Distributed Computation in the Digitized Battlefield

Susanta Sarkar & Paul Richardson
VETRONICS Laboratory, US Army.
TARDEC, AMSTA-TR-R, MS –264
Warren. MI, 48397-5000
810-574-5142
{sarkars, richardp}@tacom.army.mil

Abstract

A distributed computation is defined as several computers performing computation for a common purpose. There are three basic characteristics of this kind of systems, *multiplicity of nodes, interconnection of nodes,* and *shared states among these nodes*. This paradigm is analogous to a group of combat vehicles in a battlefield, pursuing a common goal. The three characteristics stated above have corresponding elements in a battlefield. The shared states among nodes are similar to a common goal for a group of combat vehicles. Moreover, combat vehicles correspond to multiple nodes in a distributed system. Finally, communication links among the combat vehicles are akin to interconnections of computing nodes. Due to these striking similarities, technologies developed for distributed computations may be readily applicable for a digitized battlefield of the future.

However, due to a few distinctive factors of digitized ground combat, techniques of classical distributed systems must be extended to be useful. At the U.S. Army VETRONICS Technology Center, we have started an initiative to accomplish these enhancements. In this report, essential elements of the program and some interim findings are reported.

1. **Introduction**

The computation and communication model of a distributed system is strikingly similar to a digitized battlefield in the ground domain. The three primary characteristics of a distributed system – *multiplicity of nodes, interconnection of nodes,* and *shared states among these no*des -are present in a battlefield. For example, each combat vehicle in a ground combat is similar to an independently computing node. Each vehicle exchanges message with other vehicle, through radios. This matches to multiple computing node and asynchronous message passing communication mode. Friendly forces have a common operational objective. Thus, the paradigm of distributed computation applies. From this similarity, it is apparent to conclude that technologies available for the distributed computation domain are readily applicable for a digitized battlefield. However, these techniques must be enhanced to make it more effective and efficient for a digitized battlefield.

For instance, take the example of situation awareness at each ground combat vehicle. The correct global picture is of paramount importance. However, due to loose coupling of the combat vehicles in a battlefield, it is difficult to form a correct and identical global picture at all the friendly nodes. There are existing techniques [1,2] in the field of distributed computation that applies to this global picture generation. These techniques cannot be directly applied, because there are several distinct factors that set the digitized battlefield apart from the general model of distributed computation. These distinctive factors render the techniques inefficient and sometime ineffective. There are three areas, where the digitized battlefield differs form the general model of distributed computation.

- *Transaction oriented state machine* model of a node is not always applicable in the battlefield situations
- The number of participating node in a computation is not constant in the battlefield.
- The number of messages exchanged in a protocol – message complexity, is of utmost interest in a battlefield.

We have analyzed these issues and taken suitable approaches to be more effective.

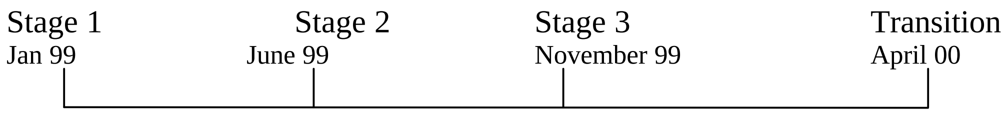
The rest of the paper is organized in five major sections. The following section presents motivation and important parts of this program. In section 3, we briefly review the background and rationalize approaches taken. Section 4 describes the technical models used. In section 5, we discuss some of the results. Section 6 outlines the path forward.

2. **Motivation & Elements of the Program**

As a direct result of digitization of the battlefield, locally sensed important information can be shared among all friendly entities. During conduct of experiments involving the last VETRONICS based ATD (Crewman's Associate), it became evident that sharing of this information would lead to significant improvement in operations. However, synthesis of all these locally sensed information to create a correct and consistent global picture is not trivial. The issues involved require careful survey, analysis, synthesis, and enhancement. Thus, the Global Predicate Evaluation task has been defined within the VETRONICS laboratory of the US Army Tank-Automotive Research & Development Center.

