

*Definition 1.* An abstract distributed sensor network is a 4-tuple  $(\mathcal{X}, \mathcal{Y}, \mathcal{S}, \mathcal{C})$ , where  $\mathcal{X}$  is the space being observed by the sensors of the Distributed Sensor Network;  $\mathcal{Y}$  is the parameter space;  $\mathcal{S}$  is a collection of sensors; and  $\mathcal{C}$  is a collection of clusters, and these clusters pick up data from their respective regions and transmit them to the processors for further processing. The collection of regions, to which the various sensor clusters are allotted, constitute the  $\mathcal{X}$ . In the space of parameter values, the parameter being monitored can be represented as a vector  $p$ , whose elements being measured values for each sensor. These parameters being measured by a cluster  $c$  can be represented by  $p_c$ , where  $c \in \mathcal{C}$ . Space  $\mathcal{X}$  is partitioned into disjoint subspaces each  $S_j$  being assigned to one sensor  $s_j \in \mathcal{S}$ . Subspace  $S_j$  is the region of interest of sensor  $s_j$ .

## A General Computational Framework for Distributed Sensing and Fault-Tolerant Sensor Integration

S. S. Iyengar and L. Prasad

**Abstract**—This paper proposes an abstract framework to address the problem of fault-tolerant integration of information provided by multiple sensors. Extending our earlier results[10], this paper presents a formal description of spatially distributed sensor networks, where i) clusters of sensors monitor (possible overlapping) regions of the environment; ii) sensors return measured values of a multi-dimensional parameter of interest; and iii) uncertainties associated with a sensor output are represented by a connected subset in the parameter space. A method to obtain interval estimates of components of the actual parameter vector is developed, wherein information from faulty sensors (whose uncertainty subspaces do not contain the true value of the parameter) are filtered out.

The problem addressed involves combining interval estimates of sensor outputs into a best intersection estimate of outputs. The sensor fault model used assumes most faults are “tame.” That is, faults cluster in the neighborhood of the correct values. The procedure of this paper, assuming this fault model, is superior to earlier work by Marzullo.

To test the theoretical analysis of the computational framework proposed in our paper, we have developed a modular parameter-driven simulator SIMD-SN for the fault-tolerant integration of abstract sensor interval estimates. The simulator uses the well-known Monte-Carlo

multi-dimensional parameters defined in  $\mathcal{Y}$  and are mapped to sensor data values the processor could receive. We use these data values to simulate the errors in sensor measurements and the communication between sensors and the processor. As the sensor network consists of several sensors, it is necessary to consider the case of the sensor being faulty. The sensors need to be successive or faulty. The technique of integration of the sensors consists of the procedure of merging the various data obtained from the various sensors. This process continues until all the components of the parameter vector are obtained at the required level of accuracy. We also present the simulation results of the performance of our algorithm in comparison to that of Marzullo’s algorithm.

technique to generate random correct and tamely faulty intervals. The results of our algorithm and Marzullo’s are compared. We conclude based on the simulation that the union of the disjoint intervals (if any are disjoint) determined by our algorithm and the gaps between these intervals together can never exceed the width of the final output of Marzullo’s method. Thus, while Marzullo’s method specifies a rather large interval as the final output of the integration process, our method subdivides this interval into (possibly) disjoint intervals each with an associated *reliability measure*.

### I. INTRODUCTION

A distributed sensor network (Distributed Sensor Network) consists of several sensors distributed spatially, which collect data from their surroundings. The data from the sensors are received by processors which put together the information obtained from the sensors into a desirable form and send the integrated information for further processing and utilization by other processors and actuators. The distributed sensor integration problem has been shown to have wide ranging applications in many areas such as surveillance and tracking of moving objects in military applications, medical imaging, space application, etc.

A Distributed sensor network consists of several clusters of sensors, each cluster feeding into a processor. Each cluster is assigned to a region in space from which it collects data. All the sensors in a cluster observe the same phenomenon, i.e. read the same value in the region to which they are assigned. This redundancy is introduced to ensure fault-tolerance since some of the sensors in a cluster may be faulty. The set of all the regions, to each of which a sensor cluster is assigned, constitutes the space under observation by the Distributed Sensor Network. This must include the cooperative solution of problems by a decentralized and loosely coupled collection of processors, each of which integrates information received from a cluster of spatially distributed sensors into a manageable and reliable output for further

Manuscript received December 16, 1992; revised March 1, 1994, April 27, 1994 and June 3, 1994. This research in part is partially supported by Board of Regents (LEQSF-RD-A-04, 1990) grant and by ONR N000014-91-J1306.

The authors are with the Robotics Research Laboratory, Department of Computer Science, Louisiana State University, Baton Rouge, LA 70803 USA. IEEE Log Number 9406641.

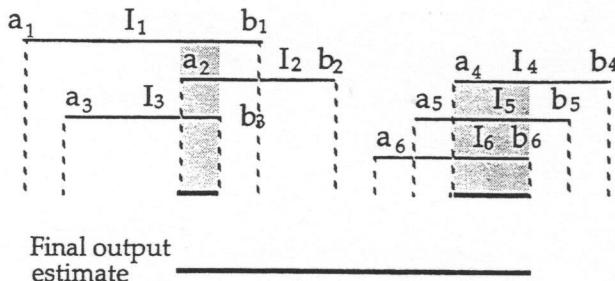


Fig. 1. Integration of interval estimates.

integration at a higher level. Integration of information at the sensor level requires techniques to be developed to abstractly represent and integrate sensor information. Further these techniques have to be robust in the sense that even if some of the sensors are faulty, the integrated output should still be reliable. One of the features that distinguishes distributed sensor processing is its demand for computational framework for fault-tolerant sensor integration.

The work described in this paper is allied to the methodology provided by Marzullo in masking failure of multidimensional sensors. Our method coincides with Marzullo's [6] as far as considering intersections of the geometric sets representing sensor outputs is concerned. Marzullo encloses these intersections by a connected minimal set, and obtains bounds for the size of this set in terms of bounds for the number of faulty sensors.

Our approach to Distributed Sensor Network problem differs from the previous work [6] in that we regard sensors as giving point-set outputs, with no probability or weight function associated with them. We then look at the problem of integrating these geometric sets to obtain a set which contains with high reliability the point corresponding to the actual value observed by the sensors, even though some of the sensors may be inaccurate or faulty.

The sensors in general have uncertainties in their readings and may not yield a definite value but 'hover' over a range of values. Thus it is appropriate to look upon a sensor's output as a contiguous set of values, rather than as a point value. The parameter being observed may in general be a vector (eg. velocity, position, etc. The processors fed by the various sensor clusters integrate the information to obtain a reliable estimate of the parameter(s) being measured in the respective regions. This information is local to the region and may be used to effect an action in that region or be integrated at higher processors to obtain a global picture or to make a global decision.

