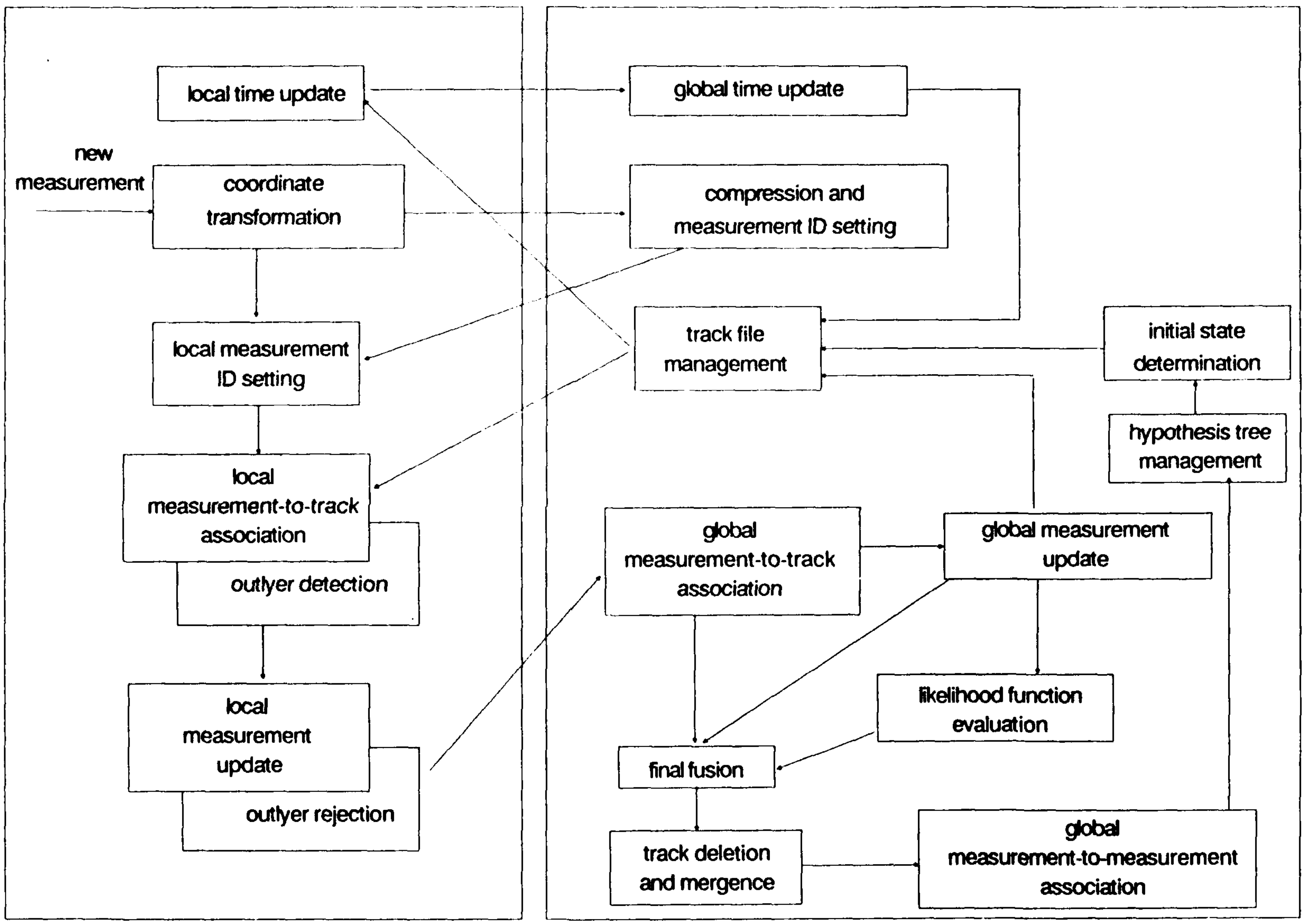

local time update
new measurement
coordinate transformation
local measurement ID setting
local measurement-to-track association
outlyer detection
local measurement update
outlyer rejection
global time update
compression and measurement ID setting
track file management
initial state determination
hypothesis tree management
global measurement-to-track association
global measurement update
likelihood function evaluation
final fusion
track deletion and mergence
global measurement-to-measurement association

Figure 22: Functional block diagram of the multisensor multitarget tracking software.

Tables 1-1 through 1-4 contain the coordinate transformed x-, y-, and z-values of data at the first 4 time steps (labeled sequentially as 50, 51, 52, and 53) from each range radar. Figures 23-1 through 23-3 show the grouped and compressed measurements at these steps in X,Y Y,Z and Z,X coordinates. Figures 23-4 and 23-5 are zoomed versions of figure 23-2. In each of these figures, 2 new measurements are formed at each time step, one from an actual measurement and the other from an outlier measurement. Figure 24 shows the hypothesis tree generated for the same 4 steps after global measurement-to-measurement association has been performed at each step.

In figure 24, at the 50th step, 2 new measurements form root nodes, and level 1 is assigned to each node. After gating with the second measurement set, which also consists of 2 measurements, measurement 1 of the 50th step is correlated with measurement 1 of the 51st step, but measurement 2 of the 50th step is not correlated with any measurements of the 51st step, i.e., the gate centered at the measurement 2 of the 50th step does not contain any measurements from the 51st step. Hence, it is regarded as clutter and deleted from the hypothesis tree. Level 2 is assigned to the leaf of branch 1. On the other hand, measurement 2 in the 51st step is regarded as a new potential target, therefore level 1 is assigned to it. The same process is repeated up to the 53rd step, where only 1 track is confirmed (which is represented by level 4), and a new potential target is designated by measurement ID 2 in the tree. For the confirmed track (with leaf of level 4), the initial state (position, velocity, acceleration in global coordinates) is estimated and stored in the track file.

(3) Coordinate Transformation and Local Time Update

After targets have been detected locally, the tracking cycle starts with two processes. Firsı, range radar measurements are transformed into the GCS and sent to the GP. In the second step, local time updating is performed by all of the LPs, but only when the track file contains at least one confirmed track.

In the following example, targets are detected by radars 393, 350, and 394. Their coordinate transformed x,y,z values are listed in table 2, where k=93 for the third shot, k=138 for the second shot, and k=183 for the first shot.

(4) Global Time Update and Global Predicted State Estimates

After receiving smoothing coefficients and their corresponding track IDs from all of the LPs, the GP starts the global time updating process. Global time updated states are obtained for all tracks and stored in a track file by invoking the track file updating process. The updated track file is broadcast to all LPs for local measurement-to-track association.

Table 1-1: Coordinate transformed measurements at step 50.

	X	Y	Z
Radar 393	-4503.9822	-72394.8189	1563.6824
Radar 350	-4428.8935	-71596.9552	1949.4430
Radar 394	-4436.4745	-71603.7353	1949.0175

Table 1-2: Coordinate transformed measurements at step 51.

	X	Y	Z
Radar 393	-4469.6811	-72360.5947	1604.5716
Radar 350	-4434.3141	-71509.6555	1975.5616
Radar 394	-4439.9353	-71516.1014	1973.6357

Table 1-3: Coordinate transformed measurements at step 52.

	X	Y	Z
Radar 393	-4407.3084	-72295.3127	1683.3286
Radar 350	-4438.0967	-71422.3913	1999.2928
Radar 394	-4443.3743	-71428.8496	1998.0419

Table 1-4: Coordinate transformed measurements at step 53.

	X	Y	Z
Radar 393	-4320.8625	-72201.0084	1785.4247
Radar 350	-4439.9678	-71335.7806	2025.2496
Radar 393	-4446.7809	-71342.0039	2022.2219

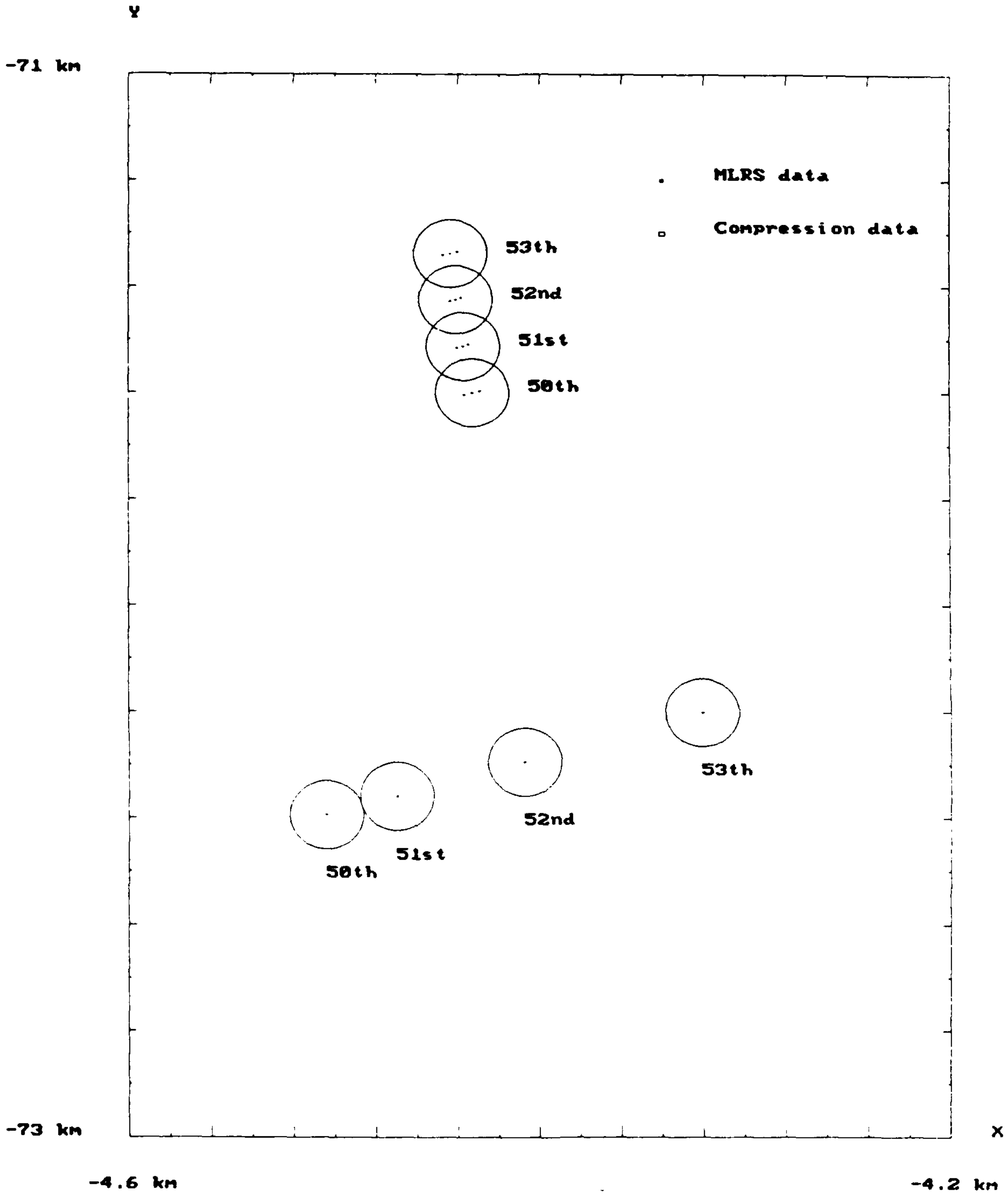


Figure 23-1: Grouped and compressed measurements at step 50th, 51st, 52nd and 53th in X,Y coordinates.

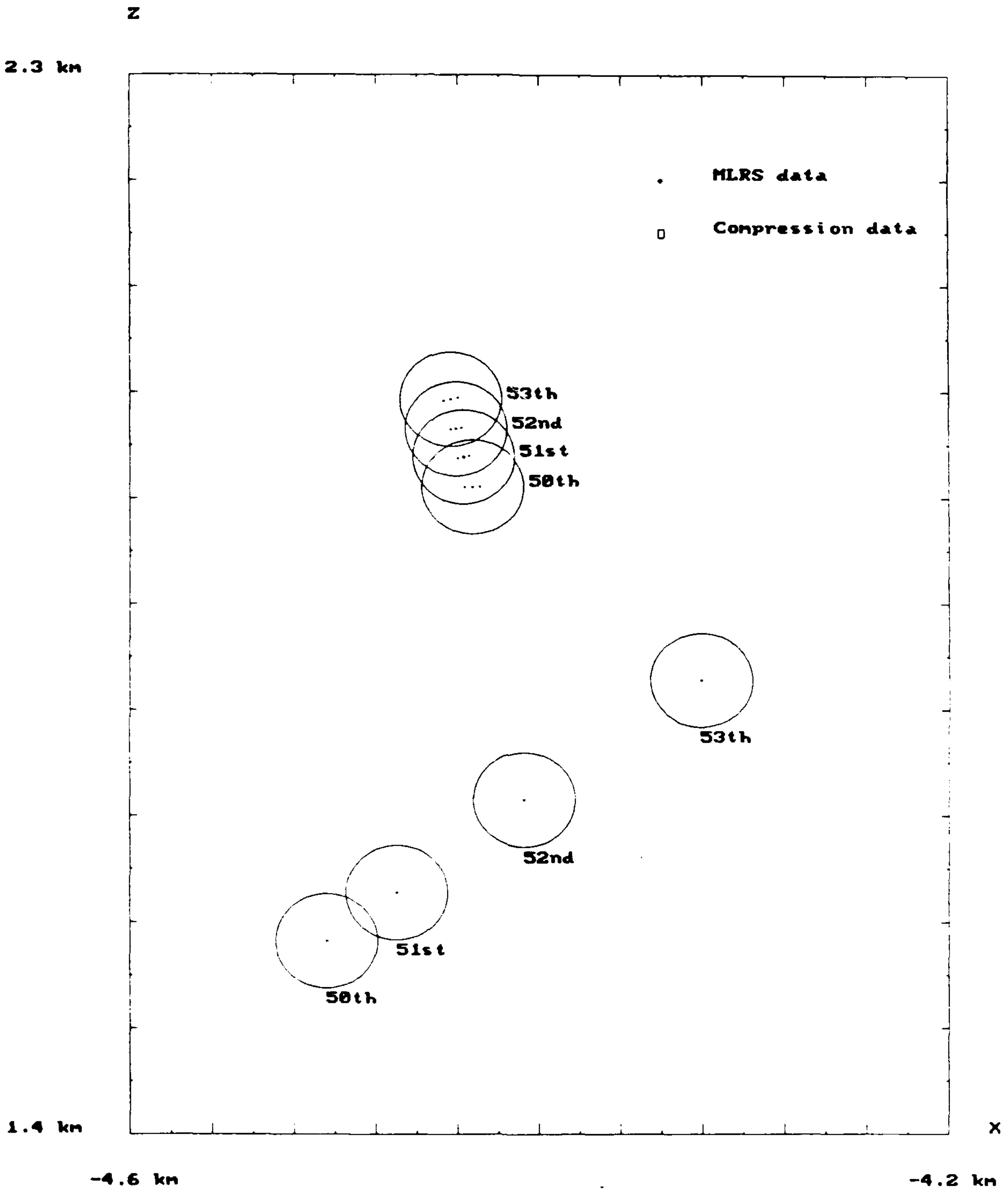


Figure 23-2: Grouped and compressed measurements at step 50th, 51st, 52nd and 53th in X,Z coordinates.

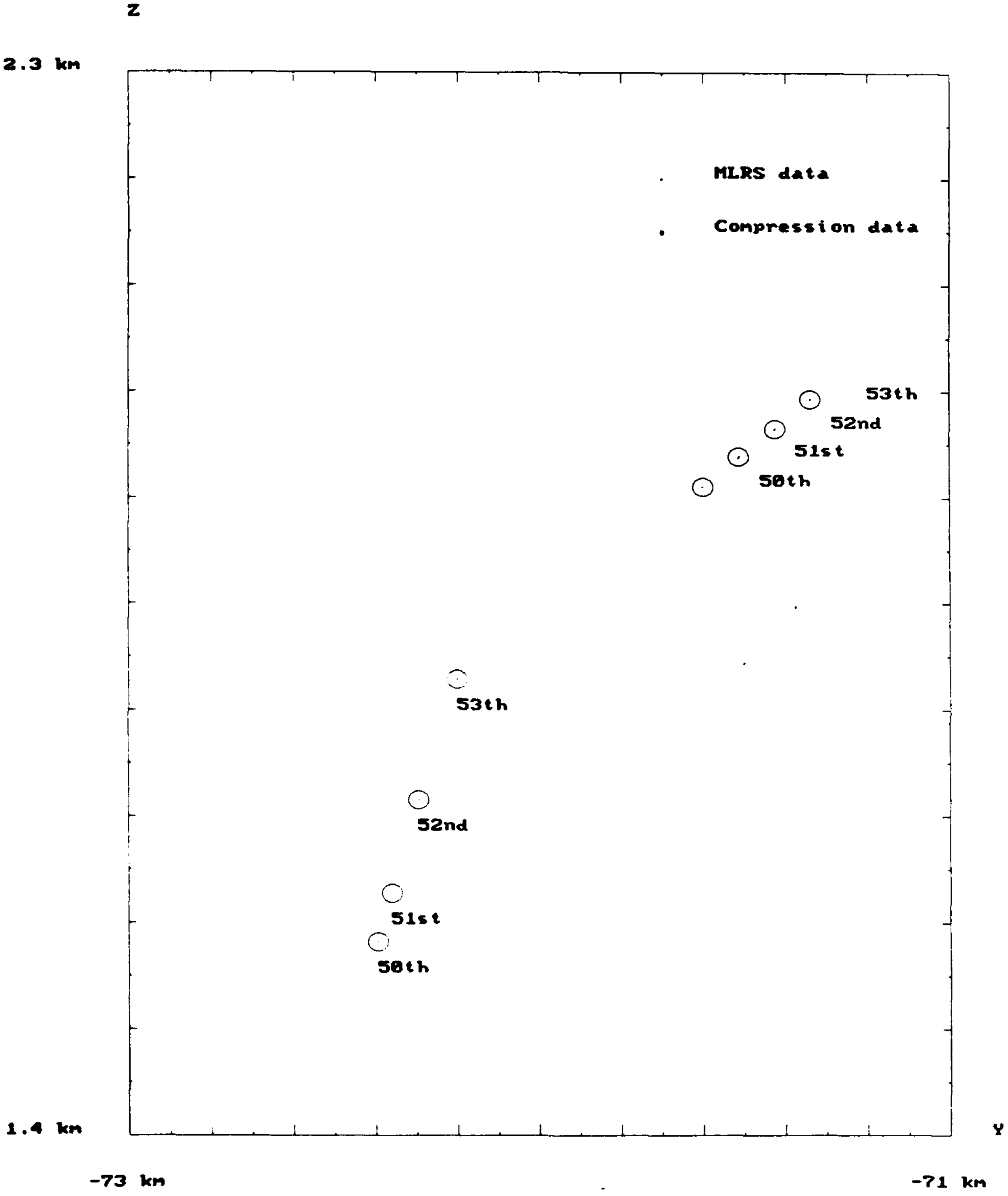


Figure 23-3: Grouped and compressed measurements at step 50th, 51st, 52nd and 53th in Y,Z coordinates.

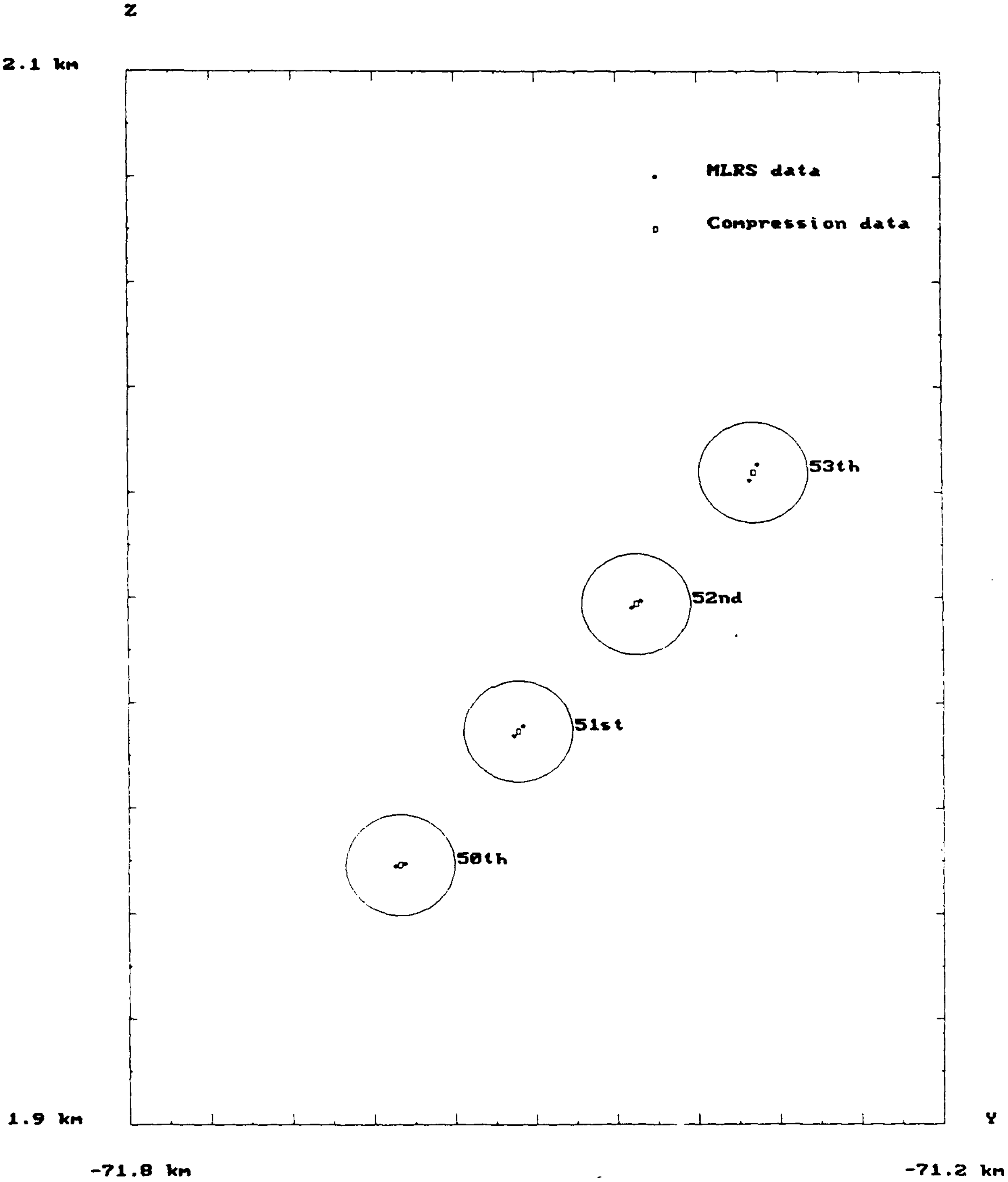


Figure 23-4: Figure 23-3 redrawn to a smaller scale.

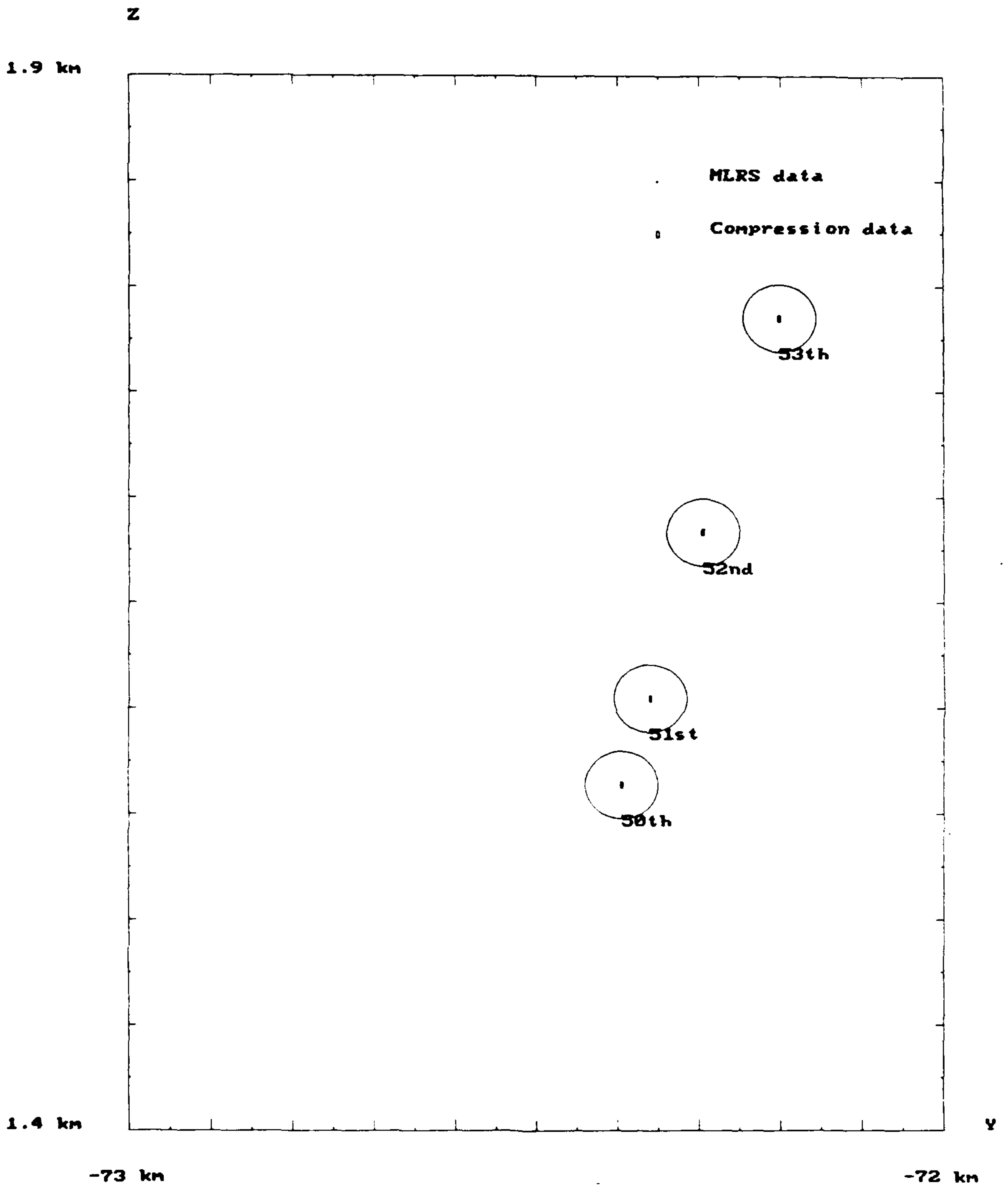


Figure 23-5: Figure 2?-3 redrawn to a smaller scale.

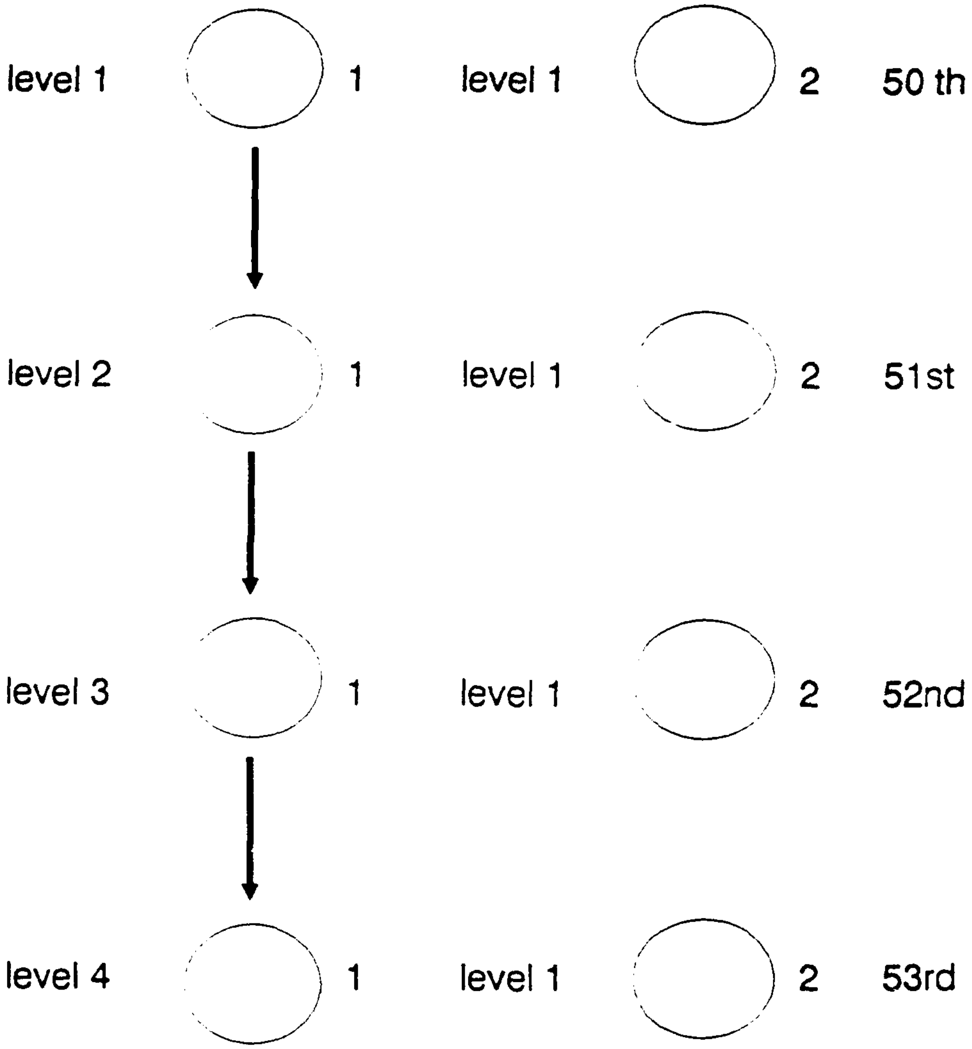


Figure 24: Hypothesis tree generated using the 50th, 51st, 52nd and 53rd measurements.

Table 2: List of measurements from 3 radars at step 183.

Radar 393

X	Y	Z
-4832.9477	-59805.2021	3912.3792
-4614.4501	-62083.0452	4057.9941
-4579.5235	-64775.5641	3496.6276

Radar 350

X	Y	Z
-4837.8757	-59807.2487	3923.8294
-4613.7217	-62084.1828	4064.8936
-4597.5922	-64714.7547	3481.2448

Radar 394

X	Y	Z
-4838.2044	-59807.4555	3925.0130
-4616.2307	-62085.3223	4071.2146
-4600.4270	-64713.6921	3482.3893

(5) Compression and Measurement ID Setting

When the coordinate transformed data is received, the global time updating process (4) begins, and a grouping process is performed concurrently. The grouping process begins withcomputation of the distance to the target(s) measured by the sensors. Next, a simple gating method is used for grouping the data. All targets whose position measurements (range, azimuth and elevation) fall within the same gate belong to a single group. This process can be described in the following pseudo-code form.

```
Define threshold value
In a loop, for number of radars
    In a loop, for number of measurements of r1
        In a loop, for number of measurements of r2
            Compute the distance between a measurement from r1
    and a measurement from r2
            If the distance is less than threshold
                Compute the mean value of measurements from r1
                and r2
                Increase COUNT by 1 and INDEX of r2 = 1
            End if
        End loop
        If COUNT is not ZERO
            In a loop, for number of measurements of r2
                If INDEX equals to 1
                    Choose the minimum distance measurement
                    and associated mean value
            End loop
            In a loop, for number of measurements of r3
                Compute the mean value of measurements from r1,
                r2, and r3
                If the mean value is less than threshold
                    Increase COUNT1 by 1 and INDEX of r3 = 1
                End if
            End loop
            If COUNT1 is not ZERO
                In a loop, for number of measurements of r3
                    If INDEX equals to 1
                        Choose the minimum distance
                        measurement and associated mean value
                        Create COMP_FILE
                    End if
                End loop
            End if
        Else
```

```
            Create COMP_FILE
        End if
    End loop
End loop
```

Here, r1,r2,r3 represent radar 393,350,394 respectively. The threshold chosen for grouping is 46 m. Distance is measured using the conventional Euclidean metric.

Now, we have several groupings of position measurements, each of which represents a different target. Each grouping of radar-based position measurements should have data from at least 2 different radars. If any radar-based position measurement is not grouped with a position measurement from another radar, then a group with only one measurement is formed. When this actually happens in our simulation, the singular radar position measurement is an outlier. By outlier we mean that the target is located outside the expected range of position based on prior knowledge of the target. This indicates that the measurement is not correct and should not be used.

In most cases, each group contains data from more than one radar. Since each grouping represents data from a single target, the position data from the sensors which have been grouped together must be combined as a single set of position measurements for that target. The compression process is used to calculate the nominal position of a target based on all the radar measurements in a specific group. Assuming that the measurement noise covariance is the same for all 3 radars in our simulation, the same matrix can be substituted for the noise covariance in equations (2) and (3). Then, the compressed measurement is the arithmetic average of all the target position measurements in that group.

Figures 25-1 through 25-3 show results of the grouping and compression processes with in X,Y Y,Z and Z,X axes. Figures 25-4 through 25-6 are zoomed versions of figure 25-1. For the 183th step, 4 groups are formed. Each group is distinguished by a circular gate. The location of the center of the circle is the compressed measurement.

In the compression process, each group is assigned a global measurement ID, which identifies a target whose location is given by the compressed measurement of the group. Assignment of global measurement ID is also shown in figures 25-1 through 25-3.

For the measurements identified as ID 1 and 2, the three dots are coordinate transformed measurements, and the square is the compressed measurement. For the measurement with ID 3, two measurements are grouped, and these two measurements are used for compression. For the measurement with ID 4, only one measurement is within the gate, and the data from radar 393 is actually an outlier. However, this determination has not yet been made at this point in the processing, and there is no way to tell, a priori, which data sets are outliers and which data sets are not. Therefore all the groups, including the outlier, must be regarded as individual targets until further analysis can make a determination regarding outliers. Hence, at this point in the analysis there are 4 different

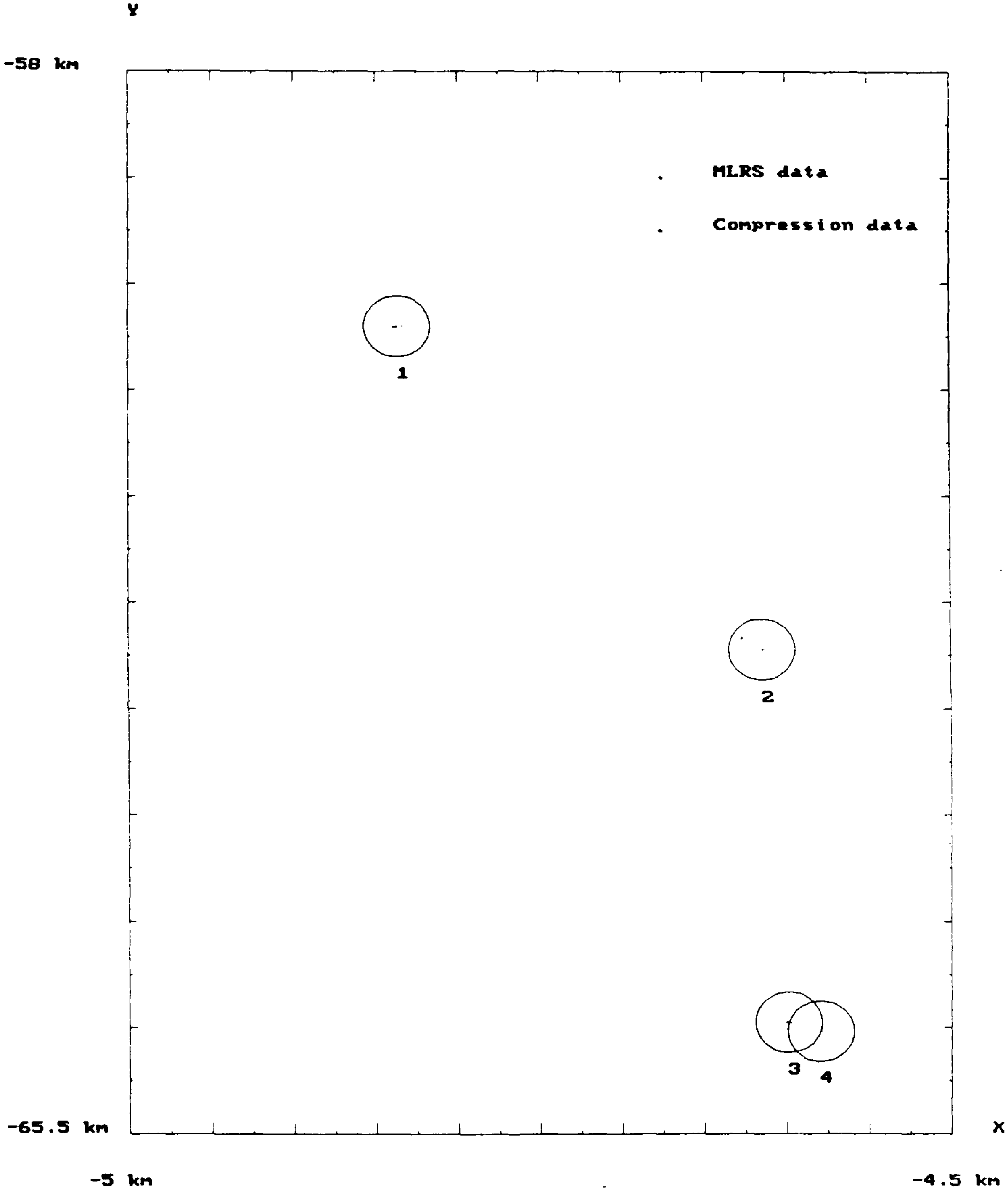


Figure 25-1: Grouped and compressed measurements at step 183 in X,Y coordinates.

