SENDROM: Sensor Networks for Disaster Relief Operations Management

Erdal Cayirci and Tolga Coplu

Computer Engineering Department
Istanbul Technical University, 34469 Istanbul, TURKEY.
erdal@cs.itu.edu.tr, tolga@be.itu.edu.tr

***Abstract*— SENDROM is a novel sensor network architecture, based on state-of the-art technologies, developed to manage the rescue operations after large scale disasters. This architecture mainly consists of sensor nodes deployed prior to a disaster and central nodes - stored nearby emergency operations centers and airports and linked to the SENDROM databases – that query the sensor nodes following a disaster. In this case, rescue teams are assigned one mobile central node and guided to a region based on the data in the SENDROM database. This paper explains the architecture, the task and data dissemination schemes, and the end-to-end reliable event transfer scheme used in SENDROM. Then it evaluates the performance of our architecture based on statistical data from the 1999 Izmit (Turkey) earthquake.**

I. INTRODUCTION

Disasters, man-made or natural, can have significant consequences on both people and the environment. Major disasters include but are not limited to earthquakes, storms, floods, fires and terrorist attacks; such catastrophes can be of very large scale, potentially resulting in detrimental outcomes.

For example, in August 1999, a massive earthquake devastated northwestern Turkey; more than 22 thousand buildings collapsed and more than 50 thousand people were trapped beneath the rubble. That was a major catastrophe and disasters of this scale seldom occur. As is usually the case in major disasters, the international reaction to this epic was instantaneous as hundreds of rescue teams from around the world raced to Turkey; however, officials lacked a clear plan related to the distribution of rescue teams and the priority of digging regions, which slowed down the emergency response. In addition, it was very hard to mobilize emergency centers to manage the relief operations as their staff members were either victims of the catastrophe or survivors desperately trying to save their family members and friends. These problems are encountered in all types of disasters due to the resulting confusion and impaired decision-making leading to less than optimal emergency response.

Disasters continually prove that an effective system to detect living human beings can help to successfully manage a disaster relief operation potentially saving hundreds of lives. Traditional techniques –including, in our example, dogs, video cameras, sound sensors or simply shouting with the faint hope of hearing a reply from a trapped person – are used in disaster relief operations. However, such methods are clearly ineffective and time-consuming especially when you consider that the first 24 hours are very critical to rescue a wounded person. We started the sensor networks for disaster relief operations management (SENDROM) project with this motivation. Our novel SENDROM architecture uses state-of-the-art wireless sensor network [1] technologies to detect living humans and continuously report their status.

In SENDROM sensor nodes are deployed prior to a disaster and the nodes that collect sensed data from the sensor nodes, i.e., data collecting nodes (cnodes), are stored nearby emergency operations centers (EOC) and the harbor areas used by rescue teams at their arrivals, e.g., airports, seaports, customs, etc. Rescue teams are assigned a *cnode* before being deployed to the disaster area. *Cnodes* are linked to a central SENDROM database, which is updated continuously by EOCs and the other *cnodes* operated by the other

teams. Rescue teams can either make their decision by themselves or be directed to a certain region by an EOC based on the SENDROM database. *Cnodes* are also used to query sensor nodes that report the existence of human trapped under ruble. In this paper we explain our novel schemes related to the following issues of SENDROM architecture:

- New approaches and techniques to deploy the nodes of SENDROM.
- A new routing scheme that fits the characteristics of SENDROM.
- A new location estimation scheme for SENDROM.
- A new end-to-end reliable event transfer [8] scheme for SENDROM.

The remainder of this paper is organized as follows: Section II illustrates the SENDROM architecture. Section III introduces the task and data dissemination techniques designed for SENDROM. Section IV describes the end-to-end reliability scheme. Performance of the designed schemes is evaluated in Section V. Section VI depicts the experimental results. Section VII concludes this paper.

II. The Architecture of Sensor Networks for Disaster Relief Operations Management

In the SENDROM architecture, individuals deploy sensor nodes randomly at home, in their offices, and in other places where they might stay. The data collecting nodes of our architecture, called *cnodes*, are initially stored close to emergency operations centers (EOC) and airports and assigned to rescue teams before they are deployed in the disaster area. These *cnodes* are linked to a central SENDROM database, which is continuously updated by the EOCs and by the other *cnodes* themselves; this database is used to direct rescue teams to specific regions based on the data collected in the operations field. *Cnodes* are also equipped with directional antennae [10] allowing them to invoke sensor nodes in specific sections of the disaster area. When sensor nodes are queried by *cnodes*, they switch from an *idle* mode to an *active* mode and start reporting the sensed data to the *cnode* that invoked them.

Sensor nodes – deployed prior to a disaster – can be classified into two classes, namely *snodes* and *inodes*. *Snodes* sense and report any living human in their vicinity. *Snodes* are thus not associated with individuals implying that they do not require unique identifications. They may be of various types such as the following:

Standalone *snodes* (*ssnodes*): These are matchbox sized sensor nodes specifically designed for SENDROM, and located in places such as inside drawers, and on top of cabinets.

Embedded *snodes* (*esnodes*): *Snodes* can also be mounted on home appliances (e.g., washing machines) by the manufacturers. This approach has a number of advantages: First, for safety purposes, it is always recommended to be near a home appliance during an earthquake. Second, sensor nodes embedded in home appliances are constantly supplied with power reducing the maintenance overhead. Third, the probability p_b that a sensor node embedded into a home appliance is damaged when a building collapses is low compared to the p_b for standalone sensors.

Inodes, on the other hand, have unique identifications

associated with the owner individuals who carry them. Therefore they do not only report the status of a nearby human being, but they are also used to locate individuals. Similar to *snodes* there may be various types of *inodes* such as:

Standalone *inodes* (*sinodes*): *Sinodes* are also matchbox sized and carried in the pockets or bags of individuals.

Embedded *inodes* (*einodes*): These are embedded into the personal belongings of the individuals such as wristwatches. *Einodes* have a number of advantages comparing to *sinodes*. Since they are in contact with the skin of the individual, simple micro sensors attached them can detect the status of the individual by sensing the heartbeat, the breadth, etc. In addition, *einodes* do not need a special care for maintenance, and can be embedded with practical energy scavenging tools that use sources such as the movement of an arm or human pulse.

A. SENDROM Architecture Prior to a Disaster

As previously stated, the sensed data collected by the sensor nodes, i.e., *snodes* and *inodes*, is communicated to *cnodes* – in the form of mobile computers – which are deployed once an earthquake occurs. Prior to a disaster, *cnodes* are stored in *cnode* pools located near the emergency operations centers (EOCs) and airports that receive rescue teams coming

from abroad or other cities as shown in Figure 1.

