Sensors & Transducers Journal, Vol.0, Issue 0, Month 2009, pp.

www.sensorsportal.com

Sensors & Transducers

ISSN 1726-5479

http://www.sensorsportal.com

Effective Management and Energy efficiency in Management of Very Large Scale Sensor Network

Moran FELDMAN[1], Sharoni FELDMAN[2]

[1]Faculty of Computer Science, Technion – Israel Institute of Technology, Haifa 32000, Israel
Phone: +972-4-8294948, E-mail: moranfe@cs.technion.ac.il

[2]Advance Technology Consulting (ATC), Haifa, Israel
Phone: +972-4-8294948, E-mail: feldsh@netvision.net.il

Received: */Accepted:* */Published:*

Abstract: One of the major challenges in deploying large scale sensor networks is to make the sensors weave dynamically and autonomously into a sensing plan. In this paper we present a novel algorithm with the following characteristics: (a) it is applicable for thousands of sensors, (b) it uses the tree based organization to present an aggregation method that aggregates discrete events into compound events that reduce the traffic, (c) it presents a set of security levels to ensure that events are transmitted to the sink, (d) it presents backup layers to ensure maximal connectivity and (e) it is applicable for sensors without GPS. The algorithm was successfully tested using the dedicated IFAS simulator on a terrain containing up to 10,000 sensors. Our results show that the sensors succeed to weave into a sensing plan, identifying multiple intruders and reporting the events to the sink in a short time.

Keywords: Large Scales Sensors; Routing; Localization, Data aggregation

1. Introduction

The evolution of microelectronics and communication technologies facilitates the manufacturing of miniature sensors comprising of a small transmitter/receiver, a processor, memory components and a low-power battery [1, 2]. Most often, the sensor, or node, is a Boolean sensing device, able to detect an event within a given sensing range, and report the event using a wireless sensor network (WSN) to one

of the sinks. The WSN is constructed via inter-communication between physically adjacent nodes.

WSNs have drawn considerable attention from the research community. The network organization of most WSNs fall into one of a few common categorize, defined according to the network structure [3]. The first category is flat routing, wherein all nodes have an identical role. The routing of events from the sensing nodes to the sinks can use any node in the network without any limitations and without interacting with any centric nodes [4, 5, 6, 7].

The second category consists of hierarchical routing protocols. In these kinds of protocols, part of the sensors possesses additional tasks. In this case, the sensors are grouped into clusters. One of the sensors in every cluster is designated as the cluster-head. If necessary, it is also possible to group cluster heads into new clusters [8, 9, 10, 11]. This organization method delegates routing responsibilities to the cluster heads, and remove this burden from the regular nodes. However, the additional tasks of the cluster heads require them to be more powerful (*i.e.*, have higher energy resources), as opposed to the regular low-power nodes which are used for sensing. Often the cluster heads aggregation data in order to reduce the number of messages transmitted in the direction of the sink. This method is applicable when the number of sensors is very large. Hierarchical routing is considered an efficient way to reduce energy consumption by transferring the aggregation process to the cluster head.

A third category is location-based routing. Such protocols assume that the sensors are aware of their position in the theater. This location information can be exploited in order to route data over the network more efficiently. The sensor location can be obtained from a GPS receiver installed in every sensor [12]. Another method uses relative coordinates that are based on information gathered from neighboring sensors. The distance between neighboring sensors can be estimated according to the strength of the incoming signals [13, 14, 15]. An interesting approach is presented in [16]. The traffic is divided into two types: high priority traffic and low-priority traffic. High priority data is routed using a dedicated congestion zone arranged as a spanning tree with the sink as a root and the low priority traffic is routed via other nodes on longer paths. This model enables each node to be connected to several trees; one tree per sink.

Another classification of WSN routing protocols is based on the amount of energy spent on network organization in advance. There are three main types under this classification: proactive, reactive and hybrid protocols. The first type requires every node to prepare the routing paths in advance, prior to the actual need. This type of protocol requires persistent maintenance of the routing tables, which demands a constant energy input. A reactive protocol creates a routing path only when it is actually needed, thus saving energy. The weakness of reactive protocols compared to proactive ones is the time needed to create a route once such a route becomes necessary. Hybrid protocols constitute a combination of the former types.

In this paper, we present the Very Large Scale Sensor Network Algorithm (VLSSNA). The VLSSNA presents a novel routing technique for very large sensor network composed of 10,000 sensors. This number is significantly higher than the number presented in literature [4]. This algorithm creates a flat network (or single cluster network) without hierarchy and specific tasks to selected sensors. Moreover, the algorithm enables the network operators to define a connectivity level to the network. This connectivity level enables the operator to assign an "importance level" to events and to transmit important events on parallel layers of the network. This feature increases significantly the probability that events to reach the sinks even in case some intermediate nodes malfunction.
The sensors dissemination process is totally a random process without any prior definitions in the sensors. The organization of the network is an autonomous procedure requiring no external intervention. In our research, we assume a sensor density that prevents an object from crossing the sensing field without being detected [17].

A sensor network faces two basic, and somewhat contradictive, requirements. The first requirement is to use as few energy as possible to transfer events to the sink. The other requirement is to ensure that critical events will not disappear due to malfunctions. Energy economization is achieved using an aggregation process that minimizes the number of messages and a communication method that combines broadcasting and sensor-to-sensor communication.
The remainder of this paper is organized as follows. Section 2 presents the theater and the network elements, section 3 presents the sensors control and management, section 4 presents the events management process, event aggregation and delivery verification, section 5 deals with energy saving methods, section 6 presents the simulator used for evaluation of the sensor network and section 7 presents the simulations and results. Conclusions follow in section 8.

2. **Theatre and Network Elements**

2.1 The Theater

Error! Reference source not found. presents a schematic view of a typical theater on which the sensors are dispersed. The sensors are dispersed so that their density ensures that every target will be detected by one or more sensor. At the edge of the theater, three base stations (BSs) are placed, arranged in an equilateral triangle. The BSs are connected by a high-speed communication link. At least one sink controls and monitor the events.

2.2 Network Elements

a. Base Stations

The BSs are identical, and they are controlled by the sink. Every BS is required to know its exact position in space (x, y, z) coordinates. This information can be obtained manually when the BS is installed or via a GPS. In addition, all BSs clocks are synchronized.

A BS is constructed of the following units:

1. A long-range downstream transmitter that cover the whole theatre.
2. A short-range downstream transmitter that cover the adjacent sensors.
3. An upstream receiver that receives messages from adjacent sensors within their transmission range.
4. A high speed LAN that connects all BSs and the sink. This LAN is used to transfer synchronization data among the BSs and the sink, alarms received from the sensors and messages from the sink to the BSs and the sensors.

b. Sensors

The sensors are scattered in the terrain. The distribution of the sensors in the theater is not required to be uniform, however, it is assumed that no sensor is isolated. A sensor is a dispensable device of a limited life expectancy. The network can continue functioning even with a certain decrease in the number of active sensors. The design of the network allows the operators to add periodically new sensors to the field to replace the sensors that ceased to operate.