The organization of this paper is as follows. Section II discusses related work in this area. In Section III, we describe an abstract representation of Distributed Sensor Network. A scheme for patching together local processor information to obtain a global picture of the parameter measured is given in Section IV. Finally, in Section V, we compare the performance of our model to that of Marzullo's.

## II. RELATED WORK

Refer to our earlier papers for preliminary definitions on sensors [4], [5]. Also, please refer Marzullo's [6] paper about his work. Fig. 1 describes integration of interval estimates using [6].

### A. Motivation and Summary of This Work:

While the problem of distributed sensing has been addressed often with reference to specific kinds of tasks and sensors, there has been no general formulation of the problem in a formal sense. Following the work in [4], [5] and [6], we propose a formal description of an abstract Distributed Sensor Network that subsumes the description suggested and employed in the above mentioned papers and includes

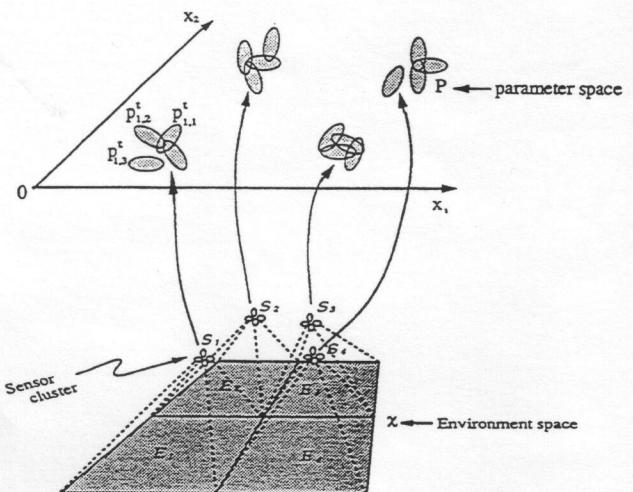
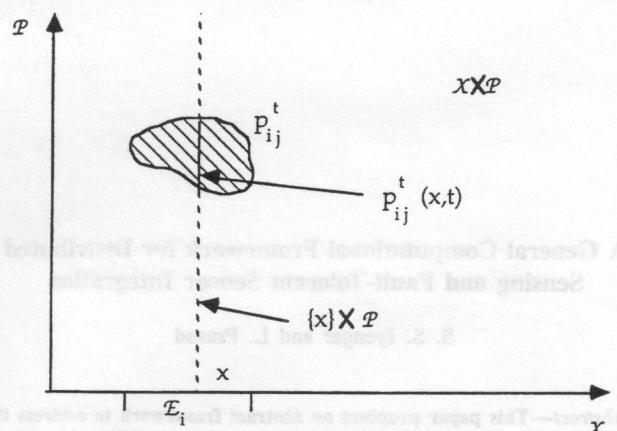


Fig. 2. Schematic representation of an abstract DSN.

Fig. 3. The subsets  $P_{i,j}^t$  corresponding to the sensor  $\sigma_{i,j}^t$ 

a wide class of Distributed Sensor Networks. It is to be noted that this formal description is a very theoretical framework for characterizing a general Distributed Sensor Network and addressing the general computational features and problem of a Distributed Sensor Network and looks upon sensor information as geometric point-sets rather than as distributions. It is a symbolic description capturing the features of a large class of Distributed Sensor Networks and not an implementation of the Distributed Sensor Networks in the class implicit in the characterization. Indeed, the exact nature of the implementation would depend upon the specific Distributed Sensor Network and the nature of the phenomenon being sensed and the sensors. Yet its abstract structure would be concordant with the characterization to be given.

Keeping the above remarks in mind, we present an abstract representation of a general Distributed Sensor Network. We first introduce the formal definitions with notation. The notation and definitions will be followed by explanations and examples.

## III. ABSTRACT REPRESENTATION OF A DISTRIBUTED SENSOR NETWORK

### A. Notation and Definitions

We formally introduce here the abstract set up for a general distributed sensor network:

$$\mathcal{X}_{i,j} = \bigcup_{x \in \mathcal{E}_i} \mathcal{X}_{i,j}(x), \quad (x \neq y \Leftarrow p_{i,j}(x) \cup p_{i,j}(y) = \emptyset)$$

Note that the characteristic function of the set  $p_{i,j}$  is

$$\chi_{\mathcal{X}_{i,j}}(y) = \begin{cases} 1 & \text{if } y \in \mathcal{P}_{i,j}(x) \\ 0 & \text{otherwise} \end{cases}$$

Consider the characteristic function of the set  $p_{i,j}(x)$ :

measured in this general case also as shown below:  
to obtain a fault-tolerant estimate of the actual physical value being  
The method of integration of our earlier paper [5] can be used here  
measured at  $x$  at time  $t$  ( $a \in \mathcal{P}$ ), else it is faulty.

An abstract sensor  $g_{i,j} \in \mathcal{G}_i$  is correct at  $x \in \mathcal{E}_i$ , at time  $t$  if  
 $(x, a) \in \mathcal{P}_{i,j}(x)$  where  $a$  is the actual value of the parameter being  
gration for general distributed sensors networks. We define correctness  
and faultiness in abstract sensors as follows:

#### D. Fault-tolerant Integration of Abstract Sensor Outputs

Distributed Sensor Network defined above.  
technique carries over easily to the sensor outputs of the abstract  
output of Marzullo's method in general. We shall show now that this  
an output that is reliable and has measure smaller than that of the  
strong bounds on the number of faulty sensor, but which produces any  
we have provided a polling technique which does not assume any  
the number of faults is not less than  $\lceil n/2d \rceil$ . In our earlier work [5],  
in most situations, a cluster of  $n$  sensors is still useful even when  
entire cluster's data when  $f > \lceil n/2d \rceil$ . It is however possible that  
(failure is never often controllable). One may have to reject an  
However, when one cannot ensure this boundedness of the faults  
estimates and  $f \leq \lceil n/2d \rceil$  for  $d$ -dimensional rectangular estimates).  
are bounded above by a function of  $n$  ( $f \leq \lceil n/2 \rceil$  for 1-dimensional  
computing a fairly accurate output when the number of faulty sensors  
Marzullo's technique of integration gives an effective way of