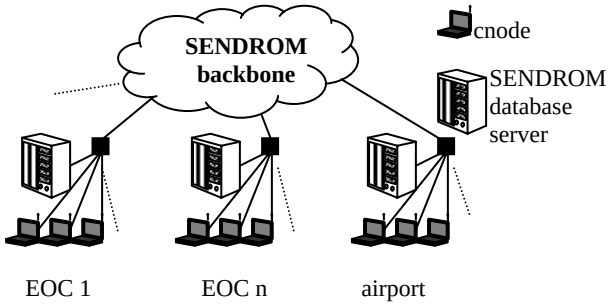


Fig. 1. SENDROM backbone prior to a disaster.

The *cnode* pools also contain a SENDROM database server (SDS). Data about the buildings, population, *inode* identifiers and their owners, and other logistical issues are stored in the SENDROM database and replicated and updated in all SDSs. Since our focus in this paper is not the SENDROM database, we do not give its design details.

Cnodes can query and download the required part of the SENDROM database. Moreover, the SENDROM database can be queried from remote sites via the
Internet allowing rescue teams to get priori information while on their way to the site of the disaster. An advantage of such an access method is that remote EOCs can more effectively manage a disaster relief operation.

B. SENDROM Architecture After a Disaster

Following a disaster, *cnodes* – which are nomadic nodes – are deployed and assigned to rescue teams and start communication with SDSs via mobile access points, manned or unmanned aerial vehicles and satellites as shown in Figure 2. The data entered by rescue teams and the sensed data collected from *snodes* and *inodes* are sent to SDSs.

As a result, a replicated SENDROM database is continuously updated by the data originating from field to effectively guide rescue teams and allow an efficient management of operations by the EOCs. Note that rescue teams and EOCs query the SENDROM database using *cnodes*.

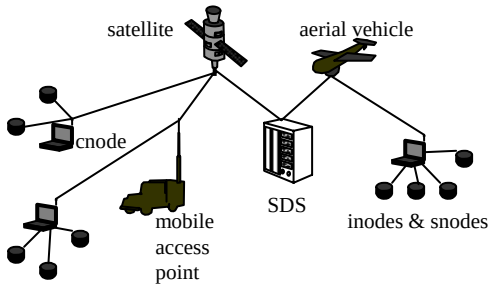


Fig. 2. SENDROM after a disaster.

III. TASK AND DATA DISSEMINATION IN SENDROM

Rescue teams use *cnodes* to invoke *snodes* and *inodes* in their field of operation. Following the reception of a task from a *cnode*, the sensor nodes switch from an *idle* mode to an *active* mode and start sensing and reporting the collected data. Three major issues are associated with this process, namely task dissemination, sensed data dissemination [2], [3], [4], [5], [6], [11], and location estimation [7], [9], [10].

A. Task Dissemination

Since *cnodes* are nomadic computers, and the distance to the sensor nodes is typically less than 100 meters, a *cnode* broadcasts sensing tasks at a single hop using a directional antenna. Figure 3 illustrates a task dissemination example where a *cnode* employs a directional antenna to broadcast a task in a specific region as shown by the pattern in the figure. Each direction is identified by a unique *task id* that is disseminated along with the sensing task. Task ids include a preamble that indicates a cnode id. Therefore, it is possible to distinguish the cnode that originates a task among multiple cnodes being operated in the same region. We will explain how this is used in the following section. The sensor nodes that receive a "*start to report*" task message start sensing and reporting the collected data: *inodes* always generate reports while s*nodes* generate reports only when they detect a living human being in their vicinity. In these reports, sensor nodes include the *task id* that they are responding to. In the case where sensor nodes receive multiple tasks, they can include all *task ids* in a single report. Hence the cnode can find out the region where the sensed data are originated.

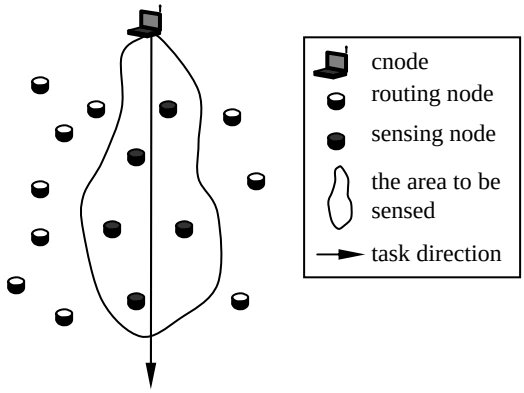


Fig. 3. Task dissemination in SENDROM.

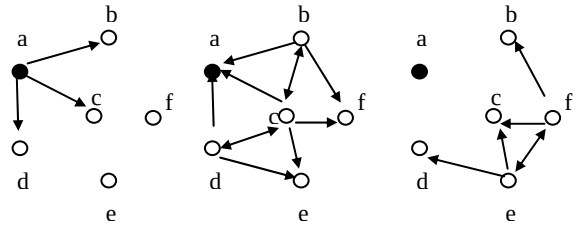
There are three major reasons to broadcast tasks by using a directional antenna:

- It allows estimating whereabouts of the sensor nodes responding to a sensing task [10]; the details of this scheme are explained in Subsection C.
- It minimizes the number of nodes unintentionally involved in a sensing task.
- Nodes are not involved in the task dissemination process.

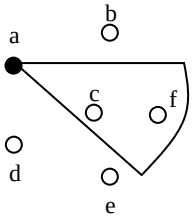
B. Data Dissemination

Most of the current routing schemes in sensor networks are based on an initial task flooding [3], [4], [5], and they are designed to route the sensed data to a single sink. However, the tasks are broadcast at a single hop by a directional antenna, and the sensed data related to a task should be forwarded to the cnode that broadcast the task in SENDROM. Please note that there may be multiple cnodes being operated close to each other, i.e., there may be multiple cnodes working on different buildings in the same street. The routing scheme in SENDROM should route sensed data related to a task to the correct cnode that disseminates the task. The new data dissemination protocol for SENDROM tackles this issue. This new scheme also selects power efficient routes through a controlled flooding process.

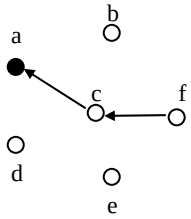
The data dissemination scheme in SENDROM has three phases as shown in Figure 4: route establishment, task dissemination and data dissemination. First the route establishment process establishes routes. Once routes are established, tasks can be assigned to the sensor nodes in any part of the sensor field by using directional antennae as explained in the previous section. After a task is disseminated, sensor nodes that have the sensed data related to the task send the data via the selected routes. In the example given in Figure 4, Node *a* is a cnode, and it disseminates a "route packet" to establish routes in the first phase. After route establishment process, tasks are disseminated by the cnode. In our example Node *f* has a sensed data for a disseminated task. Therefore, it sends the data through the selected route. Please note that tasks can be disseminated by locating the directional antennae any point around the sensor field. However, the sensed data is always routed back to the same point. If the cnode that carries out the route establishment process relocated to another point, than the route establishment process needs to be repeated. The details about the route establishment and data dissemination are explained below.



a. Route establishment.



b. Task dissemination.



c. Data dissemination

Fig. 4. Data dissemination in SENDROM.

type	RID	cnode id	TTL	echelon	total PA	min PA

a. Type=”route”: Route establishment packet format.

type	SID	RID	UID	TID	task report

b. Type=”data”: Data packet format.