A sensor is composed of the following major units:

1. A binary sensing device such as a microphone. The sensing distance is limited and usually (but not always) is shorter than the transmission range. The sensing device is able to detect an intruder within the detection range but is unable to detect the direction or distance of the event.

2. A short range transmitter. **Error! Reference source not found.** presents a sensors field. The dashed lines present the sensing range while the full lines present the transmission range. A sensor can receive messages transmitted from other sensors within their transmission range or from the BSs.
3. A processor that runs the stored program.
4. A battery that supplies energy to the processor and transmitter.

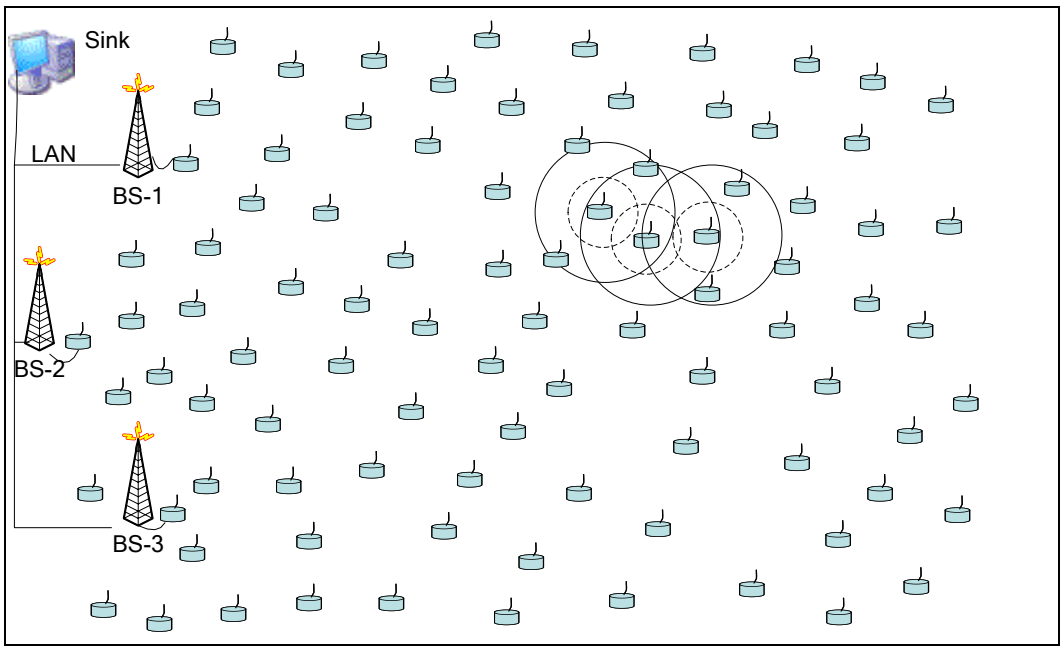

Fig. 1. A typical sensor theater. Three base stations organized as an equilateral triangle and a single sink monitor the wireless sensor netwrok.

3. Sensors Control and Management

3.1 First Network Activation

The network activation process starts after all sensors are scatterred in the field. Sensors in the field will not start to operate until the activation process is done. The activation process is used to synchronize all sensors clocks, and let the sensors determine their physical location. The activation process has 3 steps and is coordinated by the BSs:

1. BS_1 sends a beacon that covers the whole field. This beacon carries the following data elements:
   a. Time stamp used to synchronize the internal clock of every sensor and the other BSs.
   b. The name and geographical location (XBS_1, YBS_1, ZBS_1) of BS_1.
   c. Wait time (t_w) in milliseconds.
2. BS_2 waits tw milliseconds (the value t_w is set in step 1) before broadcasting a beacon. This beacon cover the whole field and carries the following parameters:
   a. Time stamp.
   b. The name and geographical location (XBS_2,YBS_2, ZBS_2) of BS_2.
3. BS_3 will wait t_w milliseconds after BS_2 transmission before broadcasting a beacon. This beacon parameters and the process are identical to these of BS_2.

After receiving the 3 beacons, every sensor starts its localization algorithm (based on trilateration). This algorithm use the information received in the beacons, and the arrival time of each beacon. The output of the algorithm is physical location of the sensor and the exact time (according to the clocks of BSs.

The network activation process runs periodically and is used to join new sensors that were added to the field or remap old sensors that were moved inside the field.

3.2 Messages and Data transfer

The communication among the network elements is performed by messages. The network elements exchange broadcast messages addressed to all listening nodes within the transmission range and directed messages that address a specific node within the transmission range. Another type of classification is based on the transmission range of the message originator. A BS can transmit short range and long range messages while a sensor is capable of transmitting short range messages only. Table 1 summarizes the possible combinations of message types and message transmission ranges. While a BS can send directly a long distance message to every node in the terrain, a node which is required to send an event to the BS, is required to use intermediate nodes to bridge the distance. The process of transferring the information from the node to the BS is based on a "store and forward" mechanism. A node that received a message will forward the message to the next leg in the chain only after the message was received completely.

Table 1. Possible Combinations of Message Types and Ranges

MSG Range / *Type*	Long Range Message	Short Range Message
Directed Message	A message sent by a BS, directed to a specific sensor in the theater. The addressed sensor is identified by unique sensor-id within the terrain.	A message sent by sensors or the short-range transmitter of the BS. This message is addressed a specific sensor identified by unique sensor-id within the transmission range.
Broadcast Message	A message transmitted by a BS. This message is targeted to all sensors in the terrain	A message sent by sensors or the short-range transmitter of the BS. This message targets all receiving nodes within the short transmission range.

Special attention was given to energy saving. We implemented "energy saving" methods in critical and demanding procedures. Details appear in the relevant sections below.

3.3 Trees formation processes (TFA)

During the trees formation phase, each BS construct a spanning tree. The constructions are done simultaneously, but independently. This phase starts after the activation process and terminates once all sensors joined all trees.

Fig. **1** presents a glance to the tree formation process. Fig. **1**A presents the first step in the tree formation process initiated by BS_2. BS_2 sends a beacon (actually a "Hello" message) via its short-range transmitter. Three nodes are within the transmission range and they join BS_2 tree and get the logical address <2.1>, <2.2> and <2.3> The prefix <2.> represents the root Base station. Now every one of the nodes <2.1>, <2.2> and <2.3> continue independently the process of building BS_2's tree. Fig. **1**B presents the next step. Nodes 1-3 send a beacon and let additional nodes join the tree as their children. For example, node <2.2> gets two children: nodes <2.2.1> and <2.2.2>, while node <2.3> gets nodes <2.3.1> and <2.3.2> as its children. The number of children of a node is unlimited and can vary from node to node. Fig. **1C** presents the ongoing process of building the tree.