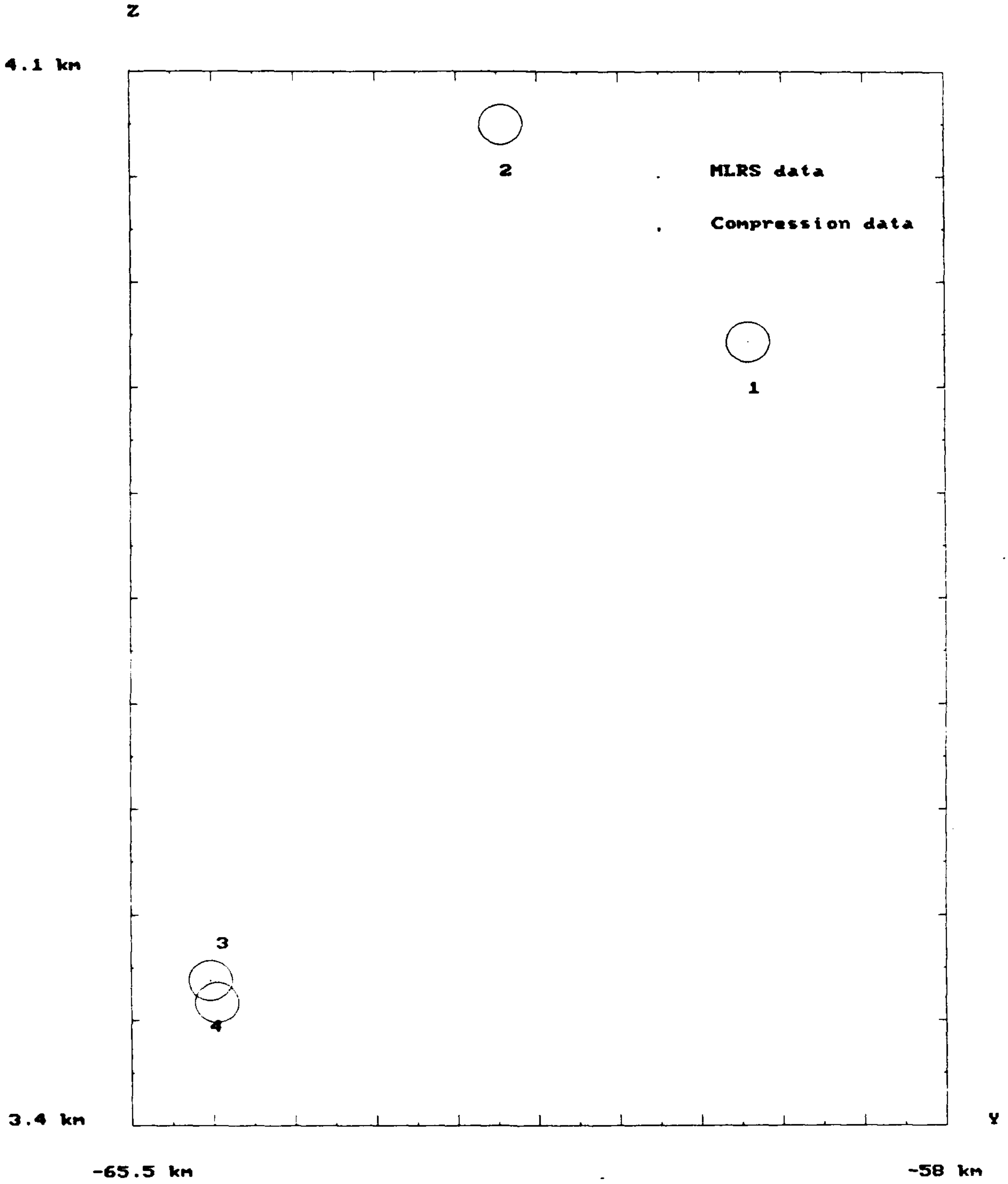


Figure 25-2: Grouped and compressed measurements at step 183 in Y,Z coordinates.

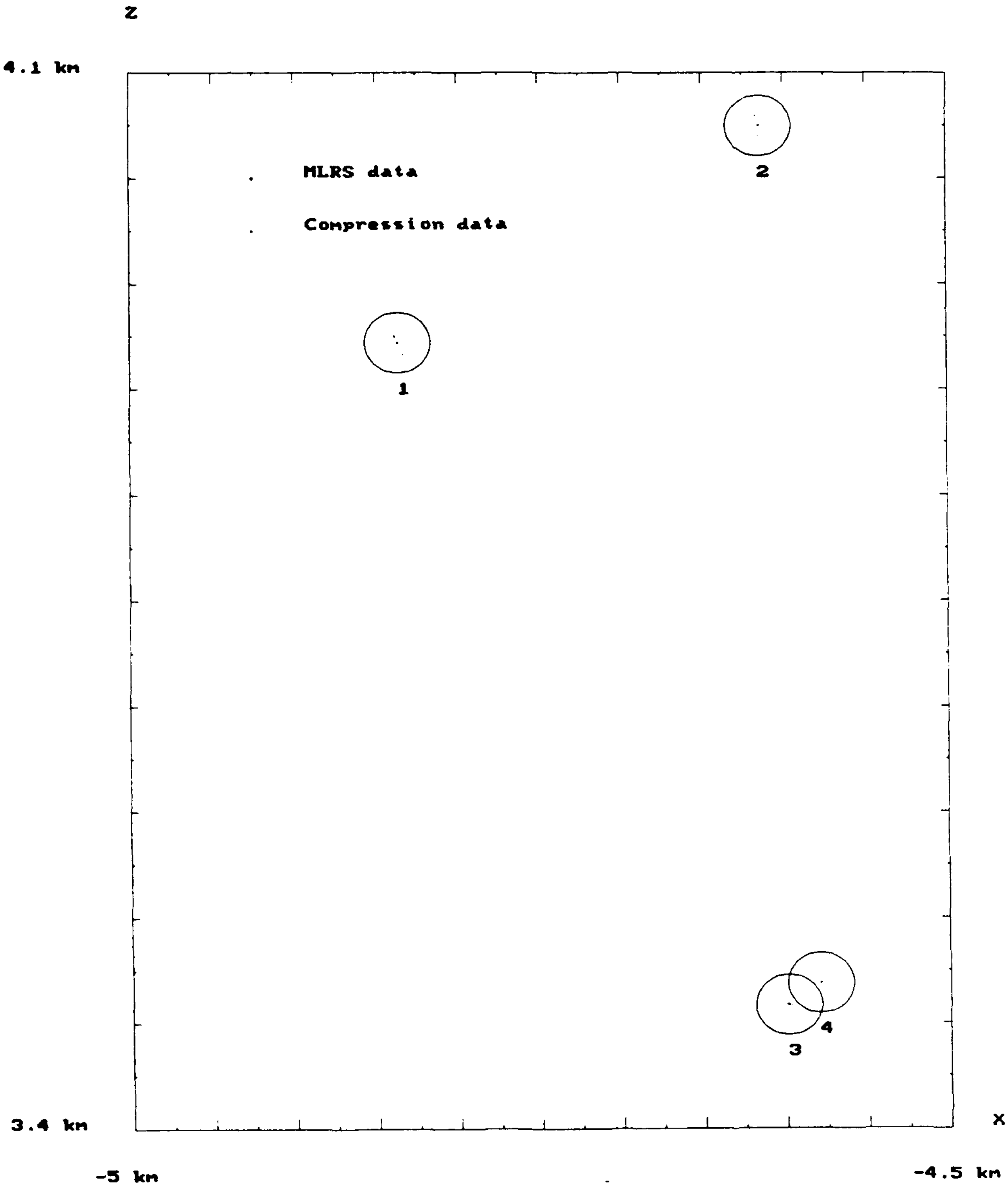


Figure 25-3: Grouped and compressed measurements at step 183 in Z,X coordinates.

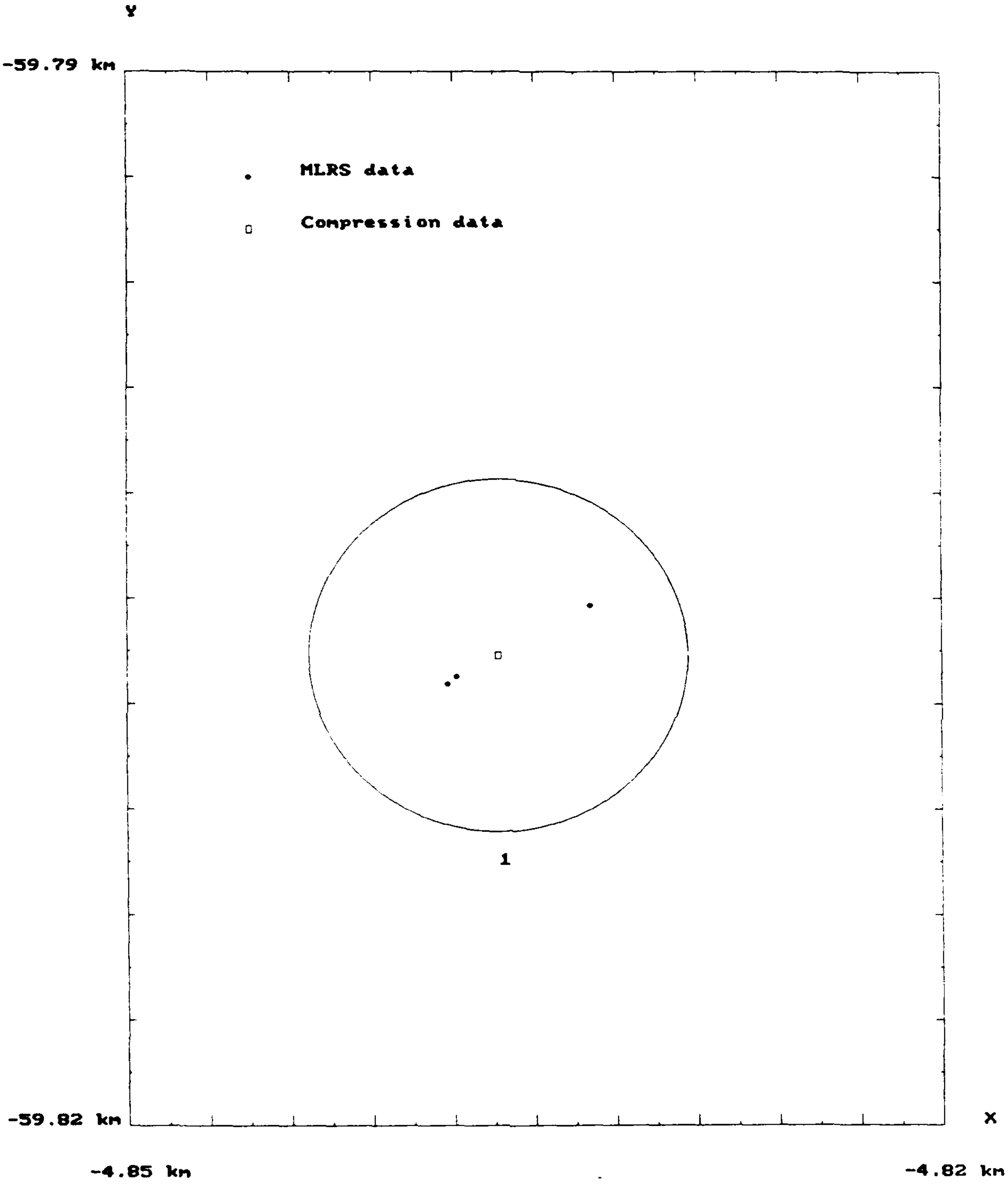


Figure 25-4: Figure 25-1 redrawn to a smaller scale.

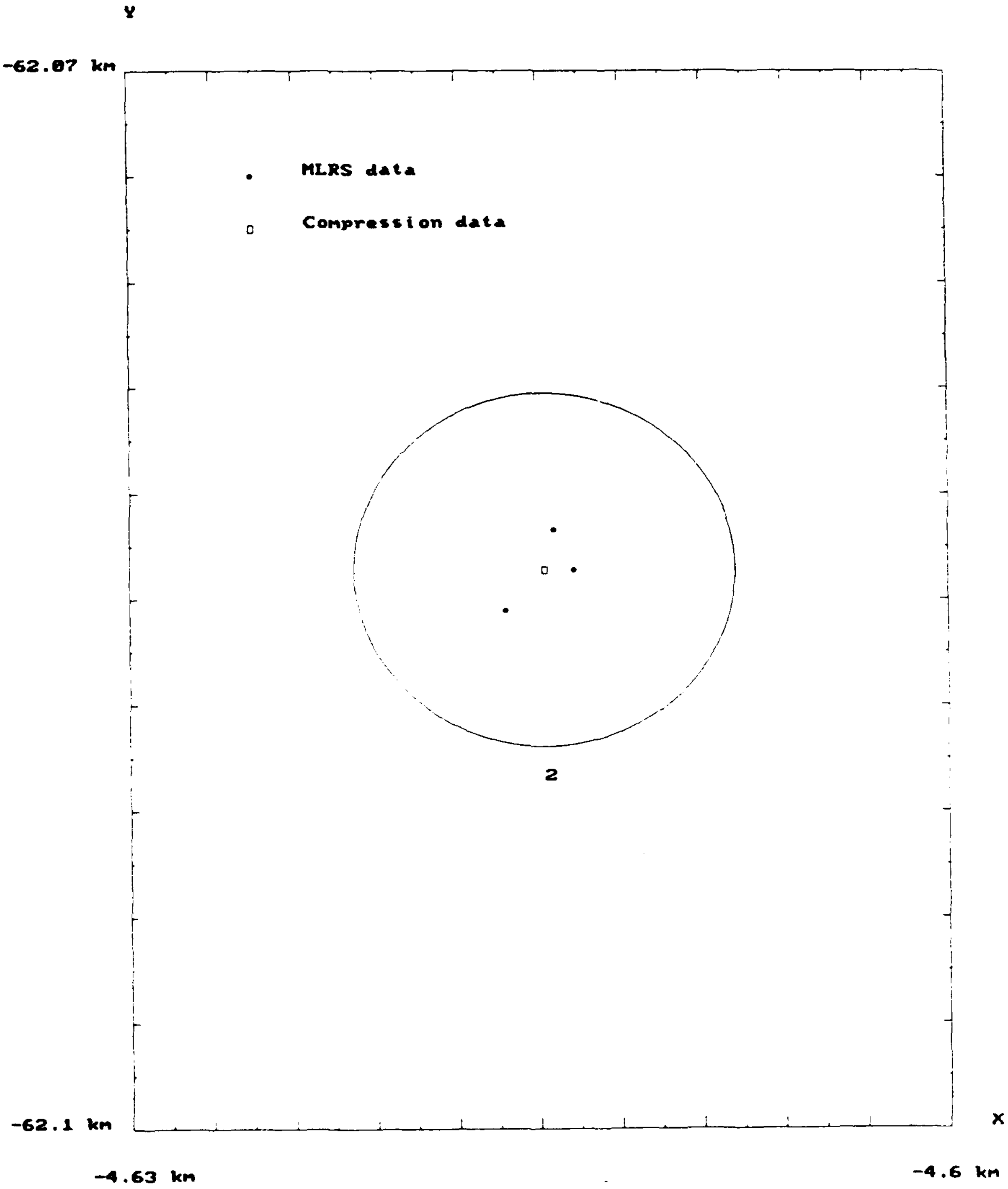


Figure 25-5: Figure 25-1 redrawn to a smaller scale.

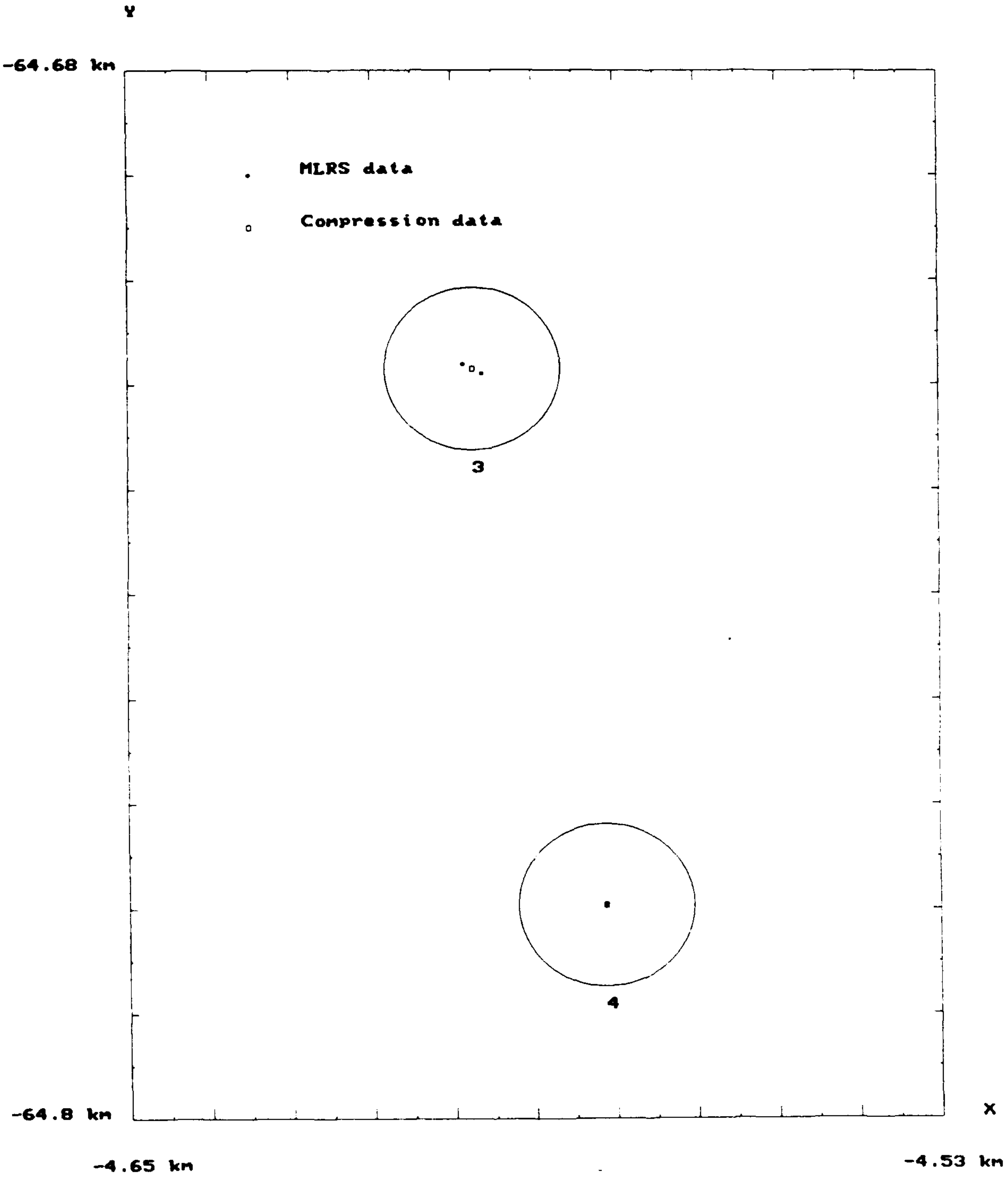


Figure 25-6: Figure 25-1 redrawn to a smaller scale.

groupings of radar measurements, implying that there are 4 targets when the actual number of targets is 3.

The global measurement IDs are sent to LPs along with their corresponding position measurements for the identified target. This enables assignment of a global measurement ID to the target being tracked at the LP. If the sensor is a range radar, the GP can simply send back the original radar measurements transmitted from that radar along with the (new) global measurement ID. If the sensor is an optical tracker, the GP will send the compressed (radar) measurements with the corresponding (new) global measurement IDs. The compressed measurements must be used to assign the appropriate global measurement ID and its corresponding compressed position measurement to the optical tracker.

(6) Local Measurement ID Setting Process

In (5), we noted that original measurements or compressed measurements are returned to LPs (along with their global measurement IDs) depending upon whether the sensor is a range radar or optical tracker, respectively. Now we shall consider the local measurement ID setting process. For the case of radar trackers, the measurement data returned after global fusion are exactly the same as those transmitted from the radar to the GP for grouping and compression, so it is simple to assign new global measurement IDs to each set of radar measurements.

In order to assign a global measurement ID to measurements from an optical tracker, the positions of identified targets must be compared to the positions given by the optical tracker. The position of a target identified by its global measurement ID is given by its associated compressed radar measurements. The compressed radar measurements for all identified targets are returned to optical trackers along with their respective global measurement IDs. Although the compressed radar measurements contain information about range, azimuth, and elevation, the optical trackers only measure azimuth and elevation. Therefore, the local compressed target locations can only be compared on the basis of azimuth and elevation.

The ID setting process is based on gating and nearest neighbor analysis. Both gating and nearest neighbor analysis compare the target position (elevation and azimuth) measured by an optical tracker with the target positions for each of the compressed (radar) measurements. A rectangular gate, centered at the compressed measurement in local coordinates, is chosen. For the optical tracker G30, .009 rad is chosen as a radius for both azimuth and elevation. For the optical tracker G80, .051 rad is chosen for azimuth, .02 rad for elevation. Now, a correlation can be made. The magnitude of the difference between the optical measurement and compressed radar measurement of azimuth is compared to the azimuth side radius. Similarly, the elevation side radius is compared to the difference between the optical measurement of elevation and the compressed radar measurement of elevation.

If the actual measurement and the compressed measurement are correlated, then the global measurement ID associated with the compressed measurement is assigned to the actual measurement. In most cases, gating will correlate one optical measurement set with only one compressed radar measurement set. In this case, the measurement data from the local optical tracker can simply be assigned the global measurement ID associated with that compressed radar measurement. However, it happens that occasionally more than one compressed measurement is correlated with an optical measurement. In this case, a nearest neighbor analysis is performed; the Euclidean metric is computed for each association, and the association with the smallest distance is chosen as a final association. Similarly, the remaining measurements which were not correlated with any compressed measurements on the basis of gating, are assigned the global measurement ID of the nearest identified compressed measurement. Note that the local measurement IDs assigned to each optical measurement is a global measurement ID which is common to a specific group of radar trackers.

Table 3 shows how the local measurement IDs of the 183th step are associated with global measurement IDs.

(7) Local Measurement-To-Track Association

Once global measurement IDs have been assigned to all measurements, local measurement-to-track association starts. The track record contains the existing target tracks up to the last cycle of processing, and now the track records must be updated to include the presently identified measurements. In the last processing cycle, global measurement updated states were calculated for each identified target. Global time updated states were calculated for each target tracked in the last complete cycle after step (4) of the current cycle. Both types of global states will be used for association. In the association process, an association matrix is created locally at each local sensor. The local association matrix indicates the association between all current measurements with the target tracks from the last processing cycle.

Gating is used to associate the current local measurements with existing target tracks. There are two different comparisons which can be made to correlate the current measurements with an existing track. First, for each existing track, a target position is calculated as a prediction based on existing knowledge as to what the target's location should be at the present time:

> compute $h^i(x_k(-))$, where $x_k(-)$ is the globally optimal time updated state estimate of the target, and h^i represents the nonlinear observation function for the sensor i.

Table 3: Local measurement ID setting.

Radar 393

Local ID	Global ID
1	1
2	2
3	4

Radar 350

Local ID	Global ID
1	1
2	2
3	3

Radar 394

Local ID	Global ID
1	1
2	2
3	3

G30

Local ID	Global ID
1	2
2	
3	

G80

Local ID	Global ID
1	1
2	2
3	4

This is the global time updated state for that target track, and was computed in step (4) of the current cycle. Now, a gate is drawn centered around the global predicted state of the target track position, $h^i(x_k(-))$. Any metric distance, $d(thr)$ may be used for the gate. Let $d(h^i(x_k(-)),y_j)$ be the distance between the measurement y_j and $h^i(x_k(-))$; if

$$d(h^i(x_k(-)),y_j) < d(thr),$$

then y_j is inside the gate. Each measurement is compared to the gate. If any of the measurements fall inside the gate those measurements are correlated with that target track. Correlations are indicated by assigning an "I" to the location corresponding to that measurement and target track in the local association matrix.

In our simulation, a rectangular volume gate is used for range radars, and a rectangular (planar) gate is used for optical trackers. The radii of the gate are 2 m for range, .1 rad for azimuth, and .1 rad for elevation. The radii of the gate for an optical tracker are .1 rad for azimuth and .1 rad for elevation.

If a measurement was not correlated with the global time updated state for any of the target tracks, a second method is tried to make the correlation. A new gate is used which is centered on the global measurement updated state for the target, $h^i(x_{k-1}(+))$. This gate represents a reasonable estimate of the maximum distance a target could travel from its last known position, based on a general analysis of target motion (see section 2.2.2). The gate radii for range radars are 100 m for range, .15 rad for azimuth, and .15 rad for elevation. On the other hand, the radii for both azimuth and elevation is .15 rad. As before, comparisons are made between each of the global measurement updated states and the measurements. If any of the measurements fall inside the gate, an "I" is assigned to the appropriate location(s) in the local association matrix.

Otherwise, the measurement is regarded as spurious, and an "O" is assigned to the appropriate location(s) in the association matrix. "O" is marked unless there aren't any measurements detected by the i^{th} sensor, then, "M" replaces the i^{th} column (or row, depending upon the representation in the matrix) entries of that measurement. Next, the outlier detection process is invoked for that target.

More explicitly, the following pseudocode is adopted:

```
If (the gate centered at the global time updated state does not contain any measurement)
then

    If (the gate centered at the global measurement updated state does not contain any
    measurement) then
```

```
        Assign "O" in the association matrix.
    Else
        Fill up the association matrix with "I" at the corresponding measurement
        position.
Else
    Fill up the association matrix with "I" at the corresponding measurement position.

    Fill up the empty position of the association matrix with "M" for missing data.

If (all measurements are assigned by "O" or "M") then

    Invoke outlier detection process
```

Tables 4-1 through 4-5 are local association matrices for radars 393,350,394 and optical trackers G30,G80, respectively.

(8) Local Measurement Updating

Local measurement updating is invoked after local measurement-to-track association has been completed. The local measurement updating process generates a local measurement updated square root information matrix and vector for each target identified by a tracker. For each target registered in the track file, the association matrix obtained through local measurement-to-track association, is scanned along the measurements. Local measurement updating is performed for all measurements marked "I" in the local association matrix. Local measurement updating of outlier measurements is also performed for targets which have "O" correlation with all measurements.

(9) Global Fusion and Global Measurement Updating

Each LP sends its local association matrix and corresponding local square root information matrices and vectors to the GP for fusion and global measurement updating. As a first step in the fusion process, the association matrices are combined by invoking the Majority Voting Method. That is, a global fusion matrix is created, whose entries are determined by the number of "I"s and "O"s from corresponding entries in the local association matrices. If the number of "I"s is greater than or equal to the number of "O"s, then the entry at that location in the global fusion matrix is "I". Otherwise, the entry is "O". In this counting procedure, "M" is not considered.

Table 4-1: Local association mtrix for radar 393.

Radar 393

Measurement ID

Target ID

	1	2	3	4
1	I	O	M	O
2	O	I	M	O
3	O	O	M	O
4	I	O	M	O
5	O	O	M	I
6	O	O	M	O
7	O	I	M	O

Table 4-2: Local association matrix for radar 350.

Radar 350

		Measurement ID			
		1	2	3	4
Target ID	1	I	O	O	M
	2	O	I	O	M
	3	O	O	O	M
	4	I	O	O	M
	5	O	O	I	M
	6	O	O	O	M
	7	O	I	O	M

Table 4-3: Local association mtrix for radar 394.

Radar 394

Measurement ID

Target ID

	1	2	3	4
1	I	O	O	M
2	O	I	O	M
3	O	O	O	M
4	I	O	O	M
5	O	O	I	M
6	O	O	O	M
7	O	I	O	M

Table 4-4: Local association matrix for optical tracker G30.

G30

Measurement ID

Target ID

	1	2	3	4
1	M	O	M	M
2	M	I	M	M
3	M	O	M	M
4	M	O	M	M
5	M	O	M	M
6	M	O	M	M
7	M	I	M	M

Table 4-5: Local association matrix for optical tracker G80.

G80

Measurement ID

Target ID	1	2	3	4
1	I	O	M	O
2	O	I	M	O
3	O	O	M	M
4	I	O	M	O
5	O	O	M	I
6	O	O	M	M
7	O	I	M	O

If all conflicts are resolved in the global fusion matrix, global time updating proceeds according to the result of association. Otherwise, the Rule is applied to the global fusion matrix to resolve conflicts as a second screening method. The Rule is based on the obseıvation that one measurement is associated with at most one target.

Rule: Scan all of the slots, row by row, to find which shot is associated with only one measurement. Reset all entries in the column of the measurement found to O (outside). When a target is associated with more than 2 measurements, the measurement with more "I"s is chosen.

By repeated application of the Rule we might be able to reach the correct association. The association result, after application of the Rule, is stored again in the global fusion matrix. If correct association is achieved, global measurement updating proceeds. However, there will be cases which cannot be resolved with this rule only. When the Rule fails to provide a definitive association, a decision based on the likelihood function is used as a third screening method.

The present simulation is comprised of 5 trackers, so the global fusion matrix combines 5 local association matrices. Table 5-1 shows the global fusion matrix after the majority voting method has been applied to the local association matrices in tables 4-1 through 4-5. In table 5-1, conflicting assignment exists for the target with ID 5. To overcome this conflict, the next step is ιo use the Rule described in section 2.2.3.

However, as shown in table 5-2, the Rule cannot resolve the conflict in this particular case, and as a last resort we must evaluate the likelihood function to resolve it. Before we can evaluate the likelihood function, a global measurement update must be performed.

(10) Global Measurement Update

The global measurement updating process is invoked based on the results returned by the global fusion matrix. If there are no conflicts in the global fusion matrix, the global measuremenι updated state for each respective target can be computed and stored in the track file. When conflicts in the global fusion matrix have not been resolved, global measurement updating is performed for all possible cases and evaluation of the likelihood function is applied to resolve the conflicts.

(11) Evaluation of Likelihood Function

Conflicts that exist in the global fusion matrix are resolved using a likelihood function generated from the global measurement updated square root information matrices for all possible cases obtained in (10). Association conflicts which existed in the global fusion

Table 5-1: Global fusion matrix using the Majority Voting Method.

		Measurement ID			
		1	2	3	4
Target ID	1	I	O	O	O
	2	O	I	O	O
	3	O	O	O	O
	4	I	O	O	O
	5	O	O	I	I
	6	O	O	O	O
	7	O	I	O	O

Table 5-2: Global fusion matrix using the RULE.

	Measurement ID			
Target ID	1	2	3	4
1	I	O	O	O
2	O	I	O	O
3	O	O	O	O
4	I	O	O	O
5	O	O	I	I
6	O	O	O	O
7	O	I	O	O

matrix can now be resolved by choosing the correlation case which gives the smallest value of the likelihood function. The resolved associations and corresponding information are sent to update the track file and complete the global measurement-to-measurement association process.

For the case described in table 5-2, we have obtained the value 13.7406575 for measurement 3, and 13.806810111 for measurement 4. The association pair which gives the smallest likelihood function value is chosen; in this case, measurement 3 is selected. This is shown in table 5-3.

(12) Track Merging and Track Deletion

Once a one-to-one mapping has been established between track IDs and global measurement updated states, the tracks are evaluated to determine which ones are duplicates and which are anomalous. Any tracks which show similar behavior (almost identical) are considered duplicates, and will be merged into one track. On the other hand, any track which shows anomalous behavior, i.e. velocity or acceleration beyond the physical limits of an MLRS rocket, will be deleted from the track file. Duplicate tracks will be merged, and anomalous tracks will be deleted.

Duplication is demonstrated by track 1 and track 4 in table 6. A comparison of the data for tracks 1 and 4 shows small differences between y-velocity, x-acceleration, y-acceleration, and z-acceleration for each track. The following explanation can be applied. Initially one target is tracked, identified as track 1. At some iteration prior to 183, an outlier was identified as a new track, labeled track 4. Then track 1 and track 4 were treated as separate targets until it was clear that both tracks were convergent. In other words, the gates of tracks 1 and 4 contained identical measurements. Since the same measurements are used in the filtering process, the global measurement updated states of both tracks converge. The criterion used for convergence is to compare the magnitude of the difference between the position vectors of two tracks with a specified threshold. If the magnitude is less than the threshold value (3 m), the tracks are considered identical and the two tracks are merged. Another example of merging with a similar explanation is shown by track 2 and track 7. Figures 26-1 through 26-3 show this case more clearly. The numbers in each figure represent track numbers.

Track deletion is determined by comparing the global measurement updated state with specific boundary values. In table 6, the magnitude of velocities for track 3 and track 6 are 3201.0421 m/sec and 3054.4420 m/sec, respectively. We have set the threshold value for deletion using a speed of 3000 m/sec. The behavior of the targets in track 3 and track 6 is anomalous because the velocity of these targets is not appropriate for the type of target we want to track. Therefore, these tracks are deleted. This is also shown in figures 26-1 through 26-3.

Table 5-3: Global fusion matrix using the likelihood function.

	Measurement ID			
Target ID	1	2	3	4
1	I	O	O	O
2	O	I	O	O
3	O	O	O	O
4	I	O	O	O
5	O	O	I	O
6	O	O	O	O
7	O	I	O	O

Table 6: Global measurement updated state at step 183.