Fig. 5. Packet formats in SENDROM.

Two types of packet formats are used in SENDROM. The format in Figure 5.a, i.e., packet type is “*route*”, is used to determine the best routes between a snode and the cnode. *RID* field in this packet format is for the identification of the sending node; every node that repeats a *task report* replaces the *RID* field with its own node id. Note that the ids of *snodes* need only be locally unique and need not be recorded in SENDROM database; this is not the case for *inodes* whose ids are globally unique and as they are associated with particular individuals in the SENDROM database. Each *cnode* has a unique *cnode id*. Multiple *cnodes* can operate in the same region. The *cnode id* indicates the *cnode* that starts the route establishment process. Since task ids also indicate *cnode* id that disseminates the task, the sensed data for the task can be forwarded to the correct cnode. TTL limits the number of hops from the cnode. The TTL value cannot be higher than the *echelon* of a sensor node that repeats a “*route*” packet. *Echelon* means the minimum number of hops required to reach a node from the source node. To further explain echelons, an illustrative topology is shown in Figure 6 with circles represent the coverage areas of nodes: since only nodes *A* and *B* are in the range of the cnode, they are the only nodes in *Echelon* 1. The nodes that are in the range of *A* and *B* represent *Echelon* 2.

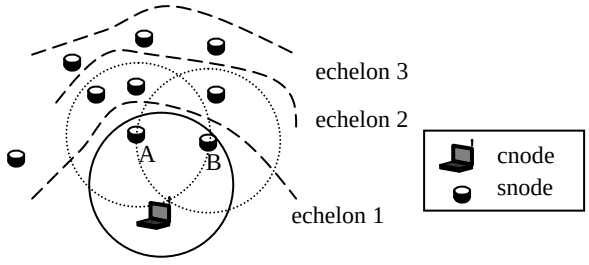


Fig. 6. Route establishment in SENDROM.

Total PA is the sum of the power available in every node along the route; the cnode initially sets this field to 0. *Min PA* is the power available in the node that has the minimum power along the route. *Echelon*, *total PA* and *min PA* fields are used to select more power efficient routes.

The format in Figure 5.b, i.e., packet type is “*data*”, is used to convey the data from a sensor node to the cnode. *SID* is the identification of the source node that generates the sensed data. *RID* field is the same as the *RID* field in the packets of “*route*” type. *UID* is the identification of the node that this data packet is forwarded to, i.e., the uplink node. Note that each relaying node unicasts a data packet to a specific node, i.e., it’s uplink node, in SENDROM. *TID* is a variable length field that includes the identification of the tasks that this report is generated for, the *cnode id* for the related task, and the received signal strength indicator (RSSI) – that reports the signal strength of the received “*start to report*” command for the related task; are also indicated in this field. *Task report* is the payload of the packet; in other words, it contains the sensed data related to the task.

```
while(1){
 waituntilreceive(packet);
 if(packet.type==”route”){
  t=currentTime;
  ts=random(t+tmin, t+tmin+θ. tmin);
  while(currenttime<ts) {
   if(newroutingdata(packet,neighbortable))
    insertintoneighbortable(packet);
   receive(packet);
  }//end while
  if(betterroute(neighbortable, parameters)){
   updateParameters(neighbor, parameters);
   if(packet.TTL>parameters.echelon)){
    prepareroutepacket(packet, parameters);
    broadcast(packet);
   }//end if packet.TTL
  }//end if better route
 }//endif packet.type==”route”
 else if(packet.type==”data”&&packet.UID==myid){
  while(not send(packet, uplinknode) {
   if(not updateparameters(neighbortable,parameters))
    startrecoveryprocess();
  }//end while
 }//end elseif
}//end while(1)
```

Algorithm 1. Data dissemination algorithm.

Algorithm 1 that runs on sensor nodes establishes

the routes and disseminates the sensed data. The *route establishment* process of our scheme is somewhat different from flooding as nodes repeat the selected incoming *route* packets only. While in the *route establishment* phase, a node that receives a *route* packet for the first time waits for the *transmission time* t_s; t_s is a random value between $t + t_{min}$ and $t + t_{min} + \theta\ t_{min}$ where t is the time that the first *route packet* is received, and t_{min} and θ are system parameters. t_{min} indicates the minimum time to wait after the *route packet* is received while θ is a random number between *0* and *1*. These system parameters must be carefully selected such that the nodes in one echelon wait long enough to receive most of the *route packets* from the previous echelon; the *transmission time* should also be selected appropriately, i.e., not too short to avoid many collisions or not too long to avoid high delays. Please note that nodes delay only "route" packets, i.e., during route establishment process, but not data packets. We examine route establishment delays in detail during our experiments.

TABLE I
AN EXAMPLE NEIGHBOR TABLE FOR NODE *C* IN FIGURE 4.

cnode id	UID	echelon	min PA	total PA
A	a	0	100	0
A	b	1	5	5
A	d	1	7	7
A	e	2	5	12
A	F	2	7	14
G	h	5	4	30

By the *transmission time* t_s, the other received *route packets* are also taken into account. Every *route packet* is recorded in the *neighbor table*. The *neighbor table* of a node is the list of nodes in its transmission range. An example neighbor table for Node *c* in the sensor network shown in Figure 4 is in Table 1. In this table cnode id and UID fields are the key fields, and a node can find out the best uplink node for a given TID based on this table because TID field in data packets includes a preamble for the id of the cnode that disseminated the related task.

The best uplink node id (UID) is determined by the *echelon*, *minimum PA*, and the total PA fields. Our algorithm first selects the routes that require the minimum hop to reach the cnode; if this number is the same for two or more routes, the route with the largest *minimum PA* is picked. If two or more routes have the same number of hops and the same *minimum PA*, then the route with the higher *total PA* is chosen. We prefer minimum hop routes to convey the sensed data because we assume that nodes transmit the same power level, and therefore the minimum hop routes are the most power efficient routes. When a power control scheme is employed, and the nodes transmit by using different signal strengths for different nodes, then a field that indicates the total power used to convey the sensed data may replace *echelon* field. This suffices to use the same scheme to find out the most power efficient routes for the networks where power control is used. If the *neighbor table* is full, and a new route is more power efficient than the least power efficient route in the *table*, the least power efficient route is replaced by the new route.

