SenSearch: GPS and Witness Assisted Tracking for Delay Tolerant Sensor Networks

Jyh-How Huang*, Lun Jiang†, Ankur Kamthe†, John Ledbetter*,
Shivakant Mishra*, Alberto Cerpa† and Richard Han*
* Department of Computer Science, College of Engineering and Applied Science,
University of Colorado at Boulder; Boulder, CO 80309
jyh.huang@colorado.edu, john.ledbetter@colorado.edu, shivakaht.mishra@colorado.edu, rick.han@colorado.edu
† Computer Science and Engineering, School of Engineering,
University of California, Merced; Merced, CA 95344
akamthe@ucmerced.edu, ljiang2@ucmerced.edu, acerpa@ucmerced.edu

***Abstract*— Mobile wireless sensor networks have to be robust against the limitations of the underlying platform. While lightweight form factor makes them an attractive choice for monitoring applications, they are hampered by energy and communication constraints, such as small lifetimes and limited bandwidth. Moreover, they have to operate under stringent conditions like occasional network connectivity and limited storage capacity. Under these constraints, the design of mobile WSNs has to account for the resource limited nature of the underlying platform.**

In this paper, we describe the implementation and performance evaluation of SenSearch, an outdoors GPS assisted personnel tracking system using mica motes. SenSearch is a mobile wireless ad-hoc network comprising of sensor nodes that are worn by users, which store and forward information about the locations of other nodes in environments with lack of communication infrastructure. A key feature of SenSearch is that it does not require a continuously connected network for its operation. It is designed for a delay tolerant network that provides only occasional connectivity between nodes. It uses the distributed storage available thru multiple nodes and the mobility provided by users to propagate the history of nodes' GPS locations to the processing center. Using this information, it is possible to drastically reduce the search space while estimating the possible location of a missing node. The main contributions of this paper include in-depth analytical, simulation and experimental results of the performance of the system as a function of memory usage, power consumption, localization error and data delivery rate.

I. Introduction

Wireless sensor networks (WSNs) have gathered a lot of attention from the research community in the past decade. They combine distributed sensing, computation, storage, and communication capabilities under the same platform. Much of the attention during this incubation period was focused on developing infrastructure for the platform. On the system hardware side, the field has experienced multiple revisions of the Berkeley [1] and Telos motes [2]. The sensing capabilities of the platform have been supplanted through the development of specialized sensor boards for measuring light, temperature, pressure, acceleration/seismic, acoustic, magnetic, and other types of data. Programming platforms like TinyOS + nesC [3], [4], MANTIS OS [5], EmStar [6] have been designed to harness the unique sensing capabilities of the underlying platform.

These days, most hardware in sensor networks is half the size of your typical cellphone. The lightweight form factor increases the portability while imposing energy, communication and memory constraints on the hardware and software. It is equivalent to having 1980s computer hardware (8Mhz CPU, 4MB memory, 200Kbps modems) packed inside a matchbox with very limited battery-life. Also, they have to often operate in intermittently connected networks, shifting the onus of the resource limited nature of the underlying platform to the system designer. In spite of all these hindrances, WSNs have been deployed in wide variety of applications such as habitat monitoring [7], detection and surveillance [8], smart spaces [9] and process monitoring applications, due to the flexible nature of the underlying platform.

With Moore's law leading to lower cost of hardware, the field has now grabbed the attention of the every-day user. This has lead to the emergence and growth of location-centric services based on the WSN platform. One of the most important location-aware service is personnel tracking, i.e., tracking the motion of people (e.g.: patients in hospitals), animals (e.g.: study of migration patterns), etc. These services are to be delivered in a timely manner and require the presence of a constantly connected network via wired (Ethernet) or wireless (satellite) medium. This requirement severely cripples the applicability of the platform for widespread deployment, in particular in environments with complete lack of communication infrastructure.

To motivate our research, consider a plausible scenario in which a person (who also happens to be a CS major) is hiking along a popular but difficult trails in the Yosemite National Park. The hiker encounters several people at the beginning of the trail who have just completed the hike. This person chats with some of them, asking about the conditions of the trail as it had rained last night. In spite of bad reports, the hiker in a rush of adventure starts his trek. Initially, he/she is accompanied by several other hikers who, later on, follow other trails forking from the same path. The hiker encounters a few people going in the opposite direction. The last of these returning hikers advices our hiker to turn back as the

trail becomes progressively slippery and that the hiker would be wise to return. Our non-chalant hiker continues. After a passage of time, our hiker's journey come to an abrupt halt, when he/she looses his footing on a slippery section, falls and breaks his leg. As it is quite late in the day, our injured hiker decides to rest in the shelter of some trees in the hope that someone passes by before he attempts the return journey on a broken foot.