	1	2	3	4	5	6	7
x	-4836.3443	-4614.8021	-4894.8793	-4836.3443	-4599.0105	-5442.2085	-4614.8021
y	-59806.6343	-62084.1823	-67625.5185	-59806.6343	-64714.2233	-66462.8272	-62084.1823
z	3920.4073	4064.7009	1715.1604	3920.4073	3481.8170	1912.7958	4064.7009
$\dot{x}$	-13.7140	-10.7801	3200.4956	-13.7140	-17.5573	-2415.4698	-10.7801
$\dot{y}$	487.1116	552.1354	-35.5327	487.1118	630.5180	184.4807	552.1354
$\dot{z}$	-1.4475	55.4267	-47.2901	-1.4475	98.3732	-1860.4002	55.4266
$\ddot{x}$	1.9432	-2.3798	4120.7427	1.9433	-1.9157	-2671.5097	-2.3799
$\ddot{y}$	-11.8729	-14.4680	-515.7403	-11.8699	-20.3208	-376.0155	-14.4651
$\ddot{z}$	-12.1609	-12.5031	755.6285	-12.1610	-13.1431	-1655.4370	-12.5043

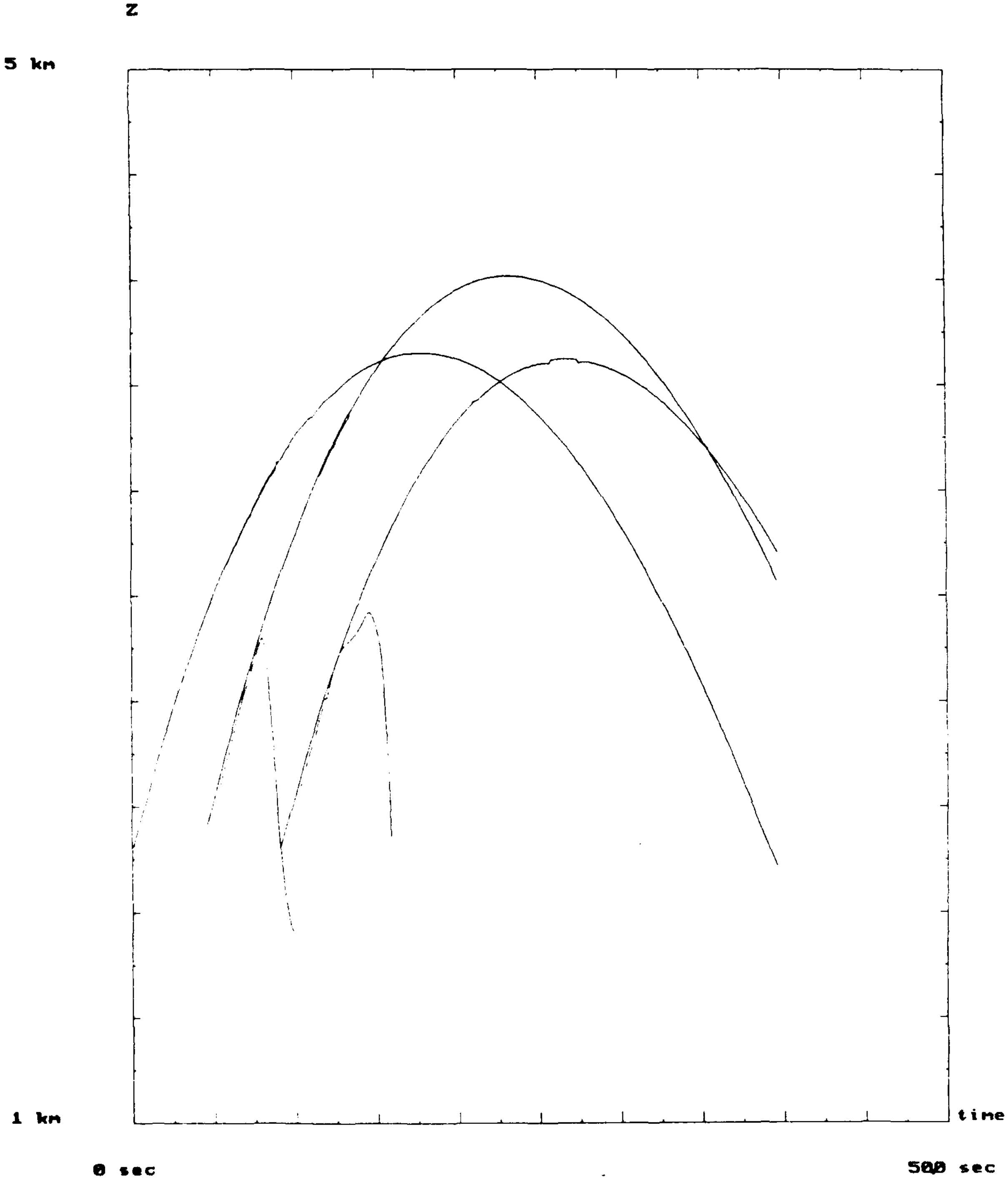


Figure 26-1: Z component of global measurement updated states.

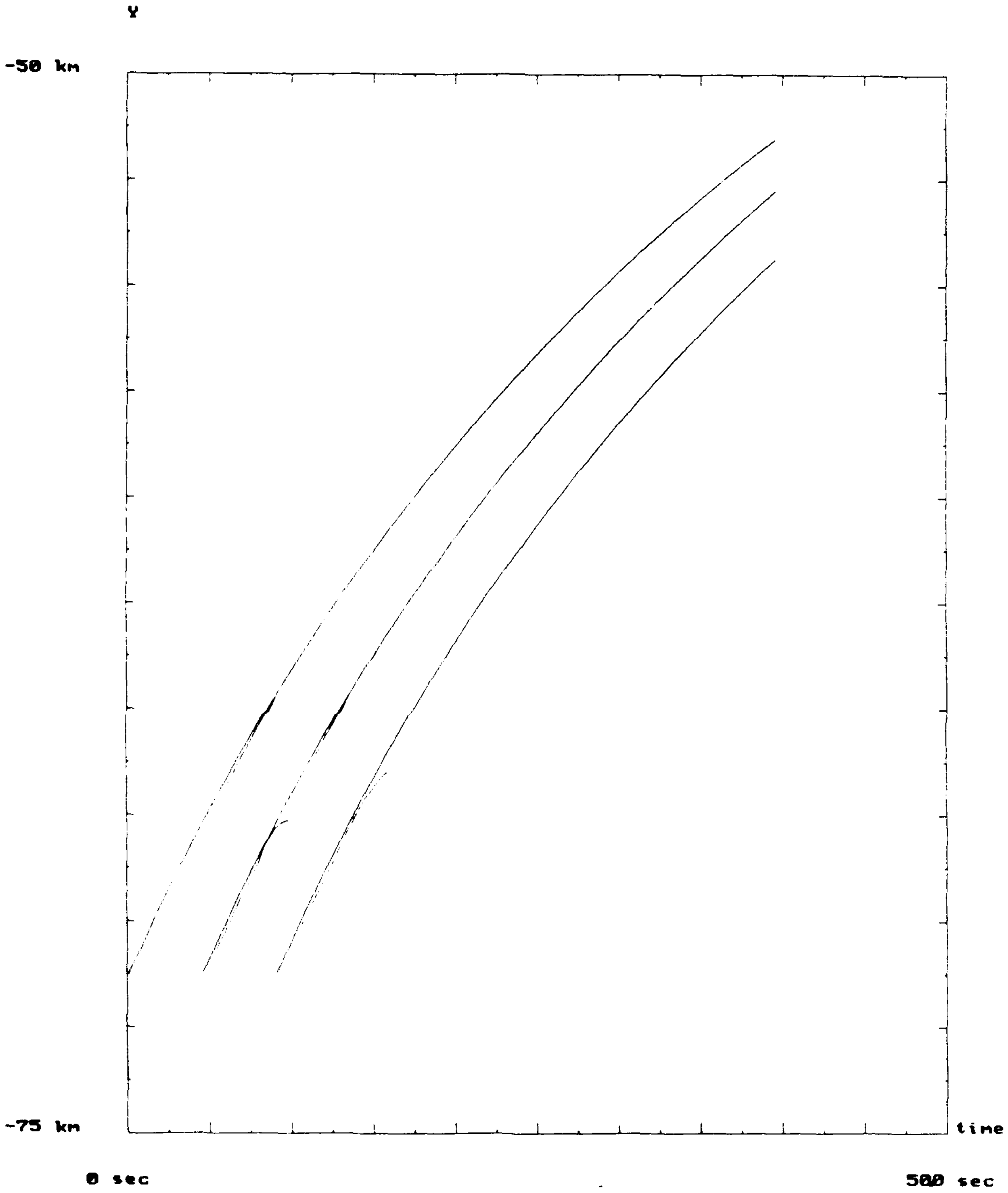


Figure 26-2: Y component of global measurement updated states.

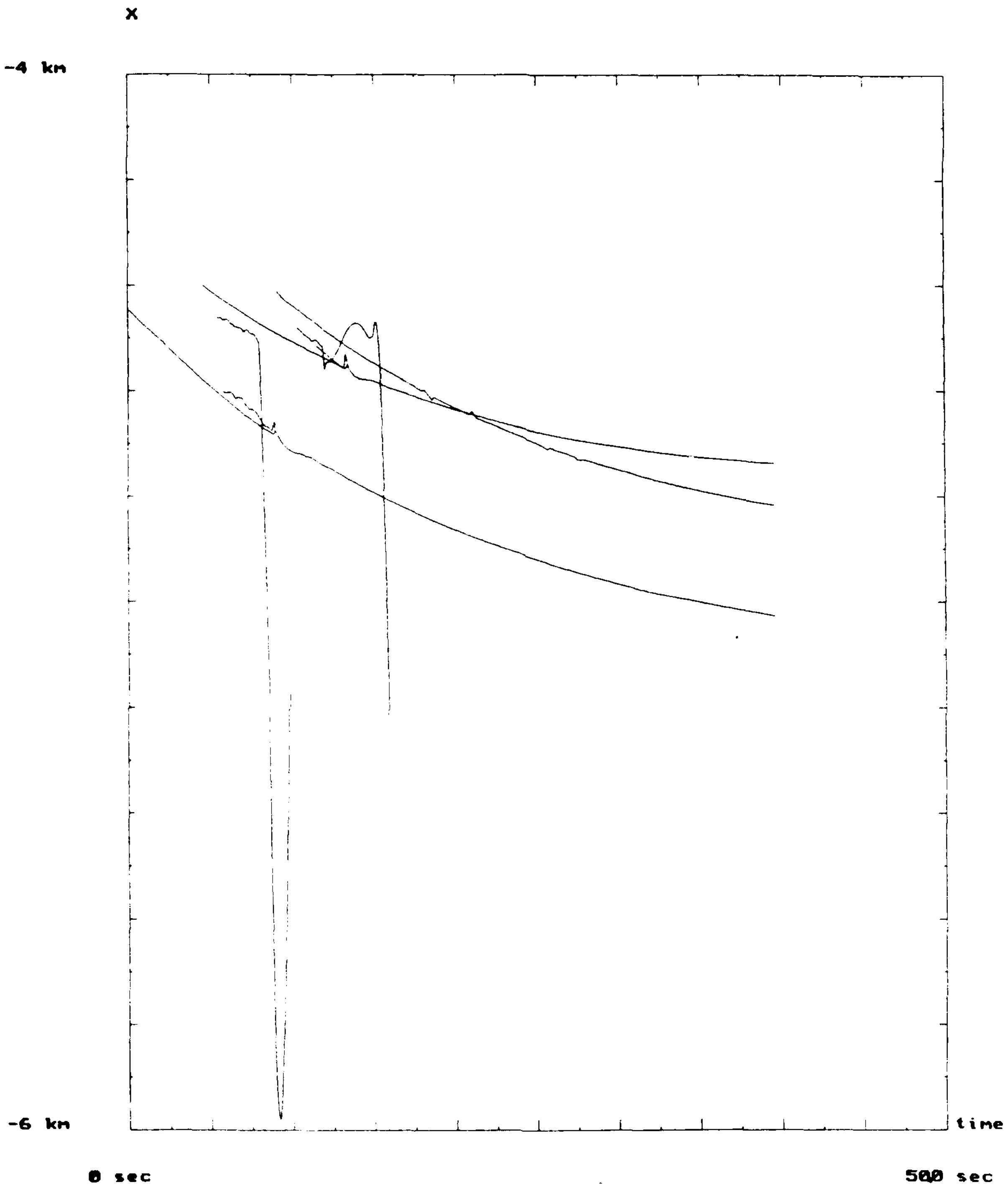


Figure 26-3: X component of global measurement updated states.

(13) Global Measurement-To-Measurement Association

The global measurement-to-measurement association process correlates the external nodes (or leaves) of the current hypothesis tree with the newly obtained compressed measurements. This process is only applied to new compressed measurements which have not yet been correlated with existing tracks. A simple gating method is used for correlation. In our simulation, a rectangular volume gate is used with radii of 10 m (x-axis), 90 m (y-axis), and 29 m (z-axis). Correlation results are sent to the hypothesis management process.

Up to now our examples have been for iteration step 183, however, it is not necessary to make a global measurement-to-measurement association at the 183rd iteration step. First, every hypothesis tree has at least 3 levels, which means every track has existed for at least three consecutive iterations. Second, all of the compressed measurements at the 183rd iteration are correlated with existing tracks.

An example of global measurement-to-measurement association can be found in the 94th step. In this case the current hypothesis tree is given by level 1, 2, and 3 branches. Among the 94th compressed measurement set, measurement 3 and measurement 5 are not correlated with any tracks that exist at the 93rd iteration. Hence, global measurement-to-measurement association is performed between the leaves of the hypothesis tree and measurements 3 and 5. This is given in table 7. In table 7, only measurement 3 is connected with leaf 3. This result is sent to the hypothesis management process.

(14) Hypothesis Management

The hypothesis management process consists of several subprocesses. First, connections (branches) must be made between external nodes in the hypothesis tree and newly obtained measurements. These connections are based on the correlation results. Second, the number of levels of new external nodes are increased by adding one level to each of their parent nodes. A new external node may have several parent nodes, and these parent nodes might have different numbers of levels. In this case, a copy of the new external node is made for each of the parent nodes. Each copy is regarded as a different node and a tentative track is formed between each parent node and its copy of the external node. The level of each copy is given by the level of the parent node to which that copy is connected. Third, there is a pruning process which removes any parent nodes with less than or equal to three levels from the hypothesis tree if they are not connected to any new measurements. Fourth, any measurement which is not connected to any previous external

Table 7: Global measurement-to-measurement association matrix at step 94

measurement ID

track ID

	1	2	3	4	5
3	O	O	I	O	O
5	O	O	O	O	O

nodes is regarded as a new target and assigned level 1. Finally, after checking the levels of all new external nodes, all possible tracks which end at leaves with more than 3 levels are sent to the initial state determination process. In figure 27, one track is formed, and compressed measurement 3 becomes the leaf of that track. This result is sent to the initial state determination process.

(15) Initial State Determination

Initial state determination process is invoked after tracks which have level 4 nodes are identified. Using the first three values and second order polynomial extrapolation, as described in section 2.2.3, the initial state of the confirmed track is estimated.

The estimated initial state is stored in the track file as a global time updated state and a global measurement updated state with track (or target) ID.

(16) Broadcast Updated Track File

The final updated track file is broadcast to all LPs. The LPs are also sent the final decisions on the global measurement-to-track associations which were chosen by the global measurement updating process. Based on this, local time updating is performed without any data association, and the next cycle of the entire process may begin.

To summarize, MTI's software for multi-sensor multi-target tracking is based on the EDSRIF. It was successfully tested using real MLRS data. Performance of the software was evaluated by comparing its track calculations with tracks formed using only the EDSRIF, but with perfect associations (see figures 28-1 through 28-3). In these figures, solid lines mark the trajectories from the new software package and the dotted lines mark the "EDSRIF-only" trajectories. The results are quite promising. Even though there were extraneous tracks early on in the simulation, all of the extraneous tracks converged in less than 16 seconds, leaving only three tracks, which corresponds to the number of targets in the simulation. Note how closely the new software system's tracks (solid) matches the "EDSRIF-only" tracks (dotted) once the extraneous tracks have converged. The small difference is due to one using all 14 sensors and the other using only 3.

2.2.4 Computing Submunition Trajectories

Debris including "chem str glass pnl", restraint strap, piston and gas generator are ejected along with each submunition. However, the drag coefficients for debris are much smaller than for submunitions, and we expect that debris will fly past the impact area. Thus, the short range network should only see true targets. Nonetheless, discrimination of submunitions from debris by long range or short range/along track sensors will be necessary

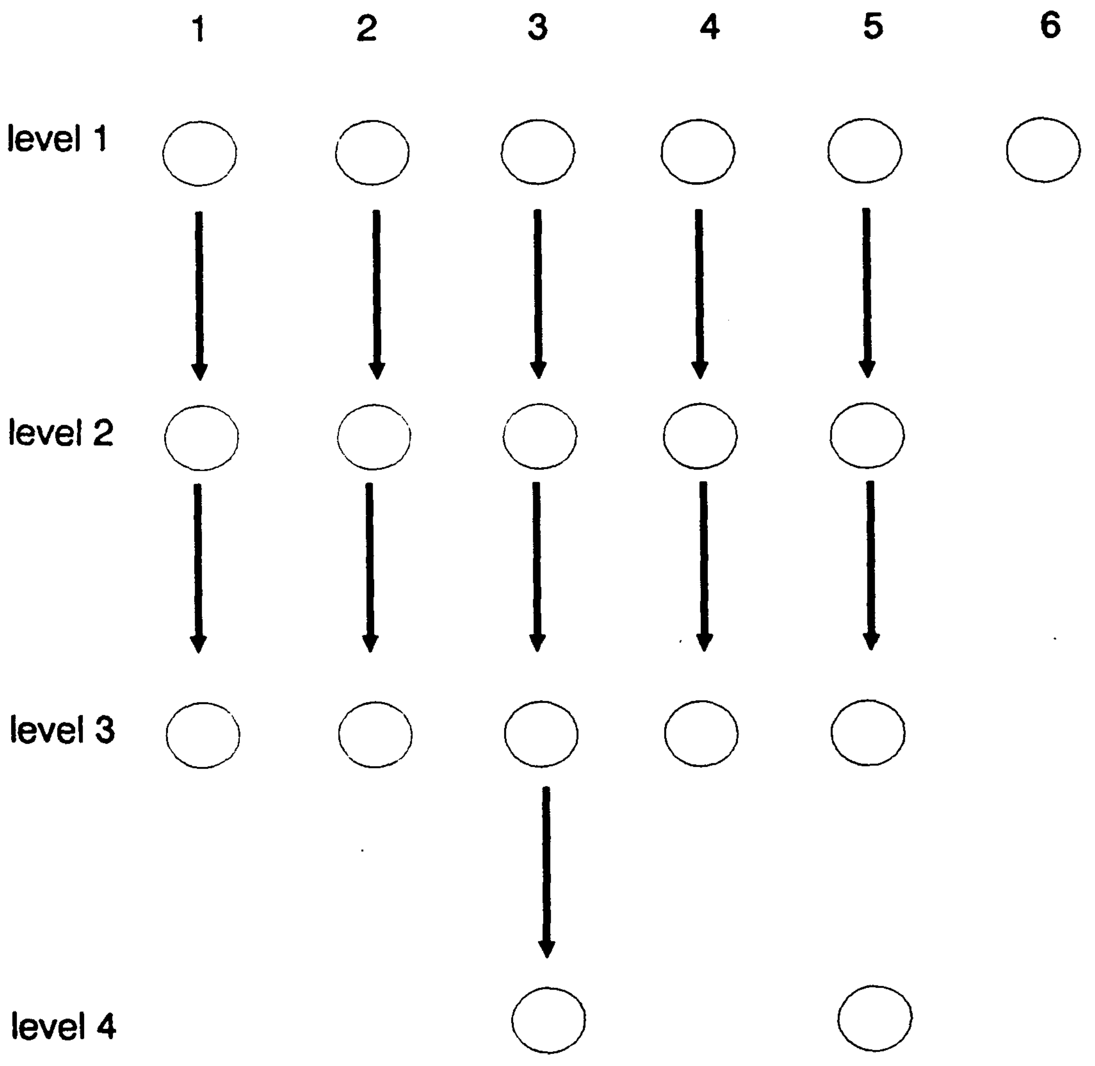


Figure 27: Hypothesis tree generated at step 94.

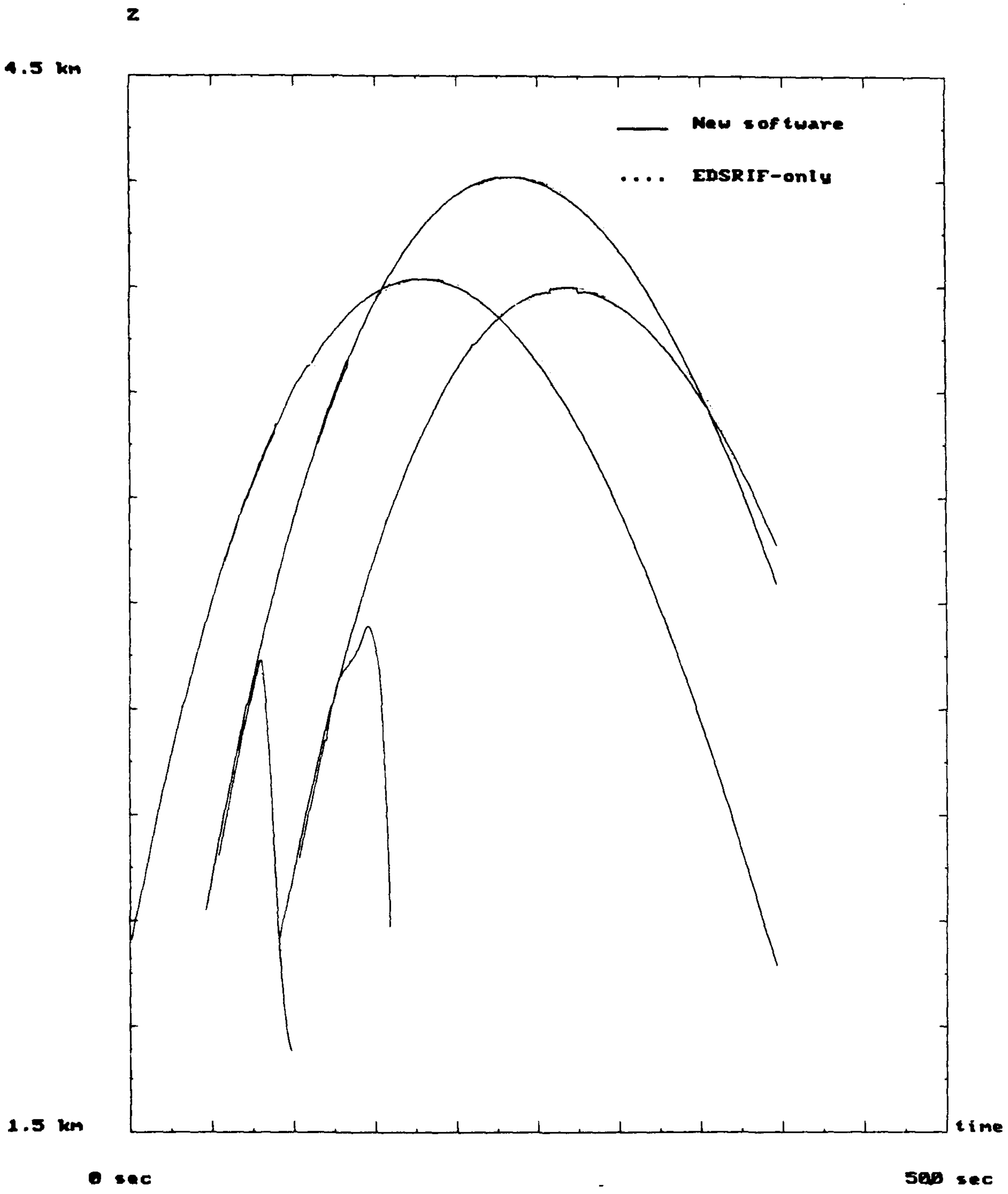


Figure 28-1: Z component of global measurement updated states using the EDSRIF.

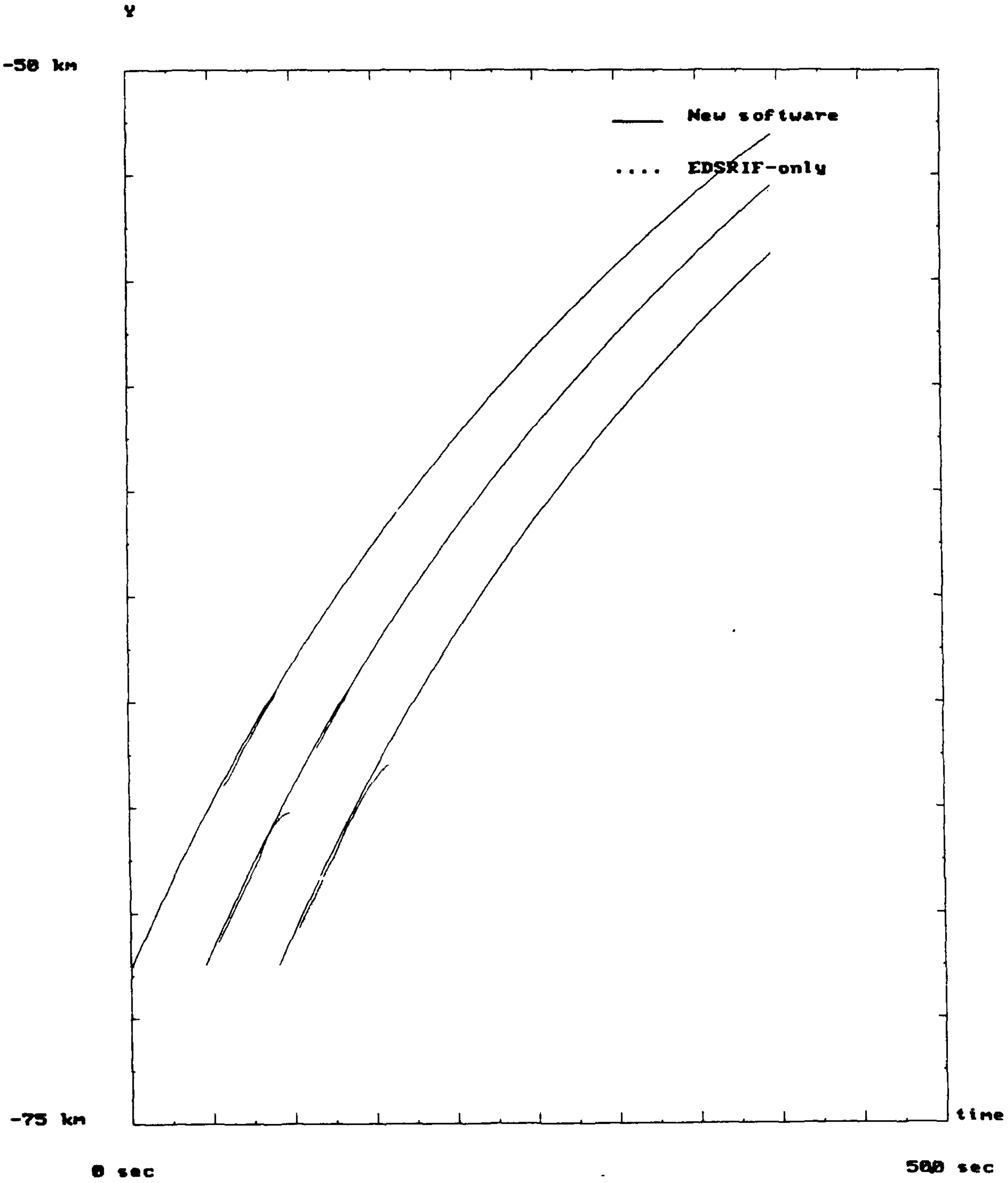


Figure 28-2: Y component of global measurement updated states using the EDSRIF.

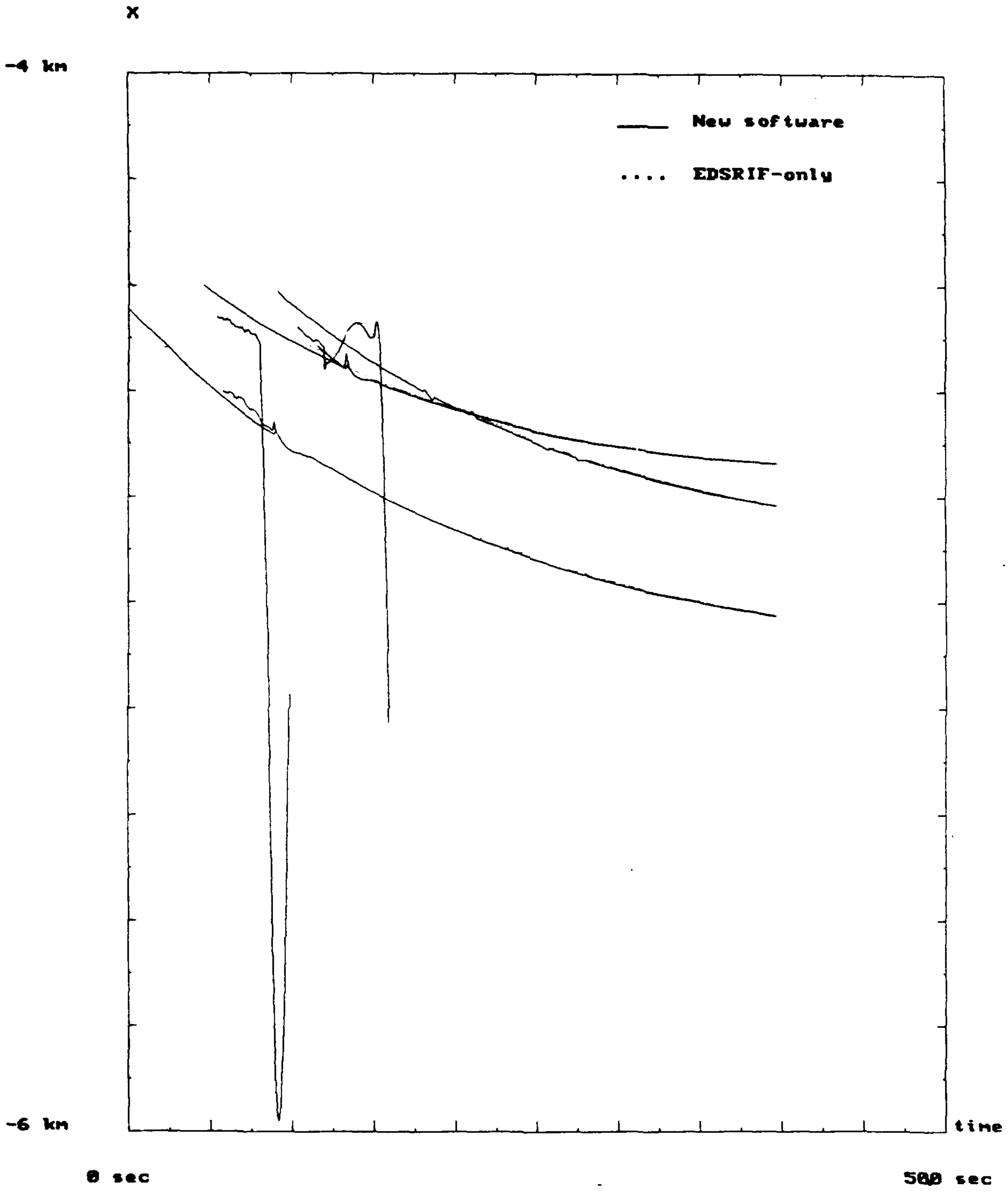


Figure 28-3: X component of global measurement updated states using the EDSRIF.

for submunition tracking during its early stages of motion. The latter problem is under investigation as part of a separate project.

Design of Submunition Trajectories: This simulator generates a set of trajectories for 1 rocket and 6 submunitions in a Cartesian coordinate system. The trajectories are used to create a set of corresponding simulated measurements for processing by the dual network. The following is a brief description of the simulator.

stage 0: This represents the rocket boost phase, during which arbitrary piecewise constant accelerations over arbitrary time periods may be imposed.

stage 1: This represents ballistic motion. At the height of 460 meters, the first 2 submunitions are ejected. Thereafter, 2 submunitions are ejected every 0.05 seconds until 6 submunitions have been deployed. The rocket and submunitions move ballistically following a 3-dimensional constant acceleration kinematic model. The initial conditions for the submunitions are the vector sum of the state vector of the rocket plus the state vector of the submunitions relative to the rocket. Different process noise levels for the rocket and submunitions are used. Also, the process noise levels for the rocket were increased after each ejection in order to compensate for non-aerodynamic motion that is likely to occur.

stage 2: This stage represents ballistic motion of the rocket and deployment of the Ram Air Inflation Decelerator (RAID) for submunitions. The RAID for each submunition is deployed at a user specified time after ejection. The RAID decelerates the submunition and reduces its spin rate. Assuming that frictional force is proportional to the square of velocity, the following system of differential equations was used to model the submunition trajectory after RAID deployment:

$$\dot{x} = -R_x\, x^2, \qquad \dot{y} = -R_y\, y^2, \qquad \dot{z} = -R_z\, z^2 - g$$

Here R_x, R_y, R_z represent coefficients for drag force in each direction due the RAID. The gravitational acceleration is represented by g.

stage 3: In this stage we assume that the rocket still undergoes ballistic motion, and the first and second Orientation & Stabilization (O&S) devices are deployed. The following system of differential equations was used to generate a submunition trajectory for the first part:

$$\dot{x} = -L_x\, x^2, \quad \dot{y} = -L_y\, y^2, \quad \dot{z} = -L_z\, z^2 - g$$

For the second part we have:

$$\dot{x} = -K_x\, x^2, \quad \dot{y} = -K_y\, y^2, \quad \dot{z} = -K_z\, z^2 - g$$

Here L_x, L_y, L_z, K_x, K_y, K_z are drag force coefficients. Thus, the 2 parts differ only in the values of their proportional coefficients. All the parameters are tuned so that submunitions move approximately 5 km downrange from launch, all submunitions land in a 1.1 km diameter circle, and the submunition's rate of descent is approximately 21 m/sec from a height of approximately 150 m.

Figures 29-1 and 29-2 show the trajectories of one rocket and 6 submunitions. Piecewise constant accelerations (10 m/sec^2, 20 m/sec^2, 50 m/sec^2), (25 m/sec^2, 15 m/sec^2, 60 m/sec^2), (25 m/sec^2, 25 m/sec^2, 70 m/sec^2), (35 m/sec^2, 35 m/sec^2, 80 m/sec^2), (45 m/sec^2, 45 m/sec^2, 90 m/sec^2), and (50 m/sec^2, 50 m/sec^2, 90 m/sec^2) are applied for 6 different but equally divided consecutive time periods from 0..sec to 3. sec. The RAID coefficients were chosen to be 2.1E-2, 2.1E-2, and 2.1E-2. 1.3E-2, 1.3E-2, and 1.9E-2 are used for the coefficients of the 1st O&S system. The second O&S system uses 1.5E-2, 1.5E-2, and 2.3E-2 as coefficients. In figure 29-1, a submunition ejection angle of 45° is chosen, and it is zoomed around the ejection point. An ejection angle of 130° is used in figure 29-2. In both cases, 3.0 sec is used as the time period for the RAID deployment.

2.2.5 Methods for Processing Submunition Data

In the remainder of this section, several methods for tracking maneuvering targets within an E-DSRIF framework are considered. Tracking SADARM submunitions during deployment requires sophisticated algorithms for maneuver detection because the submunition's speed changes from approximately 300 m/sec to approximately 20 m/sec over a very short time interval. The following methods were considered:

Method 1: In this method maneuver detection is not required. At each iteration, the process noise covariance is updated according to

$$Q_k = c\, P_k(-)$$

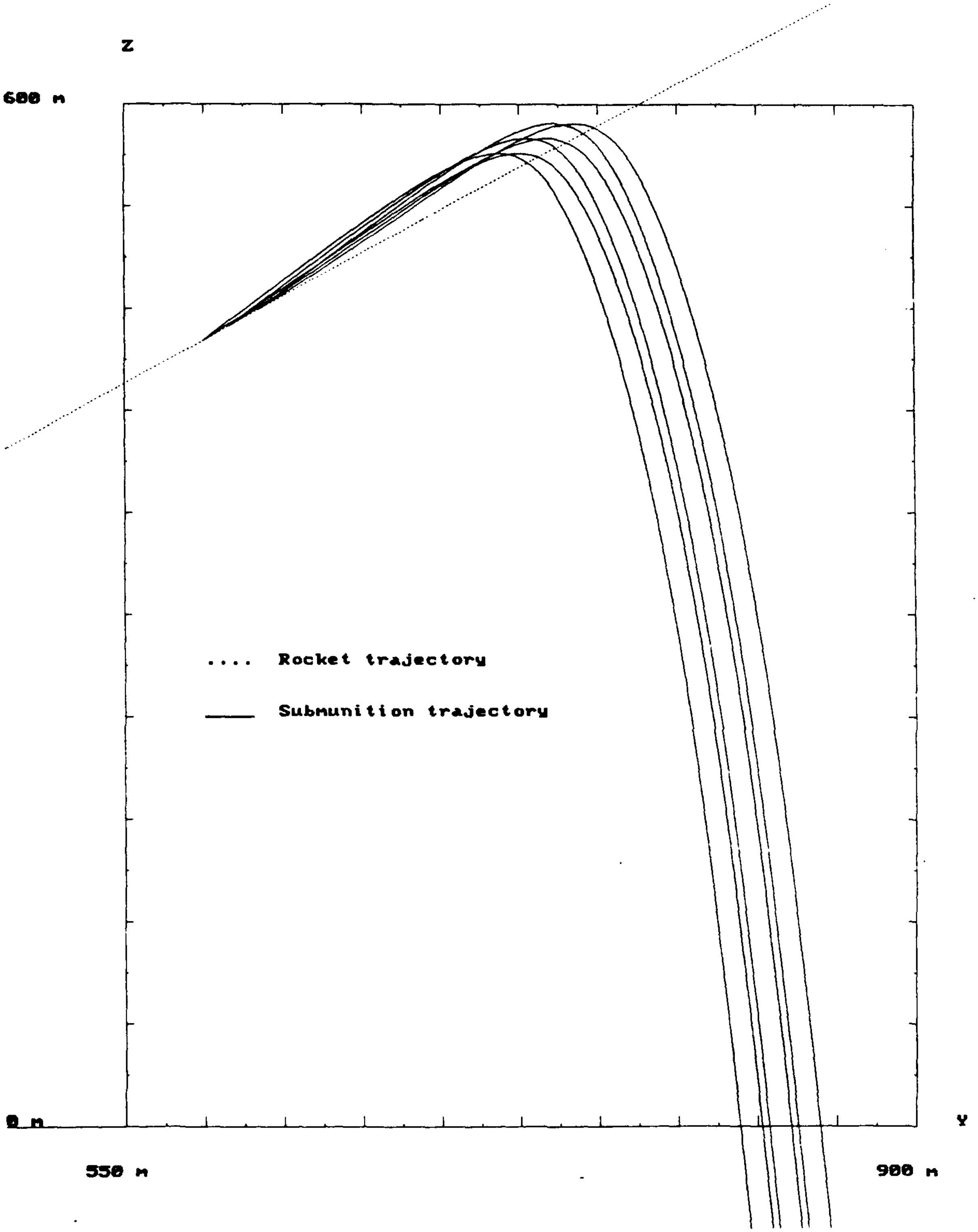


Figure 29-1: Submunition trajectories in Y,Z coordinates (45 degree ejection angle).