Based on the above criteria, we select one of the available routes as the uplink route, and repeat the *route packet* by the *transmission time* t_s. Before repeating the *route packet*, the node replaces the *RID* field by its own node id and checks whether its available power is lower than the *minimum PA* of the uplink route; if this is the case, the *minimum PA* in the *route packet* is replaced by the available power in the node. The *total PA* value in the packet is then updated accordingly.

When a node has sensed data to send, it looks up its neighbor table, and find out the best uplink node for the related cnode, then forwards the data packet to this uplink node. The node listens to the uplink node to ensure that it repeats the data packet, i.e., implicit acknowledgement. If the packet is not repeated, this indicates a transmission failure. A transmission failure invokes route recovery process.

For route recovery, first a HELLO packet is sent to the related uplink node to find out whether the node is alive or not. At the end of this, if it is detected that the uplink node is not alive anymore, then it is removed from the neighbor table, and the best uplink node is selected from the modified neighbor table. First a "route" packet is prepared according to the new uplink node and broadcasted, and then the data are sent to the new uplink node. This process is repeated until either the uplink node successfully relays the packet or the neighbor table does not have any neighbor that can relay the data for the related cnode.

If there is no node that can relay the data in the neighbor table, a HELLO packet is broadcasted for the

related *cnode*. Any node that can hear a HELLO packet replies it if it has an uplink node for the related cnode in its neighbor table. If the node does not get any reply for the HELLO packet broadcasted, this indicates that the node does not have any neighbor within its transmission range, and therefore it broadcast the sensed data with the maximum transmission power. The broadcast by the maximum transmission power is repeated until a blanket acknowledgment explained in Section IV is received. Please note that the loops are prevented because neighbor tables are modified every time that the uplink node of a node changes, and nodes always forward the sensed data to the lower echelons.

C. Location Estimation

In the TID field of the task report packets, the *task id* and *RSSI* for each task are indicated. Since a *task id* indicates a direction relative to the *cnode* and *RSSI* reports the signal strength of the received *"start to report"* command for the related task, this data gives the approximate polar coordinates of the reporting sensor node, i.e., the distance and the direction of the reporting node relative to the *cnode*. Although this location estimation scheme can intrinsically exhibit error, the result is accurate enough for rescue teams. In cases where more accurate location estimation is required, the same sensing task can be disseminated under different task ids with differences in the task dissemination directions by locating the transmitter at various points around the sensor field. Since a sensor node either generates a separate report for each task or includes the ids of all tasks that it has received in its *data packets*, the location of the node can be narrowed down to a small area representing the intersection of all reported tasks. This is shown in Figure 7 where a *cnode* uses a directional antenna to broadcast three different tasks in overlapping regions as indicated by the patterns; the location of a sensor node at the intersection of the three regions can be confined to the small shaded region based on the *task ids* – that indicate the direction relative to the *cnode* – and the *RSSIs* – that indicate the distance relative to the *cnode* – reported by that node, resulting in more accurate location estimation. This is similar to the beacon based multilateration [7], [9]. However, it is a simpler and more cost effective scheme because sensor nodes do not need any additional software or hardware components for node localization. Please note that we do not need to locate the nodes but whereabouts of alive humans reported by the nodes.

The location estimation scheme in SENDROM can also be backed up by audio- visual aids. For example, sensor nodes that detect a living human being nearby can blink and broadcast periodic audio signals, i.e., beeps, when they receive "start audio-visual aids" command from the cnode.

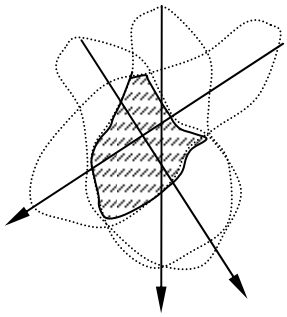


Fig. 7. Location estimation in SENDROM.

IV. End-to-end Reliability in SENDROM

End-to-end reliable transfer of the sensed data has utmost importance in SENDROM because nodes report the status of a living human being, and rescue teams make their decisions based on the received reports. In SENDROM the reliable transfer of the collected information, i.e., end-to-end reliable event transfer [8], is more important than the reliable transfer of each data packet. In other words, it is likely that more than one node will respond to the same task, and the reception of only one of the replies by the *cnode* stimulates a rescue operation in that task region; therefore, the successful transmission of only one of these reports suffices. Our task dissemination scheme perfectly fits the end-to-end reliable event transfer requirement of SENDROM.

We use our new *blanket acknowledgement* scheme where the *cnode* broadcasts only one acknowledgement for each task following the reception of the first report coming for that task. In the acknowledgement message, the *cnode* indicates the id of the acknowledged task; as a result, all the *snodes* in that task region reporting a presence of a living person can assume that they are acknowledged. This is not the case for *inodes*, which are directly associated with a particular individual and are thus responsible for indicating his status; therefore, the *cnode* acknowledges each *inode* separately by specifying

both *task id* and *inode id* in the acknowledgment message.

The acknowledgment message may also include other task parameters that could command sensor nodes in a certain task region to stop sensing or sense periodically for short time intervals.

The sensor nodes wait for an *acknowledgement time period* t_a – a fixed system parameter -after transmitting their report; if they do not receive an acknowledgment message for their task region during that period, they retransmit their report. The *acknowledgement time period* t_a can be as high as few minutes because of the nature of SENDROM. Therefore, we do not need a carefully designed timeout period selection algorithm for this parameter. Note that an implicit hop based acknowledgment scheme is also used in SENDROM as explained in the previous section.

V. Performance Evaluation

In this section the performance of SENDROM is evaluated by formulating the probability P_ω that a stimulus, i.e., a living human trapped under rubble, is within the sensing range of at least one sensor node, average hop distance m, and route establishment delay d. Since we are interested in disaster relief operations, collapsed buildings will be our new sensor fields.

Let's think about an apartment as the sensor field where the width is w and the length is l with q floors. When a building is collapsed, the height of each floor will change according to the construction techniques and materials. So the number of floors a sensor can sense or communicate, r_z, is considered as the sensing and communication range of sensors in z domain, and r_{xy} in meters is the sensing and communication range of the sensors in x-y domain.