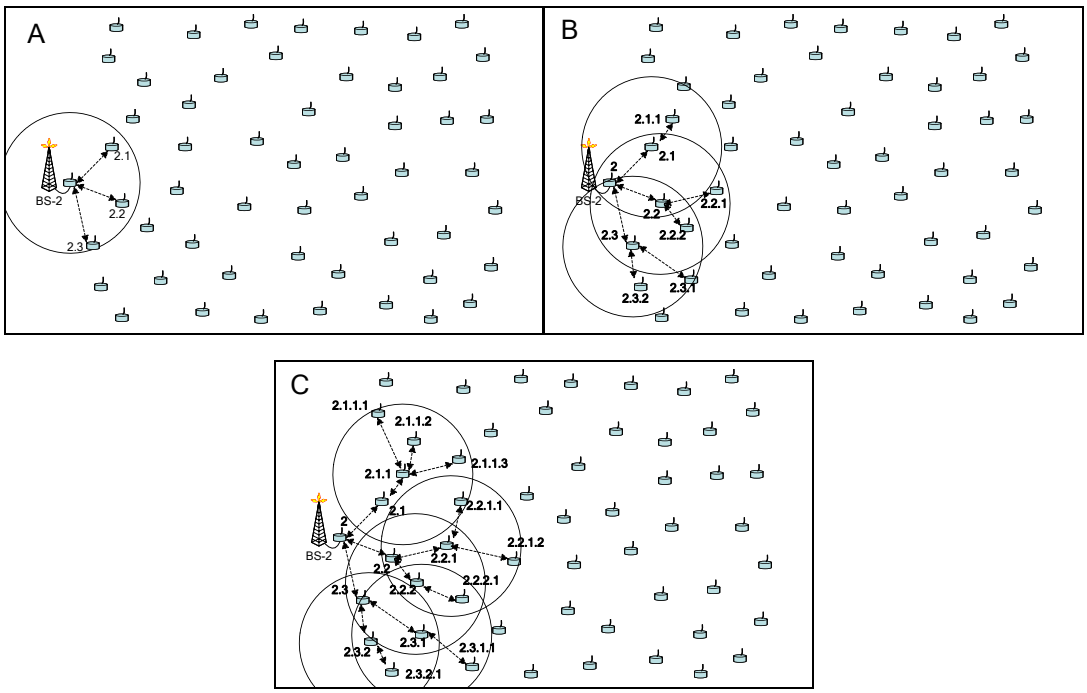

Fig. 1. Tree formation process

As a rule, a node (Except the root) does not send a become before it itself received a become and joined a. Fig. 2 presents 2 trees in theater. The tree of the root node BS_1 and the corresponding nodes addresses are presented in black using continuous line. Addresses of this tree start with <1.>. The tree of root node BS_2 and the corresponding nodes addresses are presented in blue using dashed lines. The addresses of this tree start with <3.>.

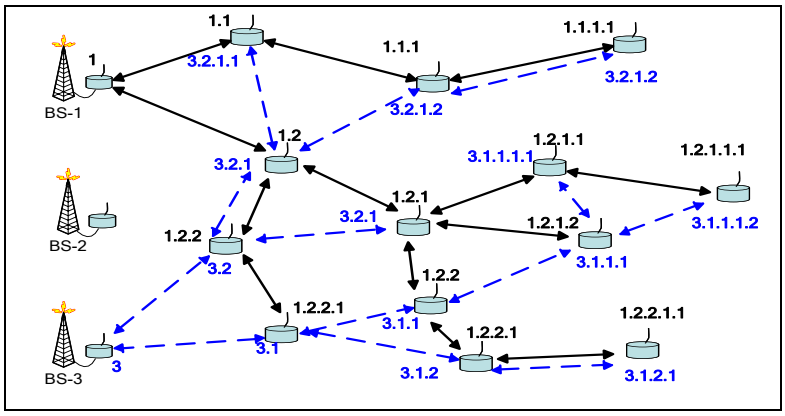

Fig. 2. Trees in the terrain

3.4 Detailed Description – TFA algorithm

In this section we describe in details a single TFA. TFA organizes the nodes in the field in a tree rooted at a given BS. Only nodes that belong to the tree can transfer events to the BS which acts as the tree root of this tree. To ensure maximal connectivity, all nodes try to join the tree. Every node in the field has a unique *node-id* (like phone number or IP address) and virtual coordinates (of the type *x.y.z..)* that are tree specific, and may change when the tree structure changes. Every tree is identified by a "*tree name*" which is the *id* of its BS root node. Nodes send periodically beacons (*hello* messages). If a node that does not belong to the tree receive a beacon from a node in the tree, it becomes a child of the beacon sender. The protocol satisfies the follow properties:

1. Eventually (assuming no dynamic changes occur) all nodes within transmission area fuse into a single tree.
2. The protocol maximizes the number of nodes joining the tree in each step (yielding a parallel fuse).
3. Nodes periodically attempt to balance the tree by improving their position in tree and joining higher-level nodes.
4. The protocol is fully distributive with no “central" bottlenecks, namely it is defined at the level of pairs of nodes. The merging node gets new coordinates in its new tree according to its new position.
5. Running alarms are not be affected (*i.e.*, do not break) in any way as trees join or rearrange themselves.
6. The parallel process of creating the trees (with the roots BS_1, BS_2…) if fully orthogonal (*i.e.*, there is no dependency between the trees).
7. When all TFAs stabilize, every node is a member of every trees.

The TFA algorithm sends Hello messages periodically. The frequency of the transmission run depends on the stability of the network and the changes in the field. The repeated transmission is required in order to bridge “holes” created by faulty sensors, and to add new sensors to the network.

4. Events, Events aggregation and Delivery Verification

In our model, every sensor has a circular events detection area around the sensor. An intruder that enters the sensors field will stimulate every node whenever it enters its detection radius.

Every node that detects an intruder within its detection range sends an event message to its father. Fig. 3 presents a section of the sensors field and the intruder path within it. The circle around each sensor presents the detection zone of the sensor. Our assumption is that every sensor is able to perform basic filtering of the detected noise, i.e. a sensor programmed to detect a noise of a car will not respond to a noise created by a walking animal or the barking of a dog.

Fig. 3 presents a case when an intruder crosses the field and triggers 4 sensors. The root of the tree is <1>, it has two children <1.1> and <1.2>. Node <1.1> has 3 children and node <1.2> has 1 child. Recall that the intruder activates only the sensors located on his path. Every node can take one of the following roles: detect an intruder (like node <1.1.1>), participate in the process of transferring the event to the sink (like node <1.1>) or being idle (like node <1.2.1.1>).