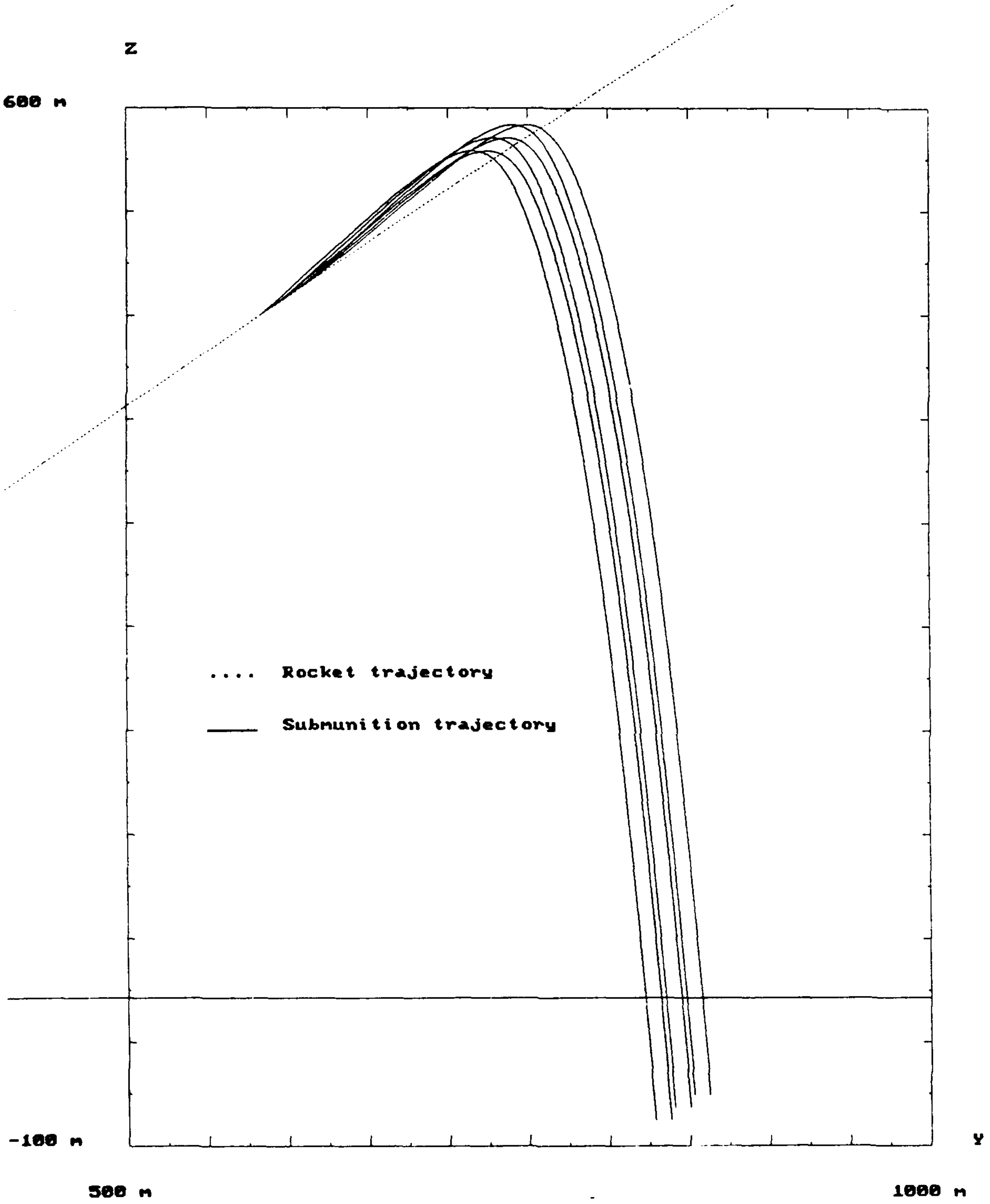


Figure 29-2: Submunition trajectories in Y,Z coordinates (130 degree ejection angle).

where Q_k and $P_k(-)$ are the process noise covariance and predicted state estimate error covariance respectively, and c is the adjustment factor which should be chosen based on the dynamics of the maneuver [13].

Method 2: Once a maneuver is detected by examining the residual or innovations process, the process noise covariance is adjusted using any one of several well-known methods for adaptive Kalman filtering. More specifically, the residual at the iteration is obtained by

$$\pi_k = y_k - H_k \, x_{k-1}(-)$$

where y_k is the measurement, $x_k(-)$ is the predicted state estimate and H_k is the measurement matrix. The residual covariance matrix B is given by

$$B^{-1} = H_k \, P_{k-1}(-) \, H_k^{tr} + R_k$$

where $P_{k-1}(-)$ is the covariance associated with $x_{k-1}(-)$, and R_k is the measurement noise covariance. Define:

$$l_k = \pi_k^{tr} \; B^{-1} \; \pi_k,$$

then, it is well known that l_k is a Chi-square random variable under the assumption that π_k forms a Gaussian distribution. Maneuver detection follows from the change in l_k (and especially from the increase in l_k). An increase in the process noise covariance levels should be performed so that l_k is reduced [8],[9].

Method 3: Using the Dyer-McReynolds smoothing coefficients, one can obtain the one step smoothed value of the process noise. The smoothed value is given by

$$w_{j|j-1}^{*} = [R_w^{*}(j)]^{-1} \, (z_w^{*}(j) - R_{wx}^{*}(j) \, x_{j+1|j})$$

Changes in $w_{j|j-1}^{*}$ which exceed a threshold might be considered to signify a detected maneuver. The process noise covariance levels should be adjusted accordingly [10].

Method 4: Among the several multiple model approaches, the Interacting Multiple Model (IMM) method may provide improved performance (over residual monitoring with adjustment of the process noise covariance). The algorithm of this method can be found in [11].

To gain experience, methods 1 and 2 were integrated with a SRIF and tested. Following an example from [6], a 2-dimensional constant velocity motion with sudden maneuvering is considered. The sampling period is 10 seconds and it is assumed that the initial state is

$$X(0) = [\ x(0)\ \ x(0)\ \ y(\dot{0})\ \ y(\dot{0})\ ] = [\ 2{,}000\ \ \ 0\ \ \ 10{,}000\ \ \ -15\ ]$$

where position and velocity are measured in meters and meters/second, respectively. A piecewise constant acceleration input

$$a_x = a_y = .075 \text{ m/sec}^2$$

is applied to the target over the interval between 400 and 600 seconds. The dotted curve in figure 30 is the corresponding trajectory. The solid curve in figure 30 represents filtered state estimates using the SRIF. The corresponding rms position estimate error is given in figure 31.

Next, we incorporated the adaptive filtering method into the SRIF as per the formula described in METHOD 2. The solid curve in figure 32 represents filtered state estimates for the SRIF and the rms position estimate error is given in figure 33.

Finally, we applied the continuous updating of METHOD 1 to the same test trajectory and obtained the filtered state estimates, depicted as a solid curve in figure 34. Figure 35 shows the position rms estimate error.

The following observations are made. The continuous updating method is to update the process noise covariance matrix at each iteration. The updated values depend upon a scale factor and the measurement updated globally optimal estimate error covariance matrix. The scale factor determines the performance of the method but we do not yet have a systematic method for determining its value. This is one disadvantage of the method.

The adaptive filtering method is to update the process noise (to a new but fixed value) only when a maneuver is detected. The decision criterion is the norm of the residual vector exceeding a threshold value which, like the scale factor, requires careful adjustment. Also, the initial choice of the process noise covariance will affect tracking performance.

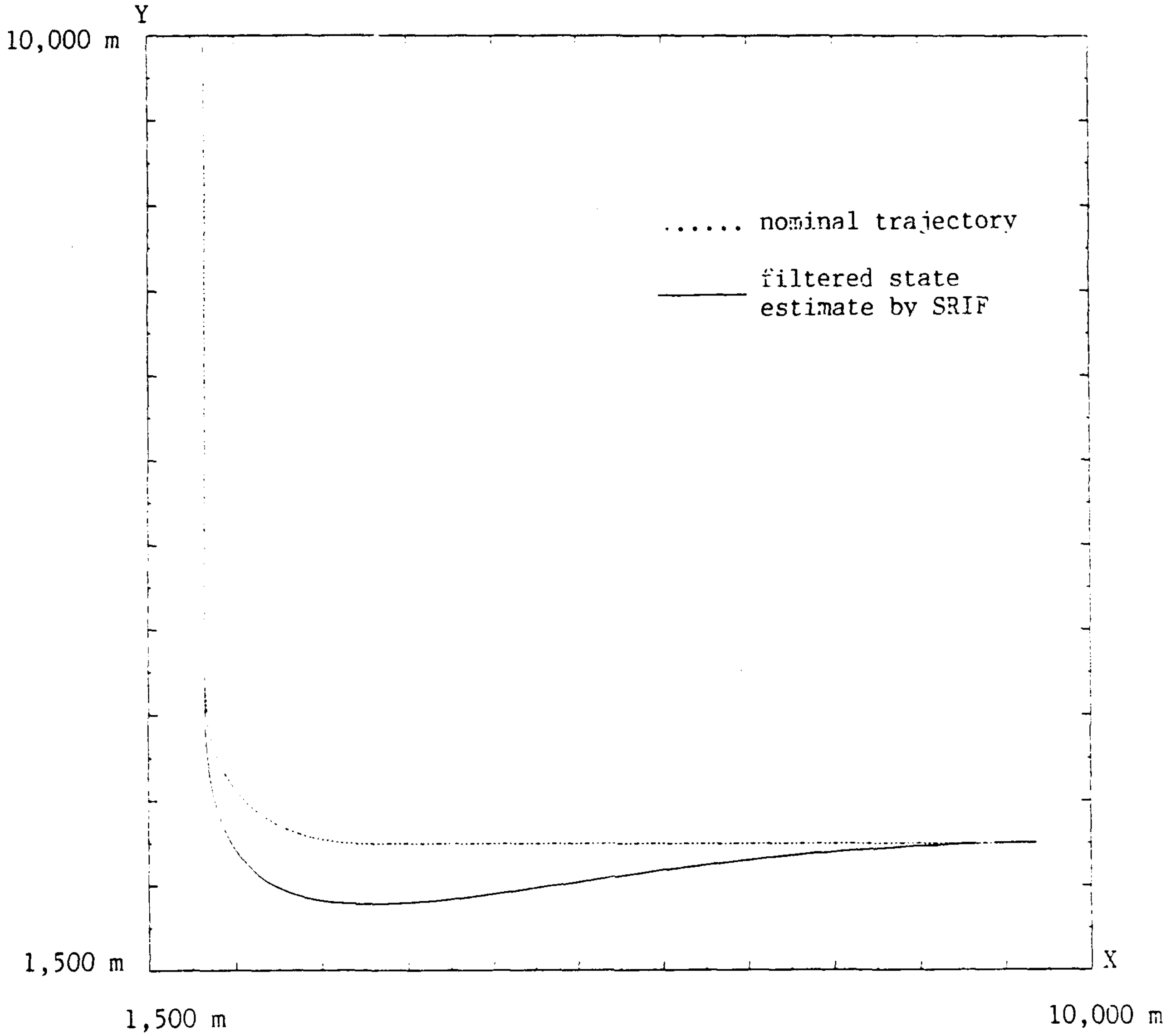


Figure 30: Nominal trajectory and measurement updated states in X,Y coordinates using the SRIF.

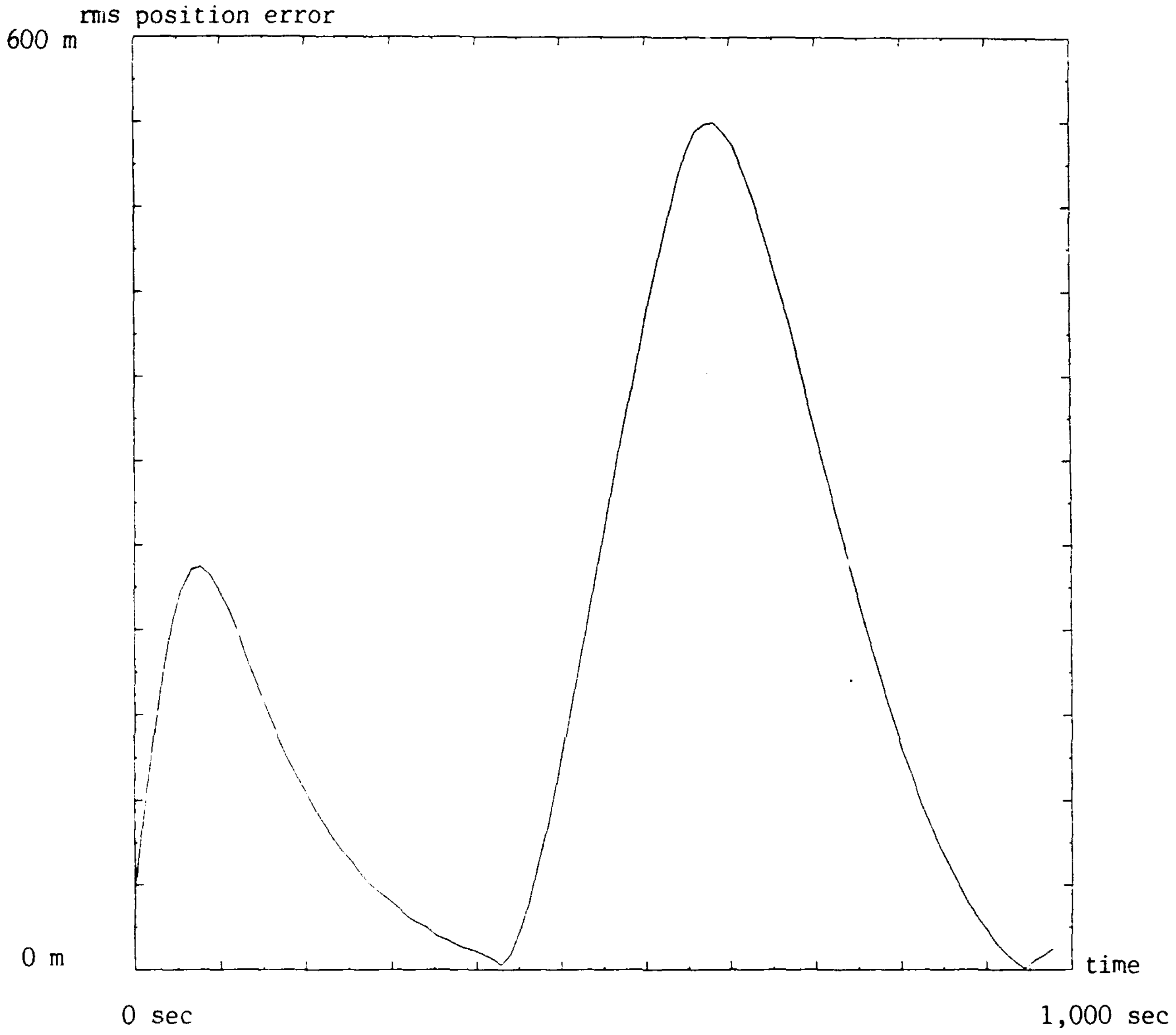


Figure 31: RMS position error for figure 30.

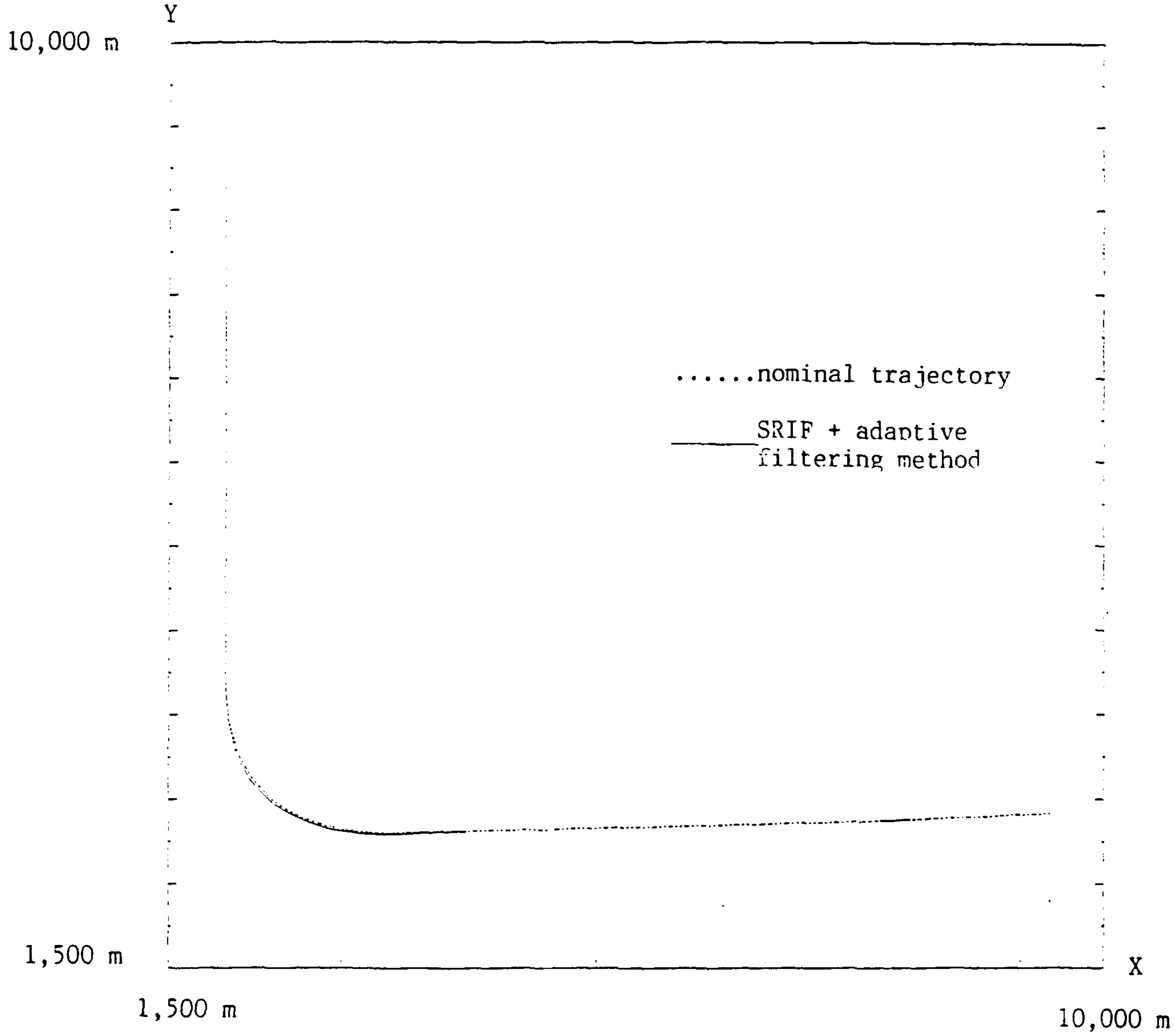


Figure 32: Nominal trajectory and measurement updated state in X,Y coordinates using an adaptive SRIF.

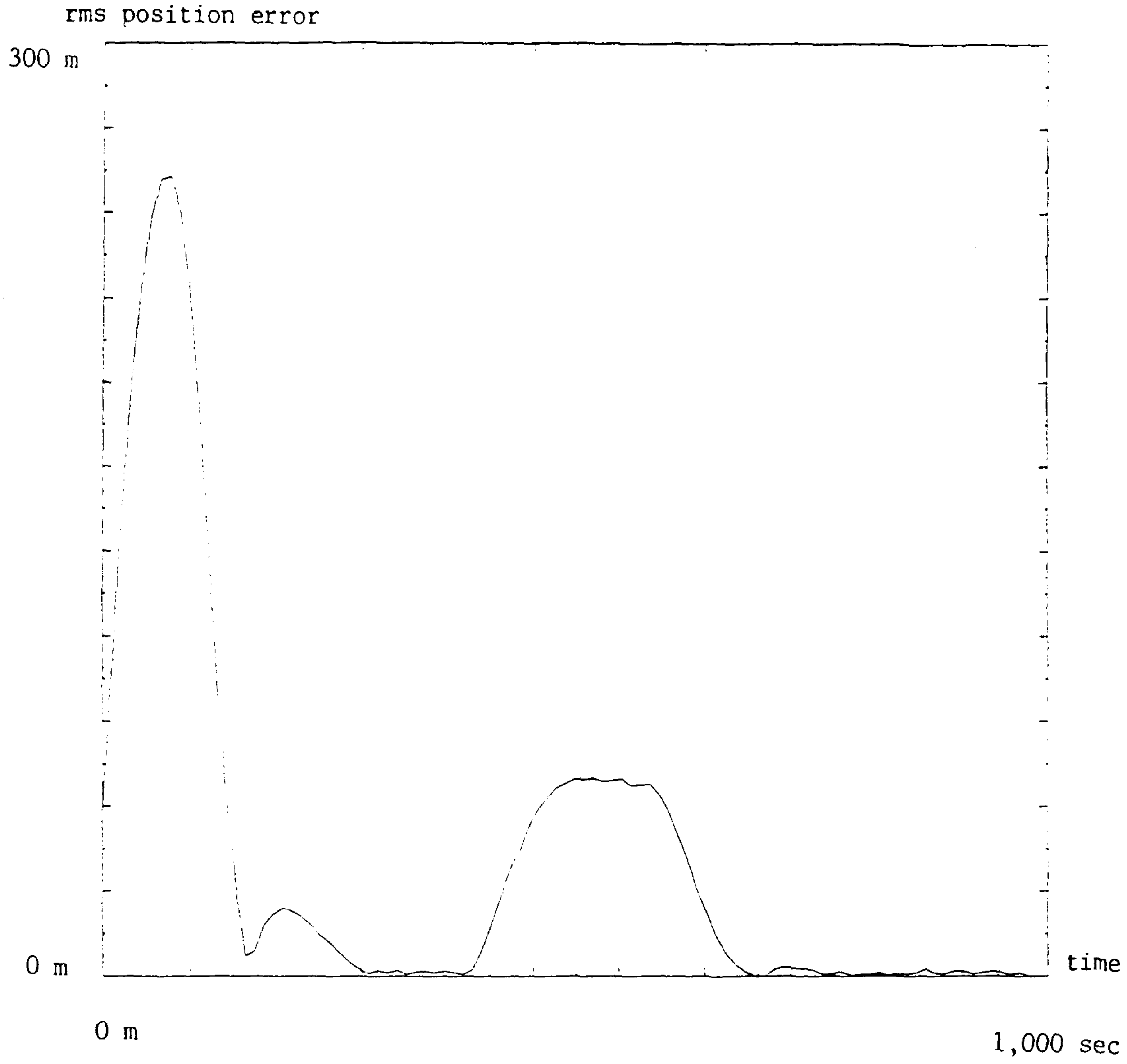


Figure 33: RMS position error for figure 32.

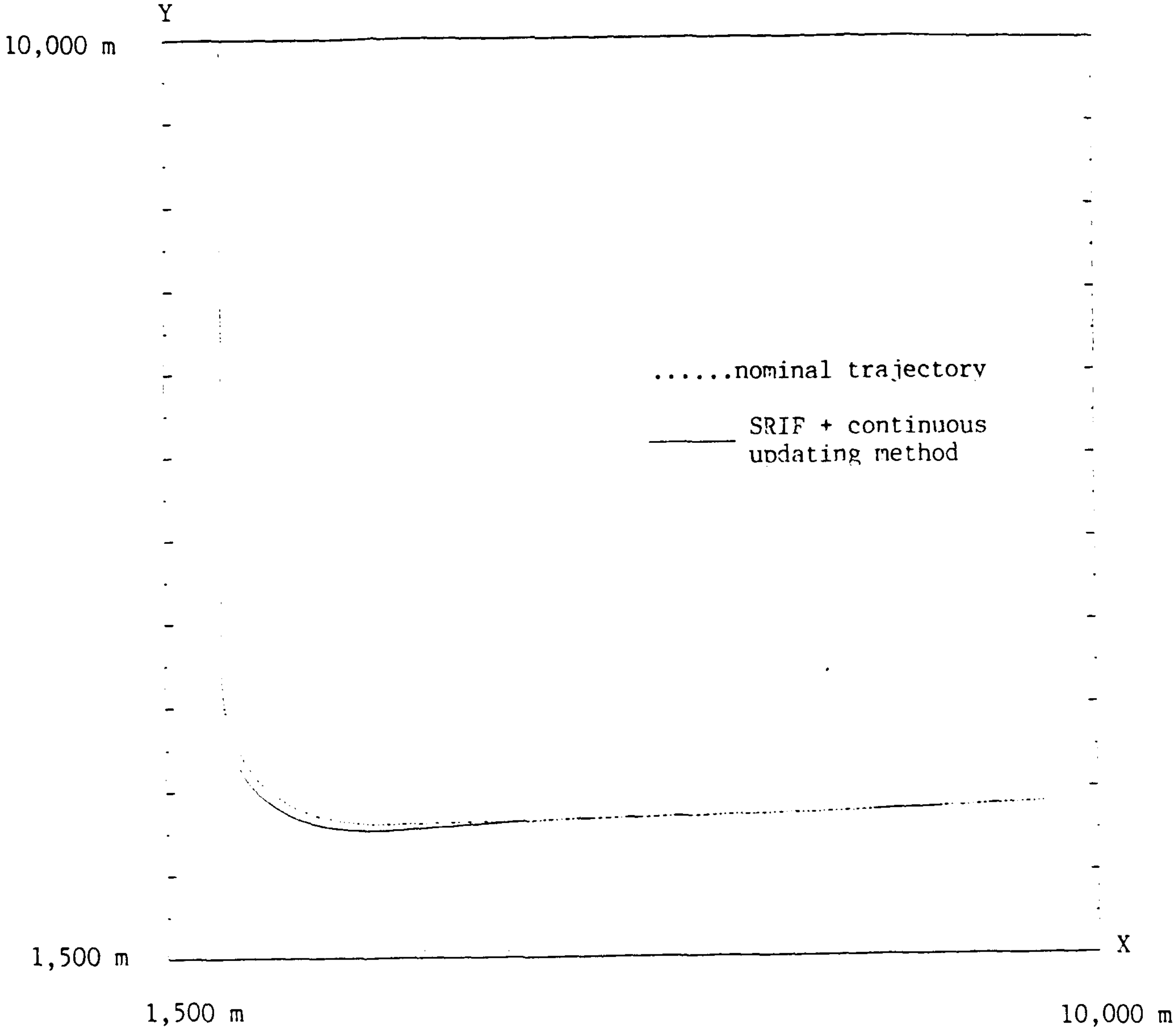


Figure 34: Nominal trajectory and measurement updated states in X,Y coordinates using a continuous updating SRIF.

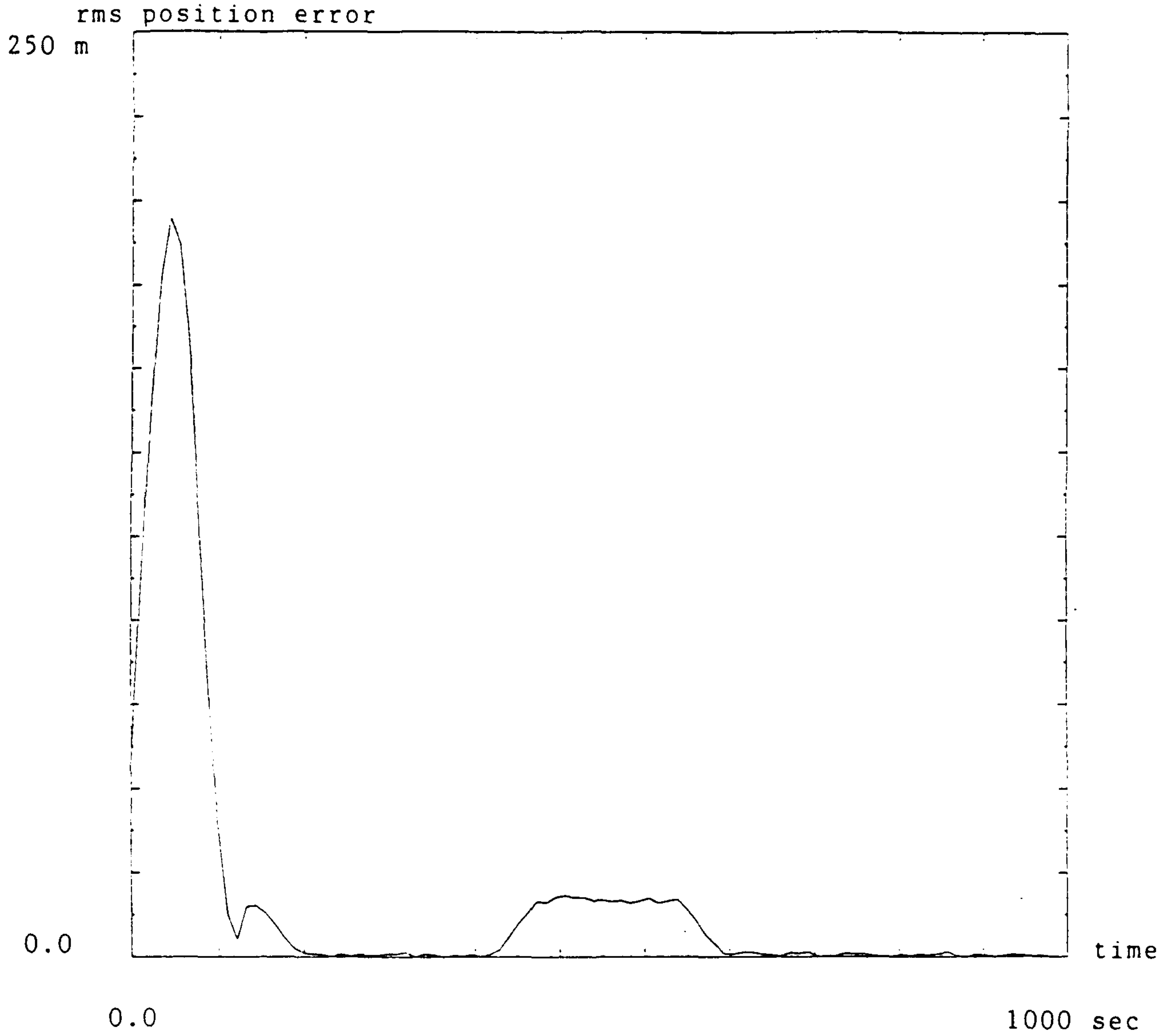


Figure 35: RMS position error for figure 34.

This is seen in the following well-known equation from [8]:

$$P_{k-1}(-) = \Phi_{k-1} P_{k-1}(+) \Phi_{k-1}^{tr} + Q_{k-1}$$

The initial choice of the process noise covariance matrix, Q, affects the predicted state error covariance, $P_{k-1}(-)$. Also, the residual covariance matrix, B, depends on the predicted state estimate error covariance via the equation (4). Since the norm of the residual is given by (5), tracking performance depends on the initial choice of the process noise covariance.

Nearly optimal parameters for each method were obtained by making repeated simulation runs. Figure 36 shows the position rms estimate errors for each method when using the nearly optimal parameters.

2.3 Design and Testing of a Specialized Processor for Integrated Tracking

Computational throughput is one of several keys to improved tracking performance. Our approach to increased throughput is to distribute the processing of data over a network of inexpensive microcomputers. However, inspection of the DSRIF algorithm shows that computation may be parallelized even within each LP and the GP. To this end, MTI entered into a collaborative arrangement with Space Tech Corporation, Ft. Collins, Colorado, who has been working on the design of a multiboard set for a similar application since approximately 1987. Much exchange of ideas between MTI and Space Tech was made, and we converged to a final design which is being manufactured by Space Tech subcontractors in small quantities. We hope to receive the multiboard set for testing and integration with our laboratory experiment.

Most of our discussions with Space Tech involved efficient architectures for implementing Fast Givens Rotations (which are free of expensive square root calculations) and Householder transformations, since matrix orthogonalization is the major mathematical operation in the DSRIF. We especially looked at GP architectures since the matrices are much larger than those for LPs, and are already block-wise upper-triangular.

Also, Space Tech promised to provide a DT-Connect interface for preprocessing of the video data, and an EISA bus for compatibility with MTI's "486" computer.