#### C. Limitation of Marzullo's Technique

correct value of the parameter from among the various sensor outputs.  
method of integration that isolates a neighborhood of the  
which sensor are faulty at any given instant of time, one has to derive  
Network to function reliably. Since it is impossible in general to know  
faulty accurate, is required at each processor for the Distributed Sensor  
most cases an output which contains the value of the parameter and is  
parameter measured by it. A technique of integration that yields in  
may be faulty in that its output may not contain the value of the  
regulations of the specific Distributed Sensor Network. A sensor  
which is decided depending upon the specific regulations and performance  
at  $x \in \mathcal{E}$ , if  $h(p_{i,j}(x))$  is greater than a preassigned number  $M$   
on  $n$ -dimensional Euclidean space). We say that  $g_{i,j}$  is inaccurate  
to some measure  $u$  defined on  $\mathcal{X} \in \mathcal{P}$  (say, the Lebesgue measure  
may assume that the geometric sets  $p_{i,j}$  are measurable with respect  
 $x \in \mathcal{E}$ , by having a  $p_{i,j}(x)$  of a "very large" size. More precisely, we  
of the parameters value. A sensor  $g_{i,j}$  could be inaccurate estimates  
to tolerate faults to a reasonable extent and provide reliable estimates  
integrated sensors tend to be inaccurate or faulty. The technique of  
The sensors tend to be inaccurate or faulty. The technique of

rectangles or circles.  
connected convex sets for the  $p_{i,j}(x)$ , and in particular,  $d$ -dimensional  
the specific problem, we could model the abstract sensors to yield  
 $d$ -dimensional parameter measured at the point  $x$ . Depending upon  
or  $d$ -dimensional circles representing abstract sensor readings of a  
motion since the  $p_{i,j}(x)$  could very well be  $d$ -dimensional rectangles  
multi-dimensional parameters defined in [6] are included in our def-

It is also clear that the abstract sensors modeling sensors measuring  
either totally connected or be a disjoint union of connected subsets.  
We can say that the  $p_{i,j}(x)$  corresponds to the abstract sensor  
B. Summary of Notations

all the  $g_{i,j}$  in  $\mathcal{S}_i$  have identical outputs.  
environment space  $\mathcal{X}$ , namely  $\mathcal{E}_i$ . Under ideal fault-free conditions,  
sensor  $g_{i,j}$  in the cluster  $\mathcal{S}_i$  collects data from the same region in the  
the point  $x$ . Here, the parameter is assumed to be 1-dimensional. Each  
returns an interval of values  $(p_{i,j}(x))$  as the abstract sensor reading at  
parameter  $x$ . The set of values  $p_{i,j}(x)$  is assumed to be 1-dimensional.  
sensors at all points of the 1-dimensional interval  $\mathcal{E}_i$ .  $g_{i,j}$   
of the sensor  $j$  in the cluster  $i$ . Here, it is assumed that it reads the  
schematic anatomy of the set  $p_{i,j}$ : The set  $p_{i,j}$  represents readings  
sensor  $g_{i,j}$  (Fig. 3).

the same as the dimension of the parameter measured by the  
of  $\mathcal{X} \times \mathcal{P}$ . The dimension of the sets  $p_{i,j}(x)$  is in general  
see that  $p_{i,j}$  is a set of finitely many connected subsets of  
of  $\mathcal{X} \times \mathcal{P}$ , or more specifically of  $\mathcal{E}_i \times \mathcal{P}$ . Thus in any case, we  
 $\mathcal{E}_i$ , then  $p_{i,j} = p_{i,j}(x_1) \cup p_{i,j}(x_2) \cup \dots \cup p_{i,j}(x_k)$ . If the  
parameter manifolds itself on finitely many connected subsets of  
 $\mathcal{E}_i$ , then  $p_{i,j} = p_{i,j}(x)$  is connected for each  $x \in \mathcal{E}_i$ .  
itself only at finitely many discrete points  $x_1, x_2, \dots, x_k$  in  
We denote  $p_{i,j} = \cup_{x \in \mathcal{E}_i} p_{i,j}(x)$ . If the parameter manifolds  
 $g_{i,j}$  at  $x$ . We assume  $p_{i,j}(x)$  is connected for each  $x \in \mathcal{E}_i$ .  
by  $p_{i,j}(x) = \{(x, z) | z \in \text{the range of values given out by}$   
Thus  $g_{i,j}$  maps  $x \in \mathcal{E}_i$  onto the subset  $p_{i,j}$  of  $\mathcal{X} \times \mathcal{P}$  given  
of values, then  $g_{i,j}$  is correct at the point  $x$ , else it is faulty.  
If the actual value of the parameter at  $x$  belongs to this set  
subsets of values representing the value of the parameter at  $x$ .  
Thus we assume that for each  $x \in \mathcal{E}_i$ ,  $g_{i,j}$  gives a single value  
the parameters, it is realistic to expect it to give a range of values for  
 $x \times \mathcal{E}_i$ , since a sensor in general has some uncertainty in its  
reading, it is realistic to expect it to give a range of values for  
readings that about a parameter at each point  
The sensor  $g_{i,j}$  senses data observed in  $\mathcal{E}_i$  at time  $t$ .

of the parameter values observed in  $\mathcal{E}_i$  at time  $t$ .  
 $\mathcal{X} \times \mathcal{P}$  as the abstract sensor estimates of the sensors  $\{g_{i,j}\}_{i=1}^m$   
any instant of time  $t$ , we have the collection of  $\{p_{i,j}\}_{i=1}^m$  of  
The values of the subsets  $p_{i,j}$  will be described soon. Thus at  
varying map which maps the set  $\mathcal{E}_i$  onto a subset  $p_{i,j}$  of  $\mathcal{X} \times \mathcal{P}$ .  
 $S_i = \{g_{i,j}\}_{j=1}^{n_i}$ , where each abstract sensor  $g_{i,j}$  is a time-  
assigned to an  $\mathcal{E}_i$ . Each  $S_i$  is a cluster of  $n_i$  abstract sensors  
4)  $\{S_i\}_{i=1}^m$  is a collection of  $m$  sensor-clusters, each  $S_i$  being  
sensors which obtain data from  $\mathcal{E}_i$ . Each subset  $\mathcal{E}_i$  is allocated a cluster of  
chairs on  $\mathcal{X}$  (Fig. 2). Each subset  $\mathcal{E}_i$  is allocated a cluster of  
 $\mathcal{X}$  such that  $\cup_{i=1}^m \mathcal{E}_i = \mathcal{X}$ . The collection  $\{\mathcal{E}_i\}_{i=1}^m$  is called a  
 $\mathcal{E}_i$  is a collection of subsets of the environment space  
discussed later.