In our model, it is assumed that sensor nodes and living human beings are deployed according to uniform distribution in each floor. On the other hand, it is easy to see that a number of sensor nodes will fail during the disaster, and when thought about earthquakes and collapsed buildings, the number of damaged nodes will increase in the lower floors. So while developing the mathematical model a function $\kappa(i)$ is used to represent the percentage of failed sensor nodes in floor i. The following function can be used for $\kappa(i)$:

$$\kappa(i) = \begin{cases} 2\sigma\dfrac{q-i}{q-1} & when\ \sigma\dfrac{q-i}{q-1} \le 0.5 \\ 1 & otherwise \end{cases} \quad (1)$$

where i is the floor that we calculate the number of failed nodes for, q is the number of floors in the building, and σ is the ratio between the total number of failed nodes and the total number of deployed nodes.

Since the probability P_ω that a stimulus is within the sensing range of at least one sensor node is determined by the sensing range of nodes, the distance between the stimulus and the nodes and the number of active nodes, P_ω can be formulated as:

$$P_\omega = 1-(1-P_\phi)^N \quad (2)$$

$$N = \delta A \sum_{i=j-r_z}^{j+r_z} 1-\kappa(i) \quad (3)$$

where N is the total number of active sensor nodes in the floors where the stimulus can be detected, δ is the node density (i.e., number of nodes per square meter), A is the base area of the building, j is the floor where the stimulus is in, r_z is the range of the sensors in z domain in the number of floors, and $\kappa(i)$ is the function that gives the percentage of failed sensor nodes in floor i.

$$P_\phi = P\left(\sqrt{(x_n-x_e)^2+(y_n-y_e)^2} < r_{xy}\right) \quad (4)$$

P_ϕ is the probability of sensing a stimulus in x-y domain, where (x_n,y_n) is the coordinate of the sensor node, (x_e,y_e) is the coordinate of the living person to be detected, r_{xy} is the sensing radius of the sensor node in x-y domain.

We find P_ϕ in two steps. First, we compute the probability density functions (pdf) of $X = X_n - X_e$ and $Y = Y_n - Y_e$; then we compute the pdf of $Z = \sqrt{(X^2+Y^2)}$. Since $X_n(0,w)$, $X_e(0,w)$, $Y_n(0,l)$ and $Y_e(0,l)$ are independent random variables, first by substituting $X_n = X + X_e$ and $Y_n = Y + Y_e$, we find

$$f_X(x)=\begin{cases}\int_0^{x+w}\frac{1}{w^2}d_{X_e}=\frac{w+x}{w^2}, & -w\le x\le 0\\ \int_x^{w}\frac{1}{w^2}d_{X_e}=\frac{w-x}{w^2}, & 0\le x\le w\end{cases} \quad (5)$$

$$f_Y(y)=\begin{cases}\int_0^{y+l}\frac{1}{l^2}d_{Y_e}=\frac{l+y}{l^2}, & -l\le y\le 0\\ \int_y^{l}\frac{1}{l^2}d_{Y_e}=\frac{l-y}{l^2}, & 0\le y\le l\end{cases} \quad (6)$$

At the second step to formulate the pdf of *Z*, an auxiliary random variable *T*, as T=*X*, is introduced. This will enable us to use the general formula of finding f_{ZT} from two functions of two random variables with n real roots, given below

$$f_{ZT}(z,t)=\sum_{i=1}^{n} f_{XY}(x_i,y_i)\left|\tilde{J}_i\right| \quad (7)$$

The equations

$$Z-\sqrt{X^2+Y^2}=0 \quad (8)$$
$$T-X=0$$

have two real roots, for $|t|<z$, namely

$$x_1=t \qquad x_2=t$$
$$y_1=\sqrt{z^2-t^2} \qquad y_2=-\sqrt{z^2-t^2} \quad (9)$$

At both roots, $\left|\tilde{J}\right|$ has the same value:

$$\left|\tilde{J}_1\right|=\left|\tilde{J}_2\right|=\frac{z}{\sqrt{z^2-t^2}} \quad (10)$$

Since X and Y are independent random variables, a direct application of Equation 7 yields

$$f_{ZT}(z,t)=\frac{z}{\sqrt{z^2-t^2}}\begin{cases}(\frac{2}{l^2})(\frac{1}{w}+\frac{t}{w^2}) & ,-w\le x\le 0,-l\le y\le l\\ (\frac{2}{l^2})(\frac{1}{w}-\frac{t}{w^2}) & ,0\le x\le w,-l\le y\le l\end{cases} \quad (11)$$

where $z>0, |t|<z$ conditions must be satisfied.

$$f_Z(z)=\int_{-\infty}^{\infty} f_{ZT}(z,t)dt$$

$$f(z)=\begin{cases}\frac{2z}{lw}\int_0^{z}\frac{dt}{\sqrt{z^2-t^2}}u(z)+\frac{2z}{lw^2}\int_0^{z}\frac{tdt}{\sqrt{z^2-t^2}}u(z), & -w\le x\le 0,-l\le y\le l\\ \frac{2z}{lw}\int_0^{z}\frac{dt}{\sqrt{z^2-t^2}}u(z)-\frac{2z}{lw^2}\int_0^{z}\frac{tdt}{\sqrt{z^2-t^2}}u(z), & -w\le x\le 0,-l\le y\le l\end{cases} \quad (12)$$

If we substitute v as $v=z^2-t^2$, so dv becomes $dv=-2wdw$ and solve the integrals, since $z\ge 0$ *and* $|w|<z$ we get;

$$f_Z(z)=\begin{cases}\frac{2z}{lw}(\frac{\pi}{2}+\frac{z}{w})u(z), & -w\le x\le 0,-l\le y\le l\\ \frac{2z}{lw}(\frac{\pi}{2}-\frac{z}{w})u(z), & -w\le x\le 0,-l\le y\le l\end{cases} \quad (13)$$

P_ϕ becomes:

$$P_\phi=F_Z(r_{xy})=\int_0^{r_{xy}} f_Z(z)dz$$

$$P_\phi=\begin{cases}\frac{\pi(r_{xy})^2}{2lw}+\frac{2(r_{xy})^3}{3lw^2}, & -w\le x\le 0,-l\le y\le l\\ \frac{\pi(r_{xy})^2}{2lw}-\frac{2(r_{xy})^3}{3lw^2}, & 0\le x\le w,-l\le y\le l\end{cases} \quad (14)$$

We can use Equation 13 also to find out the average hop distance m:

$$m=\int_0^{\varphi} z\, f_z(z)\,dz \quad (15)$$

where φ is the maximum distance that can be between two sensor nodes under a ruble, i.e., a collapsed building, and given by

$$\varphi=\sqrt{l^2+w^2+(qh)^2} \quad (16)$$

where l is the length, w is the width, and q is the number of floors of the collapsed building. The last variable h is the average height of the floors after the building is collapsed.