2.4 A Dual Camera Tracking Experiment

A "2-element" cluster (figure 37-1-d) within the short range network was built and tested. Each element contains a single video sensor mounted on a 2-axis gimballed platform. Figure 37-2 is a block diagram of the dual camera network. Each camera records

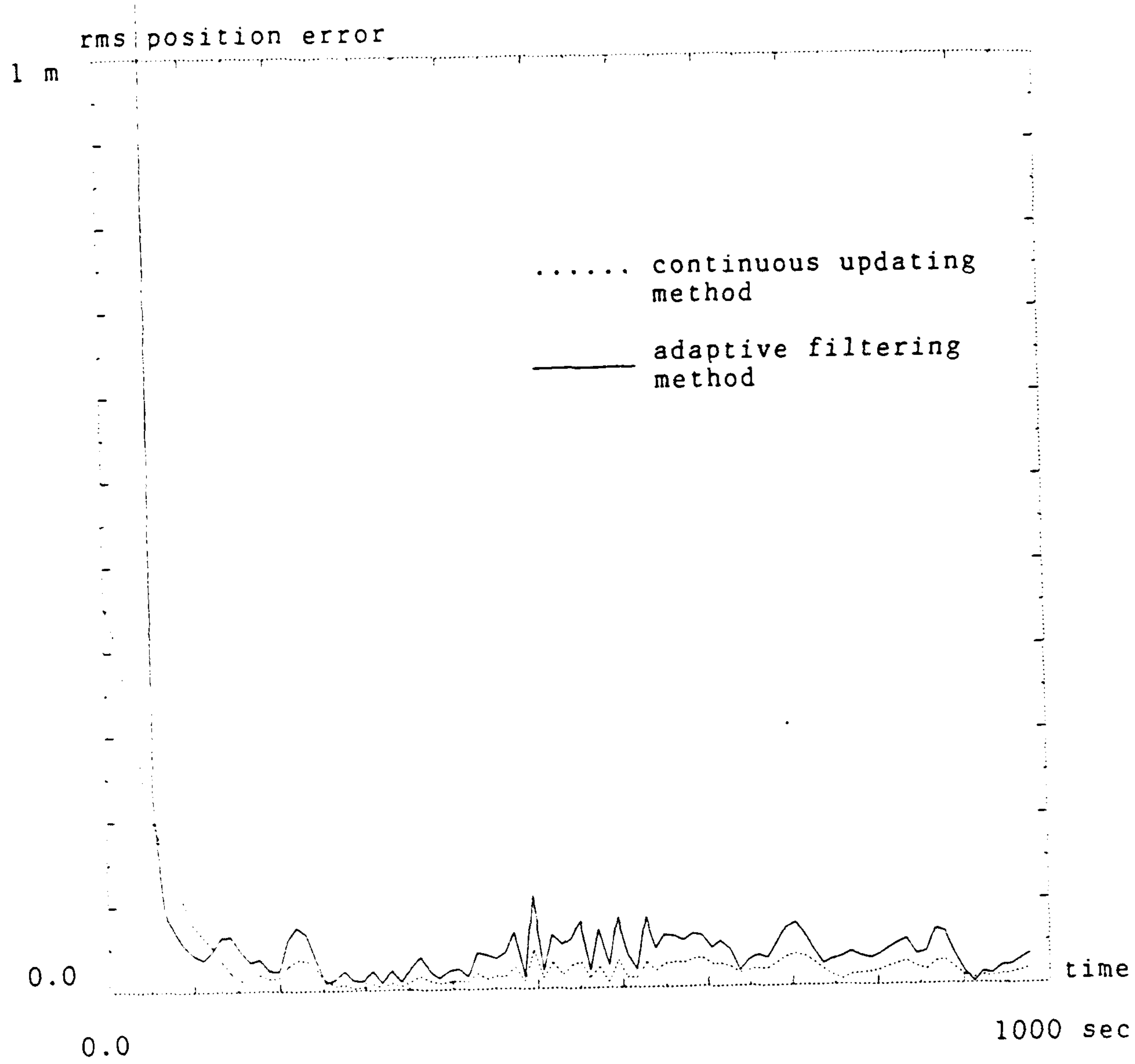
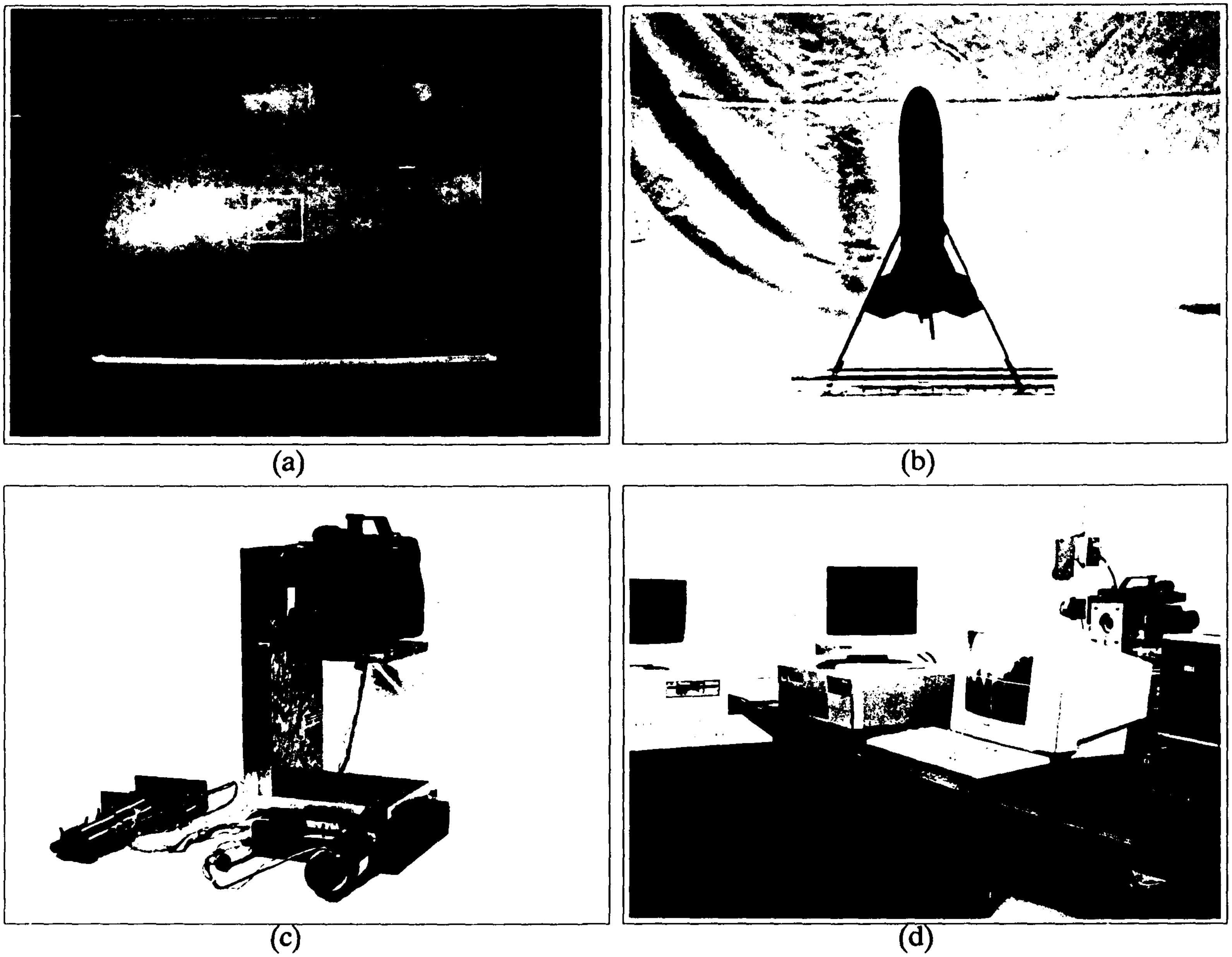


Figure 36: RMS position error for continuous updating versus adaptive SRIF.



(a)

(b)

(c)

(d)

Figure 37-1: Video Tracking System plus targets.

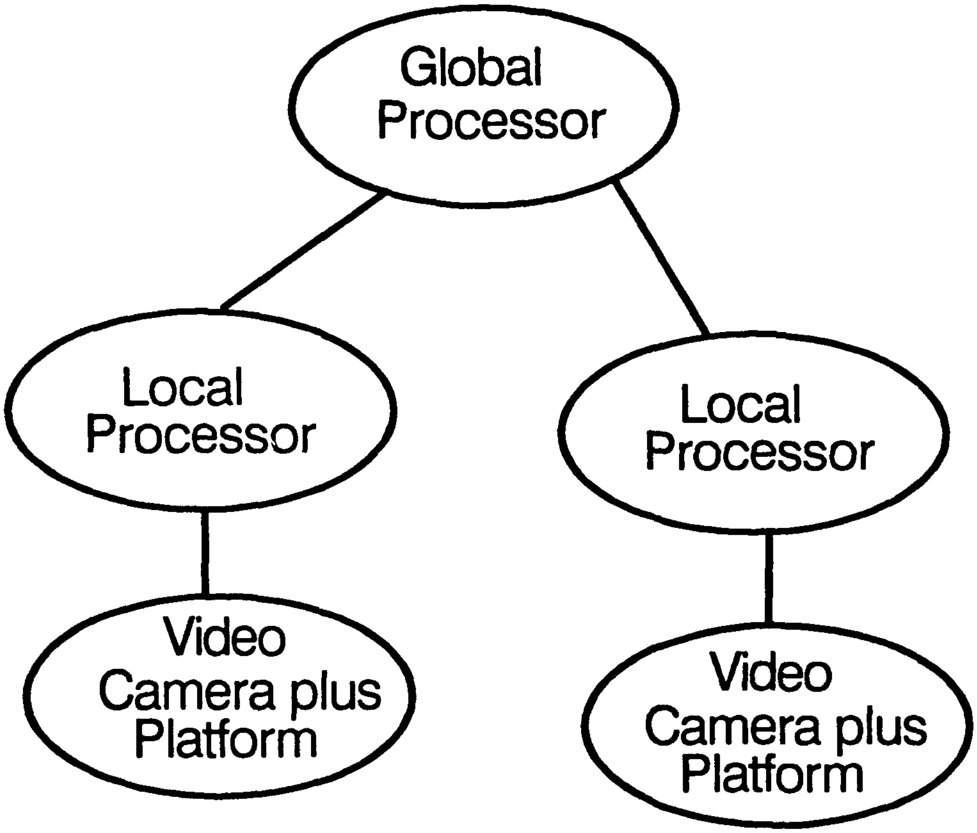


Figure 37-2: A dual camera cluster.

images within its respective field of view (adjustable from 5 to 40 degrees) at 30 frames per second. With the camera fixed, 2 consecutive frames are grabbed and then subtracted pixel by pixel. This eliminates common background noise. Each local processor then computes the centroid of its subtracted image and generates a new position feedback command for repointing its platform. Separate dc servo motors control platform azimuth and elevation, but both axes of rotation are controlled by a single board motor controller. After repointing, the target's centroid is nearly at the origin of the focal plane's coordinate system which has been preset to the center of the camera's field of view.

Various factors contribute to the observed offset between target centroid and focal plane origin. They are, in order of decreasing magnitude, a non-zero target velocity, mount rotational axes being noncoincidental with focal plane axes, finite resolution of the shaft encoders, motor coupling slippage, and variation of the mount's axes of rotation with respect to the manufacturer's specification. By far, the greatest contributor to repointing error is the movement of the target between frame subtraction and repointing of the platform. Filtering, and prediction based upon the filtered trajectory will greatly reduce this source of error in future work. Also, incorporation of velocity feedback into the motor control loop will greatly increase the maximum angular track rate of an individual element. Assuming a constant velocity trajectory, grabbing frames while continuously repointing the platform along the target's predicted velocity vector will prevent loss of target at the second frame. For uniformly accelerating targets, incorporation of velocity and acceleration estimates into the control loop will prevent track loss in a similar manner.

The remaining sources of error are relatively small and except for the second source, were neglected. The fact that mount rotational axes were not coincidental with the focal plane axes, became an issue when the ability of the cluster to coordinate target acquisition was tested. The azimuth and elevation of each platform is continuously transmitted over the network from local to global processor, and there displayed in real-time. When only 1 element is actively tracking, the global processor computes and transmits azimuth and elevation commands to the non-tracking element. Upon command by the user at the global processor's terminal, the non-tracking element will repoint using the global command, acquire and begin to track the common target. The global processor computes an equivalent azimuth and elevation in the local coordinate system of the non-tracking element, assuming a predetermined range. Thus, the "tracking volume" is an arbitrarily thin "spherical shell". Additional code which computes a full azimuth/elevation search path based upon a preselected range gate, was written but neither installed on the global processor nor tested. In this case, the "tracking volume" is a spherical shell whose thickness is equal to the range gate.

For coordinated target acquisition at very short ranges, calculation of the equivalent azimuth & elevation was also based upon geometric models of platforms and sensors. The local coordinate system of each element is defined as the mount rotational axes plus their cross product as vectors. The geometric model is a coordinate transformation which includes the displacement and orientation of the focal planes with respect to the local coordinate systems, as well as the displacement and orientation of the local coordinate systems with respect to each other. Thus, a vector drawn from the origin of 1 local coordinate system to a target, can be transformed and added to the local coordinate systems' displacement vector, in order to arrive at the vector from the origin of the other (non-tracking) element to the same target.

Laboratory and field testing of an individual element, and laboratory testing of the "2-element" cluster was carried out. The 80486 based element (figure 37-1-c) was field tested in an open area near our Rockville office. A gasoline powered generator, placed 100 feet from the instrumentation, was used to provide 120 volt, 60 cycle power. Standoff of the generator helped to reduce its rf interference with the instrumentation display. The launch area was 165 feet from the tracker. Two types of targets were assembled and used for the test. First, circular shaped balloons, filled with helium, were tethered from the launch area. The balloons were essentially 2-sided with one side being highly reflective (and specular) and the other side very dark. The distance between the balloons and the ground was approximately 50 feet, but wind moved the balloons around by as much as 20 to 30 feet from the tethered position. Once the tracker was manually pointed so that the balloon appeared within the tracker's central window, automatic tracking was initiated. In this mode, the balloon was successfully tracked (against a cloudy background sky) for many minutes, and without loss (figure 37-1-a). Then, the same experiment was repeated but with 5 of the same balloons (for more upward movement) tied together and released from the tethered position. Again, the group of balloons was tracked without loss, but only until the group appeared out of range (at which point its image was only a few pixels).

Secondly, a variety of small rockets were assembled and launched, one at a time. The 2 types were the "Alien" (figure 37-1-b) and "MTI Special" which was designed by MTI staff to closely resemble the SADARM submunition. Both types were set for launch to approximately 400 feet vertical, at which point a parachute is deployed and the rocket falls back to ground. We were unsuccessful in launching the "MTI Special" as the fuselage separated from the rocket body at lift-off. However, the "Alien" was successfully launched 5 out of 5 attempts. For 2 of the shots, we were unable to manually acquire the descending rocket at all. For 2 of the shots we were able to view the descent but only outside the central window. Finally, for the last shot we were successful in manually acquiring the descending rocket. At this point automatic tracking was initiated, and the target was tracked for approximately 9 seconds until impact (figures 37-3-a through 37-3-d)! A video tape of our field test results was delivered to WSMR for their review.

Testing of the "2-element" cluster was carried out in our laboratory at MTI. Its floor dimensions are 30 by 30 feet with a ceiling height of 10 feet. The "80386" and "80486" based elements were bolted to a 3 foot by 8 foot table top at opposite ends of the table. Each tracker's base was positioned flush with the edge of the table. Thus, the cluster's topology is similar to a pair of human eyes providing binocular vision. A variety of targets were successfully tracked. Most were hand held at the end of a hangar and led around the room. Even goldfish in a 3 cubic foot tank were tracked. Of course, loss of track could be

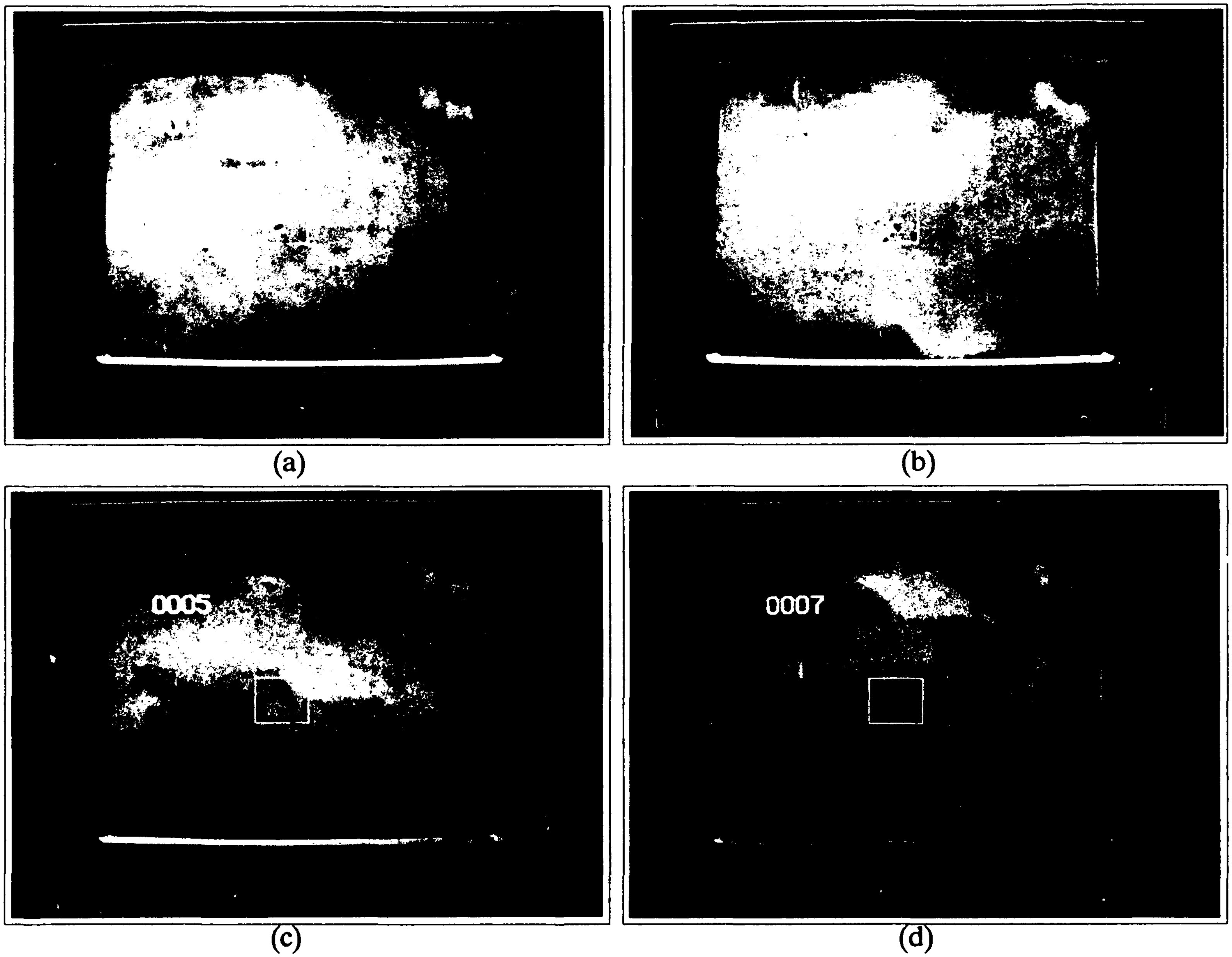


Figure 37-3: Tracking the "Alien" Rocket.

achieved by moving the target too quickly, but generally speaking, the cluster performed well and was able to coordinate its acquisition from any point in the room.

Elements which contain multiple sensors on a single platform are a natural evolution of our Phase II work. Small radar hand guns can provide range rate measurements at short range. In combination with video, the SRIF could yield reasonable estimates of range after only a few accurate measurements are processed. In this case, small initial estimate errors are needed for rapid convergence of the filter.

Figures 38-1 through 38-3 show the configuration of video tracking system and functional block diagram. In the remaining paragraphs, details about the design and operation of the cluster are provided.

2.4.1 Tracker Hardware

Each element was built using the following components:

Servo Motor Controller (Technology 80, Model 5638): It is an IBM PC/XT/AT-compatible, digitally sampled servo controller card, offering up to three axes of servo control. It performs the time-intensive computations required for closed-loop digital motion control, freeing the host computer for other tasks. The Model 5638 generates an analog voltage from an on-board 12-bit DAC to drive the amplifier.

Servo Motor Power Amplifier (Technolcgy 80, Model 6410): It is a dual axis DC servo motor amplifier capable of delivering 60 volts and -7 to +7 amps of continuous current to each motor. The amplifier accepts a -10 to +10 volt input signal which produces a corresponding amount of current in the motor. Highly efficient, power FET switching H-bridges are used to supply current to the motor.

Power Supply (Electrostatics, Model 400-28): It is an all-silicon, regulated supply with 0.01% line regulation, 0.1% load regulation, 500 microvolts maximum ripple, foldback current limiting, and of small size and weight.

DC Servo Motor (EG&G, Model ME2620-315B): It is a brush motor with an 8,000-hour brush life at 1000 rpm. It contains a shielded precision ball bearing rated for a 40 lb side load. Torque output is a direct function of current applied, while speed is a direct function of the voltage level. Its maximum torque is 54 oz-in with 4.5 amps and maximum speed is 7000 rpm with 60 volts.

Optical Encoder (EG&G, Model 1DM-1000-5L37A5): Its resolution is 1,000 pulses per revolution, however the servo controller can count 4 times for each encoder cycle. Thus, with quadrature signals, the effective resolution becomes 1/4,000 of a revolution. The servo controller keeps track of the motor position by incrementing or decrementing its actual

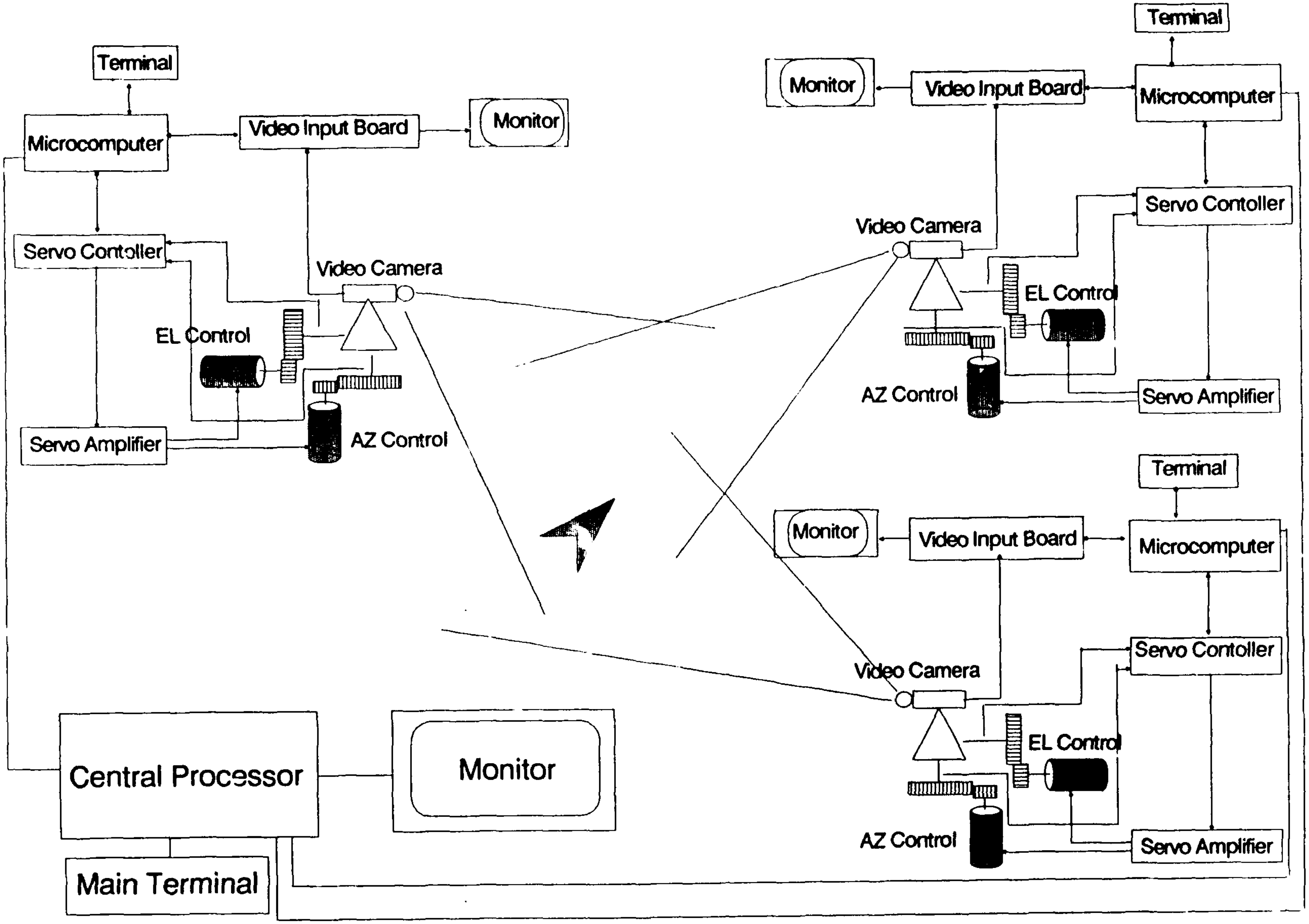


Figure 38-1: Configuration of the video tracking system.

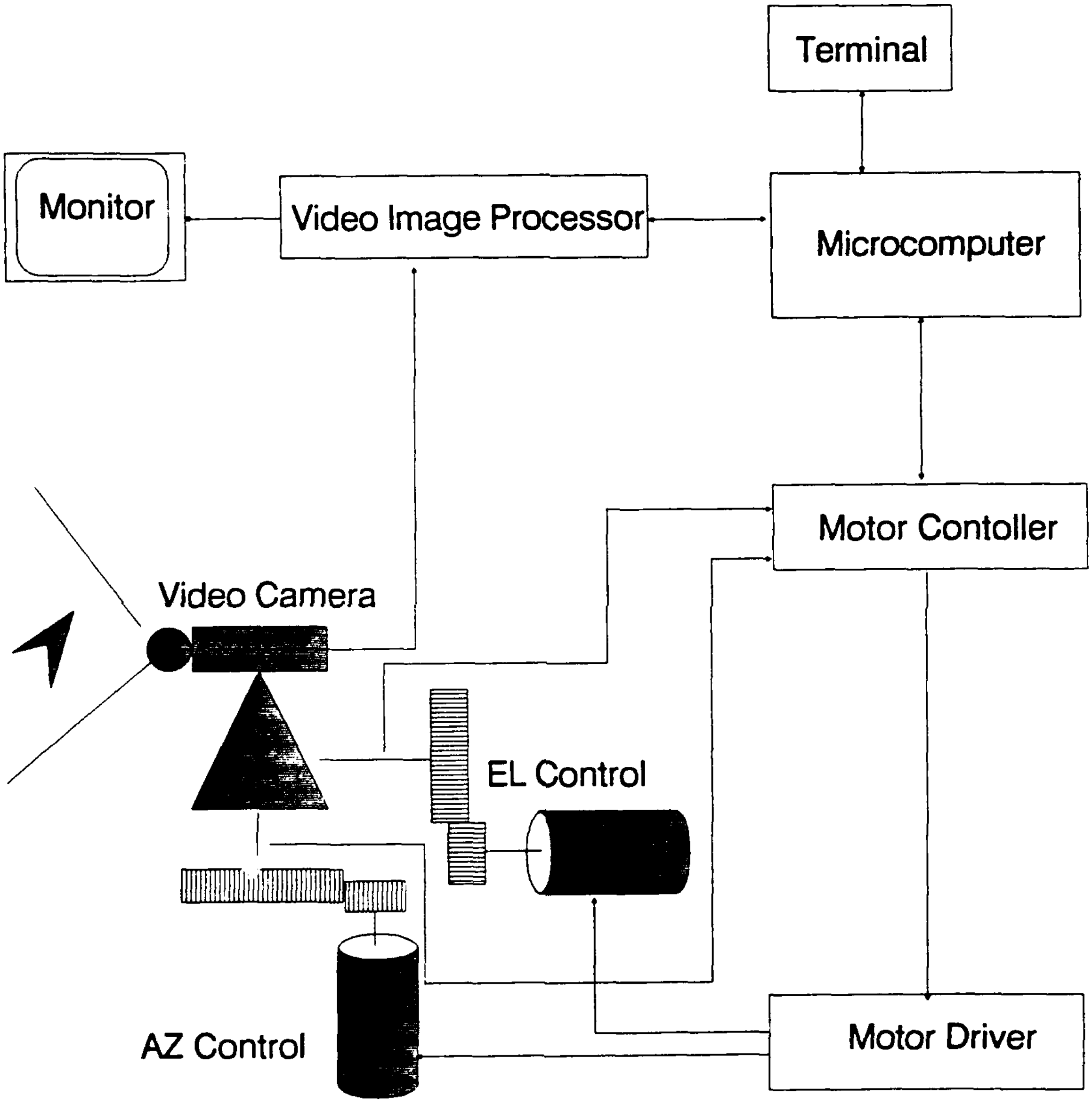


Figure 38-2: Functional block diagram of the video tracking system.

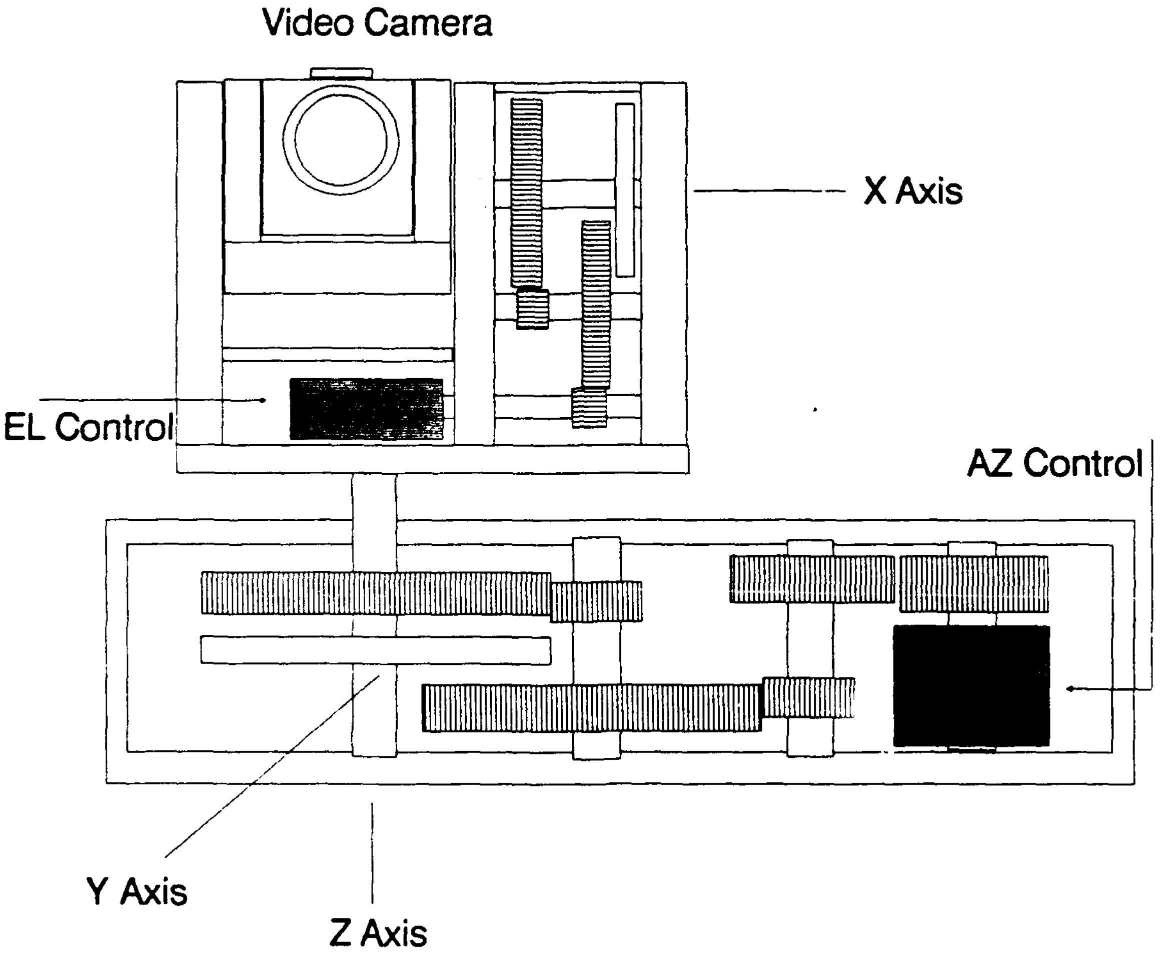


Figure 38-3: Diagram of the motor controller.

position counter accordingly.

Rotary Table (Daedal, Model 208-01-SH): It is designed for precise motor driven rotary positioning and indexing. It is low in profile and high in precision. Two rotary tables are used per tracker, 1 for azimuth (10 inch and 8 inch diameters) and 1 for elevation (6 inch and 5 inch diameters). It is stiffly pre-loaded to minimize run out, while producing smooth rotation of the table top. A precision worm gear drive provides precise point-to-point bidirectional accuracy and repeatability. A 180:1 gear ratio provides added flexibility in establishing table speed and resolution when using a servo motor to drive the table.

Mounting Brackets (designed and built by MTI): Descriptions are provided in figures 39-1 through 39-7.

Video Cameras: Magnavox Moviemaker VHS camcorder, Panasonic VHS-C Palmcorder.

Display: Magnavox 8CM873 multimode color display

IBM Compatible PCs: One "80386" motherboard, and one "80486" motherboard designed and built by MTI.

Frame Grabber (Data Translation): The DT-2851 is a 512 x 512 x 8-bit frame grabber well suited for real-time digital image processing. The DT-2851 has 512 Kbytes of on-board frame-store memory. This memory is used as two screen buffers. Each buffer has a resolution of 512 x 512 x 8-bit per frame.

Network Card (AT&T Starlan)

2.4.2 Tracker Software

The image processing part of the tracker uses a Data Translation DT-2851 high resolution frame grabber residing within a 12 MHZ IBM AT compatible computer. The video images from the CCD camera are uniformly sampled at the rate of 30 frames per second.

At first, we developed software which subdivides each buffer into four quadrants and then sums the intensities of every pixel in the same quadrant. The intensities were thresholded so that all values above threshold are set to 1 while all values below are set to 0. This speeds up the quadrant summations, and was perfectly acceptable when tracking our nominal target simulator (a flat circle moving on a computer screen with a programmable trajectory). With thresholding, approximately 9 frames per second can be processed into 4 quadrant intensity values and passed to the servo-motor control board for tracking.

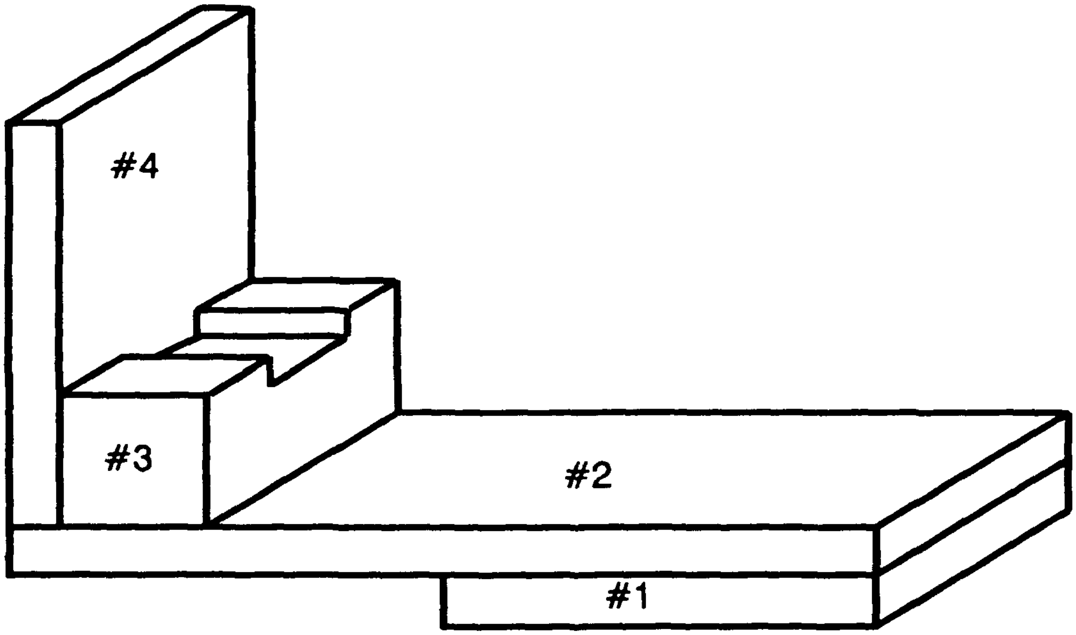


Figure 39-1: Mounting brackets.