The primary objective of this investigation is to establish a theoretical background for generating a correct and consistent global picture for all combat vehicles engaged in a digitized battlefield. The major driving force behind this objective is harnessing synergistic capabilities in operations because of digitized communication in a ground combat battlefield. For example, one important issue to consider: *is it possible to correctly synthesize a global situation from local sensing and exchanging these sensed information through digitized messages?* The other important issue relates to accuracy of the global situation. *Will this global situation awareness be accurate enough to be a basis for distributed target servicing plan within a group?* During the Crewman's Associate ATD, it became apparent that the answers to these questions are mixed. It is possible to synthesize but it is not trivial to make this synthesized global situation correct and consistent. The primary challenge is abundance of data. Every entity in a battlefield is sensed by more than one entity and this information is exchanged in a group. This phenomenon generates a large amount of data.

One of the ways the data gathered at a node can be reconciled is through recognition of pattern. More often than not, the redundancy of data relates to movement. A node receives information about other entities from more than one friendly node. For moving entities, a node receives multiple streams of data that are intermixed. It is possible to untangle these intermixed tracks through correlation. Based on this it is also possible to identify the latest position in a track. The issue to address is, *if there is pattern of movement in actual vehicle, then, is there an algorithm that can recognize it identically at each node?* In the first stage of this task, the specific goals are to develop a class of algorithms to synthesize identical picture at all nodes. The critical metrics to consider at this stage are: number of messages exchanged in a protocol and completeness of the global picture. In the second stage, we plan to develop a family of algorithms that can recognize complex pattern at each node. For this stage, the VETRONICS laboratory is collaborating with Wayne State University of Detroit, Michigan. In third stage, we plan to find existing patterns of movement in a battlefield. For this stage, initial work has already begun by enhancing capabilities of the simulation tool ModSaf. Using this enhanced package, one can construct a formation of a platoon or company and record their positions for analysis. At this stage, we plan to establish some closed form equations for patterns of movement obtained from ModSaf and collect real vehicle movement data from National Training Center to adjust parameters in the equations. Finally, we plan to marry the algorithm developed earlier with the closed form representation of pattern. The following is a tentative of schedule of events.



3. **Background**

Several approaches are available for synthesizing global predicate from local states in the domain of classical distributed computation. Excluding approaches, where underlying commutations are blocked, there are three known ways of observing global properties. In the *first* method [6] local states and messages in the communication channels are collected from each participating node. The collection point is either one of the computing nodes or an external observer. These states are then combined to form a global state. This can be done actively - meaning the external observer proactively requests each node to return their local state. Alternatively, [3] the external observer passively collects local states from the participating nodes as they send messages to other nodes for the underlying computation. The former approaches can only detect *stable global properties*. A stable global property remains true once it becomes true.

The *second* approach [9] is designed to detect unstable properties. First, a lattice of all probable global states is constructed from possible local states. Next, presence or absence of the global property under consideration is detected for each of these global states in the lattice.

In the third approach [10], specific attributes of a global property are used to narrow down the number of possible global states to be considered. For example, if a global property is a conjunction of a set of local properties, only the local states in which the local property holds are considered. These local states are combined to arrive at a global state.

However, as mentioned earlier a few distinct characteristics of a typical digitized battlefield renders these [*2*] approaches ineffective. Specifically, there are two characteristics that impact effectiveness. The transaction oriented state machine model of a node is not always applicable in battlefield situations, and the number of participating nodes in a computation is not constant in the battlefield.

3.1 ***Nodes as State Machine***

To reason about global parameters, [3] each node can be modeled as a state machine. This means that the type of stimulus is important as well as the order of a sequence of stimuli sensed by a node. The global picture suffers from inconsistency if these sequences of stimuli occur in different order at different nodes. The concept of causal precedence is generally used to reason about this out of order problem. If there is a precedence relation between two events, these events must be processed in the same order at each node. Otherwise, computations at various nodes will diverge.