by  $\mathcal{P}$ . The exact nature of these subsets will be  
Space of dimension  $k$ . The value of each sensor is represented  
measured is a  $k$ -dimensional vector, then  $P$  is the Euclidean  
measured, called the parameter space. If the parameter being  
are allocated constitute the space  $\mathcal{X}$ .  
The collection of regions to which the various sensor clusters  
transmit them to the processors to their respective regions and  
these clusters pick up data from various regions of  $\mathcal{X}$ , and  
Clusters of sensors are allocated to various regions of  $\mathcal{X}$ .  
Distributed Sensor Network, called the environment space.  
 $\mathcal{X}$  is the space under observation by the sensors of the

Definition 1: An abstract distributed sensor network is a 4-tuple  
( $\mathcal{X}, \{\mathcal{E}_i\}_{i=1}^m, \{S_i\}_{i=1}^m, P$ ), where:

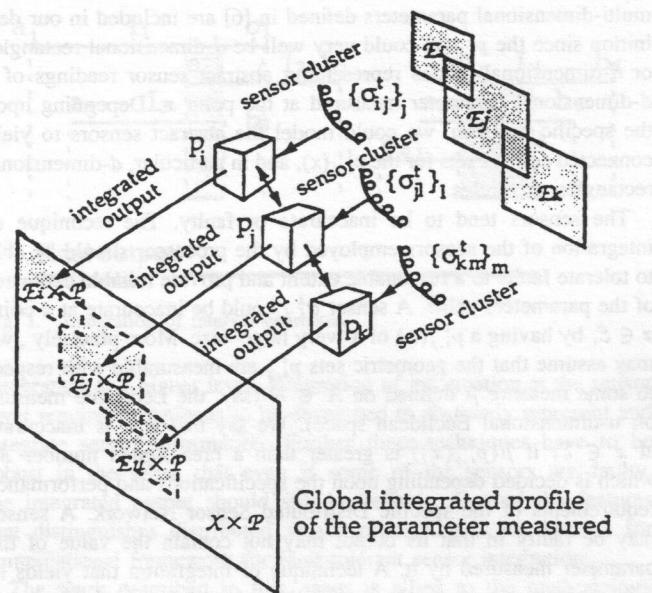


Fig. 4. Schematic representation of information patching to obtain a global picture.

If atmost  $f_x$  of the  $n_i$  sensors in the cluster  $\mathcal{S}_i$  are known to be faulty at  $x$ , then following the method of integration employed in our earlier paper [5], the correct value of the parameter being measured by the sensor cluster  $s_i$  at  $x \in \mathcal{E}_i$  and at time  $t$  is contained in the subset corresponding to the characteristic function  $\mathcal{X}_{x,t}^i(y)$  given by

$$\mathcal{X}_{x,t}^i(y) = \mathcal{X}_{[n_i - f_x, \infty)}(O_{x,t}^i(y))$$

where

$$\mathcal{X}^{[a,\infty)}(x) = \begin{cases} 1 & \text{if } x \in [a, \infty) \\ 0 & \text{otherwise} \end{cases}$$

and  $O_{x,t}^i(y) = \sum_{j=1}^{n_i} \mathcal{X}_{x,t}^{i,j}(y)$  is the overlap function for the cluster  $\mathcal{S}_i$  at the point  $x \in \mathcal{E}_i$  and at time  $t$  i.e.  $O_{x,t}^i(y)$  gives the number of sensors  $\sigma_{ij}^t$  whose estimates  $p_{ij}^t(x)$  at the point  $x \in \mathcal{E}_i$  and at the time  $t$  contain the point  $(x, y) \in \{x\} \times \mathcal{P}, y \in \mathcal{P}$ .

Consequently,  $\mathcal{X}_{x,t}^i(y)$  is the characteristic function of the set of all those points of  $\{x\} \times \mathcal{P}$  which lie in the intersection of at least  $n - f_x$  intersections of the sets  $p_{ij}^t(x)$ .

The correct value of the parameter being measured at  $x \in \mathcal{E}_i$  at time  $t$  must lie in the subset  $\Sigma^i(x, t) = \{y \in \mathcal{P} | \mathcal{X}_{x,t}^i(y) = 1\}$ , since the assumption that atmost  $f_x$  sensors are faulty at  $x$  implies at least  $n_i - f_x$  sensors are correct and hence must overlap since their estimates  $p_{ij}^t(x)$  must all contain the point  $(x, a)$  where  $a$  is the actual value of the parameter being measured.

As there is no reason to assume that  $f_x$  is fixed or bounded quantity, we may as well treat it as a parameter which we may change at will to obtain larger or smaller intersection as desired. For instance, we may look upon  $f_x$  as

$$\min\{f | \mathcal{X}_{x,t}^i(y) = \mathcal{X}_{[n_i - f, \infty)}(O_{x,t}^i(y)) \neq 0\},$$

and thus obtain those intersections which are a result of the largest number of sets intersecting to give a nonempty set.

Faulty sensors could have random or "wild" faults, in which event there is little correlation between the sensor's estimate and the correct physical value of the parameter being measured. On the other hand, the fault may be "tame," in which case the faulty sensor's estimate although does not contain the actual physical value of the parameter being measured, lies sufficiently close to it.

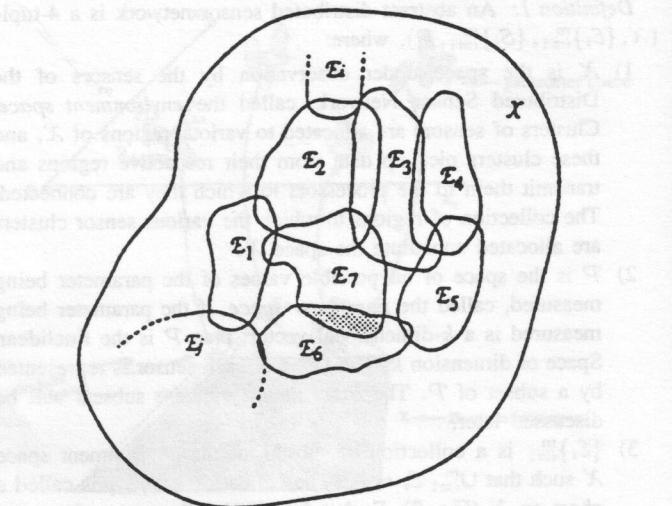


Fig. 5. A general coordinate chart  $\{\mathcal{E}_i\}_{i=1}^n$  on an environment space  $X$ .