PIECE #1

Thickness = 3/4" ALUMINUM

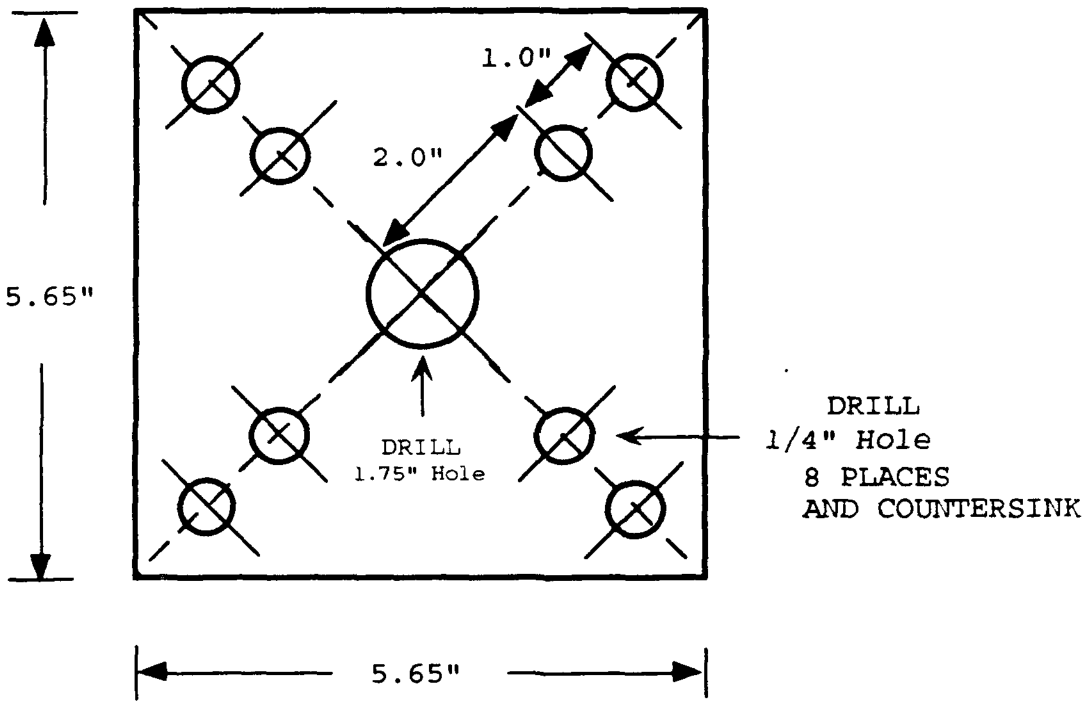
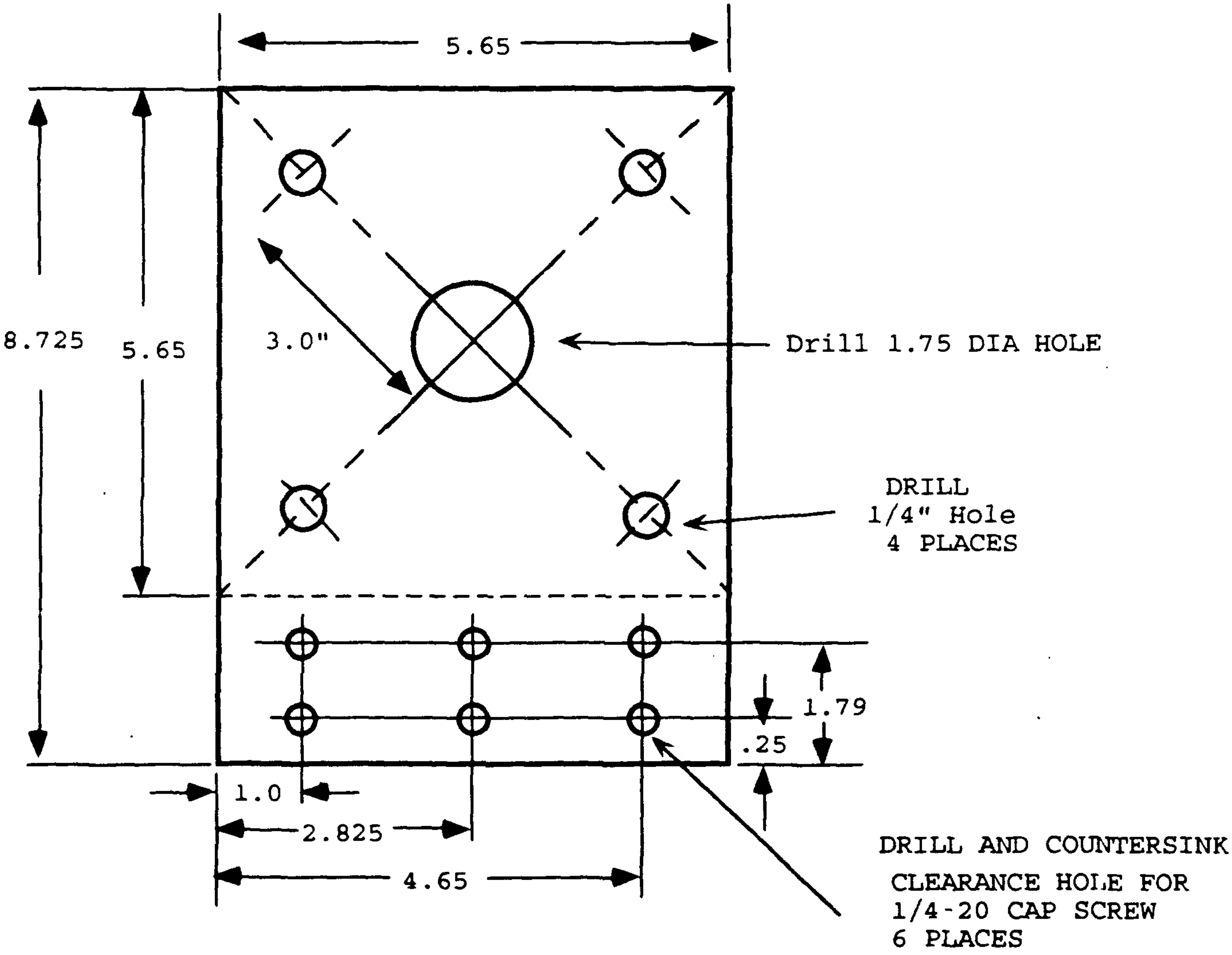


Figure 39-2: Mounting brackets.

PIECE #2

Thickness = 3/4" Aluminum



Drill 1.75 DIA HOLE

DRILL
1/4" Hole
4 PLACES

DRILL AND COUNTERSINK
CLEARANCE HOLE FOR
1/4-20 CAP SCREW
6 PLACES

Figure 39-3: Mounting brackets.

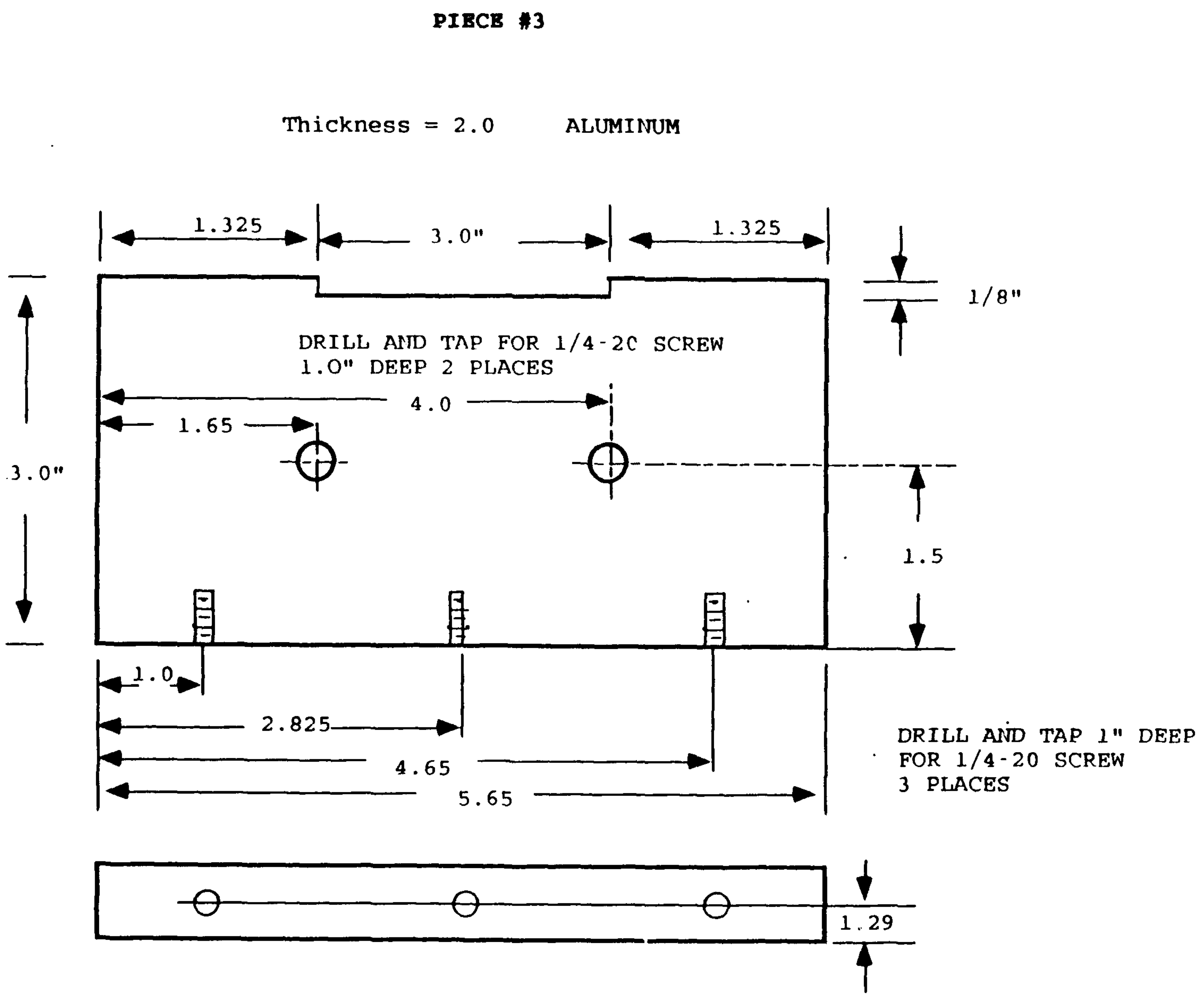


Figure 39-4: Mounting brackets.

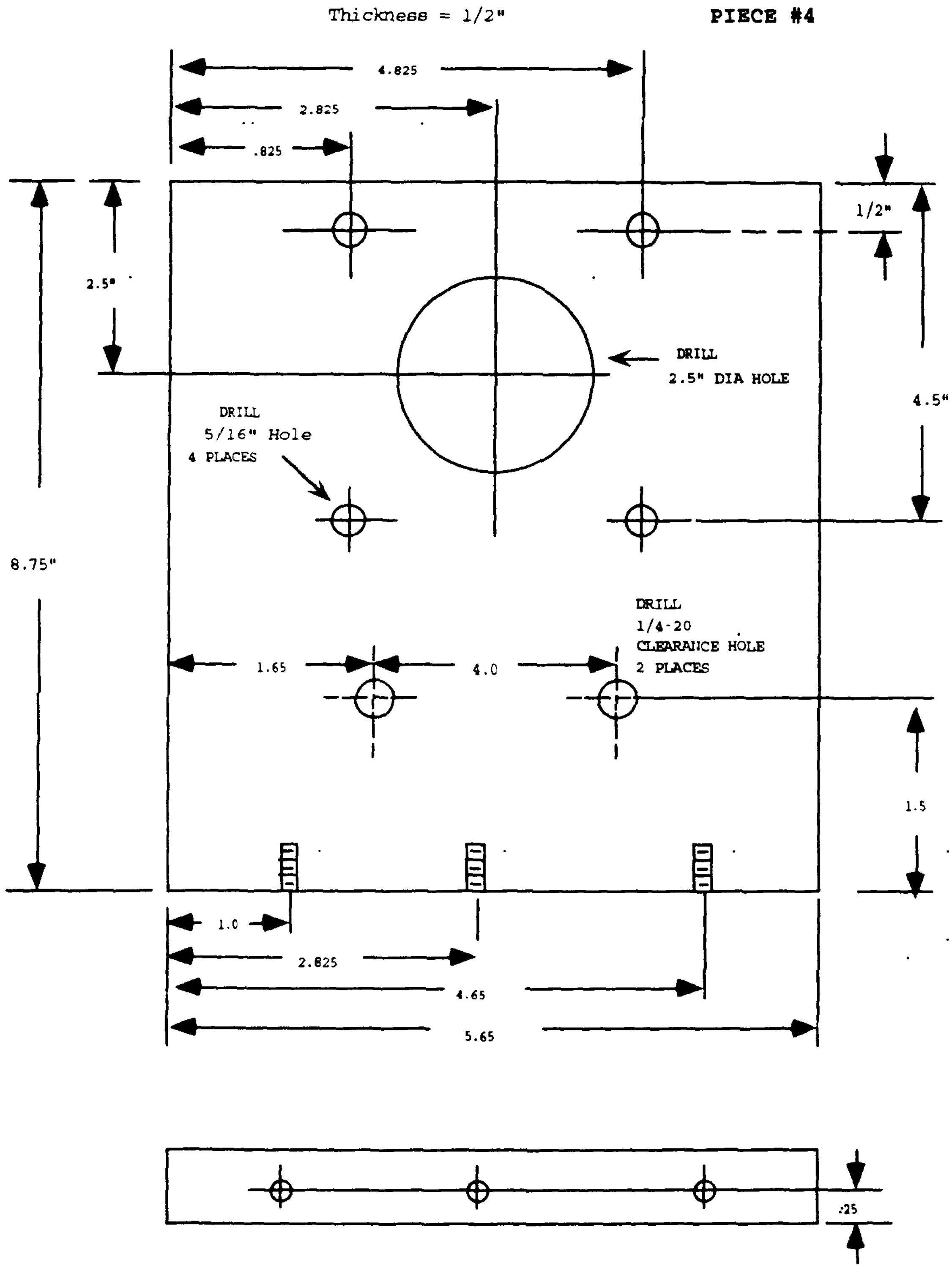


Figure 39-5: Mounting brackets.

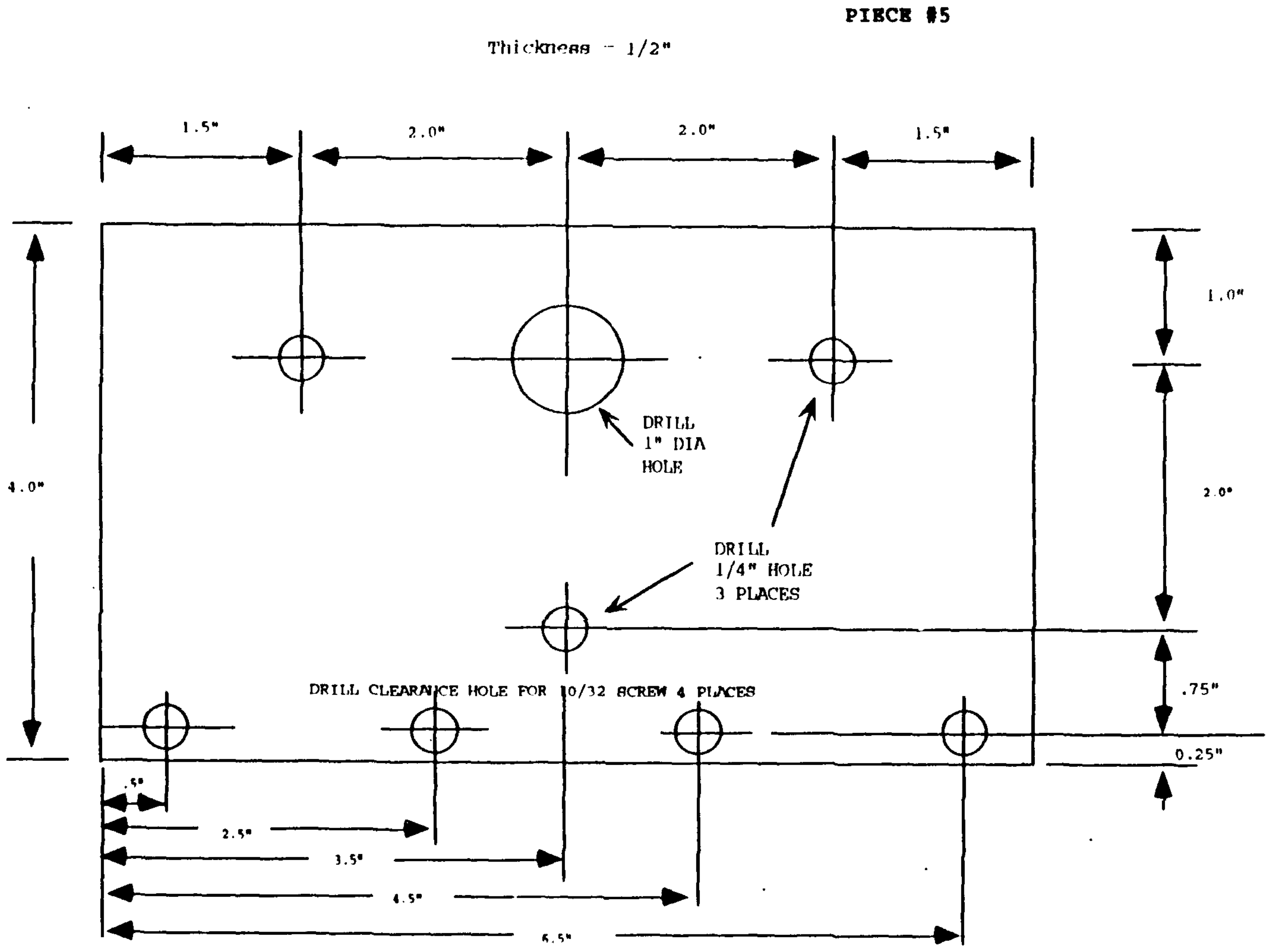


Figure 39-6: Mounting brackets.

PIECE #6

Thickness = 1/2" ALUMINUM

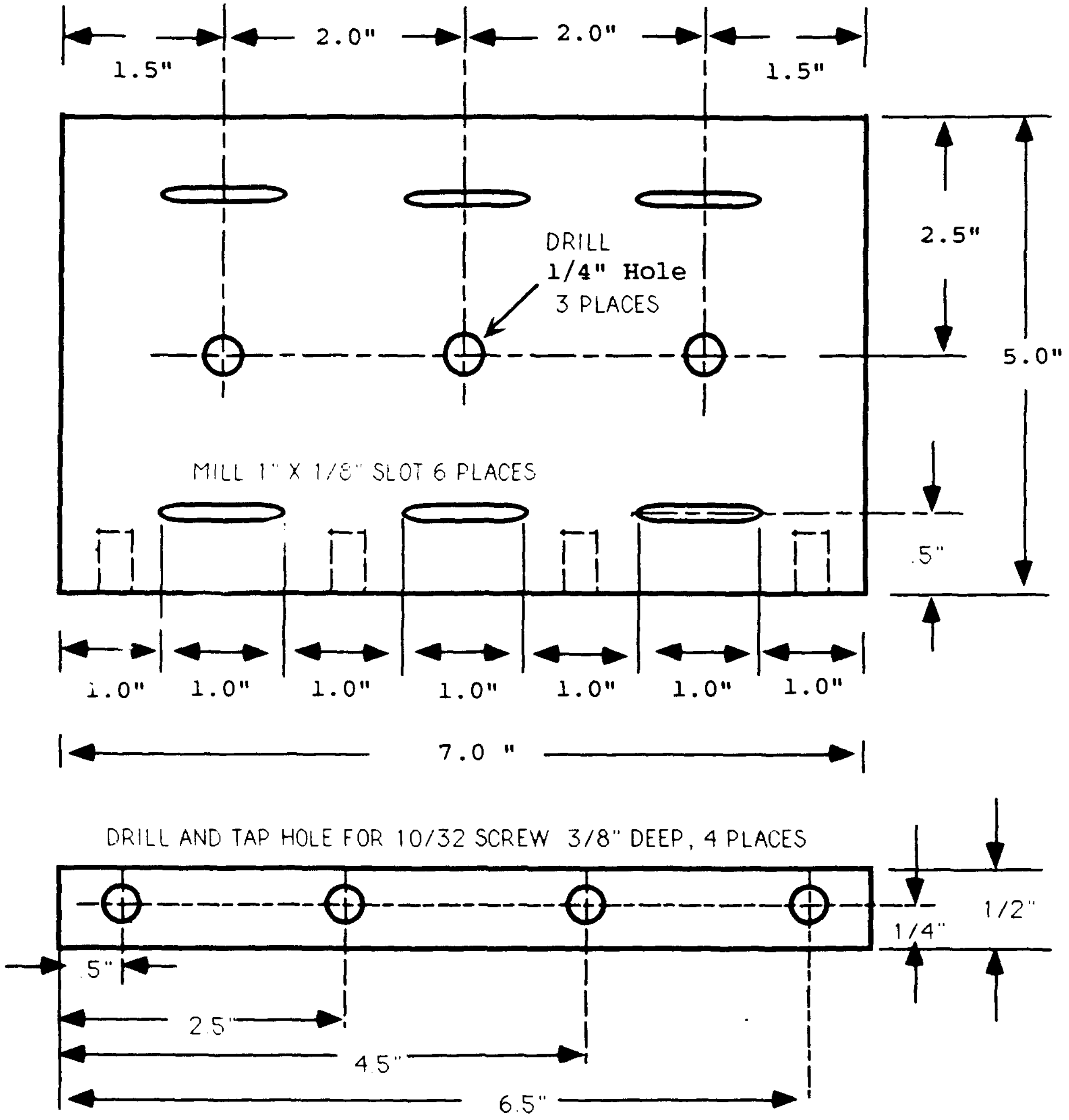


Figure 39-7: Mounting brackets.

Simple motor control software for the Technology 80 programmable controller was also developed and successfully tested.

Next, we began to improve the image processing algorithm. Windowing of the screen was added, and various equations for calculating the target's true centroid (as opposed to this iterative quad tree approach) were developed.

Changing the window from 512 x 512 pixels (full screen) to 96 x 96 pixels allowed new servo-motor commands to be generated every 0.1 seconds, based upon recursive quad-tree processing. Adding a true centroid calculation improved tracker performance but increased the processing time. The centroid was calculated using the following equation:

$$X_c = \frac{\Sigma\ X_i\ I_i}{\Sigma\ I_i}$$

$$Y_c = \frac{\Sigma\ Y_i\ I_i}{\Sigma\ I_i}$$

where X_i (or Y_i) is the X (or Y) axis coordinate of the center of pixel and I_i is its measured intensity. The increase in processing time is due to the long buffer interface library routine, provided by Data Translation.

MTI's Intel 80486 CPU based PC was substituted for our existing PC-AT computer. As a result, the new MTI-486 based video tracker system is able to compute centroids and execute motor control commands at 8 frames per second. This represents a speedup in data rates by a factor of approximately 4 times.

Two changes in the software were made. They are the addition of a direct memory block access scheme to the DT-2851 buffer, and the addition of sub-window subtraction using a direct memory block access scheme.

The DT-2851 has two 256 K-bytes frame grabbing buffers which are synchronized to the external video input at the rate of 30 frames per second. At the start, the software establishes a subwindow of 64 x 64 pixels and creates 2-dimensional buffers corresponding to the 2 frame buffers. Then, the software accesses each buffer in order to download pixel intensities to the PC destination buffers.

Once pixel intensities in the subwindow have been completely transferred form DT-2851 buffers to the destination buffers, the program then subtracts the current window intensity from the reference window intensity for each pixel.

Currently, only the central 64 x 64 window pixels within the full 512 x 512 frame of pixels, is used for calculation of the target's centroid. The software for centroid calculation is being upgraded to include an algorithm for selecting alternate 64 x 64 windows at different locations within the full frame. This will improve tracking performance during periods in which the target maneuvers or its velocity exceeds the tracker bandwidth. In the long term, selection of the window location will be based upon globally optimal track predictions (transformed back into local coordinates); in the short term, it will be based upon locally optimal track predictions.

Along these lines, software for selecting 1 of 8 alternate 64 x 64 windows inside of the full frame was completed but not fully tested. That is, in addition to the 64 x 64 window located in the center of the full frame, 8 alternate 64 x 64 windows surrounding the central window may be sequentially searched when the central window is empty.

The following describes the software structure for a video tracker.

Input: mp_para.dat (fj.dat for Fujinon camera, mg.dat for magnavox)
Library:
1) Microsoft-C compiler large model library
2) Floating point library with Math Coprocessor (80387)
3) dtir.lib
4) t5638.lib

Algorithm:

```
main( )
{
        initial_value_setting;
        motor_initialization;
        image-grabber_initialization;
        read data from the file mp_para.dat;
        move motors to find target with cursor keys manually;
        for (;;)
        {
                dt_target_direction(old_delta_x,old_delta_y,&direction);
                dt_monopulse(delta_X,delta_y);/*monopulse*/
                if (delta_x==0 and delta_y==0) then
                        switch_direction(&direction, delta_x,delta_y);/*relocate window*/
                        profile(0,delta_x, vel, acc);/*move x_axis motor*/
                        profile(1,delta_y,vel, acc);/*move y-axis motor*/
                        net(delta_x,delta_y);*send data*/
        {
        free(memory);
}
```

The following are a list of functions.

```
dt_target_direction(old_delta_x, old_delta_y, &direction)
/*find target direction and relocate window to the new area */
{
        switch_direction(&direction, ... ,delta_x,delta_y);
}
dt_monopulse(delta_x,delta_y)
/* monopulse main body*/
{
        setup_window:
        dt_centroid: /*calculate the centroid* /
        return quadrant_number;
}
dt_change_window(delta_x,delta_y)
/* if no target inside of window, change_window, and */
/* move motors until if finds target */
net(delta_x,delta_y)
{
        initialize_graphic_mode:
        receive_cmd;/*receive data from client 1 and 2*/
        setpixel(delta_x,delta_y); /*plot delta_x and delta_y*/
        if (key= ='s') then exit;
}
```

2.4.3 Network Communications

A network program for communicating between MS-DOS based local and global processors was created. It is based on the AT&T Starlan Network. The 2 local processors were named "client1" and "client2", and the global processor was named "center". Each client contains a network card in addition to the other cards previously described. Each client performs centroid calculations and motor commands. Also, each sends delta_x and delta_y values to the center, which receives data from both clients, and plots their values using real-time graphics. This section describes basic networking concepts, the network components, and explains how to operate the full cluster from a networking point of view.

Network Components: The AT&T Starlan Network is a baseband, 1-megabit per second, local area network that provides simple, reliable communications (logical and physical) between two or more devices on the network. Devices on the network include minicomputers, personnel computers, video display terminals, printers, and intelligent workstations.

The logical aspect of an AT&T Starlan Network connection is established and maintained via network software and firmware within each device involved in the connection.

The physical aspect of an AT&T Starlan Network connection consists of AT&T Starlan Network hardware and the transmission medium of the network; standard 24-gauge, building telephone wire. Figure 40 illustrates the components involved in a typical AT&T Starlan Network for video tracking system.

Basic Networking Concept: The following basic networking concepts explain the operation of the AT&T Starlan Network.

Peer Oriented - The AT&T Starlan Network is a peer network. All devices on the network are recognized as equal with regard to access of network services and the order of servicing network devices. Requests for network services are handled on a first-come, first-served basis with no need for a single, centralized point of network control.

Network Names - Each device on the AT&T Starlan Network is assigned unique name. That name can then be used to identify the device on the network for all subsequent network operations. Names can be locations (room number), a person's name (Bob), or any other easy-to-remember identifiers. Names can also be assigned to local groups.

Communicating via a session - Devices on the AT&T Starlan Network are connected with reliable, point-to-point connections known as virtual circuits. The establishment of a virtual between two network names and the subsequent use of that virtual circuit to communicate is known as a session. Sessions are equivalent to the reliable, point-to-point, two-way connections that exist in traditional telecommunications networks.

Video Tracking Network Programs: The video tracking networking program contains routines that add network names, connect to a network name, listen on a network, send (transmit) data on a network, disconnect a network using the SLI (Session Layer Interface) program interface.

To use the video tracking network program, the following network environment is needed. Three MS-DOS version 3.3 PC workstations, either IBM-PC 386 or 486 machines, each equipped with an AT&T Starlan Network NAU (Network Access Unit). A working AT&T Starlan Network physical connection must exist between the three workstations. Each workstation must have the AT&T Starlan Network client version 3.2 program software installed and operational. Each workstation must have the client1, client2, center program files installed.

Running Client Programs - The following procedure is used to install program files.

Step 0: when you boot system, system asks you "Load AT&T Starlan program? (y/n)".

then, type 'y' to load Network program.

Step 1: Change directory for example (C:\> CD \NET).

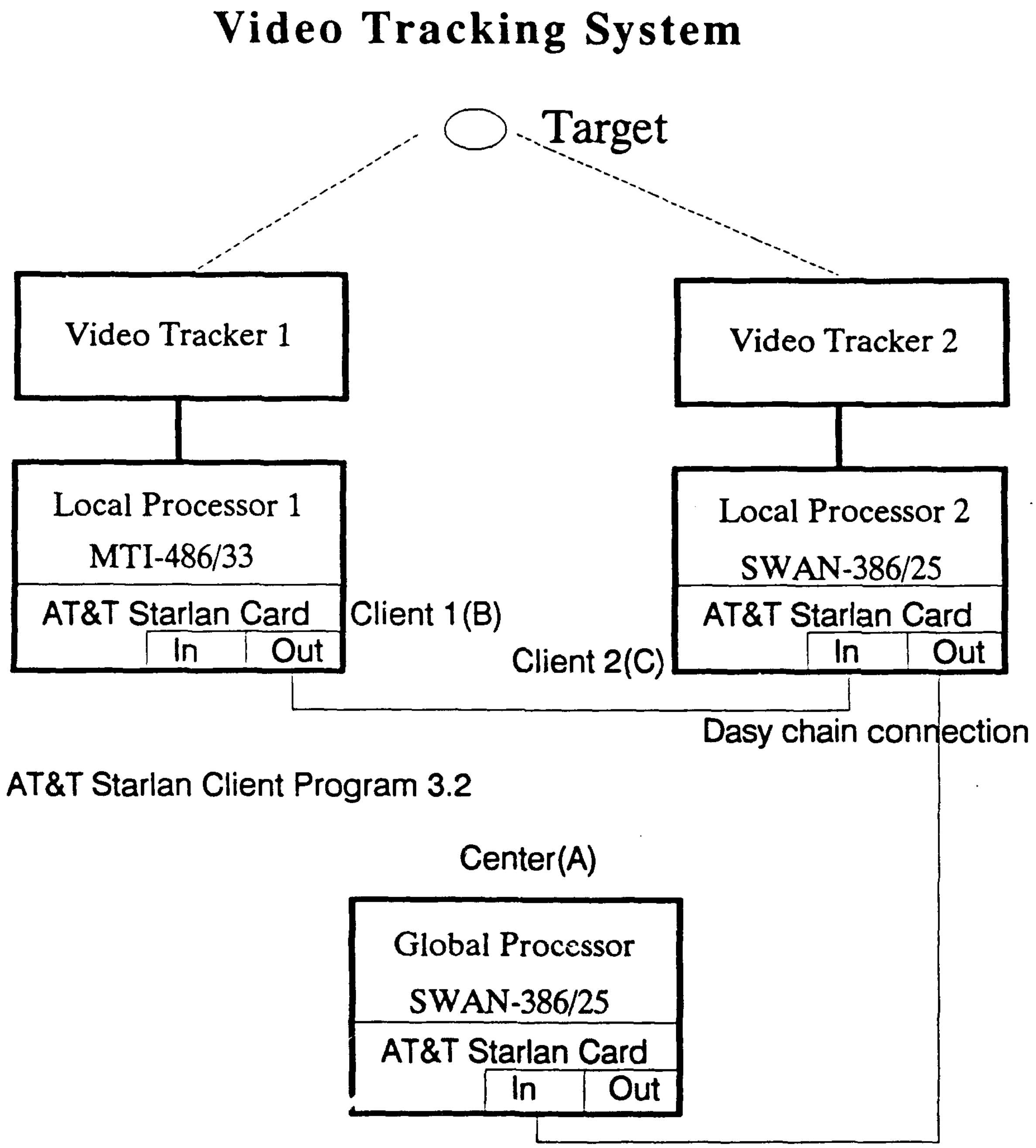


Figure 40: Starlan network for the video tracking system.

Step 2: type 'client1'.

The video tracking network program is now loaded. The Main Menu screen should now be displayed on the screen. This screen presents a list of available network operations. To select a particular network operation, simply enter the letter of the network operation from the menu. It will issue a series of screen prompts that request information necessary to perform the network operation selected. See figures 41 and 42.

Establishing A Network: Each client displays the following after you type "client1" or "client2" at the DOS prompt.

```
                    VIDEO TRACKING TEST COMMAND MENU

        a - Add a Unique Network Name

        c - Call and Establish a Session

        1 - Listen and Establish a Session

        m - Redisplay This Menu

        ? - Help

        q - Quit

menu selection --->
```

Center Menu is as follows (when you type "center" at the center computer).

```
                    VIDEO TRACKING TEST COMMAND MENU

        a - Add a Unique Network Name

        c - Call and Establish a Session

        l - Listen and Establish a Session

        v - Run Video Tracking

        m - Redisplay This Menu

        ? - Help
```

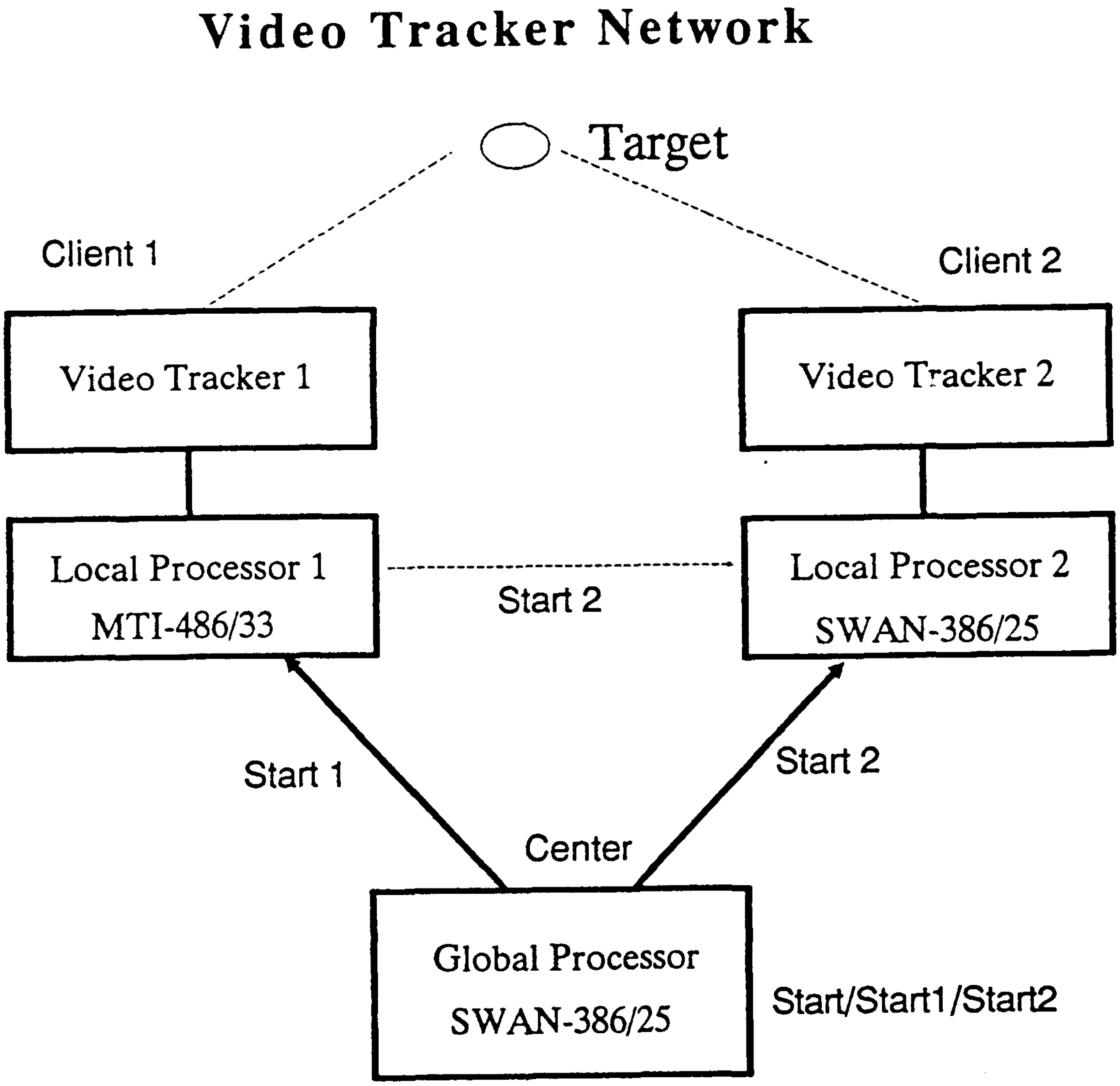


Figure 41: Operation of the network.