Also, there could be a pair of events (state) where none causally precede each other. This means, that processing order of these events in the state machine model does not have any effect on the global parameters. These kinds of event pairs are called *concurrent* events. In a digitized battlefield, two consecutive positions from the same node cannot be modeled as strictly causally related events. In a targeting or situation awareness context, use of the latest position will make the underlying state machine more desirable to the user. However, in a tracking context, the causal sequence of positions is a valuable tool for prediction at a node.

3.2 ***Number of Nodes***

In a digitized battlefield, the number of nodes at the beginning and at the end of global state synthesis may change. The number of nodes in a computation may change due to inability to connect to the battlefield network (out of range) or unwillingness to connect to battlefield network (silent mode). Thus, any global state synthesis must not base algorithmic steps on the number of nodes.

In view of these differences from the classical model, applicability of existing active and passive approaches to a digitized battlefield is reviewed below. In a battlefield, a combat vehicle’s own lethality and survivability issues take precedence over external requests for information. Therefore, the nodes in a network may not remain responsive at all times. In an *active snapshot* approach, local channels and states of all nodes are exchanged for synthesis of a global state. This method requires O(n2, n3) message passing during local state collection but guarantees consistency if completed. However, this protocol may not see completion due to lack of responsiveness mentioned earlier. Thus, the primary benefit of active approach is compromised due to uncertainty of strong connectivity property.

Also, this intense message passing will be called for whenever a global state is needed, for example, siting of a new entity. This external event will also trigger computations related to lethality and survivability. These two competing activities will necessitate a dynamic resource allocation. For real-time embedded combat vehicles, deterministic behavior is mandatory.

Dynamic resource allocation prevents deterministic behavior. The required additional processing by this dynamic load might impact timeliness.

However, the active approach is an excellent method of starting with a common picture before commencing activities, when complete connection can be assured and the required timeliness is not severe.

On the other hand, a passive approach, by not requesting information, uses a fewer number of messages and does not dynamically change the computing resource requirement. By a steady stream of message passing, the passive approach avoids resource contention and contributes towards a deterministic combat system design. So the passive approach is adopted as the basis for experimentation. *This approach is augmented by selectively updating the message recipient based on temporal information.*

4. **Technical Model of the Context**

We model friendly combat vehicle as active nodes, that exchanges message with others and enemy vehicles as passive nodes. Let there be 'n' independently computing active nodes, A_1, A_2….A_n connected by asynchronous communication. Let there be also '*m*' passive nodes, P_1,P_2….P_n in the vicinity. Normally, size of this set (*m*) will be unknown.