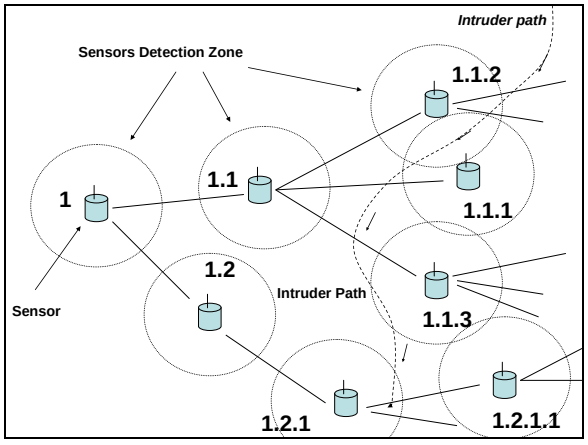

Fig. 3. Sample Intruder Path

The software that runs in all nodes is identical and must cope with the detection, basic analysis and transfer of the event towards the sink. The tree structure creates a natural organization of the nodes, which allows each subtree root to “aggregate” events from the subtree, and minimize the amount of

messages transferred towards the sink. For example, node <1.1> is able to aggregate the events detected by its children <1.1.X> and send it as a unified event to its father. Fig. 4 presents the basic automaton that runs in every node. This automaton enables the node to act in one of two possible ways: detect an event or receive and handle event messages from its children and transfer them either transparently or with some updates toward the sink.

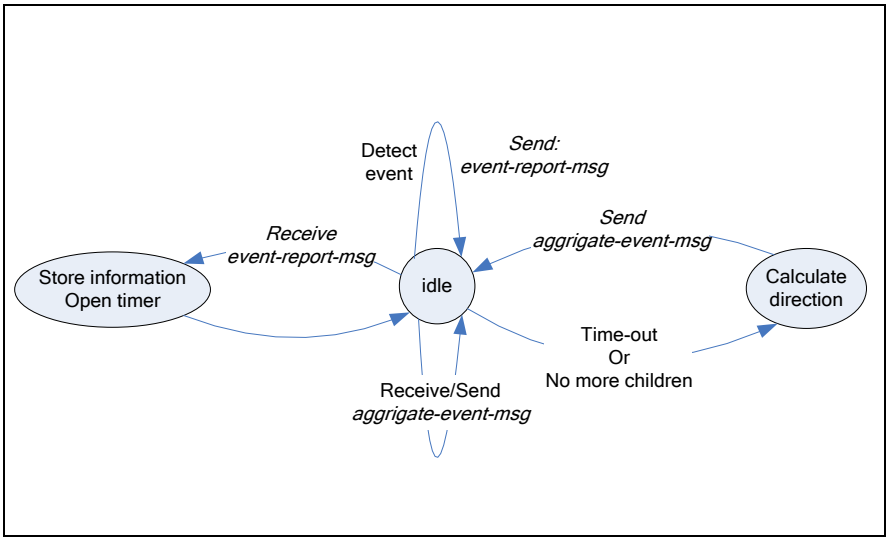

Fig. 4. node basic automaton

The automaton is composed of 3 states.

1. **Idle.** This is the stable state of the node. In this state, the node is waiting for an event created by the intruder or to a message from one of its children. When the node detects an intruder, it will send the message *"event-report-message"* with the event details to its father. The main event details are the geographical location of the node and the event timestamp. After detecting an event, the node will not report any additional event for a short time span. This prevents the node from creating a flood of messages toward the sink caused by a single short timed stimulation.
2. **Store information – open timer**. As soon as the node receives an "*event-report-message"* from one of its children, it assumes that its other children may detect this same event. The node stores the event data and waits for additional messages from the other children. A short wait timer is activated in order to limit the wait time. If the node receives an "*event-report-message*" from all its children, the timer is cancelled because it becomes redundant.
3. **Calculate direction**. This state is activated once the node decides to stop waiting for additional "*event-report-messages*", either because of timer expiration or because of the node received a message from all its children. . The node tries to calculate the local direction of the intruder, based on the messages it received. The results are then sent towards the root using the message "*aggregated-event-message"*. It is possible to calculate the direction only if two or more children reported the event. In case that only a single child detected the event, the aggregated message will carry the location of the detecting node. In case 2 or more nodes have detected the event, the message will carry the locations of the two most distant detecting nodes.

   When a node receives a message "*aggregate-event-message"* from its subtree, it acts as a pure router and sends it to its father.

For example, assume that node <1.1.2> in Fig. 3 detects an event. This node sends an *event-report-message* to its father with its location and the time stamp of the detection. Node <1.1> receives this message, and enters the "**Store information – open timer**" state. As the intruder moves in the field, nodes <1.1.1> and <1.1.3> will also send *event-report-message* to node <1.1>. As node <1.1> does not have more children, after it receives the last of these messages, it enters the "**calculate-direction**" state and calculates the maximal distance between the child-nodes which detected the event. In the current example, assuming that nodes <1.1.1> and <1.1.3> are of greater distant than any other pair of

children (<1.1.1>--<1.1.2> and <1.1.2>--<1.1.3>), Node <1.1> sends an *aggregated-event-message* to its father, carrying the locations of <1.1.1>, <1.1.3> and the time stamps of detection in these two nodes.

4.1 Data Aggregation

The purpose of the aggregation process is to reduce the amount of data transferred from the detecting nodes toward the root, however, as discussed later, this process slows the speed in which events are transferred toward the sink. The aggregation process can be nested, for example, node <1> in Fig. 3 can aggregate the data from its children <1.1> and <1.2> into a single message. A *k-level* aggregation process performs the aggregation process in the *k* lowest levels of the tree (starting from the leaves). The aggregation level can be adjusted according to the required performance of the network and the type of expected intruders. Fig. 5 presents the data structures of an event transmitted using the *"Event-Report-Message"* message which is composed of two fields identifying the event time and location. The *"Aggregated-event-message"* is composed of two identical substructures – *event1* and *event2*. These fields will be populated with the details of the two of the nodes that participates in the aggregation process and have maximal Euclidean distance.

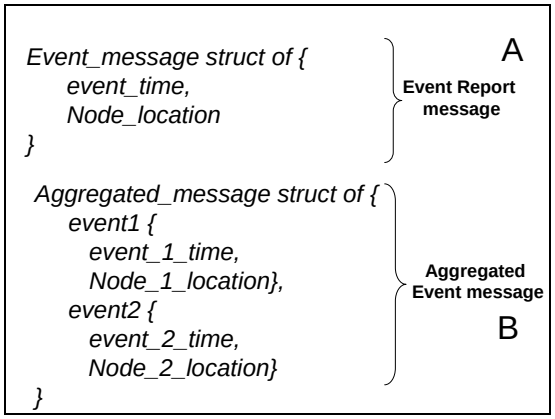

Fig. 5. Event Report Message and Aggregated message structures

For example assume that node <1.1> in Fig. 3 aggregates the *"event-report-messages"* arriving from nodes <1.1.2>, <1.1.1> and <1.1.3>. Assuming that the geographical (*x, y*) coordinates of the nodes are (10, 100), (30, 120) and (90, 100) respectively. The substructure "event 1" in the "*aggregated-event-message"* will be populated with the details of node <1.1.2> and the substructure "event 2" will be populated with the details of node <1.1.3> .