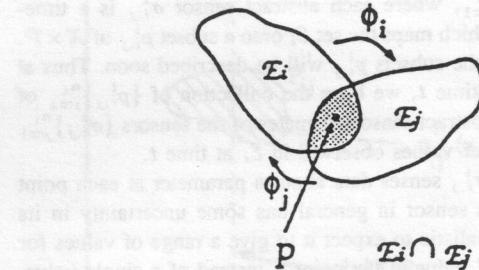


Fig. 6 The transition function  $\phi_i$  and  $\phi_j$  defined on the intersection  $\mathcal{E}_i \cap \mathcal{E}_j$ .

In order to obtain a reliable estimate of any of the above mentioned parameters, it is important to have redundancy in reading the parameter value. The parameter value is sampled in a local neighborhood of the point in question, the value of the parameter being assumed to be stable in a small neighborhood. The values these obtained are compared with each other to yield a consensus value, this is achieved by a process of sensor integration.

Thus sensor failure can result in the failure of the electronic component or the physical interface or due to local variations in the environment.

Electronic failure, normally spells permanent failure. The nature of electronic failure is uncorrelated and hence is "wild." The failure resulting from a defective physical interface or a locally fluctuating environment (e.g. turbulence in fluids, statistical variations in the sample space observed etc.) results in values which are close to correct values and hence are "tame."

It is reasonable to assume that sensors more often fault tamely due to physical perturbation than wildly due to random electronic failures, In which case faulty sensors tend to "cluster" in the neighborhood of the correct value of the parameter being measured.

We may make use of this clustering to predict with reliability the subset with highest chance of containing the actual value of the parameter among those subsets which belong to  $\Sigma^i(x, t)$ . In order to do this, we need to make the notion of a tame fault more rigorous. The following definition is derived from above characteristics of tame faults.

*Definition 2:* A faulty sensor is *tamely* faulty at  $x$  at time  $t$  if it intersects with a sensor that is correct at  $x$  at time  $t$ .

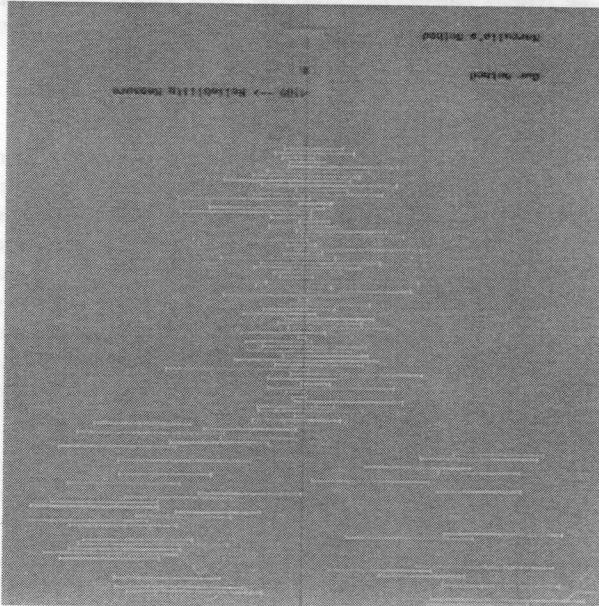
$L_j(x, t)$  of  $\Sigma^j(x, t)$  containing the correct value of the parameter. It is clear that the larger the reliability of the connected set  $L_j(x, t)$ , the greater the clustering of sensors about  $L_j(x, t)$ , and hence the output  $H_k(x, t)$  is the sum of the populations of all those sensors' outputs which contain the set  $L_j(x, t)$ .

$$H_k(x, t) = \sum_{i=1}^{n_k} \|A_{i,j}^x A_{i,k}^x\| \pi_{i,k}(x, t)$$

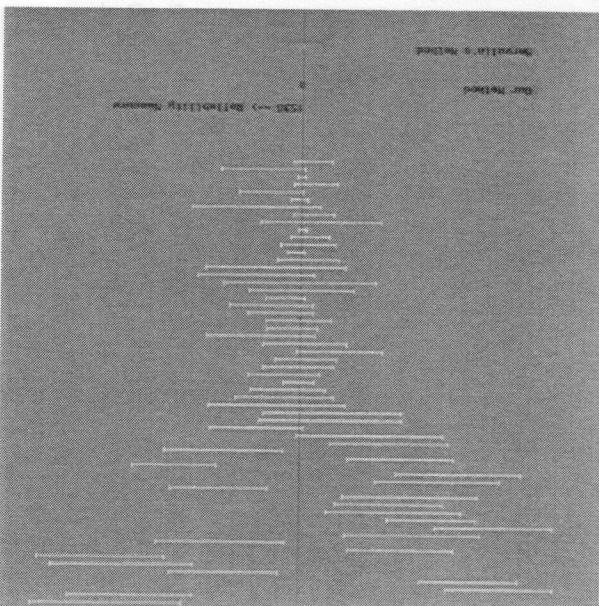
Then define the reliability  $H_k(x, t)$  of the set  $L_j(x, t)$  to be the nonnegative integer given by:

Fig. 9(a). Sensors: 60, tame faults: 23. (b) Sensors: 100, tame faults: 40.

(b)



(a)



where  $\|\cdot\|$  is the maximum value of the function  $f_{\pi_{i,k}}(x, t)$  given by:

$$\pi_{i,k}(x, t) = \left( \sum_{i=1}^{n_k} \|A_{i,j}^x A_{i,k}^x\| - 1 \right)^{-1}$$

in the cluster  $S_i$  to be the normative measure  $\pi_{i,k}(x, t)$  given by:

Define the popularity of the  $k$ th sensor ( $1 \leq k \leq n$ )

physical value of the parameter measured at  $x$  at time  $t$  ( $x \in E$ ).

subsets of  $P$  whose union is the subset  $\Sigma^j(x, t)$  containing the correct value  $L_j(x, t), \dots, L_k(x, t)$  be the disjoint maximal connected clusters of  $P$  whose union is the subset  $\Sigma^j(x, t)$  containing the correct value  $L_j(x, t)$ .

Let  $L_i(x, t), \dots, L_k(x, t)$  be the disjoint maximal connected clusters with high reliability:

to reduce the measure of the polling technique introduced in an earlier paper [5] or less evenly. In the case where most sensor faults are tame, we may resort to the polling technique which will wild or random faults more by their very nature, they are uncorrelated and hence distributed more unevenly than sensors with wild or random faults since it is unlikely that sensors with wild or random faults cluster since

#### E. Reduction of the Output Measure When Most Faults are Tame

Fig. 8. Block diagram of simulation.