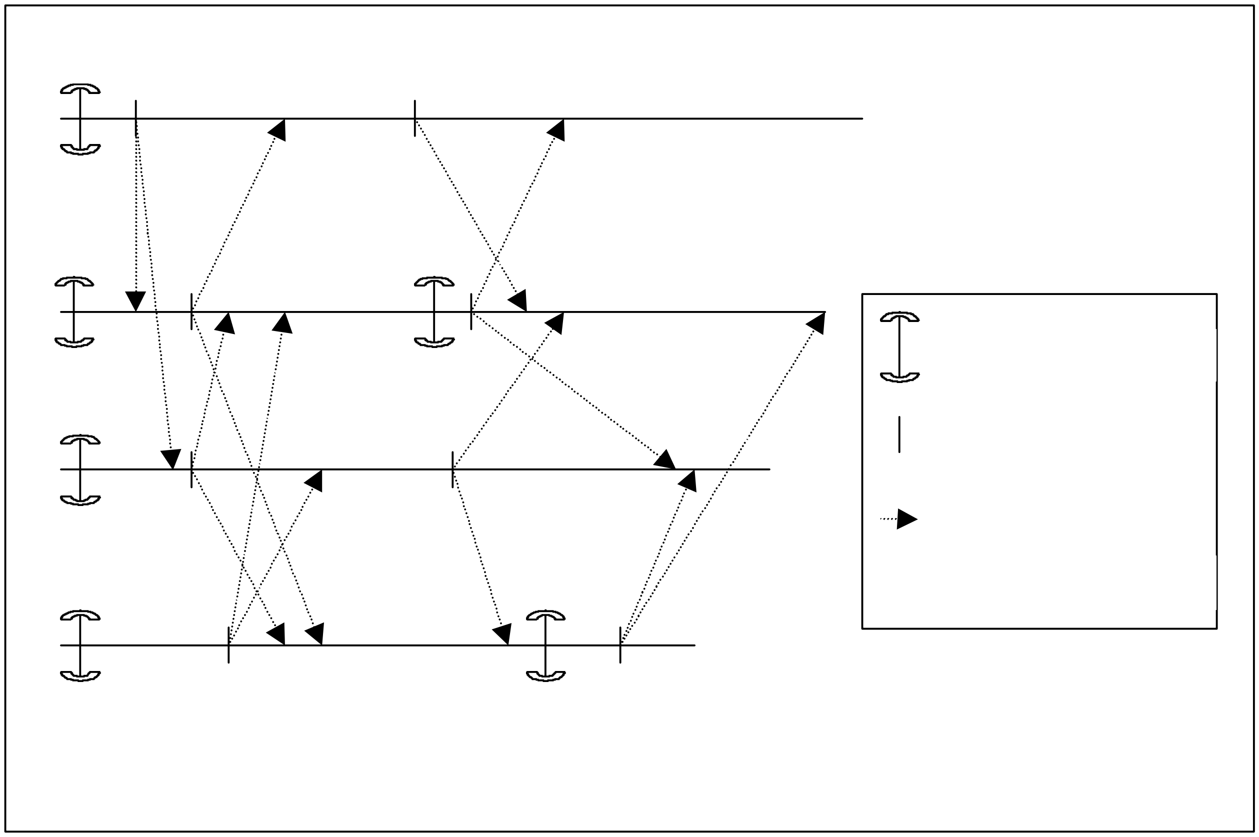


Fig 1: Illustrated System Model

Each of these nodes, active or passive is mobile in a terrain. In this investigation, we want to reason about positions of active and passive nodes. Let position be a record of three fields (x, y, z) of floating point numbers with respect to a predefined world co-ordinate system.

4.1 ***Illustrated System Model***

An illustrated system model, in the notation of distributed computation, is shown in Fig 1. The diagram shows four nodes, computing and communicating with each other. Let us represent execution of a node by a horizontal line. The subscripts designate state change. For example, node A_1 is shown to advance from state A_{11} to A_{15}. Each node changes it state, if a message is sent (A_{12}), or a message is received (A_{22}) or the last sensing event is completed (A_{26}).

4.2 ***Communication Model***

Any two active nodes, (i, j) can communicate with each other through two unidirectional channel C_{ij} or C_{ji} by exchanging messages. Channels are reliable. The communication network is strongly connected but not necessarily completely connected. There is a maximum bound for communication delays, but the actual delays are not deterministic.

4.3 ***Message Exchange Protocol***

To maximize probability of constructing identical global state at all the participating nodes following rules are utilized.

1.) A node periodically senses its surrounding. Immediately after completion of this sense event, it sends a message to its neighboring peers. A sent message includes its own position and a bag (G) of positions created from the last sense event and all the received messages.
2.) If a predefined distance threshold 'd' is exceeded during movement, a node sends its position along with a newly synthesized value of (G).