The need to aggregate messages requires that inbound messages will be delayed in the node for a period of time. This delay enables other inbound event messages to arrive and be aggregated into a single aggregated message. Fig. 6 presents two different cases of the aggregation process. Fig. 6A presents the case when an inbound message enters a sensor. This message waits for the "*delay time*" before being outbounded by the node. The delay times of the 3 inbound messages do not overlap, and as a result 3 outbound messages are transmitted towards the next level. Fig. 6B presents the opposite case where two message arrive during the "delay time" of the first message. All three messages are aggregated into a single outbounded message.

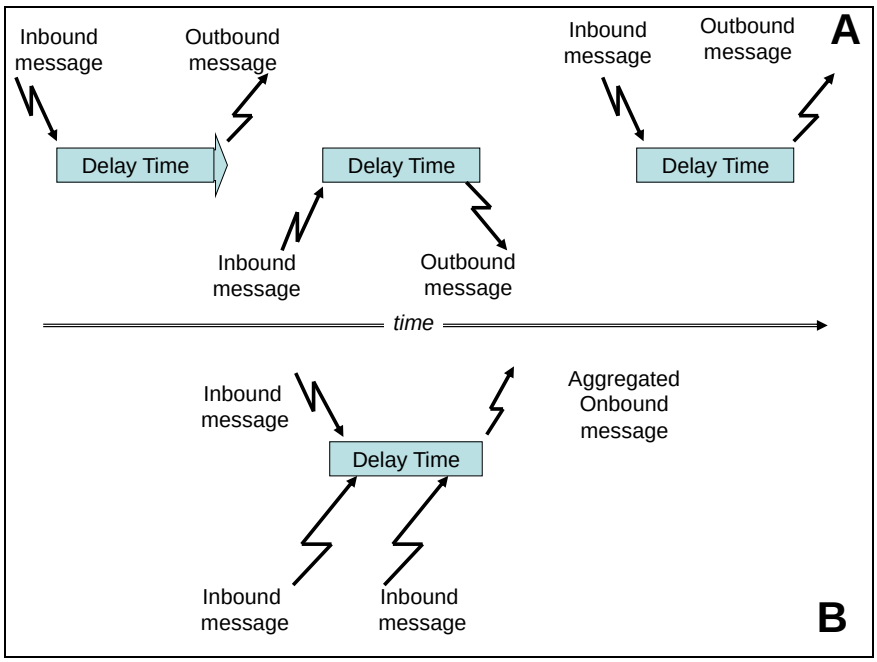

Fig. 6. SchematicDescription of Aggregation Process

4.2 Event Delivery Verification

The verification feature ensures with high level of certainty that the reported event has been transferred successfully from the sensing nodes via the routing nodes to the Base Stations and the sink. The network architecture enables the use two types of verification levels. The basic method is "Report & Forget" which does not require the BS to acknowledge the acceptance of the event report. The enhanced method "Report & Acknowledge" requires the receiving Base Station to send an acknowledgement to the reporting nodes. The acknowledgement need not use the sensors network to get to the sender. Instead, it is broadcastted as a "Shout" message transmitted directly from the BS to the sensor. In case the event originator does not receive an acknowledgment within a predefined period, it resends the event report.

Another capability of the architecture is to use two levels of energy saving delivery methods. The economized method uses a "Load-Sharing" mechanism where the reporting node selects cyclically one of the trees and sends the event report over this tree to a single Base Station. The other method, known as "Active-All" mechanism, is more energy consuming. Under this method, the node sends a copy of the event report over all possible trees in parallel. The replicated events are propagated to the Base Stations and the sink. Using the events time and locations, the sink fuses the replicated events into one single event.

5. Energy Saving Methods

The energy resources of the nodes are very limited. A critical factor in the design of the VLSSNA algorithm was to reduce the energy utilization as much as possible and to enable the network operators to select an operation mode that fits both the needs and the energy resources of the nodes.

The following list presents the major energy saving methods used by the algorithm.

1. The tree formation process is sensitive to the distances between the potential father and the children. The purpose is to balance between the number of relay nodes that an event message has to cross, the latency and robustness of the tree. In case a node is able to control its transmission

power, and has more than a single candidate to become its father, it will select the most "economical" father that is not too close.

2. Optional Acknowledge (Ack) via "*shouting"*. A node that generates an *event-message* is able to ask for an acknowledgment indicating that the event message arrived successfully to the sink. An absence of this confirmation will trigger the sensor to send again the *event-message* after a predefined wait time. A common method is to use the sensors network and send an Ack message back to the originating node via the path of the *event-message* or a similar solution. Our method takes a different direction. The *Ack-message* is sent via the "shouting" mechanism. A message is sent by the sink via one of the Base stations. This message includes the ID of the originating sensor. A successful reception of the *Ack-message* by the sensor will cancel the retransmissions of the *event-message*.
3. Using an adjustable replicated transmissions. A node that generates an *event-message* selects the number of replications according to the criticality of the event. The number of replications can take any value between 1 and the number of trees. The selection of value 1 implies that the originating node will select one tree and send the *event-message* on this tree. In case of more critical events, the node selects two or more trees and send the *event-message* in parallel on these trees.

   The ability of the initiating sensor to select different trees contributes also to an even utilization of energy in the network. For example, if a specific sensor sends more often an *event-message,* the trees selection disperses the burden of transmitting these messages between different routing path.

6. The IFAS: Interactive Flexible Ad Hoc Simulator

For testing and evaluating the protocols described in this research (VLSSNA and TFA) we used the IFAS simulator [18]. IFAS was originally developed for evaluating the performance of ad-hoc protocols and was later adapted for sensor networks. In this section, we shall describe the simulator and the simulation scenarios.

The simulator was originally designed and developed for testing ad hoc protocols and running comparative tests. Special attention was given to the following aspects: (i) enhanced visualization tools that give a full online view of the theater, zooming of selected zones, node movements and voice channels in ad hoc networks, and specific node status including queue status; (ii) tracing the formation of trees ; (iii) tracing the events transfer and sessions in real time; (iv) configuration and simulation definition via online screens; (v) definition and tracking of intruders paths and pace and (vi) support of logging, debugging and analysis tools.

The enhanced visualization capabilities, unique to this simulator, contributed to the understanding of the protocols behavior, as we were able to view the progress in the field and detect unexpected behavior.

The simulator enables the user to get detailed online reports on a single node behavior while the system runs. These capabilities set afloat disruptions in specific nodes behavior as a result of their location in the field. Fig. 7 presents the initial status of entities management screen with 3 Base stations and 3000 sensors. The location of every sensor is selected randomly within the theater. It is possible to "kill" nodes, zoom in and zoom out selected areas and trace explicitly certain events.