The route establishment delay d is

$$d=\sum_{i=1}^{\lceil \varphi/m \rceil} d_i \quad (17)$$

where d_i is a random variable distributed according to Uniform distribution .

$$d_i \approx U(t_{\min},2t_{\min}) \quad (18)$$

VI. Experimental Results

We evaluate the performance of our architecture based on statistical data from the 1999 Izmit (Turkey) earthquake. This data include the effects of the earthquake in terms of the ensuing state of the

buildings, the number of people trapped beneath the buildings, etc. For example, the first two experiments that follow are based on such information as the resulting height of a building when it totally collapses. This type of statistical data proved very valuable in determining some important parameters such as the expected hop distance and expected number of hops between a sensor node and a *cnode*.

We first examine the average hop distance for varying node density as a function of the percentage, σ, of failed nodes, i.e., the nodes damaged during the earthquake, and the number of floors in a collapsed building. The results of these experiments for *node density* δ=*0.15* node/m^2 are shown in Figure 8. When 2 nodes are deployed in each room of a 5 floor building with 4 apartments per floor, the average hop distance is 3.25 meters assuming that the building totally collapses and 50% of sensors are destroyed during the earthquake.

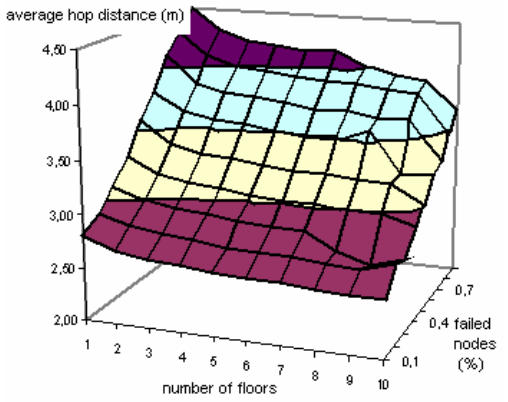


Fig. 8. Average hop distance for *node density* δ=*0,15* node/m^2.

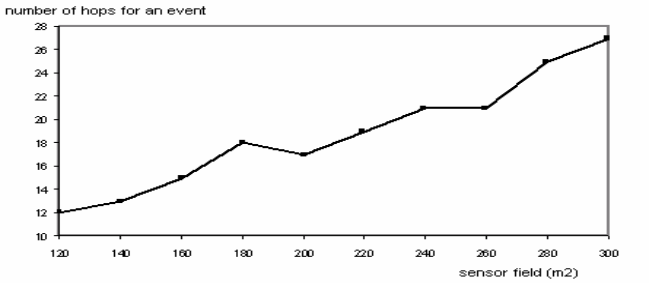


Fig.9. Average number of hops required to transmit a single detection.

Figure 9 shows the average number of hops required to convey a sensed data packet from a sensor node to a *cnode* for varying sensor field dimensions assuming an average hop distance is of 3 meters. For example, for a sensor field of 200 m^2, the average number of hops between a sensor node and *cnode* is 17.

In Figures 10 and 11, the number of active nodes and the probability that a stimulus is within the sensing range of at least one sensor node are depicted. In these experiments 200 sensor nodes are uniformly deployed to a building of 5 floors and the number of active nodes are determined by Equation 3, for σ is equal to 0.4. When we assume that sensor nodes one floor below or higher can also detect a stimulus, the probability that the stimulus has at least one sensor node in less than 5 meters from it is almost 1.

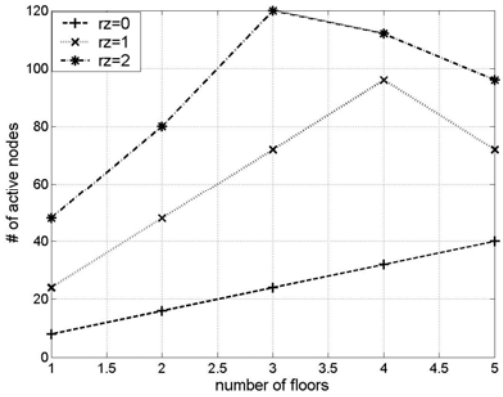


Fig.10. Average number of active nodes that can detect a stimulus for the percentage of nodes failed during an earthquake σ=0.4.

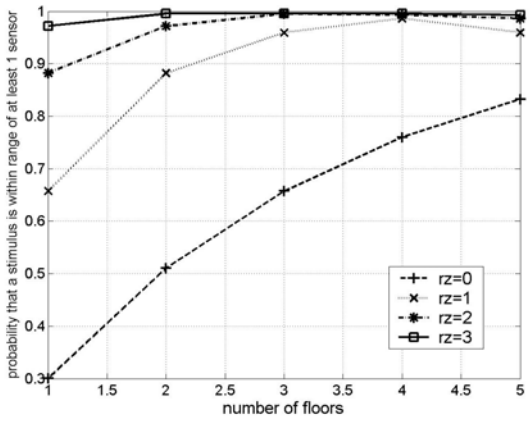


Fig.11. The probability that a stimulus is within the sensing range of at least one sensor node for the percentage of nodes failed during an earthquake σ=0.4.

We also evaluate the performance of the routing and end-to-end reliability schemes of SENDROM via simulation. In our simulations, a fixed workload based on the traffic for one stimulus, i.e., an alive human trapped under ruble, is generated. The stimulus is deployed at a point randomly selected according to

Uniform distribution in the sensor field and the *cnode* is located near one of the corners. Other 50 nodes are uniformly distributed to an aggregate floor area of 160x160 meters. In our experiments, congestion and physical factors effecting the radio transmission are not modeled for both SENDROM and the compared algorithms. The data transmission rate is set to 1.6 Mbps and *ns-2* radio energy model is used as in [5] where the power dissipation is 35 mW in idle mode, 395 mW in receive mode and 660mW in transmit mode.

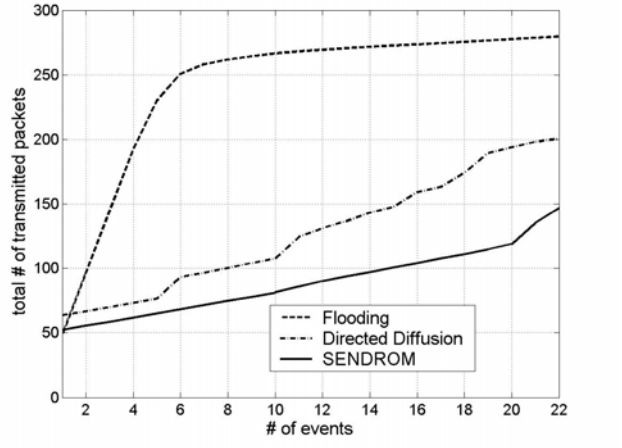


Fig.12. Total number of packet transmissions.