5. **Selected Results Of This Investigation**

There are two issues to be addressed. First, if a group of passive vehicles follow a pattern of movement in a battlefield, is it possible to correctly find out about that pattern in each active vehicle? The established pattern must be identical at all the active vehicles. Second, is it possible to reduce redundant data, based on the fact that combat vehicles use certain pattern in movements? One of the important sub-issues in this context is that, is it possible to derive a closed form equation about patterns of movements in combat vehicles?

5.1 ***Stage One Result***

This investigation proved and validated through simulation that it is possible to construct a class of algorithm to address the first issue. Briefly, we restate below the concept of *vector clock[2]*, that is critical to this development and outline of the algorithm.

5.1.1. *Vector Clock*

A vector clock is an array of '*n*' integers, where 'n' is the number of nodes in a distributed computation. Each *Vector Clock,* present at each active node, follows a set of ticking rule. Initially, value of each array element is zero, except the array element corresponding to the node which is one. After initialization, a vector clock keeps track of events by ticking it. Each node increments its own element of the array whenever a send, or receive event occurs. Each message includes a copy of the sender's vector clock array. Each receiver updates its own local copy of the vector clock to a clock value that is constructed by taking the maximum element from either the vector clock received in a message or the local vector clock.

5.1.2 *Algorithm*

Given the notion of vector clock, solution to the above problem is straightforward. In a digitized battlefield, recent positions must be immediately delivered to keep abreast of the current situation. Each node keeps 'n-1' separate *stacks,* S_1,S_2, .. S_n, of messages received so far. The most recent received message from node 'k' will be on top of stack 'k'. Let 'G' be the global picture we are trying to construct at a node 'i'. Following the steps outlined below each node periodically creates 'G' and a list of recipient requiring update. The list of recipient is based on the last vector clock value available at a node.

1. Start with an empty set 'G'.
2. Include own position in the set.
3. Include all locally sensed positions in the set.
4. Order the stacks, mentioned above based on the vector clocks of the received messages.

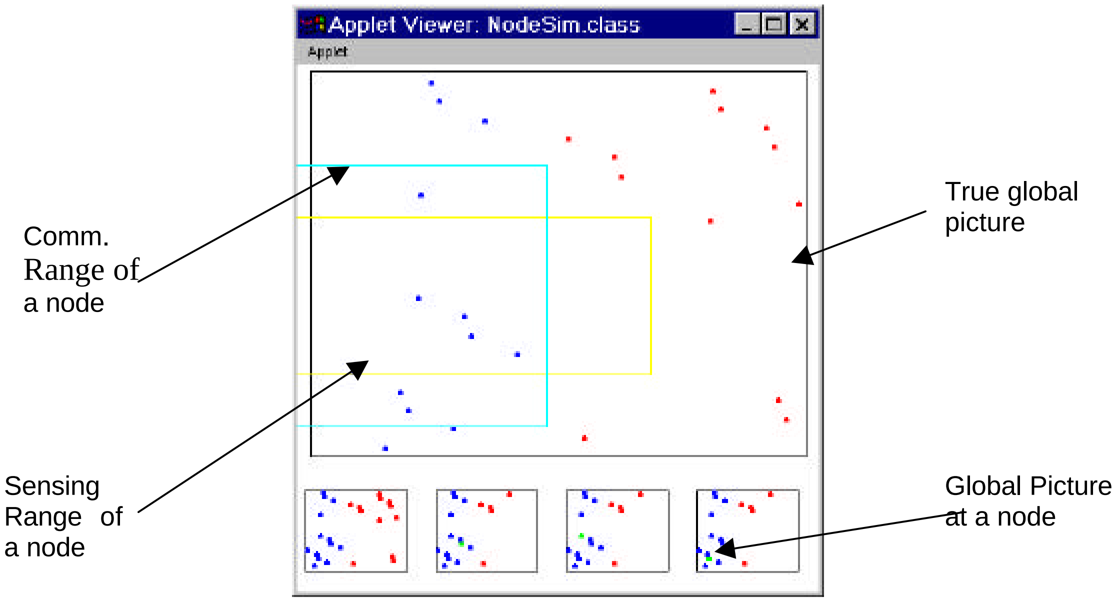


Fig 2: True Global Picture
& Global Picture At Selected Nodes

5. Starting from the stack with the highest vector clock value, for each message (m) on top of the stack selectively add positions from the message to the set of positions created in step 3.