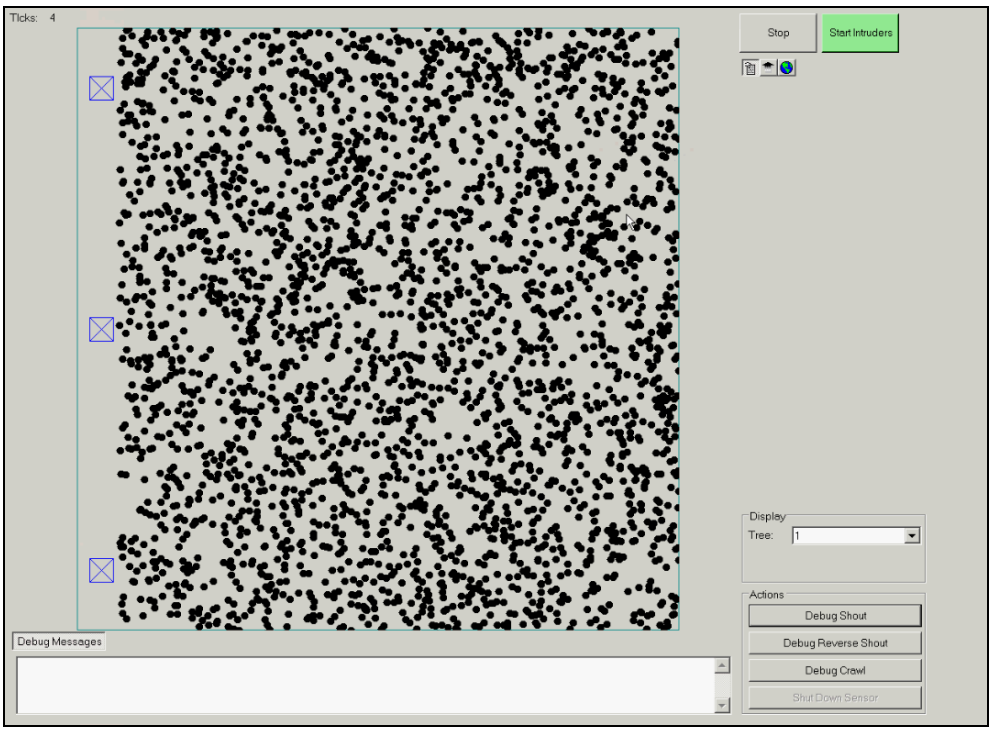

Fig. 7. Sample Testing Terrain with 3000 sensors and 3 Base Stations

Fig. 8 presents one of the trees. The root tree is Base Station 2. The intruders, which are not visible in this view, are crossing the field and activating the sensors. In the bottom of the screen we see the events received by the BS.

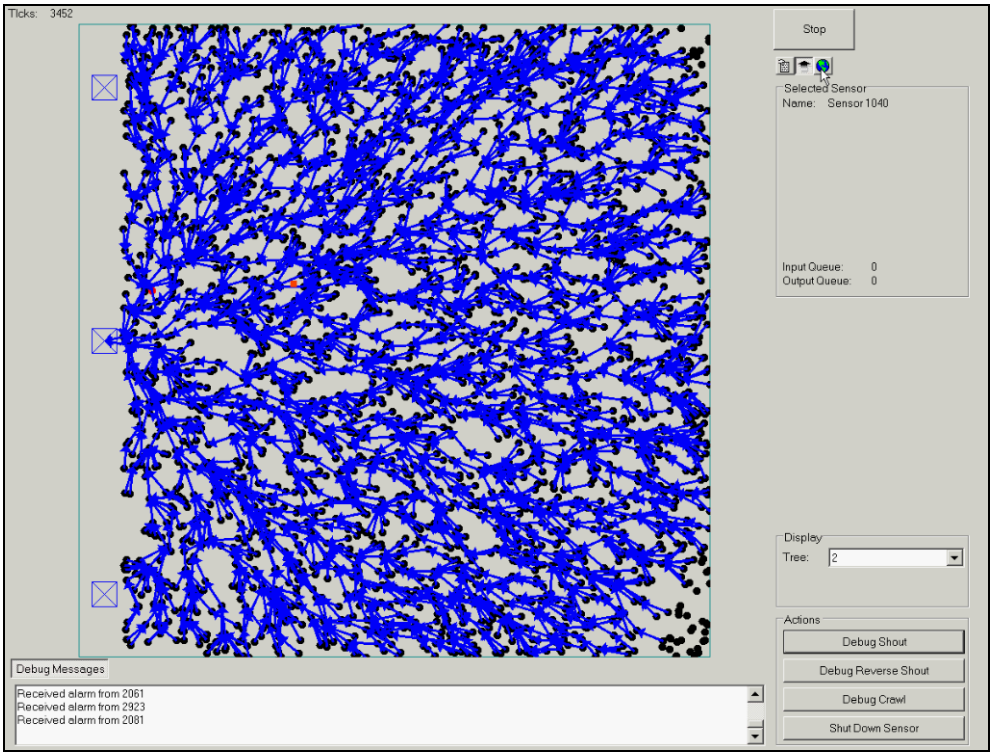

Fig. 8. Sample Tree with BS-2 as a root

7. Simulations and Results

The simulation environment creates the infrastructure to analyze the following elements:

1. The efficiency and scalability of the TFA algorithm.
2. The performance of the network in the following aspects:
   a. The contribution of the aggregation process to reduce the number of messages
   b. The delay created by nodes as a result of the aggregation process

The tests were performed using the parameters presented in Table 2:

Table 2: Quantities details

Parameter	Value
Number of sensors	3000 sensors (except for the scalability tests)
Field size	1Km x 1Km
Sensor transmission range	60 meters
Event detection range	10 meters

Fig 9 presents the BSs organization. BS-1, BS-2 and BS-3 act as the trees roots and are able to broadcast *shout-messages* that cover the whole terrain. In addition, the events are received via these BS.

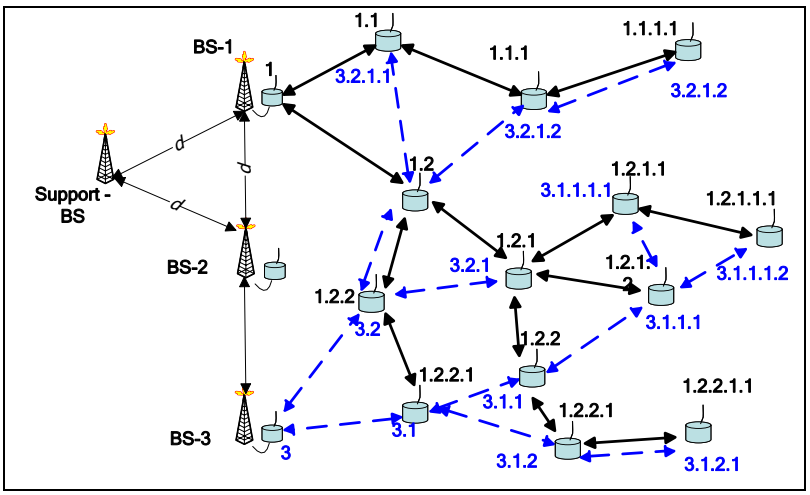

Fig 9. Base stations Organization

The localization algorithm requires using a minimum of three synchronized beacons that broadcast periodically a time event. The best organization for the beacons is in an equilateral triangle. To show an applicative possibility, we combined the functionality of two beacons with BS-1 and BS-2 and created a Support Basestation (SBS) that participates only on the localization process. The distance between SBS, BS-1 and BS-2 is *d*. Note that it is possible to use the SBS as a replacement for BS-3, however, it requires dissemination of sensors also in the triangle created by the SBS, BS-1 and BS-2.