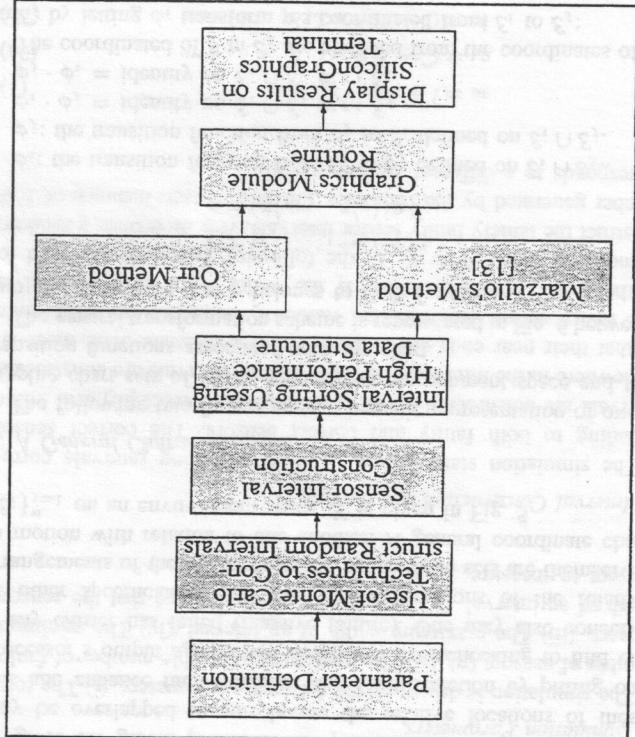
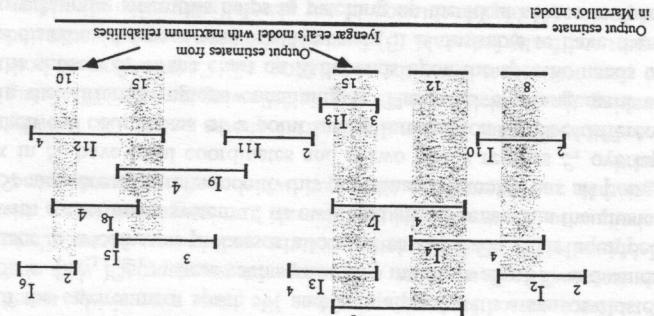
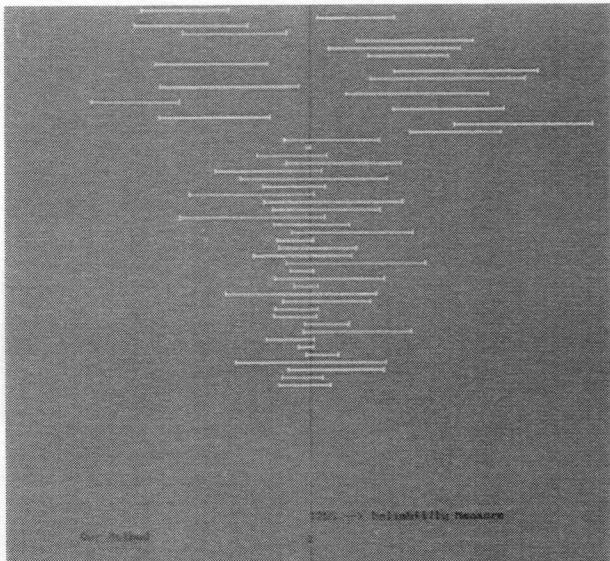
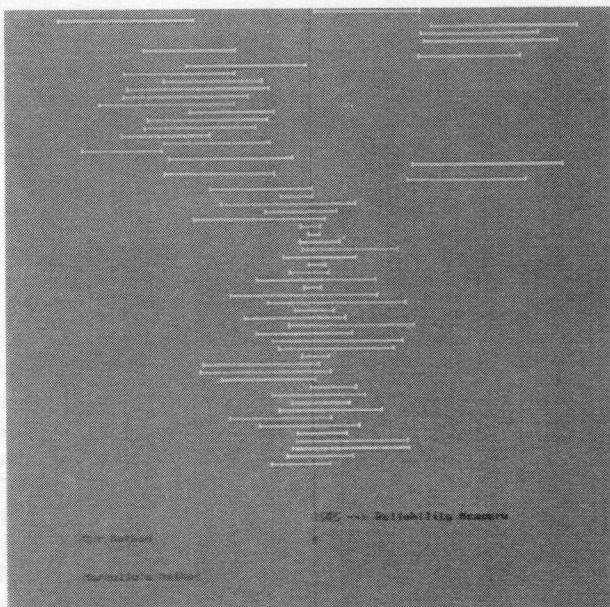


Fig. 7. A comparison of performance.





(c)



(d)

Fig. 9(c). Sensors: 50, tame faults: 17. (d) Sensors: 60, tame faults: 23.

Thus, it is seen that the techniques employed by us in the paper [5] though used on 1-dimensional interval estimates, hold for a much more general class of sensor outputs.

#### IV. A SCHEME FOR PATCHING TOGETHER LOCAL PROCESSOR INFORMATION TO OBTAIN A GLOBAL PICTURE OF THE PARAMETER MEASURED

We now look at the problem of addressing the global behavior of a parameter over the space  $\mathcal{X}$ . This involves the interaction of the processors allocated to the subsets of the chart and comparison of common information for smooth patching of local information to obtain a global picture of the phenomenon being observed over  $\mathcal{X}$  (Fig. 4).

Each of the subsets  $\mathcal{E}_i$  ( $1 \leq i \leq m$ ,  $\bigcup_{i=1}^m \mathcal{E}_i = \mathcal{X}$ ) covers a region of the environment space  $\mathcal{X}$ , and is equipped with a sensor-cluster  $S_i = \{\sigma_{i,j}^t\}_{j=1}^{n_i}$ , where each sensor  $\sigma_{i,j}^t$  monitors all of  $\mathcal{E}_i$  and sends data to a common processor allocated to  $\mathcal{E}_i$ . Each  $\mathcal{E}_i$  is equipped with a coordinate system of its own, and all the sensors in the cluster  $S_i$  measure with reference to this coordinate system. Thus all points  $x$  in  $\mathcal{E}_i$  have local coordinates and if two of the regions  $\mathcal{E}_i$  overlap then the coordinates of a point in the intersection will be different in the different regions containing it. The relative arrangement of the subsets  $\mathcal{E}_i$  of the chart on  $X$  depends upon the specific needs of a distributed sensor network. However, it is desirable to have them overlapping since this helps in patching up the local scenes to form a global picture as well as in increasing fault-tolerance and aiding fault-detection in sensors by comparison of data at common points.