- If two positions are not within an error margin then add the position from message ‘m’ to the set ‘G’.
- Always update positions of an active entity in ‘G’, if the value of vector clock is higher in message ‘m’.
- Update positions of passive entities if the vector clock value of active nodes that detected these passive entities is higher in ‘m’.

6. Since all received messages contain a complete vector clock array, it is possible to find out from a received message the state of temporal knowledge the sender has about a third active node. If the sender’s clock value corresponding to this third active entity is less than similar value at the recipient then the sender must have older information. This information is used to prepare a list of recipients that have the need to be updated.

7. Send the set of positions created in step 5 to at least all the recipients established in step 6.

The significance of this enhancement is in step 6, shown above. *Identification of active entities that are contributing towards incorrectness and inconsistency of the global picture at each node enables one to improve the situation by providing these entities with required information.*

5.1.3 ***Simulation***

The illustration above shows pictorial representation of global and local picture in a simulated ground combat environment. The large rectangle represents a view of the battlefield for an omnipresent observer. The nodes within the large rectangle represent relative positions of active (blue) and passive (red) nodes. The smaller rectangle at the bottom represents local view at a node, colored in magenta, of the global picture created through message exchange. The yellow rectangle within the larger rectangle represents sensing range of one of the nodes. Similarly, the light blue rectangle represents the communication range of a node.

This algorithm has been simulated in a JAVA test bed environment. In this completely object oriented simulation, each active or passive node has been implemented as a separate thread of execution. A *Battlefield Network Object* can simulate *point to point* or *multicast* message exchange among all active entities. In the simulation, a user can define the number of active and passive nodes, the sensing range of active nodes, and range for massage passing. Each message includes the current global picture at the sender and its latest vector clock. Sensing is modeled by creating a *Global Position* object. Whenever an active node senses its surrounding it refers to this *Global Position* object. This Object is *monitor controlled* as there are many reader and writer involved with its data. All the entities report their position to an Applet, which in turn pictorially represents the true global position and global positions as seen by selected active entities. With this pictorial representation, similarity of global pictures at each node can be verified. Also, this simulation can generate numerical position and message data to verify conjectures.

5.2 *Stage Two Results*

For the second stage, researchers at the Wayne State University have developed a correlation method that can correctly identify simple pattern at each node given a global pattern. The data is gathered through the algorithm developed at stage one.

5.3 *Stage Three Results*

On the surface, it seems straightforward to construct a closed form equation from the position data of a group of moving combat vehicles. To wards this goal, ModSAF has been enhanced to gather position data. Given below is the result of this search for closed form equation.

5.3.1 *Closed Form Symbolic Expression*

In order to search for a closed form symbolic expression that relates to positions of a group of combat vehicles, position data from a platoon in movement has been collected. As data from NTC has not been available, data from simulation is used in this endeavor. The data consists of about 600 points for each of the four vehicles in a platoon. This data is utilized to fit in classic predictors such as ARX and Kalman filters. The results of the data and the fitting are not very encouraging. The closeness of fit for this trial is not acceptable.

The rationale for this misfit are as follows. The temporal pattern of movement is influenced by several parameters. The physical parameters of the vehicle - such as size of the vehicle, maximum speed, maximum turning radius, defines certain characteristics of motion. In addition, the context - such as the nature of terrain (muddy, uphill etc.) contributes to the pattern of movement. Moreover, the state in which a vehicle is traveling also defines a few characteristics of motion. For example, if a group of vehicles are in bound and over watch pattern it moves different way than when it is in a road march. Based on this analysis the plan to search for a closed form equation has been changed as outlined below.