7.1 Scalability and connectivity

The basic requirement from the TFA algorithm is to connect all sensors in the field into one single network. This requirement inspires two sets of tests.

1. Salability tests. We run the TFA algorithm on a field with a minimal population of 1000 sensors and a maximal population of 10,000 nodes. The trees creation process succeeded to fuse all sensors into a single tree without any measurable impact on the performance.
2. Connectivity tests. The connectivity tests included two groups of tests. The first group verified that all nodes dispersed in the field are merged into a single tree. The second group of tests checked what happens if sub-tree nodes within the tree stops functioning. The tests show that if the last condition happens, the nodes that belong to the subtree of the faulty node discover the fault and leave the tree. Once the nodes left the tree, a fusion process starts fusing them back into the tree and results in a new fully connected field.

7.2 Average nodal delay

Intuitively, two factors contribute to the efficiency of the aggregation process – the nodal delay time and the number of the tree levels that participate in the aggregation process. Fig. 10 presents the impact of the nodal delay time on the aggregation process. In this test, we measured the percentage of aggregated messages out of all event messages triggered by the intruder as a function of the nodal delay time. In this test, every inbound message received by a sensor on its way to the sink is delayed for a fixed period before it is outbounded to the next tree level. We expect that the number of aggregated messages to a single message will grow as the internal delay grows. The delay allows more inbound messages to overlap the delayed message and to be aggregated into a single message. One can observe that a delay of 700ms produce 33% of aggregation. A small increase to 40% is achieved when the delay grows to 1200ms. Additional delay time greater than 1200ms does not contribute to the performance. Note that the increase in the performance costs a significant delay in the arrival of the alarm message to the sink.

7.3 The practically of k-level aggregation process

Figure 12 presents the contribution of every level in a *k-level* aggregation process. It is clear that the first and second levels have the most significant contribution. The number of matched messages decreases significantly as the level grows. This experiment shows that 2-3 aggregation levels are enough, as the contribution of additional levels is very marginal. Moreover, as discussed above, adding more aggregation levels contributes to the event transfer delay, and therefore, should be avoided.

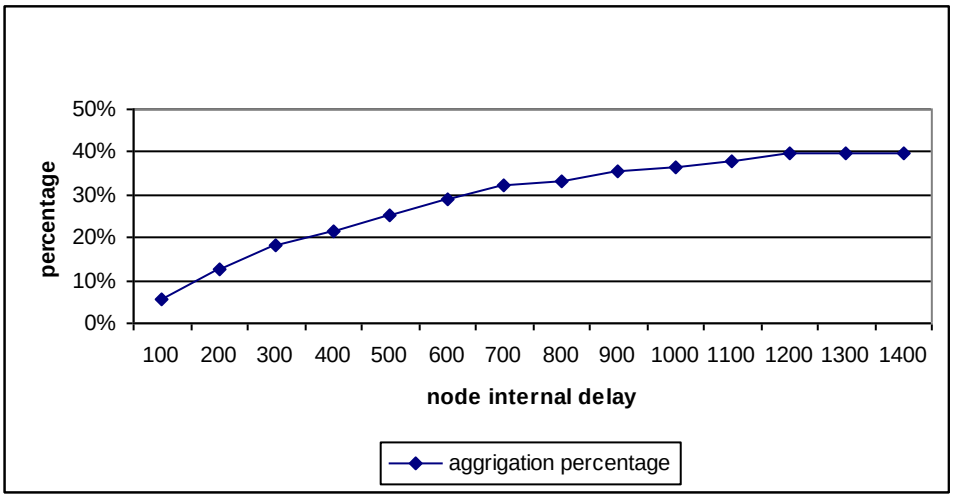

Fig. 10. Average aggrigation percentage Vs. Nodal Delay

7.4 The efficiency of the aggregation process

Fig. 11 presents a subset of our tested scenarios. In our testing environment, the base stations S_1, S_2 and S_3 are located along one side of the theater. The movement direction of the intruders is presented by arrows. In the scenarios presented here, all intruders' speeds were identical and equal to 8 meters/second.

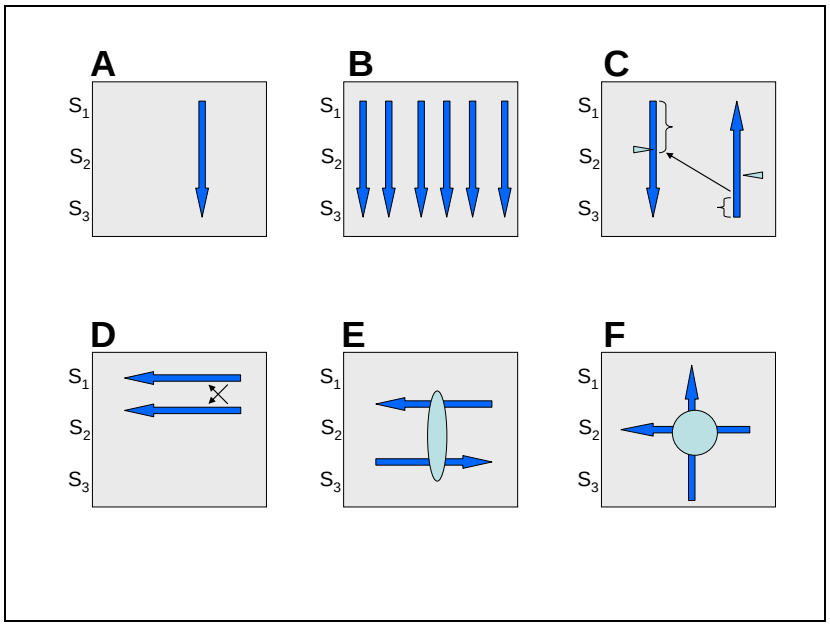

Fig. 11. Scenarios for testing Aggregation Efficiency

Let us discuss the 6 cases of Fig. 11 in more details.

(a) Fig. 11A presents the simplest case where there is a single intruder moving in a straight line along the "y" axis of the terrain. In this case, the network succeeded to reduce the total number of reported events by 60% by aggregating events.