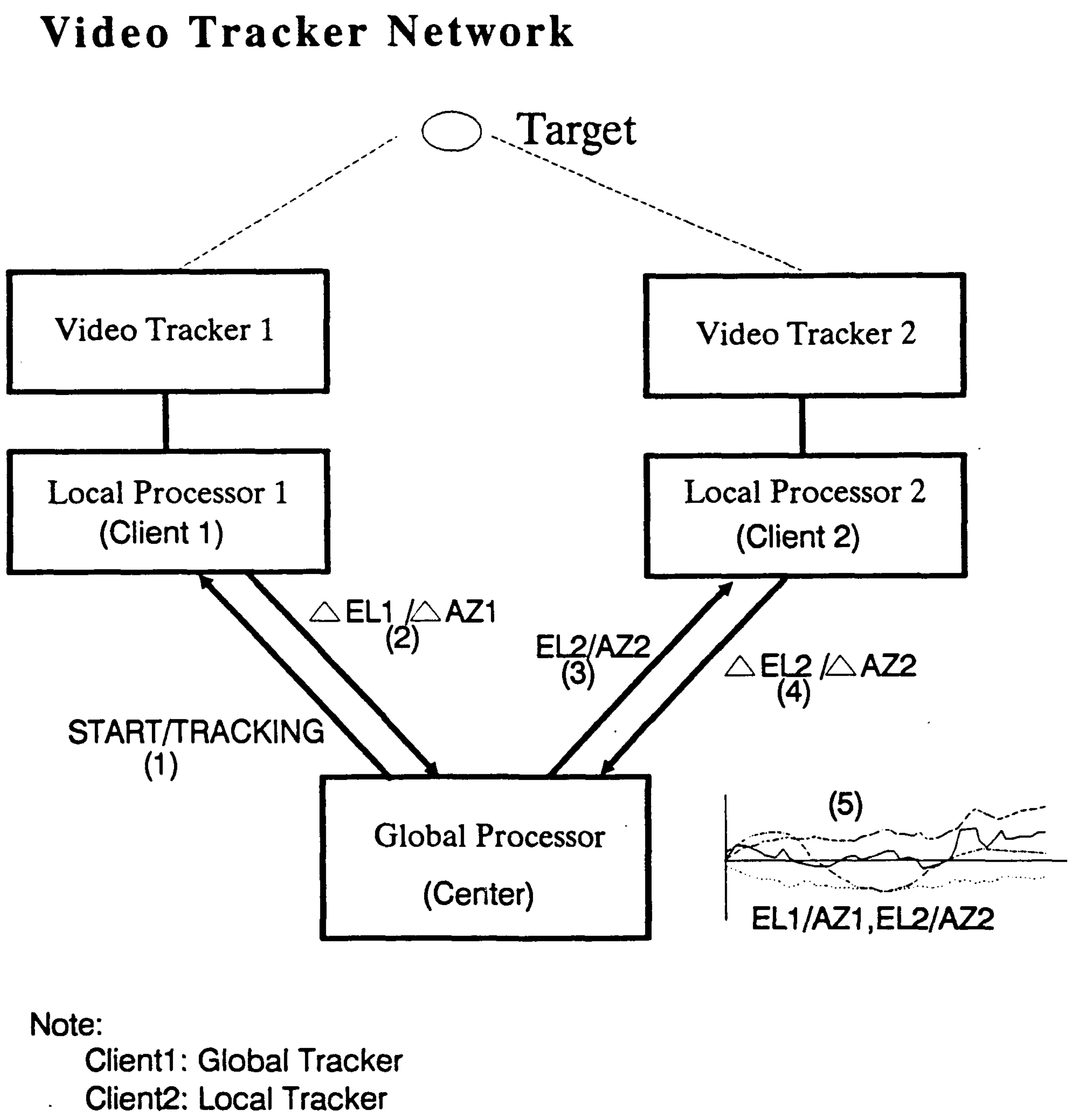


Figure 42: Diagram of the video tracking network.

```
        q - Quit

menu selection --->
```

"ESTABLISHING A NETWORK" establishes a session between these two workstations by issuing a listen ("1") on one then a call ("c") on the other. The 'remote name' is that of the other workstation. The 'local name' is that of the local workstation. When the session is established, a local session number is displayed.

Adding A Network Name: The first step you must perform in order to communicate via this program is to add a network name for each workstatiou as follows:

At workstation 1:

1. Enter ‘a’ at the menu selection --> prompt.

2. a name ("client1").

3. The program then adds the name and notifies you with the following prompt: "Add name completed successfully".

4. Repeat this procedure for workstation2 ("client2"), workstation3 ("center").

Example: In order to send and receive data between two workstations on the network, first establish a network connection between the two systems. To do this, you must do a LISTEN at the center workstation and a CALL at the client1 workstation.

At the center workstation,
1. enter ‘1’ at the menu screen,
2. enter remote name to listen for: "client",
3. enter local name to listen from : "center",
then it returns the local session number,

At the client1 workstation,
1. enter ‘c’ at the menu screen,
2. enter remote name to call : "center",
3. enter local name to call from : "client1",
then it returns the local session number.

At the client1 workstation,
1. enter 'l' at the menu screen,
2. enter remote name to listen for : "center",
3. enter local name to listen from : "client1",
then it returns the local session number.

At the center workstation,
1. enter ’c’ at the menu screen,
2. enter remote name to call to : "client1",
3. enter local name to call from : "center",
then it returns the local session number.

At the center workstation,
1. enter ’l’ at the menu screen,
2. enter remote name to listen for : "client2"
3. enter local name to listen from : "center",
then it returns the local session number.

At the client2 workstation,
1. enter ’c’ at the menu screen,
2. enter remote name to call to : "center",
3. enter local name to call from : "client2"
then it returns the local session number.

At the client2 workstation,
1. enter ’l’ at the menu screen,
2. enter remote name to listen for : "center"
3. enter local name to listen from : "client2"
then it returns the local session number.

At the center workstation,
1. enter ’c’ at the menu screen,
2. enter remote name to call to : "client2"
3. enter local name to call from : "center"
then it returns the local session number.

Running Dual Camera Tracking on the Network: Before you run video tracking, one should run client1 and client2 workstations using the ‘client1’ and ‘client2’ command at each station.

At the client1 system, type ’r’ to receive data (to wait for command). At the client2 system, type ’r’ to receive data (to wait for command). Then each station waits for the sender’s command. At the center workstation, run ‘center’ command, then you will see the Main Menu. Use ‘v’ command to run video tracking.

The screen asks questions as follows. "Type ‘y’ to move client1 motor from center workstation (y/n) ? [n]" If you say ’n’ or hit enter key, then it will skip. If you answer ‘y’, then "Enter data to send the ’move’ command. The syntax ’MOVE delta_x delta_y:’" will be displayed. Then you can answer: "move delta_x (azimuth) delta_y (elevation)", for example, ("move 10-20" or "move -5 10").

Next, you have another question like the following: "Do you want to move motor for CLIENT 2? (y/n) [n]". Then "Enter data to send syntax: 'Move delta_x delta_y':" will be seen. One can provide the same answer as above. If you say 'n', then it will skip to the next.

Finally, system asks "Enter data to send (starting command):" If you type "start" command at the center workstation, then the first workstation begins tracking. At the same time, you can see the real time graphics (azimuth and elevation for client1 video system working at the center system monitor). At this point, only the client system and video camera are working.

If you want to give the client2 workstation a command to start tracking in the middle of working, type 'c' at the center system's keyboard. Then you will see that both client1 and client2 systems are working at the same time.

If you type a "start2" command at the center system, only the client2 workstation and video camera work. If you type a "start" command at the center system, both client1 and client2 systems start tracking at the same time.

If you want to stop tracking, you can type 's' at the client2 system first while client2 system is working, then only client1 system will stop. And type 's' at the client1 system while the client1 and center system are working together. Then client1 and center system will stop working.

3. Estimates of Technical Feasibility

Surrounding ground targets with a microcomputer based network of low cost video trackers is an effective means for acquiring and tracking SADARM submunitions. At a 1,000 foot range, the submunition is at the top of the tracking volume, and will subtend only a few pixels square when the video camera's field of view is fully extended to 45 degrees. Both of the video trackers developed in Phase II were able to track high contrast targets of this size. About 8 seconds are needed to zoom from full extension to the minimum field of view (about 5 degrees). Unfortunately, after ejection, the submunition is airborne for only 10 to 12 seconds; higher speed zooms could be used to increase track time with reasonably sized images.

The Decentralized Square Root Information Filter (DSRIF) has several very unique features which have been incorporated into the design of a dual network software package. The package is approximately 50% complete, at a combined Phase I/II cost of less than $500,000. Software modules to control both the short and long range network processors, as well as the dual network interface are needed.

In future research, attitude measurements from the video camera, along with conventional range, azimuth and elevation measurements could be processed in separate filters and merged centrally. Process noise levels could be easily adjusted during target maneuvers when attitude measurements become important. The contribution of the attitude filter to the global solution is continuously available and may prove useful in maneuver detection (besides being useful in predicting the translational motion of the targets).

References

[1] G. Bierman, Factorized Methods for Discrete Sequential Estimation, Academic Press, 1977.

[2] G. Bierman and M. Belzer, "A Decentralized Square Root Information Filter/Smoother", Proceedings of the IEEE Control and Decision Conference, 1985.

[3] M. Belzer and Y. Cho, "Acoustic Data Processing Using the Decentralized Square Root Information Filter, MTI final report to the Defense Advanced Research Projects Agency, 1989.

[4] M. Belzer and G. Bierman, "Maximum Likelihood Estimation Using Square Root Information Filters", IEEE Transactions on Automatic Control, 1990.

[5] Y. Bar-Shalom and T. Fortmann, Tracking and Data Association, Academic Press. 1988.

[6] S. Blackman, Multiple-Target Tracking with Radar Application, Artech House, 1986.

[7] D. Willner, C. Chang and K. Dunn, "Kalman Filter Algorithms for a Multisensor System," Proceedings of the IEEE Control and Decision Conference, December, 1976.

[8] C. Chang, R. Whiting and M. Athans, "On the State and Parameter Estimation for Maneuvering Reentry Vehicles," IEEE Transactions on Automatic Control, February, 1977.

[9] A. Jazwinski, Stochastic Processes and Filtering Theory, Academic Press, 1970

[10] P. Bogler, "Tracking a Maneuvering Target Using Input Estimation," IEEE Transactions on Aerospace and Electronic Systems, May, 1987.

[11] A. Houles and Y. Bar-Shalom, "Multisensor Tracking of a Maneuvering Target in Clutter," IEEE Transactions on Aerospace and Electronic Systems, March, 1989.

[12] Y. Bar-Shalom, K. Chang and H. Blom, "Tracking a Maneuvering Target Using Input Estimation versus the Interacting Multiple Model Algorithm," IEEE Transactions on Aerospace and Electronic Systems, March, 1989.

[13] L. Chin, "Advances in Adaptive Filtering," in Advances in Control and Dynamic Systems, Volume 15, Academic Press, 1979.

[14] M. Belzer and Y. Cho, "Micro-Computer Network Architecture for Range Instrumentation Applications", MTI Phase I final report to the White Sands Missile Range, 1988.

Appendix A
Simulation Subroutines

1. WSMR_TM

Assuming a linear time invariant target model $x(k+1) = Fx(k) + Bw(k)$, in this routine the matrices F and B are initialized, and the inverse of F is computed using the Gauss-Jordan method.

INPUT

- F and B matrices

OUTPUT

- Inverse of F matrix

SUBROUTINES CALLED

ZERO DNVERT MULT

2. INPAR

Reads the initial parameters, which include the initial state estimate error covariance matrix, the process noise mean vector and covariance matrix, and the measurement noise covariance matrix, from a file to initialize the DSRIF. These parameters are changed to their corresponding square root information matrix and vector forms using the upper triangular Cholesky decomposition.

INPUT

- Initial state estimate error covariance matrix
- Measurement noise covariance matrix
- Process noise mean vector and covariance matrix

OUTPUT

- Initial estimate error square root information matrix
- Process noise square root information matrix and vector
- Measurement noise square root information matrix

SUBROUTINES CALLED

ZERO FTOSYM COV2RI UTINV MULTUF

3. COORTR

Calculates a set of Euler angles for each local coordinate system, as well as matrix transformations between global and local coordinate systems.

INPUT

- Radius of the Earth
- Latitude and longitude of the GCS
- Location of each sensor in the GCS

OUTPUT

- The Euler angles and matrix transformations

SUBROUTINES CALLED

MULT MADD

4. RECEIVE

Opens measurement data file(s) and stores the array YMD. Range radar measurement data are rewritten in terms of global coordinates, and stored in temporary files.

INPUT

- Matrices for transformation from local to global coordinates
- Local measurement data (Range, Azimuth, Elevation)
- Location of all sensors

OUTPUT

- Radar (position) measurements for each target in global coordinates
- Number of targets per scan

5. COMPR

Groups and compresses the coordinate transformed range radar data. A global ID is assigned to each compressed measurement. Compressed measurements and their global IDs are stored in a "compressed measurement" file.

INPUT

- Coordinate transformed range radar data

OUTPUT

- Compressed measurement data with IDs

6. GTOL

Transforms compressed measurements into local coordinates. New measurement IDs are transferred with corresponding transformed measurements.

INPUT

- Compressed measurement data with IDs
- Matrices for transformation from global to local coordinates

OUTPUT

- Compressed measurements in local coordinates with new measurement IDs

7. SETID

Resets each local measurement ID to a global measurement ID.

INPUT

- Local measurements
- Compressed measurements in local coordinates with new measurement IDs

OUTPUT

- Local measurements with new global IDs

8. ARRANGE

Sort and store local measurement data in correspondence with the global measurement IDs.

INPUT

- Local measurements with new global IDs

OUTPUT

- Sorted local measurements with new IDs

9. GTUP

Performs global time updating.

INPUT

- Smoothing coefficient matrices from all local processors
- Measurement updated global square root information matrix and vector
- Process noise square root information matrix
- F and B matrices

OUTPUT

- Time updated global square root information matrix and vector

SUBROUTINES CALLED

MULTUF_M MULT TDHHT RINZ_M

10. MEASN_M

Transforms the global state into equivalent local measurements, such as range, azimuth, elevation or azimuth, elevation depending upon the type of senso:.

INPUT

- Global state vectors (for all targets)
- Matrices for transformation from global to local coordinates

OUTPUT

- Equivalent local measurement vectors

11. CMAT-M

Computes the observation matrix.

INPUT

- Location of all sensors
- Matrices for transformation from global to local coordinates
- Time updated global state

OUTPUT

- Observation matrices for all local processors

12. MTOTA

Performs the first step of local measurement-to-track association. Gating is performed using time updated global estimates which are represented in terms of local variables.

INPUT

- Local measurements with IDs
- Time updated global estimates in terms of local variables

OUTPUT

- Local association matrices

SUBROUTINES CALLED

UTINV MULT MADD MSUB_M2 MULT_M1

13. CR_XGP

Performs the second step of local measurement-to-track association. Gating is performed using time updated global estimates which are represented in terms of local variables. This routine also detects outlying data.

INPUT

- Local measurements with IDs
- Time updated global estimates in terms of local variables

OUTPUT

- Local association matrices

14. ADD

Receives and fuses the local association matrices from the subroutines MTOTA and CR_XGP. This routine is called by each local processor.

INPUT

- Local association matrix from MTOTA
- Local association matrix from CR-XGP

OUTPUT

- Fused local association matrix

15. LMUP

Performs local measurement updating based on the local association matrix.

INPUT

- Local measurements
- Local time updated square root information matrix and vector
- Local observation matrix
- Global time updated state
- Matrices for transformation from global to local coordinates
- Measurement noise square root information matrix

OUTPUT

- Local measurement updated square root information matrix and vector

SUBROUTINES CALLED

ZERO TDHMT

16. FUSION

This routine performs global fusion of local association matrices using the Majority Voting Method and Rule.

INPUT

- Local association matrices

OUTPUT

- Global fusion matrix

SUBROUTINE

RULE

17. GMUP

Performs global measurement updating.

INPUT

- Local predicted state information matrices and vectors
- Global predicted state information matrix and vector

OUTPUT

- Global measurement updated square root information matrix and vector

SUBROUTINES CALLED

ZERO TDHHT RINZ_M

18. BGMUP

Performs the final fusion of local association matrices using the likelihood function.

INPUT

- Local association matrices

OUTPUT

- Global fusion matrix

SUBROUTINES CALLED

ZERO LKHD

19. LKHD

This routine evaluates the likelihood function.

INPUT

- Global time updated square root information matrix
- Global measurement updated square root information matrix
- Measurement noise square root information matrix
- Innovations vector

OUTPUT

- Value of likelihood function

SUBROUTINES CALLED

SYMTOF DET

20. LTUP

Performs local time updating.

INPUT

- F and B matrices
- Local measurement updated square root information matrix and vector
- Process noise square root information matrix

OUTPUT

- Smoothing coefficients from all of the local processors
- Local time updated square root information matrices and vectors

SUBROUTINES CALLED

MULTUF MULT TDHHT

21. MSTOMS

This routine performs global measurement-to-measurement association using gating.

INPUT

- Leaves of hypothesis tree (position information)
- Compressed measurements

OUTPUT

- Measurement-to-measurement a.sociation matrix

22. DELET

Deletes leaves of hypothesis tree which are not associated with any new compressed measurements.

INPUT

- Hypothesis tree
- Measurement-to-measurement association matrix

OUTPUT

- Hypothesis tree

23. MKNEW

Adds new leaves to hypothesis tree with level 1s.

INPUT

- Hypothesis tree
- Measurement-to-measurement association matrix

OUTPUT

- Hypothesis tree with new leaves

24. STORE

Stores nodes of hypothesis tree in different files according to levels.

INPUT

- Hypothesis tree

OUTPUT

- Level files

25. MKINT

This routine generates initial state estimates for confirmed tracks and stores them in a track file.

INPUT

- Level files
- Compressed measurement file
- Track file

OUTPUT

- Track file with initial states of new tracks

26. DEL

Once a track is confirmed from the set of potential tracks, this routine deletes the unnecessary tracks from the hypothesis tree.

INPUT

- Hypothesis tree

OUTPUT

- Hypothesis tree

SUBROUTINES CALLED

FINC

27. CMTOTR

Copies compressed measurement file into track file at the first iteration step.

INPUT

- Compressed measurement file

OUTPUT

- Track file

28. UPTRA

Updates the track file.

INPUT

- Track file

OUTPUT

- Updated track file

Appendix B
Listing of References [1] and [2]

Proceedings of 24th Conference
on Decision and Control
Ft. Lauderdale, FL · December 1985

A DECENTRALIZED SQUARE ROOT INFORMATION FILTER/SMOOTHER

Gerald J. Bierman
Factorized Estimation Applications, Inc.
15445 Ventura Blvd., Suite 10-410
Sherman Oaks, CA 91413

Mitchell R. Belzer
Business & Technological Systems, Inc.
10210 Greenbelt Road - Suite 440
Seabrook, MD 20706

SUMMARY

In this paper we present a new method for combining linear least squares estimates obtained from independent data sets. A bank of Square Root Information Filters (SRIF) is used to generate these "local" estimates as well as their corresponding smoothing coefficients which can be merged after each predictive step to obtain globally optimal smoothing coefficients. Additionally, the merging algorithm recursively computes a global information vector and *square root information matrix* which can be merged with their local counterparts to obtain globally optimal values. Globally optimal smoothed estimates and covariances are obtained from a backwards recursion using either the smoothed estimates and covariances directly [1] or a data equation Square *Root Information Smoother* (SRIS) [2] which uses the globally optimal Dyer-McReynolds smoothing coefficients as input.

A major advantage of our approach over a decentralized covariance approach is its ability to add effects of the *a priori initial estimate* covariance and process noise to the results obtained with these effects omitted. In the covariance based case, the effects have to be subtracted (after they have been included twice). An additional feature of the approach is that it is not even necessary to reprocess the data when the a priori initial state covariance and process noise variances are changed. This is especially attractive when one is trying to "tune" the filter for problems with large amounts of data.

I. INTRODUCTION AND PROBLEM FORMULATION

Recently, there has been considerable interest in decentralization of linear least squares estimators. A major motivation is the impending emergence of VLSI technology, the realization of parallel processing, and the need for algorithmic ways to speed the solution of dynamically decoupled, high dimensional estimation problems.

A survey of decentralized filtering techniques was made by T. Kerr et.al. in [3]. In this paper we present a new method for combining SRIF estimates obtained from independent data sets. After the problem statement that follows, the new technique is derived in Section II. The new technique turns out to be an orthogonal transformation based algorithmic implementation that generalizes an information matrix filter "homework" problem in [4]. Section III contains a discussion of the decentralized SRIF in which merits of the proposed approach are described, along with concluding remarks.

For the discrete-time linear dynamical model

$$x_{j+1} = \phi_j x_j + B\omega_j \quad ; \; j = 0, T-1 \qquad (1)$$

with two local discrete measurement models

$$\begin{aligned} z_j^{(1)} &= H_j^{(1)} x_j + \nu_j^{(1)} \\ z_j^{(2)} &= H_j^{(2)} x_j + \nu_j^{(2)} \end{aligned} \quad ; \; j = 0, T \qquad (2)$$

the goal is ([5], p.42) to find $x_0, \ldots, x_T$ that minimizes the least-squares likelihood performance functional

$$J_T(x_0, \ldots, x_T) = \| \tilde{R}_0 x_0 - \tilde{z}_0 \|^2 + \sum_{j=0}^{T} (\| H_j^{(1)} x_j - z_j^{(1)} \|^2 + \| H_j^{(2)} x_j - z_j^{(2)} \|^2) + \sum_{j=0}^{T-1} \| R_\omega(j) \omega_j \|^2 \qquad (3)$$

where the a priori estimate $\tilde{R}_0^{-1} \tilde{z}_0$ has covariance $\tilde{P}_0 = \tilde{R}_0^{-1} \tilde{R}_0^{-T}$.

At this point we do not assume any special structure on ϕ_j or the H terms other than that ϕ_j invertible and $\{\omega_j\}$, $\{\nu_j^{(1)}\}$, $\{\nu_j^{(2)}\}$ are zero mean white Gaussian noise processes that are statistically independent of one another with $E[(\nu_j^{(1)})^2] = I_{m_1(j)}$, $E[(\nu_j^{(2)})^2] = I_{m_2(j)}$ (i.e., local data sets can have arbitrary dimensions but noise-free measurements, or noise-free measurement combinations are not allowed), and $E(\omega_k \omega_k^T) = R_\omega(k)^{-1} R_\omega(k)^{-T}$. We only deal with two data sets, but it is clear from the methodology presented that any number of data sets can be accomodated.

The estimates of $x_0, \ldots, x_T$ that minimize this performance functional are the smoothed minimum variance estimates and, for covariance analysis purposes, we are interested in smooth estimate error covariances. The approach described by Willsky et al, [6] is to combine the estimates obtained by separately processing the data sets, viz. for $k = 1, 2$

This work was supported in part by the Jet Propulsion Laboratory, and by the Applied Physics Laboratory, Johns Hopkins University, under Contract 602196-S.

$\{x^{(k)}_{j|T}\}_{j=0}^{j=T}$ and

$$J_T^{(k)} = \|\bar{R}_0 x_0 - \bar{z}_0\|^2 + \sum_{j=0}^{T} \|H_j^{(k)} x_j - z_j^{(k)}\|^2 \tag{4}$$

$$+ \sum_{j=0}^{T-1} \|R_\omega(j)\, \omega_j\|^2$$

where the slash subscript notation is suggestive of conditional expectation. Since both performance functionals use the same a priori on x_0, and the same process noise $\{\omega_j\}$ it is easy to see that the estimates produced will be correlated. Another way to see this is to note that the filter estimate is a linear combination of the data and the model, so that

$$x^{(k)}_{j|T} = \sum_{i=0}^{T} F^{(k)}_{i|T}\, z_i^{(k)} + \Gamma^{(k)}_{0|T}\, \bar{x}_0 + \sum_{i=0}^{T-1} \bar{F}^{(k)}_{i|T}\omega_i \tag{5}$$

where the weighting term multipliers $\Gamma^{(k)}_{0|T}$, $\{F^{(k)}_{i|T}\}$ and $\{\bar{F}^{(k)}_{i|T}\}$ depend on the minimization of (4). The approach taken by Willsky et al is to combine the estimates as if they were independent, and then to subtract the correlation effects. The formulas resulting are elegant (i.e., compact and a posteriori intuitive) but appear ill suited for computational purposes. The problem seems to be that the Willsky et al formulation requires considerable storage and computation; moreover, from a numerical point of view it is bad practice to subtract the correlated portion of the estimate, viz. because of word length and roundoff considerations $Ax - Ay \neq A(x-y)$.

An alternate covariance based approach, Andrisani/Gau [7], appears to overcome the numerical concerns present in the Willsky et al approach. However, Andrisani and Gau only address the filtering problem, and a preliminary assessment of their formulation leads us to believe that their approach involves considerably more storage and computation than does our information based approach.

The information array approach proposed here has (among others) the following desirable attributes:

I-1 Data and estimate combinations are accomplished using orthogonal transformation techniques that have a well earned reputation for numerical reliability.

I-2 Processing is arranged so that arrays to be combined are statistically disjoint; numerically questionable subtractions and cancellations are avoided.

I-3 The formulas needed for smoothing require only modest storage requirements, computer implementation and testng are yet to be done, but the scenario envisioned makes heavy use of existing certified software [8] that seems far more efficient than the approaches suggested in [6] and [7].

II. THE DECENTRALIZED SRIF/SRIS

Let us turn attention to development of our approach; *we rely on the SRIF orthogonal transformation* methodology described in [5], and assume the reader is familiar with both the method and jargon. Suppose that data type-1 were processed using the standard SRIF mechanization:

Measurement Processing

$$T^{(1)}_{j|j}\begin{bmatrix} R^{(1)}_{j|j-1} & z^{(1)}_{j|j-1} \\ H_j^{(1)} & z_j^{(1)} \end{bmatrix} = \begin{bmatrix} R^{(1)}_{j|j} & z^{(1)}_{j|j} \\ 0 & e_j^{(1)} \end{bmatrix} \tag{6}$$

Time Update

$$T^{(1)}_{j|j-1}\begin{bmatrix} R_\omega(j) & 0 & z_\omega(j) \\ R^{(1)}_{j|j}\,\Phi_j^{-1}B & R^{(1)}_{j|j}\,\Phi_j^{-1} & z^{(1)}_{j|j} \end{bmatrix} \tag{7}$$

$$= \begin{bmatrix} R_\omega^{*(1)}(j) & R_{\omega x}^{*(1)}(j) & z_\omega^{*(1)}(j) \\ 0 & R^{(1)}_{j+1|j} & z^{(1)}_{j+1|j} \end{bmatrix}$$

where

$$[R^{(1)}_{0|-1}\ \ z^{(1)}_{0|-1}] = [\bar{R}_0\ \bar{z}_0] \quad .$$

The matrices $T^{(1)}_{j|j}$ and $T^{(1)}_{j|j-1}$ are, as discussed in [5], implicitly defined orthogonal transformations that introduce zero blocks on the right side of (6) and (7), respectively, as indicated.

Remark: We avoid detail here, but note that for satellite applications one can use a pseudo-epoch state formulation in which Φ_j has a special, simple structure. Also, when the process noise is white or colored, step (7) is carried out one-component-at-a-time (see [5] and [8] for particulars).

In terms of the filter results (6)-(7), the performance index J_T can be rewritten as

$$J_T(x_0,\ldots,x_T) = \sum_{j=0}^{T} [e_j^{(1)}]^2$$

$$+ \sum_{j=0}^{T-1} \|R_\omega^{*(1)}(j)\,\omega_j + R_{\omega x}^{*(1)}(j)\,x_{j+1} - z_\omega^{*(1)}\|^2$$

$$+ \|R^{(1)}_{T|T}\,x_T - z^{(1)}_{T|T}\|^2 + \sum_{j=0}^{T} \|H_j^{(2)}\,x_j - z_j^{(2)}\|^2 \tag{8}$$

If one peruses the relation of the filter process (6)-(7) and compares how it is used to transform the performance index, it follows that the same structure can be used to transform the type 2 data terms appearing in (8).

Measurement Processing

$$T^{(2)}_{j|j}\begin{bmatrix} R^{(2)}_{j|j-1} & z^{(2)}_{j|j-1} \\ H^{(2)}_{j} & z^{(2)}_{j} \end{bmatrix} = \begin{bmatrix} R^{(2)}_{j|j} & z^{(2)}_{j|j} \\ 0 & e^{(2)}_{j} \end{bmatrix} \quad (9)$$

Time Update

$$T^{(2)}_{j+1|j}\begin{bmatrix} -R^{(2)}_{j|j}\,\Phi_j^{-1}B & R^{(2)}_{j|j}\,\Phi_j^{-1} & z^{(2)}_{j|j} \end{bmatrix}$$

$$= \begin{bmatrix} R^{*(2)}_{\omega}(j) & R^{*(2)}_{\omega x}(j) & z^{*(2)}_{\omega}(j) \\ 0 & R^{(2)}_{j+1|j} & z^{(2)}_{j+1|j} \end{bmatrix} \quad (10)$$

where now, $[R^{(2)}_{0|-1}\ z^{(2)}_{0|-1}] = [0\ 0]$. In the case of (9)-(10) we have not used the a priori on ω_j or on x_0 and as a result it is not necessarily true that the SRIF coefficient arrays $R^{(2)}_{j|j}$ and $R^{(2)}_{j+1|j}$ are non-singular; indeed, often they will be singular. This is not a limitation because at times that a solution is wanted we will combine the results of (9)-(10) with the corresponding results from the data type-1 processing. (Incidentally data type-1 can be empty so that the result of that processing would be just the process noise and initial condition a priori effects.)

Return now to the transformed functional (8), and recast it one step further in terms of the results of (10);

$$J_T(x_0,\dots,x_T) = \sum_{j=0}^{T} [e_j^{(1)}]^2 + [e_j^{(2)}]^2$$

$$+ \sum_{j=0}^{T} \left\| \begin{bmatrix} R^{*(1)}_{\omega}(j) \\ R^{*(2)}_{\omega}(j) \end{bmatrix} \omega_j + \begin{bmatrix} R^{*(1)}_{\omega x}(j) \\ R^{*(2)}_{\omega x}(j) \end{bmatrix} x_{j+1} - \begin{matrix} z^{*(1)}_{\omega}(j) \\ z^{*(2)}_{\omega}(j) \end{matrix} \right\|^2$$

$$+ \left\| \begin{bmatrix} R^{(1)}_{T|T} \\ R^{(2)}_{T|T} \end{bmatrix} x_T - \begin{matrix} z^{(1)}_{T|T} \\ z^{(2)}_{T|T} \end{matrix} \right\|^2 \quad (11)$$

To be able to create filter estimates and covariances at time T, and smooth estimates and covariances for the entire time span one needs the star terms and terminal SRIF data array. Observe that the dimensionality of what needs to be saved for later use is independent of the number of measurements in the observation batches. This is especially important for (batch sequential) applications with large data sets.

To proceed further, start with the star sum in (11), and consider the following algoritnm

$$T_j^{*}\begin{bmatrix} R^{*(1)}_{\omega}(j) & R^{*(1)}_{\omega x}(j) & z^{*(1)}_{\omega}(j) \\ R^{*(2)}_{\omega}(j) & R^{*(2)}_{\omega x}(j) & z^{*(2)}_{\omega}(j) \\ -H^{*}(j-1)\,\Phi_{j-1}^{-1}B & H^{*}(j-1)\,\Phi_{j-1}^{-1} & z^{*}(j-1) \end{bmatrix}$$

$$= \begin{bmatrix} R^{*}_{\omega}(j) & R^{*}_{\omega x}(j) & z^{*}_{\omega}(j) \\ 0 & H^{*}(j) & z^{*}(j) \\ 0 & 0 & \# \end{bmatrix} \quad (12)$$

for $j = 0,\dots, T-1$; where $[H^{*}(-1)\ z^{*}(-1)] = [0\ 0]$ and $H^{*}(j)$ is upper triangular. This recursion is very much like the standard SRIF time update algorithm, and is therefore easy to implement with off-the shelf software [8].

To obtain the globally optimal information vector $z_{j|j}$ and square root information matrix $R_{j|j}$, we solve the following equations using $H^{*}(j)$ and $z^{*}(j)$ from (12):

$$\hat{T}\begin{bmatrix} R^{(1)}_{j|j} & z^{(1)}_{j|j} \\ R^{(2)}_{j|j} & z^{(2)}_{j|j} \\ H^{*}(j-1) & z^{*}(j-1) \end{bmatrix} = \begin{bmatrix} R_{j|j} & z_{j|j} \\ & \\ 0 & \# \end{bmatrix} \quad (13)$$

where $\hat{T}$ is an orthogonal transformation triangularization operator. Globally optimal filter estimates and covariances are then given by

$$x_{j|j} = R^{-1}_{j|j}\, z_{j|j}$$
$$P_{j|j} = R^{-1}_{j|j}\, R^{-T}_{j|j} \quad (14)$$

To obtain the globally optimal smoothed estimates and covariances using the globally optimal Dyer-McReynolds smoothing coefficients from (12), a Covariance *Smoother* (15) or SRIS (see [1], [2]) may be used:

$$\omega_{j|T} = [R^{*}_{\omega}(j)]^{-1}\,[z^{*}_{\omega}(j) - R^{*}_{\omega x}(j)\, x_{j+1|T}]$$

$$x_{j|T} = \Phi_j^{-1}[x_{j+1|T} - B_{\omega j|T}] \quad (15)$$

$$P_{j|T} = \Phi_j^{-1}\{(I + L_j^{*}R^{*}_{\omega x}(j))\, P_{j+1|T}(I+L_j^{*}R^{*}_{\omega x}(j))^{T}$$
$$+ L_j^{*}(L_j^{*})^{T}\}\Phi_j^{-T}$$

where $L_j^{*} = B[R^{*}_{\omega}(j)]^{-1}$.

When the one component-at-a-time process noise methodology is applied one can avoid the R^{*}_{ω} triangular matrix inversions; more important is the observation that in the one at a time case the smooth covariance update will then be a rank-2 adjustment that is readily implemented in terms of UD covariance factors [5], [8].

III. DISCUSSION AND CONCLUSIONS

In this paper a procedure was described for the separate processing of independent data sets, after which the results are combined. The idea is to extend an approach that has been successfully used to solve least-squares process noise free problems. To keep the analysis and notation tractable our presentation uses a single (total) state model rather than assume separate local models, each of which is a subset of the total system state. The latter case is accommodated by means of standard matrix partitioning to account for the problem structure. Removing unnecessary global states from the local models dramatically reduces storage and computational requirements.

When there are several data types to be processed and combined one can simply cumulatively apply the methodology pairwise, and then repeat the process on these results. For example, with eight data types, four computing blocks can be used to calculate four sets of smoothing coefficients, information vectors, and square root information matrices. Each block solves (12) and (13). The four sets can then be divided into two pairs with each pair an input to one of two additional computing blocks. The final output pair is then merged using one final block to obtain globally optimal values.

Merits to the approach proposed here include:

III-1 Estimates based on independent data sets are often themselves of interest.

III-2 The methodology lends itself to networking (viz. concurrent processing on VAXes or (in the future) hypercube processing.

III-3 Separating the data sets is useful for operational applications, as this approach may more readily detect faulty or inconsistent data from one of the data groups.

III-4 The SRIF based decentralized algorithm can be arranged so that it relies, in the main, on established, well tested, numerically reliable FORTRAN subroutines [8].

III-5 Our methods allow one to change the a priori initial state covariance and process noise variances; it is not necessary to reprocess the data. This feature is especially attractive for problems with large amounts of data, for which we are trying to "tune" the filter.

III-6 In applications where each data set involves only a small portion of the total filter state vector, decentralization of the SRIF can significantly reduce computation.

III-7 Depending on observability (and/or rank deficiency) it may be possible to identify bias parameters and initial state a posteriori uncertainties.

The basic formulation described here has been refined so that the SRIF/SRIS process noise inclusion algorithm due to Bierman, [5], is used. That formulation allows for singular colored noise transitions and impacts on both the time update and the smooth recursions, and though it requires a lengthier explanation, it corresponds to a more compact, efficient and reliable computer implementation. Tests are underway to demonstrate the methodology described in this paper and compare numerical results with estimates and covariances obtained by other decentralized and centralized algorithms. In addition, these tests will quantify the relative amounts of algorithm complexity and efficiencies.