6. **Path Forward**

The stage one and two effort will be continued with the existing approach as they demonstrate encouraging results. The third stage requires some revision. The position data gathered will be segmented, initially by detaching from terrain and considering type of formation and motion. Motions of a group of vehicle will be categorized. The formation will also be taken into consideration. If there is a closed form equation available then it will extended to positions with terrain. After developing the first integrated prototype, we intend to explore the possibility of incorporating this algorithm in the Embedded Battle Command development process.

7. **References**

1. V.K. Garg, "*Methods for Observing Global Properties in Distributed Systems*", Oct-Dec, 1997, IEEE concurrency
2. Distributed Systems, edited By Sape Mullender, Addison-Wesley, second edition, 1994

3. F. B. Schneider, "*Implementing Fault –tolerant Services using the State Machine Approach*", ACM Computing Surveys , Dec 1990, 22
4. R. Prakash & M. Singhal, "*Low-Cost Checkpointing and Failure recovery in Mobile Computing Systems*", IEEE Transactions on parallel and distributed systems, Vol 7, No 10, Oct 1996
5. S.T. Huang, " *Detecting Termination of Distributed Computations by external Agents*", Procedure Ninth International Conference on Distributed Computing Systems, pp 79-84, 1989.
6. K.M. Chandy & L. Lamport, "*Distributed Snapshots: Determining Global States of Distributed Systems*", ACM Transactions on Distributed Systems, Vol. 3, No. 1, February 1985, pp 63-75
7. V. K. Garg, "*Principles of Distributed Systems*", Kluwer Academic Publisher, 1996.
8. Global States and Time in Distributed Systems, edited by Zhonghua Yang and T. Anthony Marshland, IEEE Computer Society Press.
9. R. Cooper, K. Marzullo, "*Consistent Detection of Global Predicates*", Report TR 91 –1200, April 1991, Department of Computer science, Cornell University, Ithaca, NY.
10. V.K.Garg and B. Waldecker, "*Detection of Weak Unstable Predicates in Distributed Programs*", IEEE Transactions on Parallel and Distributed Systems, !994.
11. M.Spezialletti and P. Kern, "*Efficient Distributed Snapshots,*" Proceedings of the 6 th IEEE International Conference on Distributed Computing Systems, 1986, 382-388.
12. S.P. Sarkar & P Richardson, *Developing efficient algorithm for digitized battlefield through simulation*, Summer Computer Simulation Conference, July, Chicago, 99 (submitted).
13. S.P. Sarkar & P. Richardson, *Towards Consistence Snapshot of the Digitized Battlefield*, AeroSense, SPIE's 13th Annual International Symposium, Florida, April 99.

BRIEF BIOGRAPHY: Dr. Susanta Sarkar is a research scientist at the VETRONICS Laboratory of the US Army Tank Automotive Armaments Command, Warren, Michigan. He is the leading expert in vehicle electronics & embedded software. He is currently working in the, Digitization of the Battlefield, Joint Technical Architecture, and Robotics. With his initiative and expertise, the Weapon System Technical Architecture Working Group, consisting of technical experts from Aviation, Missile, Ground, & Soldier domain came to a consensus to develop an Army Technical Architecture Framework to address interoperability. In the digitization area, he is currently involved in development of algorithm to reconcile multiple images of battlefield entities. He has worked in the fault tolerance, visualization, and soldier machine interaction area. In the soldier machine interface area, he has successfully completed several major experimental investigations. As an adjunct professor of Wayne State University, Detroit, Michigan, and Oakland University, Rochester Hills, Michigan he regularly teaches computer science & engineering courses. He has been selected as one of the co-chairs of an IEEE conference, ANNIE. He has a Ph.D. in Computer Science and masters in Computer Engineering.