Thus the union of the outputs of the processors over the sets  $\mathcal{E}_i$  gives the global profile of the parameter over  $\mathcal{X}$ . The sets  $\mathcal{E}_i$  may be overlapped to corroborate the relative locations of these sets and enhance fault-tolerance and fault-detection by pitting one processor's output against the other, and crosschecking to find out if any cluster has failed (massive failure). One may also conceive of other specification and need based variations of the relative arrangements of the  $\mathcal{E}_i$ , for instance, when these sets are themselves in motion with relation to one another. A general coordinate chart  $\{\mathcal{E}_i\}_{i=1}^n$  on an environment space  $\mathcal{X}$  is given in Fig. 5.

##### A. A General Characterization

The following two figures are a schematic representation of overlapping chart sets of an  $m$ -dimensional environment space and the transition functions associated with the chart.

The general transformation scheme is represented in Fig. 6 between two sets  $\mathcal{E}_i$  and  $\mathcal{E}_j$ . Let  $p$  belongs to  $\mathcal{E}_i \cap \mathcal{E}_j$ , the

$$p = (x_1^i, \dots, x_m^i) \text{ in } \mathcal{E}_i$$

$$p = (x_1^j, \dots, x_m^j) \text{ in } \mathcal{E}_j$$

$\phi_i$ : the transition function from  $\mathcal{E}_i$  to  $\mathcal{E}_j$  defined on  $\mathcal{E}_i \cap \mathcal{E}_j$ .

$\phi_j$ : the transition function from  $\mathcal{E}_j$  to  $\mathcal{E}_i$  defined on  $\mathcal{E}_i \cap \mathcal{E}_j$ .

$\phi_i \cdot \phi_j = \text{identity on } \mathcal{E}_i \cap \mathcal{E}_j \text{ w.r.t } \mathcal{E}_j$

$\phi_j \cdot \phi_i = \text{identity on } \mathcal{E}_i \cap \mathcal{E}_j \text{ w.r.t } \mathcal{E}_i$ .

The coordinated of  $p$  in  $\mathcal{E}_j$  are obtained from the coordinates of  $p$  in  $\mathcal{E}_i$  by letting  $\phi_i$  transform  $p$ 's coordinated from  $\mathcal{E}_i$  to  $\mathcal{E}_j$ :

$$\phi_i(x_1^i, \dots, x_m^i) = (x_1^j, \dots, x_m^j)$$

Similarly

$$\phi_j(x_1^j, \dots, x_m^j) = (x_1^i, \dots, x_m^i)$$

Further analysis in this direction can be done in specific distributed sensor network situations, but the general interprocessor interaction and communication must involve the use of transition function for meaningful comparison of data. The coordinated changes are effected by local processors and only those processors that are connected have transition functions defined. In the case of moving chart sets it is possible that two sets whose processors are not connected directly may overlap for some time. It is possible to perform local coordinate changes by routing data through other intermediate processors since the transition functions are transitive and hence may be composed to obtain transformations between unconnected processors.

##### V. SIMDSN: SIMULATION OF FAULT TOLERANT INTEGRATION IN DISTRIBUTED SENSOR NETWORKS

To test our theoretical analysis of the general computational framework proposed earlier in our paper, we have developed a

the number of overlapping intervals, for each interval, the sensors of which 23 are faulty, Table I summarizes the populations produce the final output. For the first simulation run, involving 60 by summing the populations of the sensor intervals that overlap to the result of our technique. This reliability measure is calculated shown above our output is the colored blue. The integer the result of Marzullo's approach and is shown just above interval with the highest reliability in our method is shown just below technique is colored red and appears at the bottom of the screen. The correct intervals include the correct value. The output of Marzullo's fault sensor intervals are located away from this line while the correct physical value being measured by the sensors. Notice that combined results of the two interval integration techniques compared. The green line down the middle of the display screen indicates the parameters of simulation enables particular runs of the simulation faultless have been considered. A minimal amount of change in noise of graphical visualization of the integration algorithms, only involving various numbers of correct and faulty sensors. For convenience, we present here the details of four complete runs of the simulator.

#### E. Simulation Results

The interval determined by Marzullo's approach, method, the figure also shows, for the same test data, the final output with the reliability of each disjoint interval. For comparison with our below shows one set of test data and the result of our method along with several sets of random data generated dynamically and figure host accessible in the Internet domain. Our algorithm has been tested transparent and allows simulations to be carried out at any compatible of the simulation. The simulator we have developed is network 4D/3D GTX Superworkstation to compute and display the results and uses the powerful Graphics Library of the Silicon Graphics The simulator is developed entirely in the C programming language and uses the Powertex Graphics Library in the C implementation.

#### D. Graphics Module and Display Details:

and so that their near ends are located within a distance of at most  $L$ . The faulty sensor intervals are randomly scattered on either side bounded so that  $l \leq w \leq L$ , the following equations are used to construct the correct physical value. If the width  $w$  of each interval is of the correct value and all sensor intervals are located at most  $L$ , the figure corresponds to a different random number.

The algorithm we have used for implementing our scheme for the integration of overlapping intervals is as follows.

#### C. Algorithm for Integration for the Simulator

This stage is followed by sorting the randomly generated intervals in ascending order of location of start points and interval end points. Further, the worst case complexity of this sort is limited to  $O(n \log n)$  which is the best worst-case complexity for a sorting algorithm on a sequential machine.

$$a_i = C - R^{\#} * w; b_i = a_i + w \quad (4)$$

$$w = R^{\#} * (L - l) \quad (5)$$

of the correct value and during simulation, one of the equations is picked randomly to locate correct intervals equal numbers of intervals on both sides. The correct intervals are constructed according to the following equations.

$$b_i = C - R^{\#} * L; a_i = b_i - w \quad (2b)$$

$$a_i = C + R^{\#} * L; b_i = a_i + w \quad (2a)$$

$$w = l + R^{\#} * (L - l) \quad (1)$$

The simulation starts (stage I) by constructing intervals correctly generated by the simulator. Of course, each instance of  $R^{\#}$  number corresponds to a different random number. So that near ends are located within a distance of at most  $L$ . Some where in the interval, the faulty sensor intervals are constructed so as to include the correct physical value intervals are constructed so that both faulty and correct sensors. The correct sensor spanning to both faulty and correct sensors. The correct intervals correspond to a different random number.

#### B. Interval Construction and Sorting

The simulation is driven by the following parameters: (i) The total number of sensors ( $n$ ). (ii) The maximum allowable number of faulty sensors. (iii) The minimum width of an interval. (iv) The minimum width of an interval. (v) The maximum width of an interval that the sensors attempt to measure.