(b) Fig. 11B presents a complex case of multiple intruders that can be divided into two separated sub cases:

a. The distance between the intruders' paths enables the aggregation algorithm to join discrete events into a unified event.

b. The distance between the intruders' paths does not enable a reasonable separation between events generated by different intruders. In this case, the aggregation algorithm may aggregate events from foreign paths.

It is impossible to define specific rules to avoid the last case. A good separation between the paths depends on the moving speed of the intruders which may change, the connectivity of the trees, the exact direction of the path and the locations of the base stations.

(c) Fig. 11C presents a case where two intruders move along to "y" axis, but in opposite directions. Note that the triangles represent the positions of the two intruders in a given time. This case differs significantly from the two intruders case corresponding to (b). When the intruders move in the same direction, and the paths are far, the aggregation process does not mix events produced by the two different intruders. On the other hand, in the case presented in Fig. 11C, the intruders move in opposite directions and events created by the left intruder may meet the events created by right intruder. Mixing of events may happen in case that "old" events on the right path meet "new" events on the left path. It is possible in some cases to filter the mixed events, however, the implementation of such a filter requires adding extra computational power to the nodes which utilizes extra energy.

(d) Fig. 11D presents a case where the intruders move in parallel paths, in same direction but along to "x". This case presents a situation where the simple algorithm can not distinguish between

the separate events in case that the paths are adjacent. The correctness of the algorithm grows as the distance between the paths increases.

(e) Fig. 11E presents a case where the intruders move in parallel paths along the "x" axis, but in opposite directions. The algorithm works very nicely as long as the intruders are outside of the elliptic area. However, the quality of the algorithm drops as soon as the intruders enter this area. Again, we notice that the algorithm works correctly more often when the paths are well separated.

(f) Fig. 11F presents a case where the intruders move in perpendicular paths. It is possible to distinguish between events created by both intruders as long as they are not within the circular area. Some support to the separation capabilities can be added by using additional information describing the direction and speed of the intruder. However, adding this kind of information to the packet transferred between nodes costs additional transmission and processing energy.

8. Conclusions

1. The tree based connection between sensors presents a very efficient and practical way to connect a very large number of sensors in a sensor network. The method presented in this paper allows also a replication tree mechanism that increases the redundancy of the network and ensures a very high level of connectivity.
2. The aggregation algorithm is targeted to reduce the number of events that are transferred from the network to the sinks. The main purpose of this algorithm is to save transmission energy. A major consideration in the design of such algorithms is the tradeoff between the complexity of the algorithm and its memory size and processing power requirements. In addition, complex algorithms tend to require additional data transmission, which increases significantly the energy consumption of the sensors. A good algorithm should save as many messages as possible without introducing significant extra burden on the resources of the sensor.

9. REFRENCES

[1] G. Pottie and W. Kaiser, "Wireless integrated network sensors," Communications of the ACM, vol. 43, May 2000, pp. 51-58.

[2] J. Kahn, R. Katz, and K. Pister, "Mobile networking for smart dust," In preceedings of The Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom), 1999, pp. 271-278.

[3] J. N. Al-Karaki and A. E. Kamak, "Routing Techniques in Wireless Sensor Networks: A Survey," IEEE Wireless Communications Vol. 11 Issue 6, 2004, pp. 6-28.

[4] Y. Fan, A. Chen, L. Songwu, and Z. Lixia, "A scalable solution to minimum cost forwarding in large sensor networks". Proceedings of Tenth International Conference on Computer Communications and Networks, 2001, pp. 304-309.

[5] C. Intanagonwiwat, R Govindan, and D. Estrin, "Directed diffusion: a scalable and robust communication paradigm for sensor networks", Proceedings of the ACM MobiCom, Boston, Ma, 2000, pp. 56-67.

[6] J. Kulik, W. Rabiner, and H. Balakrishnan", "Negotiation-Based Protocols for Disseminating Information in Wireless Sensor Networks", Wireless Networks, vol. 8, Num. 2-3, 2002, pp. 169-185.

[7] D. Braginsky and D. Estrin", "Rumor Routing Algorithm for Sensor Networks", in proceedings of the First Workshop on Sensor and Applications (WSNA), Atlanta, GA, Oct. 2002, pp. 1-12.

[8] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy Efficient Communication Protocol for Wireless Mi-crosensor Networks," Proceedings of the Hawaii International Conference on System Sciences (HICSS '00), Jan. 2000, pp 323-327.

[9] L. Subramanian and R. H. Katz, "An Architecture for Building Self Configurable Systems", in the Proceedings of the IEEE/ACM Workshop on Mobile Ad Hoc Networking and Computing, Boston, MA, August 2000, pp. 63-73.

[10] Q. Fang, F. Zhao, and L. Guibas, "Lightweight Sensing and Communication Protocols for Target Enumeration and Aggregation", Proceedings of the ACM international symposium on Mobile ad hoc networking and computing (MobiHoc), 2003, pp. 165-176.

[11] F. Ye, H. Luo, J. Cheng, and S. Lu, L. Zhang, "A Two-tier data dissemination model for large-scale wireless sensor networks", proceedings of the annual ACM/IEEE MobiCom, 2002, pp 148-159.

[12] Y. Xu, J. Heidemann, and D. Estrin, "Geography-informed Energy Conservation for Ad-hoc Routing," In Proceedings of the Seventh Annual ACM/IEEE International Conference on Mobile Computing and Networking, 2001, pp. 70-84.

[13] A. Savvides, C-C Han, and M. Srivastava, "Dynamic fine-grained localization in Ad-Hoc networks of sensors," Proceedings of the Seventh ACM Annual International Conference on Mobile Computing and Networking (MobiCom), July 2001, pp. 166-179.

[14] N. Bulusu, J. Heidemann, and D. Estrin, "GPS-less low cost outdoor localization for very small devices", IEEE Personal Communications Magazine, vol. 7. no. 5, Apr. 2000, pp 28-34.

[15] S. Capkun, M. Hamdi, and J. Hubaux,"GPS-free positioning in mobile ad-hoc networks", Proceedings of the 34th Annual Hawaii International Conference on System Sciences, 2001 pp. 3481-3490.

[16] R. Kumar, R. Hosam, and C. Guohong, A. Farooq, Y. Aylin and T. La Porta, "Conestion Aware Routing in Sensor Networks", Technical Report, http://nsrc.cse.psu.edu/tech_report/NAS-TR-0036-2006.pdf, 2006.

[17] B. Liu and D. Towsley, " A Study of the Coverage of Large-scale Sensor Networks," in IEEE International conference on Mobile Ad-Hoc and Sensor Systems, , 2004, pp. 475-483.

[18] Y. Ben-Asher, M. Feldman, S. Feldman, and P. Gurfil. IFAS: Interactive flexible ad hoc simulator. Simulation Modeling Practice and Theory, 15(7), 2007, pp. 817–830.