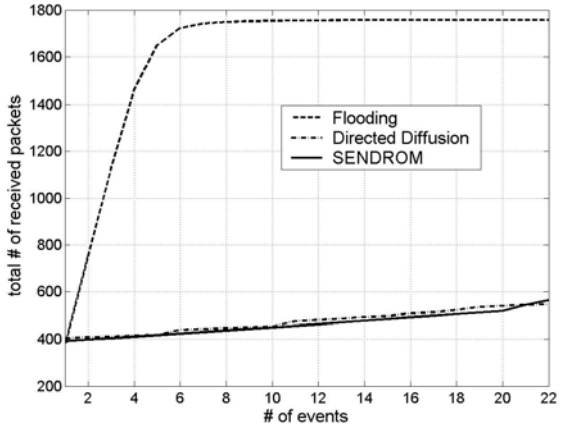


Fig.13. Total number of packet receptions.

In Figures 12 and 13 the total number of packets transmitted and received by the sensor nodes are shown respectively. Our SENDROM routing scheme outperforms flooding and directed diffusion. When number of events reported is less than 2, flooding performs better than directed diffusion and SENDROM because of the route establishment costs of these schemes. The performance gains of SENDROM gets higher as the number of events reported increases comparing to flooding. After a point the difference in the routing costs of flooding and SENDROM decreases. However this is not because the performance of flooding gets better after a point. The nodes deplete their energy sooner in flooding. Since the number of nodes involved in routing decreases due to failed nodes, the number of packets transmitted and received starts decreasing sooner in flooding comparing to the other techniques. In these figures, it is also shown that the difference between the number of received and transmitted packets is much higher in SENDROM and directed diffusion because nodes do not relay every packet in SENDROM and directed diffusion. Figures 12 and 13 also proves that SENDROM is more selective in repeating the packets received.

In Figure 14 the energy available in the network, and in Figure 15 the number of nodes failed due to energy depletion are shown. These figures prove that SENDROM also increase the lifetime of the network comparing to the other two techniques. The extended lifetime also indicates higher number of events that can be reported by the network as shown in Figure 16.

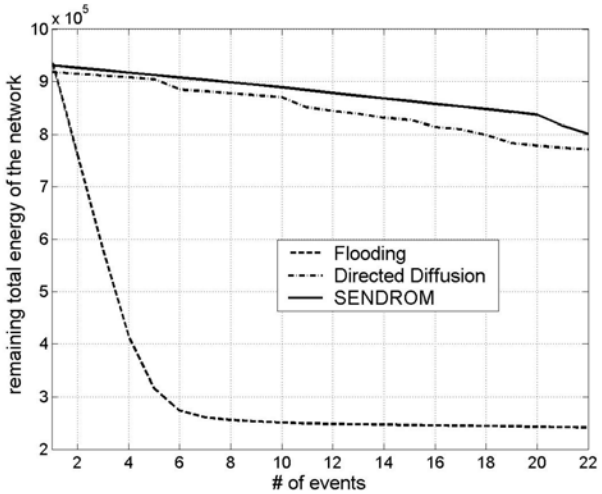


Fig.14. The energy available in the network.

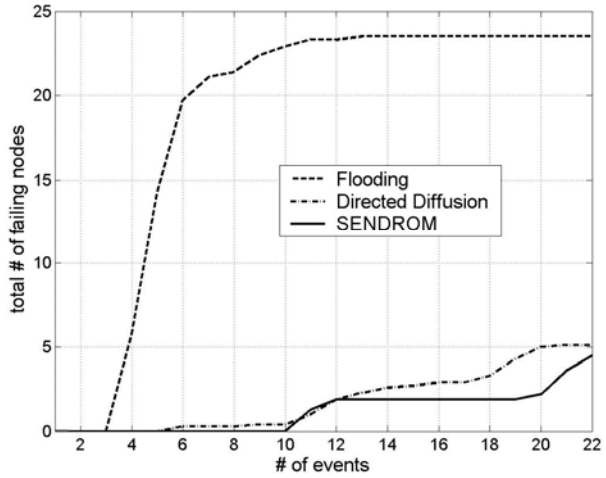


Fig.15. The number of nodes failed due to energy depletion.

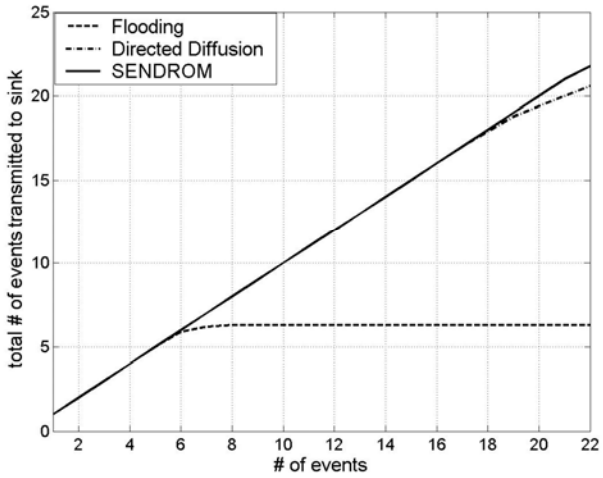


Fig.16. The number of events reported by the network.

In Figures 17 and 18, the impact of minimum delay t_{min}, i.e., the minimum time that a node waits before relaying a "*route*" packet, to the route establishment process is shown. In Figure 17, the route establishment delay d is depicted for varying t_{min}. It shows that routes are established in less than 2 seconds even when t_{min} is 100 msec. Few minutes delay can be tolerable for SENDROM. Therefore, this delay in route establishment process is negligible. We also show how many "*route*" packets are transmitted, i.e., the total number of "route" packets transmitted by sensor nodes during route establishment process, after a route establishment process is started by a *cnode* in Figure 18. The average number of "*route*" packets transmitted by a single node is as low as 1.18 for t_{min} is 100 msec.

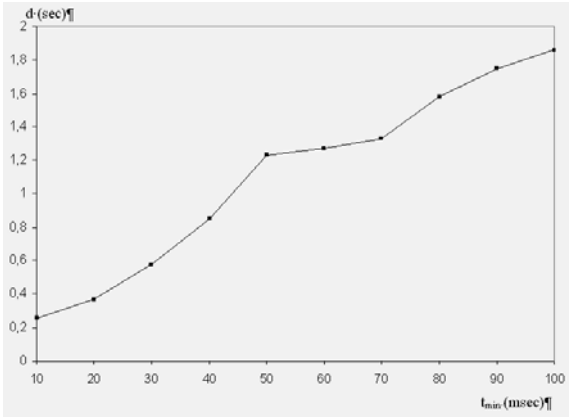


Fig. 17. Route establishment process delay.