#### A. Simulation Parameters

```

begin
    Output: Intergated output estimate
    Input: Intervals  $I_1, I_2, \dots, I_n$ ,  $f$  (parameter denoting
           modulus parameter-driven simulator for the fault tolerant integration
           of abstract sensor interval estimates. We have extensively tested
           our framework on test data and compared our approach with that
           of Marzullo [2]. The simulator uses the well-known Monte Carlo
           technique to construct random intervals for integration using our
           algorithm and Marzullo's technique which are compared subsequently.
           A block diagram of the simulator is shown in Fig. 8. Broadly
           speaking, the simulation consists of the following stages: (1) Defining
           the intervals that drive the simulation. (2) Construction of both
           correct and faulty intervals. (3) Sorting the intervals on both start and
           end points. (4) Implementing each of the intervals using a graphics
           module. (5) Display of the simulation results using a graphics interface.
    begin
        1. Determine all  $(n - f)$ -intersections of the
           intervals to yield intervals  $L_1, \dots, L_k$ ,
           each of which is an  $(n - f)$ -intersection;
        2. For each  $i$  ( $1 \leq i \leq k$ )
           a. Compute the number of intervals
              intersecting each of  $L_i$ ,
              of  $L_i$  — a measure
              of the reliability of  $L_i$ ,
              intervals involved in the formation
              of number  $r_i$  (the number of
              intervals intersecting with
              number  $r_i$  (the number of
              intervals to obtain a
              b. Add these numbers to obtain a
              intersection with  $L_i$ ,
               $I_j$ , ( $1 \leq j \leq n$ ) having a nonempty
              intersection of each of the intervals
              computing each of the intervals
              intersecting each of the intervals
              with each other;
        3. Choose the maximum of the  $r_i$  ( $1 \leq i \leq k$ )
           of the reliability of  $L_i$ ;
        4. Let  $m = \min\{r_i | r_i = r\}$  and  $M = \max\{r_i | r_i = r\}$ 
           and denote it  $r$ 
           5. Assign  $I_p = [a_m, b_M]$  to be the integrated
              output estimate
    end

```

TABLE I  
INTERVALS AND POPULARITIES FOR A PARTICULAR SAMPLE RUN

| Interval | Popularity | Interval | Popularity | Interval | Popularity | Interval | Popularity |
|----------|------------|----------|------------|----------|------------|----------|------------|
| 1        | 19         | 16       | 15         | 31       | 40         | 46       | 36         |
| 2        | 13         | 17       | 15         | 32       | 37         | 47       | 45         |
| 3        | 38         | 18       | 19         | 33       | 41         | 48       | 45         |
| 4        | 24         | 19       | 13         | 34       | 37         | 49       | 41         |
| 5        | 13         | 20       | 24         | 35       | 46         | 50       | 46         |
| 6        | 21         | 21       | 24         | 36       | 37         | 51       | 37         |
| 7        | 24         | 22       | 19         | 37       | 37         | 52       | 40         |
| 8        | 23         | 23       | 6          | 38       | 40         | 53       | 42         |
| 9        | 9          | 24       | 40         | 39       | 41         | 54       | 43         |
| 10       | 13         | 25       | 37         | 40       | 38         | 55       | 40         |
| 11       | 7          | 26       | 37         | 41       | 37         | 56       | 40         |
| 12       | 9          | 27       | 40         | 42       | 44         | 57       | 50         |
| 13       | 20         | 28       | 46         | 43       | 42         | 58       | 36         |
| 14       | 6          | 29       | 46         | 44       | 41         | 59       | 37         |
| 15       | 21         | 30       | 42         | 45       | 38         | 60       | 43         |

We conclude this section on the simulation by noting that the union of the disjoint intervals (if any are disjoint) determined by our algorithm and the gaps between these intervals together can never exceed the width of the final output of Marzullo's method. Thus, while Marzullo's method specifies a rather large interval as the final output of the integration process, our method subdivides this interval into (possibly) disjoint intervals each with an associated *reliability measure*. In a real-world situation, a suitable rule may be chosen judiciously depending upon our faith in the tameness of abstract sensor faults. It is clear that the worst-case width for the output interval as determined by our algorithm can be at most as large as that indicated by Marzullo's approach.

## VI. CONCLUDING REMARKS

We have proposed an abstract framework to address the problem of fault-tolerant integration of sensor information in a general distributed sensor network. We have also described a general computational scheme for obtaining a reliable integrated output from the inputs of several sensors in a cluster, at its processor. Our method involved a generalization of the polling technique for tamely faulty sensors presented in our earlier work [5]. A comparison of performance is given in Fig. 7.

The time complexity of finding the  $(n - f)$ -intersection of  $n$  sensors (in the case of intervals and higher dimensional rectangles) is  $O(n \log n)$  using Bentley's segment-tree. For an implementation of the algorithm see [7]. Also see [6].

Based on our simulation results, we note that the union of the disjoint intervals (if any are disjoint) determined by our algorithm and the gaps between these intervals together can never exceed the width of the final output of Marzullo's method. Thus, while Marzullo's method specifies a rather large interval as the final output of the integration process, our method subdivides this interval into (possibly) disjoint intervals each with an associated *reliability measure*.

## ACKNOWLEDGMENT

The authors would like to thank all the referees, especially to Referee A, for their comments, and Professor Marzullo, Cornell University and Professor Brady, Oxford University for their advices.

## REFERENCES

- [1] S. S. Blackman and T. J. Broida, "Multiple sensor data association and fusion in aerospace applications," *J. Robot. Syst.*, vol. 7, no. 3, pp. 445-485, 1990.

- [2] H. F. Durrant-Whyte, "Sensor models and multisensor integration," *Int. J. Robot. Res.*, vol. 7, no. 6, pp. 97-113, 1988.
- [3] S. S. Iyengar, "Distributed sensor networks: A computational perspective," *J. Comp. Sci. Informat.*, vol. 21, no. 1, 1991.
- [4] S. S. Iyengar, M. B. Mohan, and R. L. Kashyap, "Information routing and reliability issues in distributed sensor networks," to appear in *IEEE Trans. Acoust., Speech, Signal Processing* vol. 40, no. 12, pp. 3012-3021, Dec. 1992.
- [5] Lakshman Prasad, S. S. Iyengar, R. L. Kashyap, and R. N. Madan, "Functional characterization of sensor integration in distributed sensor networks," *IEEE Trans. Syst. Man Cyber.*, vol. 22, pp. 1082-1087, Sept./Oct. 1991.
- [6] K. Marzullo, "Tolerating failures of continuous-valued sensors," *ACM Trans. Comput. Syst.*, vol. 8, no. 4, pp. 284-304, Nov. 1990.
- [7] K. Marzullo and S. Owicki, "Maintaining the time in a distributed system," *Proc. of 3rd Symp. on Principles of Distributed Computing*, pp. 295-305. ACM SIGPLAN/SIGOPS, 1983.