REFERENCES

[1] G.J. Bierman, "Sequential square root filtering and smoothing of discrete linear systems," Automatica, Vol. 10, pp. 147-158, 1974.

[2] P.G. Kaminski and A.E. Bryson, "Discrete square root smoothing," Proc. 1972 AIAA Guidance and Control Conf., AIAA Paper No. 72-877, 1972.

[3] T. Kerr and L. Chin, "The theory and techniques of discrete-time decentralized filters," in Advances in the Techniques and Technology of the Application of Nonlinear Filters and Kalman Filters, ed. C.T. Leondes, AGARDograph No. 256, 1982.

[4] F.C. Schweppe, Uncertain Dynamic Systems, Prentice-Hall, 1973.

[5] G.J. Bierman, Factorization Methods for Discrete Sequential Estimation, Academic Press, 1977.

[6] A.S. Willsky, M.G. Bello, D.A. Castonon, B.C. Levy, and G.C. Verghese, "Combining and updating of local estimates and regional maps along sets of one-dimensional tracks", IEEE Trans. Automat. Contr., Vol. AC-27, pp.799-813, Aug. 1982.

[7] D. Andrisani and C.F. Gau, "Multistage linear estimation using partitioning," IEEE Trans. Automat. Contr., Vol. AC-30, pp. 182-185, 1985.

[8] G.J. Bierman and K.H. Bierman, "Estimation subroutine library (preliminary, user-guide - August 1984)," Factorized Estimation Applications, Inc., Report No. 81584.

Maximum Likelihood Estimation Using Square Root Information Filters

GERALD J. BIERMAN, FELLOW, IEEE, MITCHELL R. BELZER, MEMBER, IEEE, JAMES S. VANDERGRAFT, AND DAVID W. PORTER, MEMBER, IEEE

Abstract—**The method of maximum likelihood has been previously applied to the problem of determining the parameters of a linear dynamical system model. Calculation of the maximum likelihood estimate may be carried out iteratively by means of a scoring equation which involves the gradient of the negative log likelihood function and the Fisher information matrix. Evaluation of the latter two requires implementation of a Kalman filter (and its derivative with respect to each parameter) which is known to be unstable. In this paper, we derive equations which can be used to obtain the maximum likelihood estimate iteratively but based upon the square root information filter (SRIF). Unlike the conventional Kalman filter, the SRIF avoids numerical instabilities arising from computational errors. Thus, our new algorithm should be numerically superior to a Kalman filter mechanization.**

I. Introduction

The maximum likelihood approach to parameter estimation is a very general method which has been applied to the problem of determining the parameters of a linear dynamical system model as described, for example, by Astrom [1] or Goodwin and Payne [4]. Let the discrete-time dynamics be given as

$$x_{k+1} = \Phi_k x_k + G_k w_k; \qquad k = 0, 1, \cdots, N \tag{1.1}$$

where the state vector $x_k \in R^n$, the state transition matrix $\Phi_k \in R^{n \times n}$, $G_k \in R^{n \times q}$ and w_k is a zero-mean white Gaussian noise process with covariance Q_k, i.e., $w_k \sim N(O, Q_k)$. Let the discrete-time measurement model be given as

$$y'_k = H'_k x_k + v'_k \tag{1.2}$$

where the measurement vector $y'_k \in R^m$, the measurement matrix $H'_k \in R^{m \times n}$, and the noise is $v'_k \sim N(O, P_{v_k})$. Additionally, the matrices Φ_k, G_k, H'_k, Q_k, and P_{v_k} are functions of the unknown parameter vector $\theta \in R^p$. The maximum likelihood estimate is the value of θ which maximizes the joint probability that y'_k, for $k = 0, 1, \cdots, N$, is equal to the actual measurements in hand. This is equivalent to minimizing the negative logarithm of the latter conditional probability density function, i.e., the negative log likelihood function which is given as

$$J = \frac{1}{2} \sum_{k=1}^{N} \{\nu_k^T B_k^{-1} \nu_k + \log \det B_k\} \tag{1.3}$$

where ν_k and B_k are the *a priori* residuals and residual covariances, respectively, from a Kalman filter. That is

$$B_k = H'_k P_k(-) H_k'^T + P_{v_k} \tag{1.4}$$

where $P_k(-)$ is the predicted estimate error covariance. Other Kalman filter quantities that will be needed in the sequel are: the Kalman gain

$$K_k = P_k(-) H_k'^T B_k^{-1}, \tag{1.5}$$

the updated error covariance

$$P_k(+) = (I - K_k H'_k) P_k(-), \tag{1.6}$$

the residual (or innovation)

$$\nu_k = y'_k - H'_k x_k(-), \tag{1.7}$$

and the updated estimate

$$x_k(+) = x_k(-) + K_k \nu_k. \tag{1.8}$$

The calculation of the maximum likelihood estimate may be carried out iteratively by means of the scoring equation

$$\hat{\theta}_{i+1} = \hat{\theta}_i - (F|_{\hat{\theta}_i})^{-1} \left(\frac{\partial J}{\partial \theta}\bigg|_{\hat{\theta}_i} \right) \tag{1.9}$$

where i denotes iteration number and F is the Fisher information matrix. Iterative algorithms such as (1.9) will, in general, converge in fewer steps than ones which involve evaluation of J exclusively. On the other hand, algorithms which incorporate $\frac{\partial J}{\partial \theta}$ and F will require more computations in each step. In this paper, we derive equations which can be used to obtain the maximum likelihood estimate iteratively but based upon the square root information filter (SRIF) as developed by Dyer and McReynolds [3] and extended by Bierman [2]. The SRIF measurement update is

$$T_k \begin{bmatrix} R_k(-) & z_k(-) \\ H_k & y_k \end{bmatrix} = \begin{bmatrix} R_k(+) & z_k(+) \\ 0 & e_k \end{bmatrix} \tag{1.10}$$

where T_k is an orthogonal matrix such that $R_k(+)$ is upper triangular. Also, y_k and H_k correspond to normalization of (1.2) so that

$$y_k = R_{v_k} y'_k \text{ and } H_k = R_{v_k} H'_k$$

where

$$P_{v_k} = R_{v_k}^{-1} R_{v_k}^{-T} \tag{1.11}$$

and the measurement noise now has identity covariance. The SRIF predictive step is

$$\begin{aligned} \tilde{T}_{k+1} &\begin{bmatrix} R_{w_k} & 0 & 0 \\ -R_k(+)\phi_k^{-1}G_k & R_k(+)\Phi_k^{-1} & z_k(+) \end{bmatrix} \\ &= \begin{bmatrix} R_{w_{k+1}}(-) & R_{wx_{k+1}}(-) & z_{w_{k+1}}(-) \\ 0 & R_{k+1}(-) & z_{k+1}(-) \end{bmatrix} \end{aligned} \tag{1.12}$$

Manuscript received September 21, 1988, revised June 29, 1989. Paper recommended by Past Associate Editor at Large, D. M. Wiberg. This work was supported in part by the Johns Hopkins University Applied Physics Laboratory.

G. J. Bierman was with Bierman and Associates, Studio City, CA 91604. He is now deceased.

M. R. Belzer is with Mentor Technologies, Inc., Rockville, MD 20850.

J. S. Vandergraft and D. W. Porter are with Coleman Research Corporation, Laurel, MD 20708.

IEEE Log Number 9039470.

0018-9286/90/1200-1293$01.00 © 1990 IEEE

where

$$Q_k = R_{w_k}^{-1} R_{w_k}^{-T} \tag{1.13}$$

and $\bar{T}_{k+1}$ is an orthogonal diagonal matrix such that $R_{w_{k+1}}(-)$ and ($R_{k+1}(-)$ if desired) are upper triangular. The SRIF data array is $[R_k(\pm), z_k(\pm)]$. It is related to the Kalman filter state estimate $x_k(\pm)$ and estimate error covariance $P_k(\pm)$ by

$$x_k(\pm) = R_k^{-1}(\pm) z_k(\pm) \tag{1.14}$$

$$P_k(\pm) = R_k^{-1}(\pm) R_k^{-T}(\pm). \tag{1.15}$$

The quantity e_k in (1.10) will be shown below to be the square of the normalized measurement residual. The submatrix $R_{wx_{k+1}}(-)$ in (1.12) is useful for smoothing but will not be used further here.

The purpose of this paper is to present a new algorithm for evaluating the likelihood function and its gradient by using quantities that are generated naturally by the SRIF. This avoids altogether the use of the Kalman filter with its inherent numerical instabilities. In Section II, we derive expressions for the likelihood function in terms of SRIF quantities. Sections III and IV show how to obtain the gradient vector, and Section V gives an outline of the complete algorithm for using the SRIF to evaluate the likelihood function and its gradients. Some numerical results are shown in Section VI. We have also given, in Section IV, a first attempt at a formula for the Fisher information matrix. Limited numerical experiments have indicated that this may be a useful result; however, further work needs to be done in this direction.

It is worth noting that another approach to this ML estimation problem is to use a stable covariance filter, such as the UD filter [6] to generate the quantities needed to evaluate the likelihood function according to (1.3). However, it is not known, at this time, how to use the UD filter to generate the gradient. This, too, is an area for future research.

II. Likelihood Function in Terms of SRIF Variables

In order to evaluate the likelihood function efficiently when a SRIF has been used to filter the data, it is necessary to express the quantities that occur in J directly in terms of quantities that are computed by the SRIF. That is, the two parts of (1.3)

$$\nu_k^T B_k^{-1} \nu_k \ \text{(data dependent part)} \tag{2.1}$$

$$\log\det B_k \ \text{(model dependent part)} \tag{2.2}$$

must be expressed in terms of the quantities that appear in the SRIF formulas in (1.10) and (1.12). The two results of interest are as follows:

$$\|e_k\|^2 = \nu_k^T B_k^{-1} \nu_k \tag{2.3}$$

and

$$2\log\left(\frac{\det R_k(+)}{\det R_k(-)}\right) = \log\det B_k. \tag{2.4}$$

The proof of (2.3) is based on the eq''tion

$$T_k\left[\begin{bmatrix} R_k(-) \\ H_k \end{bmatrix} x_k(+) \quad \begin{matrix} z_k(-) \\ y_k \end{matrix}\right]\begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

$$= \left[\begin{bmatrix} R_k(+) \\ O \end{bmatrix} x_k(+) \quad \begin{matrix} z_k(+) \\ e_k \end{matrix}\right]\begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

where T_k is an orthogonal transformation. Thus, taking norms gives

$$\left\|\begin{bmatrix} R_k(-) \\ H_k \end{bmatrix} x_k(+) - \begin{bmatrix} z_k(-) \\ y_k \end{bmatrix}\right\|^2$$

$$= \left\|\begin{bmatrix} R_k(+) \\ O \end{bmatrix} x_k(+) - \begin{bmatrix} z_k(+) \\ e_k \end{bmatrix}\right\|^2$$

$$= \|R_k(+) x_k(+) - z_k(+)\|^2 + \|e_k\|^2.$$

Equation (1.14) implies that

$$R_k(+) x_k(+) - z_k(+) = 0$$

hence

$$\|e_k\|^2 = \|R_k(-) x_k(+) - z_k(-)\|^2 + \|H_k x_k(+) - y_k\|^2.$$

Next, we use (1.8) to write this as

$$\|e_k\|^2 = \|R_k(-)(x_k(-) + K_k \nu_k) - z_k(-)\|^2$$
$$+ \|H_k(x_k(-) + K_k \nu_k) - y_k\|^2.$$

A simple rearrangement gives

$$\|e_k\|^2 = \|R_k(-) K_k \nu_k\|^2 + \|H_k K_k \nu_k - \nu_k\|^2$$

where (1.14) has again been used. Hence

$$\|e_k\|^2 = \nu_k^T [K_k^T R_k^T(-) R_k(-) K_k$$
$$+ (H_k K_k - I)^T (H_k K_k - I)] \nu_k. \tag{2.5}$$

Finally, by using (1.4), (1.5), and (1.15), it can be shown that the matrix inside the square brackets of (2.5) is just B_k^{-1}, which proves (2.3).

The proof of (2.4) is based on the existence of an orthogonal transformation between certain SRIF variables and residual covariances. Specifically, let

$$U = \begin{bmatrix} I & H_k R_k^{-1}(-) \\ O & R_k^{-1}(-) \end{bmatrix}$$

and

$$V = \begin{bmatrix} B_k^{1/2} & O \\ K_k & R_k^{-1}(+) \end{bmatrix}.$$

Then simple matrix algebra, again using (1.4), (1.5), (1.6), and (1.15) to simplify terms, shows that

$$UU^T = VV^T.$$

Hence, if we let

$$T = U^{-1} V \tag{2.6}$$

then

$$TT^T = U^{-1} V V^T U^{-T} = U^{-1} U U^T U^{-T} = I$$

so that T is an orthogonal matrix. By writing (2.6) as

$$UT = V$$

we have

$$\det(U) \cdot \det(T) = \det(V).$$

Hence

$$\det(U) = \det R_k^{-1}(-)$$

$$\det(T) = 1$$

$$\det(V) = \det(B_k^{1/2}) \cdot \det R_k^{-1}(+)$$

so that

$$\det R_k^{-1}(-) = \det B_k^{1/2} \det R_k^{-1}(+)$$

or

$$\det B_k^{1/2} = \frac{\det R_k(+)}{\det R_k(-)}.$$

By taking logarithms, and using the fact that

$$\det B_k = \det\left(B_k^{1/2}(B_k^{1/2})^T\right) = \left[\det(B_k^{1/2})\right]^2$$

we obtain (2.4). A simple modification of the above arguments, where the measurement noise is also parameter dependent, gives the following general formula for the negative log likelihood function in terms of SRIF variables:

$$J = \sum_{k=1}^{N} \frac{1}{2}\|e_k\|^2 + \log\left(\frac{\det R_k(+)}{\det R_{v_k} \cdot \det R_k(-)}\right).$$

III. Gradient of the Likelihood Function

To evaluate the gradient of J, note that the gradient of the model-dependent part is

$$\begin{aligned}\frac{\partial}{\partial\theta}&\{\log\det R_k(+) - \log\det R_k(-) - \log\det R_{v_k}\}\\ &= \operatorname{tr}\left[R_k^{-1}(+)\frac{\partial R_k(+)}{\partial\theta}\right]\\ &\quad - \operatorname{tr}\left[R_k^{-1}(-)\frac{\partial R_k(-)}{\partial\theta}\right]\\ &\quad - \operatorname{tr}\left[R_{v_k}^{-1}\frac{\partial R_{v_k}}{\partial\theta}\right].\end{aligned}$$

Thus, since all of these are triangular matrices, only the diagonal elements of $R_k^{-1}(+)$, $R_{v_k}^{-1}$, $R_k^{-1}(-)$, $\frac{\partial R_{v_k}}{\partial\theta}$, $\frac{\partial R_k(+)}{\partial\theta}$, and $\frac{\partial R_k(-)}{\partial\theta}$ need to be computed. Diagonal elements of the first three matrices are obtained by an inexpensive partial inversion of their SRIF values. Diagonal elements of the latter two matrices can be computed by using the method described in Section IV.

For the gradient of the data-dependent part of the likelihood function, we use the SRIF measurement update relationship

$$T_k\begin{bmatrix} z_k(-) \\ R_{v_k} y_k' \end{bmatrix} = \begin{bmatrix} z_k(+) \\ e_k \end{bmatrix}$$

where T_k is an orthogonal matrix. By taking norms, we find that

$$\|e_k\|^2 = \|z_k(-)\|^2 + \|R_{v_k} y_k'\|^2 - \|z_k(+)\|^2.$$

Thus

$$\begin{aligned}\frac{\partial\|e_k\|^2}{\partial\theta} &= 2z_k^T(-)\frac{\partial z_k(-)}{\partial\theta}\\ &\quad + 2y_k'^T R_{v_k}^T \frac{\partial R_{v_k}}{\partial\theta} y_k'\\ &\quad - 2z_k^T(+)\frac{\partial z_k(+)}{\partial\theta}.\end{aligned}$$

Again, $\frac{\partial z_k(\pm)}{\partial\theta}$ can be obtained by differentiating the SRIF, as in Section IV. $\frac{\partial R_{v_k}}{\partial\theta}$ can be obtained by differentiating

$$R_{v_k} P_{v_k} R_{v_k}^T = I$$

to obtain

$$R_{v_k}\frac{\partial P_{v_k}}{\partial\theta}R_{v_k}^T = -\frac{\partial R_{v_k}}{\partial\theta}R_{v_k}^{-1} - \left(\frac{\partial R_{v_k}}{\partial\theta}R_{v_k}^{-1}\right)^T. \quad (3.1)$$

However, the matrix

$$-\frac{\partial R_{v_k}}{\partial\theta}R_{v_k}^{-1}$$

is upper triangular, so it must equal $U + \frac{1}{2}D$, where U is the upper triangular part of the matrix on the left of (3.1), and D is the diagonal part. Then $\frac{\partial R_{v_k}}{\partial\theta}$ is found by backsolving a triangular system. To summarize, the gradient of the negative log likelihood function may be written as

$$\begin{aligned}\frac{\partial J}{\partial\theta} = \sum_{k=1}^{N} & z_k^T(-)\frac{\partial z_k(-)}{\partial\theta} + y_k'^T R_{v_k}^T \frac{\partial R_{v_k}}{\partial\theta} y_k'\\ & - z_k^T(+)\frac{\partial z_k(+)}{\partial\theta}\\ & + \operatorname{tr}\left[R_k^{-1}(+)\frac{\partial R_k(+)}{\partial\theta}\right]\\ & - \operatorname{tr}\left[R_k^{-1}(-)\frac{\partial R_k(-)}{\partial\theta}\right]\\ & - \operatorname{tr}\left[R_{v_k}^{-1}\frac{\partial R_{v_k}}{\partial\theta}\right].\end{aligned}$$

where al' 'he quantities involved here are produced directly by the SRIF, or are easily computed by solving triangular systems.

Finally, to express the Fisher information matrix in terms of SRIF variables, recall that the l, m element of the Fisher information matrix is

$$F_{l,m} = E\left\{\frac{\partial J}{\partial\theta_l}\frac{\partial J}{\partial\theta_m}\right\}.$$

It is straightforward to show that $\frac{\partial J_k}{\partial\theta}$ is a white sequence. Thus

$$F_{l,m} = \sum_{k=1}^{N} E\left\{\frac{\partial J_k}{\partial\theta_l}\frac{\partial J_k}{\partial\theta_m}\right\} \quad (3.2)$$

where J_k is the kth term in the time series summation for J. The remaining steps are to rewrite (3.2) using the SRIF form of J as well as the facts that $E\left\{\frac{\partial J}{\partial\theta}\right\} = 0$, $e_k \sim N(0, I)$, and $\frac{\partial e_k}{\partial\theta_l}$ is also Gaussian. The l, m element of the Fisher information matrix becomes

$$F_{l,m} = \sum_{k=1}^{N} \operatorname{tr} E\left\{\frac{\partial e_k}{\partial\theta_l}\frac{\partial e_k^T}{\partial\theta_m}\right\} - \operatorname{tr} E\left\{e_k\frac{\partial e_k^T}{\partial\theta_l}\right\}E\left\{e_k\frac{\partial e_k^T}{\partial\theta_m}\right\}. \quad (3.3)$$

This formula could be used by replacing the expected values with sample values. This same replacement is regularly done when using the standard Kalman filter approach to the problem. (See, for example, [5]). However, in the Kalman formulation, part of the expression for the Fisher matrix can be written without expectations. It is possible that parts of (3.3) can be evaluated exactly also.

IV. Derivatives of SRIF Variables

In this section, we describe a numerically efficient and accurate method for computing the quantities $\frac{\partial R_k(\pm)}{\partial \theta}$ and $\frac{\partial z_k(\pm)}{\partial \theta}$ that are needed for the formulas of Section III. To simplify the discussion, we observe that the SRIF transformations in (1.10) and (1.12) have the form

$$QA = R \tag{4.1}$$

where A is a rectangular matrix, and Q is an orthogonal matrix which when multiplied by A gives an upper trapezoidal matrix R. The elements of A are differentiable functions of a parameter α. Then, given the matrix of derivatives $A_\alpha = \frac{da_{ij}}{d\alpha}$, we wish to determine the matrix R_α. The ideas involved in deriving the necessary formulas can be readily explained by studying this simple situation. The generalization to the actual SRIF transformations, and to the case where α is replaced by a vector of parameters θ is very straightforward. In fact, for now we will even assume that A, hence R, are square and nonsingular.

For completeness, we first describe two fairly obvious ways to solve this problem which, however, may lead to numerical difficulties and are not recommended. The first of these methods is based on using (4.1) to write the equation $R^TR = (QA)^T(QA) = A^TA$.

Then straightforward differentiation gives

$$R^TR_\alpha + \left(R^TR_\alpha\right)^T = A_\alpha^TA + A^TA_\alpha. \tag{4.2}$$

By using the fact that R, and hence R_α, is upper triangular, (4.2) can be used to compute the elements of R_α. This involves a forward substitution algorithm that is similar to the standard algorithm for solving a linear system. The numerical problem that might arise here occurs when A, hence R, is ill-conditioned (i.e., nearly singular). The forward substitution algorithm may then give inaccurate results.

The second questionable method for computing R_α is to differentiate (4.1) directly to get

$$R_\alpha = QA_\alpha + Q_\alpha A. \tag{4.3}$$

The matrix Q is a product of Householder matrices (elementary reflectors) whose derivatives can be determined recursively from A_α. Thus, Q_α can be computed by a simple modification of the algorithms used to compute Q itself. Notice, however, that R_α is upper triangular whereas the matrices QA_α and $Q_\alpha A$ on the right side of (4.3) are full. Therefore, total subtractive cancellation occurs when computing the lower triangular part of R_α from (4.3).

The method of choice is based on the observation that since Q is orthogonal, $QQ^T = I$. Hence, by differentiating this equation, we find that

$$Q_\alpha Q^T + QQ_\alpha^T = 0 \text{ or } Q_\alpha Q^T = -\left(Q_\alpha Q^T\right)^T. \tag{4.4}$$

A matrix S that satisfies the equation $S^T = -S$ is said to be skew symmetric. It is easily seen that such a matrix has the form $L^T - L$, where L is strictly lower triangular. Thus, from (4.4), we can write

$$Q_\alpha Q^T = L^T - L \tag{4.5}$$

for some lower triangular matrix L. In fact, the matrix L is related through (4.1) to A_α as follows.

Lemma: The lower triangular matrix L in (4.5), where Q satisfies (4.1) with A nonsingular, is in fact the lower triangular part of the matrix $QA_\alpha R^{-1}$. That is

$$QA_\alpha R^{-1} = L + D + U \tag{4.6}$$

where L, D, and U are, respectively, lower triangular, diagonal, and upper triangular, and L satisfies (4.5).

Proof: From (4.3) we have

$$R_\alpha R^{-1} = QA_\alpha R^{-1} + Q_\alpha AR^{-1}.$$

Hence

$$R_\alpha R^{-1} = QA_\alpha R^{-1} + Q_\alpha Q^T \tag{4.7}$$

where (4.1) has been used to replace $Q_\alpha AR^{-1}$ by $Q_\alpha Q^T$. Now, R_α and R are upper triangular and, therefore, so are R^{-1} and $R_\alpha R^{-1}$. Thus, the lower triangular part of $QA_\alpha R^{-1}$ must exactly cancel the lower triangular part of $Q_\alpha Q^T$. Hence, if $Q_\alpha Q^T = L^T - L$, then $QA_\alpha R^{-1} = L + D + U$.

Equation (4.7) leads easily to a method for computing R_α. By substituting (4.6) and (4.5) into (4.7), we find that

$$R_\alpha R^{-1} = (L + D + U) + (L^T - L) = L^T + D + U \tag{4.8}$$

and therefore, $R_\alpha = (L^T + D + U)R$. That is, to determine R_α: compute $QA_\alpha R^{-1}$ and write it as $L + D + U$. Then compute $(L^T + D + U)R$. The result is R_α.

Equation (4.8) clearly shows the danger inherent in using (4.3) directly. That is, from (4.8) we find that

$$\begin{aligned} R_\alpha &= (L + D + U)R + (L^T - L)R \\ &= \underbrace{LR + (D + U)R}_{QA_\alpha} + \underbrace{L^TR - LR}_{Q_\alpha A}. \end{aligned}$$

The first term here is QA_α, the second is $Q_\alpha A$. Written this way, it is seen that the matrix LR occurs in both terms, but with opposite sign. Moreover, LR is a full matrix, so that cancellation occurs throughout the matrix sum $QA_\alpha - Q_\alpha A$. If LR is small, compared to the elements of $(D + U)R$ and L^TR, then this sum will be computed accurately. However, if some elements of LR are large, then the corresponding elements of the sum will suffer from severe cancellation and may be totally inaccurate.

V. Summary of the Algorithm

The ideas of Section IV can be used to develop a very concise algorithm for computing the likelihood function, its gradient, and the terms that appear in (3.3) for the Fisher matrix. Let $\underline{\theta} = (\theta_1, \theta_2, \cdots, \theta_m)$ denote the vector of parameters with respect to which the likelihood function is to be differentiated. To compute $\frac{\partial R_k(+)}{\partial \theta_i}$ and $\frac{\partial z_k(+)}{\partial \theta_i}$, for each i, we modify the measurement update (1.10) as follows.

M1: Replace (1.10) by

$$T_k \begin{bmatrix} R_k(-) & z_k(-) & \frac{\partial R_k(-)}{\partial \theta_i} & \frac{\partial z_k(-)}{\partial \theta_i} \\ H_k & Y_k & \frac{\partial H_k}{\partial \theta_i} & O \end{bmatrix}$$

$$= \begin{bmatrix} R_k(-) & z_k(+) & A_i & B_i \\ O & e_k & C_i & E_i \end{bmatrix}$$

where the last two columns are repeated for each θ_i, $i = 1, 2, \cdots, m$.

M2: Compute, for each $i = 1, 2, \cdots, m$,

$$\begin{bmatrix} A_i & B_i \\ C_i & E_i \end{bmatrix} \begin{bmatrix} R_k(+) & z_k(+) \\ O & e_k \end{bmatrix}^{-1} = L_i + D_i + U_i.$$

M3: Compute, for each i

$$\begin{bmatrix} \dfrac{\partial R_k(+)}{\partial \theta_i} & \dfrac{\partial z_k(+)}{\partial \theta_i} \\ O & \dfrac{\partial e_k}{\partial \theta_i} \end{bmatrix} = (L_i^T + D_i + U_i) \begin{bmatrix} R_k(+) & z_k(+) \\ O & e_k \end{bmatrix}.$$

Note that in step M2, the matrix to be inverted is upper triangular. Hence, this product can be computed by backsolving.

The ideas of Section IV cannot be applied directly to the SRIF time update (1.12) because the matrix to be triangularized is not square, hence, not invertible. However, by working with submatrices, the following algorithm results.

T1: Replace (1.12) by the following:

$$\tilde{T}_{k+1} \begin{bmatrix} R_{w_k} & O & O & \dfrac{\partial R_{w_k}}{\partial \theta_i} & O & O \\ S_1 & S_2 & z_k(+) & \dfrac{\partial S_1}{\partial \theta_i} & \dfrac{\partial S_2}{\partial \theta_i} & \dfrac{\partial z_k(+)}{\partial \theta_i} \end{bmatrix}$$

$$= \begin{bmatrix} Rw_{k+1}(-) & Rwx_{k+1}(-) & z_{w_{k+1}}(-) & X_i & Y_i & M_i \\ O & R_{k+1}(-) & z_{k+1}(-) & V_i & W_i & N_i \end{bmatrix}$$

where the last three columns are repeated for each θ_i, $i = 1, 2, \cdots, m$, and

$$S_1 = -R_k(+)\Phi_k^{-1}G_k \qquad S_2 = R_k(+)\Phi_k^{-1}$$

as in (1.12).

T2: Compute, for each i the following:

$$[V_i \quad W_i] \begin{bmatrix} Rw_{k+1}(-) & Rwx_{k+1} \\ O & R_{k+1}(-) \end{bmatrix}^{-1} = [*, L_i + D_i + U_i]$$

where * denotes the first n columns of this product, which are of no interest here.

T3: Compute, for each i, the following:

$$\frac{\partial R_{k+1}(-)}{\partial \theta_i} = [L_i^T + D_i + U_i] R_{k+1}(-)$$

and

$$\frac{\partial z_{k+1}(-)}{\partial \theta_i} = \left(\frac{\partial R_{k+1}(-)}{\partial \theta_i} - W_i \right) S_2^{-1} z_k(+) + N_i.$$

The equation for $\dfrac{\partial z_{k+1}(-)}{\partial \theta_i}$ is obtained by differentiating the equation

$$\tilde{T}_{k+1} \begin{bmatrix} O \\ z_k(+) \end{bmatrix} = \begin{bmatrix} z_{w_{k+1}}(-) \\ z_{k+1}(-) \end{bmatrix}$$

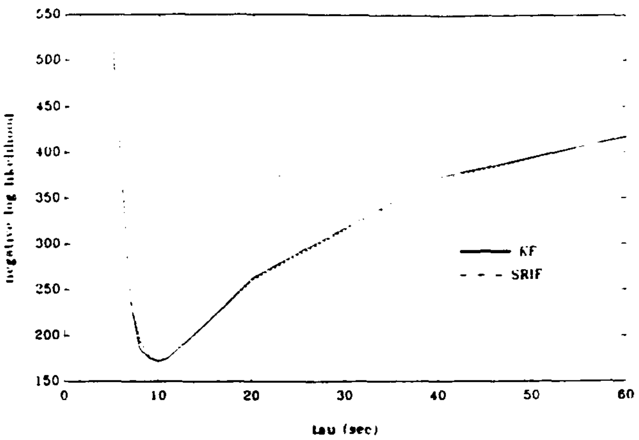


Fig. 1 Likelihood function for the conventional KF and the SRIF.

and then using (4.5) to replace the derivative of $\tilde{T}_{k+1}$ with quantities computed in step T1. Note also that S_2^{-1}, needed for this final equation, is easily obtained since

$$S_2^{-1} = \Phi_k R_k^{-1}(+)$$

where $R_k(+)$ is upper triangular.

Finally, it is interesting to observe that the quantities $\dfrac{\partial e_k}{\partial \theta_i}$ needed in (3.3) for the Fisher matrix are automatically produced by step M3 of the measurement update.

In the Kalman formulation of this gradient evaluation, it is necessary to run a "differentiated" Kalman filter for each of the parameters θ_i to be estimated. In the SRIF formulation, this "bank" of filters is replaced with augmented arrays to which orthogonal transformations are applied.

VI. Numerical Results

In order to check the aforementioned derivations, the algorithm given in Section V was applied to a simple example. The results were then compared to those produced by the standard Kalman filter approach. The example used for this comparison has dynamics equation

$$\begin{pmatrix} x_{k+1} \\ y_{k+1} \end{pmatrix} = \begin{pmatrix} 1 & \Delta t \\ O & e^{-\Delta t/\tau} \end{pmatrix} \begin{pmatrix} x_k \\ y_k \end{pmatrix} + \begin{pmatrix} O \\ w_k \end{pmatrix}$$

and measurement equations

$$z_k = (1 \quad 0) \begin{pmatrix} x_k \\ y_k \end{pmatrix} + v_k.$$

Here, τ is a parameter to be estimated. The likelihood function and its derivative, with respect to τ, were evaluated for several values of τ. Figs. 1 and 2 show the result of these evaluations using both the Kalman filter and the SRIF. Note that the two methods produce slightly different values for the derivative. However, it appears that both methods give the same zero point for the gradient.

Further numerical experiments need to be done to determine the accuracy and efficiency of this new algorithm.

VII. Conclusion

The SRIF has, in many applications, proven to be a numerically reliable formulation of the discrete Kalman filter. In this paper, we have shown how a very natural extension of the SRIF time and measurement updates can generate the likelihood function and its gradient with respect to unknown parameters. Preliminary numerical experience indicates that this method is both accurate and efficient.

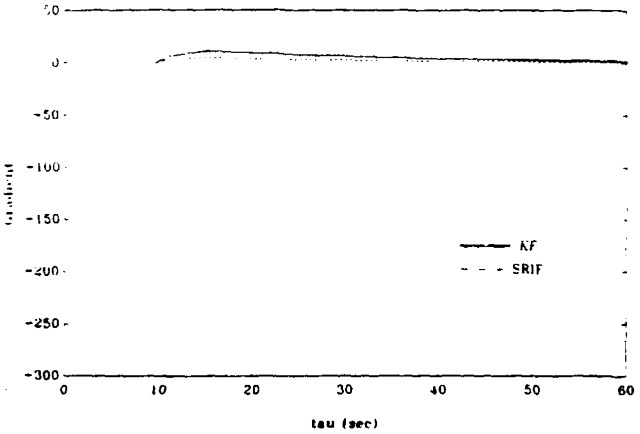


Fig. 2. Gradient for the conventional KF and the SRIF.

Acknowledgment

The authors would like to thank T. Mavromatis for the numerical resuits of Section VI.

References

[1] K. J. Astrom, "Maximum likelihood and prediction error methods," *Automatica*, vol. 16, 1980.

[2] G. J. Bierman, *Factorization Methods for Discrete Sequential Estimation*. New York: Academic, 1977.

[3] P. Dyer and S. McReynolds, "Extensions of square root filtering to include process noise," *J. Opt. Theory Appl.*, vol. 3, no. 6, pp. 444–459, 1969.

[4] G. Goodwin and R. Payne, *Dynamic System Identification*. New York: Academic, 1977.

[5] N. K. Gupta and R. K. Mehra, "Computational aspects of maximum likelihood estimation and reduction in sensitivity function calculations," *IEEE Trans. Automat. Contr.*, vol. AC-19, no. 6, pp. 774–783, 1974.

[6] C. L. Thornton and G. J. Bierman, "Filtering and error analysis via the UDUT covariance factorization," *IEEE Trans. Automat. Contr.*, vol. AC-23, no. 5, pp. 901–907, 1978.

Gerald J. Bierman (F'86), for a photograph and biography, see this issue, p. 1292.

Mitchell R. Belzer (S'80–M'82–S'82–M'84) was born in Bayside, NY, in 1953. He received the B.S. degree in physics from the State University of New York, Stony Brook, in 1976, the M.S. degree in bioengineering from Pennsylvania State University, University Park, and the Ph.D. degree in electrical engineering from the University of Maryland, College Park, in 1984.

Presently, he is Vice President and Principal Scientist at Mentor Technologies Inc., Rockville, MD, where he is currently managing the design and construction of a real-time tracking experiment based upon distributed estimation theory. His interests are in the application of estimation, identification and control theory, statistical pattern recognition, and image processing techniques for the research and development of advanced instrumentation systems.

He has published nine papers as conference proceedings, two papers in refereed journals, and has reviewed papers for publication in the IEEE Transactions on Automatic Control and the *Proceedings of the IEEE Decision and Control Conference*.

James S. Vandergraft was born in Gooding, ID. He received the B.S. degree, with distinction, and the M.S. degree, both in mathematics from Stanford University, Stanford, CA, and the Ph.D. degree in mathematics from the University of Maryland, College Park.

He was on the faculty of the Department of Computer Science at the University of Maryland from 1966 to 1979. Since then, he has worked as a consultant for government and commercial research centers. Currently, he is a Principal Scientist with Coleman Research Corporation, Laurel, MD. His research interests are in the areas of numerical analysis and computational algorithms. He has published numerous papers :nd reports in these fields, and is the author of a widely used textbook on numerical methods. He is currently involved in the development of stable and efficient algorithms for digital filtering and maximum likelihood parameter estimation.

David W. Porter (S'67–M'68–M'75) was born in Mitchellville, IA. He received the B.S., M.S., and Ph.D. degrees in electrical engineering from Iowa State University of Science and Technology, Ames, in 1968, 1970, and 1972, respectively.

Since 1978, he has been with Coleman Research Corporation, Laurel, MD, where he is Technical Director. His current research interest is microprocessor-based products and services for simulation, state estimation, parameter identification, signal processing, and control for industry and government. He has published widely in the areas of stability theory, state estimation, and parameter identification.

Distribution List

Commander
US Army White Sands Missile Range
ATTN: STEWS-ID-T
White Sands Missile Range, NM 88002-5143

Defense Technical Information Center
Cameron Station
Alexandria, VA 22304-6145