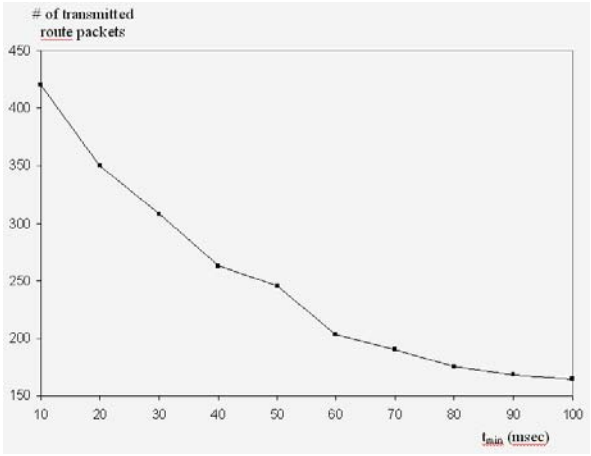


Fig.18. Number of transmitted "route" packets.

In Figure 19, the gain in end-to-end event transfer reliability of the blanket acknowledgment scheme is evaluated for two cases: 1) the hop-by-hop implicit acknowledgement is enabled and 2) the hop-by-hop implicit acknowledgment is disabled. We first run the simulation with the blanket acknowledgment disabled, and then run the same simulation after enabling it. We count the number of successfully transferred events in both runs and divide the results obtained from the first run by those obtained from the second run. The plots in Figure 19 show these results. We observe that, especially at low node density with the implicit acknowledgement disabled, the performance gains of the blanket acknowledgement scheme are very high. For instance, when the implicit acknowledgement is disabled and the node density is 2, the end-to-end reliability is 5 times higher when using the blanket acknowledgement scheme.

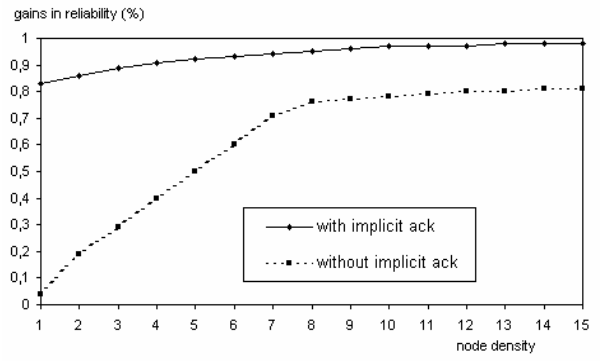


Fig.19. Gains in reliability of the blanket acknowledgement scheme.

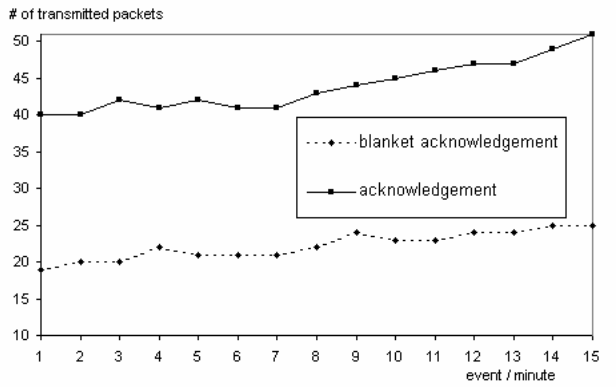


Fig.20. Performance gains of the blanket acknowledgement scheme.

Figure 20 shows the average number of packets relayed by sensor nodes to transfer a sensed data packet. The "*acknowledgement*" plot shows the performance of the conventional acknowledgement scheme, where acknowledgement packets are propagated through the sensor network back to the source node. We observe that the cost, i.e., the number of packet transmissions, of this scheme is more than 2 times higher than the *blanket acknowledgement*.

VII. CONCLUSION

SENDROM is developed for the effective management of rescue operations after large scale disasters. In SENDROM, sensor nodes – snodes and inodes - are randomly deployed before a disaster occurs. Central nodes, i.e., cnodes, are stored close to EOCs and airports; they are assigned to rescue teams once the catastrophe takes place. Rescue teams use these cnodes both to detect the presence of a living person in the disaster area – by querying the sensor nodes – and to communicate with a central SENDROM database. In this paper, we examine novel task and data dissemination, location estimation and end-to-end reliable event transfer schemes developed specifically for SENDROM. We also evaluate the performance of SENDROM through simulation based on the statistical data from the 1999 Izmit earthquake.

REFERENCES

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam and E. Cayirci, "*Wireless Sensor Networks: A Survey*," Computer Networks Journal (Elsevier), Vol. 38, No. 4, pp. 393-422, 2002.

[2] A. Boulis, S. Ganeriwal, and M.B.Srivastava, "*Aggregation in Sensor Networks: An Energy Accuracy Tradeoff,*" to appear in 1st IEEE International Workshop on Sensor Network Protocols and Applications, 2003.

[3] W. R. Heinzelman, J. Kulik, and H. Balakrishnan, "*Adaptive Protocols for Information Dissemination in Wireless Sensor Networks*," ACM Mobicom'99, pp. 174-185, Seattle, Washington, 1999.

[4] L. Li, and J. Y. Halpern, "*Minimum-Energy MobileWireless Networks Revisited*," IEEE International Conference on Communications '01, Helsinki, Finland, 2001.

[5] C. Intanagonwiwat, R. Govindan, and D. Estrin, "*Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks*," ACM Mobicom'00, pp. 56-67, Boston, MA, 2000.

[6] S. Madden, R.S. Szewczyk, M.J. Franklin, and D. Culler "*Supporting Aggregate Querries over Ad-hoc Wireless Sensor Networks,*" 4th IEEE Workshop on Mobile Computing Systems and Applications, 2002.

[7] A.Nasipuri and K.Li, "*A Directionality Based Location Discovery Scheme for Wireless Sensor Networks*," Proceedings of the First ACM Workshop on Wireless Sensor Networks and Applications, pp. 105-111, Atlanta, 2002.

[8] Y. Sankarasubramaniam and I. F. Akyildiz, "*ESRT: Event-to-Sink Reliable Transport in Wireless Sensor Networks*," to appear in MobiHoc'2003, 2003.

[9] A.Savvides, H.Park and M.B.Srivastava, "*The Bits and Flops of the N-hop Multilateration Primitive for Node Localization Problems*," Proceedings of the First ACM Workshop on Wireless Sensor Networks and Applications, pp. 112-121, 2002.

[10] A. Erdogan, E. Cayirci and V. Coskun, "*Sectoral Sweepers for Task Dissemination and Location Estimation in AdHoc Sensor Networks*", Proceedings of MilCom'2003, Boston, 2003.

[11] E. Cayirci, "Data Aggregation and Dilution by Modulus Addressing in Wireless Sensor Networks", IEEE Communications Letters, August, 2003.