While he waits, he starts thinking about his situation. He wonders whether a system could be in place to help stranded hikers like himself. Such a system could possibly utilize wireless sensors with some kind of location inferring capability, possibly GPS. Since there is no wireless network or established communication infrastructure in the park, how would the information be passed on? These sensors could use their built in radios to transfer his location information via the hikers he encounters to the park rangers. But then such a system would have to last for hours in spite of having the radios and GPS working together. This would necessitate an energy conservation policy for the radio and GPS. Lastly, what would they call such a system, SenSearch?

In this paper, we describe the design and implementation of SenSearch, a GPS and witness-assisted tracking solution for delay tolerant sensor networks in outdoors environments. In SenSearch, we propose the use of GPS modules to enable a sensor unit to infer its position information, independently. It uses the concept of witnesses to convey an entities movement and location information to the outside world. This helps overcome the constraint of having a constantly connected network to convey information. Different entities exchange their location data when they encounter each other along the way. This information is subsequently routed to the base station (central repository) through a series of data transfers between nodes. This data which consists of the history of past locations of a node can be used to estimate the location of a missing entity. The use of the GPS data drastically reduces the search space, by increasing the accuracy of the system.

However, the use of GPS along with the transmission/reception of data in an energy-constrained system will reduce the lifetime of the system. In order to save energy, we propose a duty cycling scheme for both the GPS unit and the radio. The duty cycling may lead to a decrease in the localization accuracy of the system. This along with other trade offs is discussed in detail in the performance evaluation section of this paper.

The main contributions of our paper are as follows:

- The design and implementation of an outdoor personnel tracking system based on GPS and the notion of witnesses, which facilitates the localization of stranded personnel and assists Search and Rescue operations in environments with lack of communication infrastructure.
- An extensive performance evaluation of the system using analytical, simulation and experimental results, including a comprehensive analysis of the different parameters that affect SenSearch's performance.

The rest of this paper is organized as follows: in Section II, we compare SenSearch with other approaches for the tracking of mobile entities. We also discuss the issue of routing messages in SenSearch in the context of Delay Tolerant Networks (DTNs). In Section III, we describe the hardware and software architecture of SenSearch. In Section IV, we provide analytical, simulation and experimental results of the performance of the system with respect to localization error, power consumption, memory usage and data delivery rate as a function of many different parameters in the system. Next, in Section V we discuss the different lessons learned based on the results of the previous section. Finally, in Section VI we discuss future work and conclude.

II. Related Work

A majority of networks [8], [10], [11] used for tracking movements of mobile entities comprise of hundreds of small, densely distributed wireless sensor nodes deployed in the field. They involve simple algorithms that detect and track a moving target, and alert sensor nodes along the projected path of the target. This approach is useful for tracking movement in well-designed areas with expected traffic patterns. It is not suited for tracking the movement of entities with random mobility patterns on ill-defined paths as it would require a prohibitively large number of nodes to cover all possible locations. A distributed approach wherein individual nodes carry devices that have localization capability and can communicate this information through relays to the base-station, is more flexible and can scale to different environments. In this section, we compare SenSearch with other proposed distributed approaches for tracking of mobile entities.

SenSearch is similar in intent to the ZebraNet and Electronic Shepherd project, which are designed for the tracking of mobile entities. The ZebraNet Project [12], [13] at Princeton University is a system designed for tracking the mobility patterns of zebras. The zebras are fitted with collars which have a GPS unit, wireless transceivers, CPU, flash memory for data storage and rechargeable solar batteries. It utilizes a history-based and flooding protocols for propagating their location information to the base station. In the history-based protocol, the nodes maintain a hierarchy information with respect to their proximity to the base station. A node will transfer its information to another only if the other node is closer to the base station as indicated by the hierarchy of the node. Here the history based protocol requires that the nodes be fairly static, so that a node can acquire the hierarchy information, compare it and decide whether it wants to transfer data, only after which the actual data transfer occurs. In the flooding protocol, a node dumps its information to its neighbors hoping that it is relayed eventually to the base station. The flooding protocols increases the memory requirements of neighboring nodes as it cannot control the amount of data transferred to its neighbors.

The SenSearch architecture differs from ZebraNet as it utilizes motes with small memory and limited battery life. It would be impractical to have solar cells to power such a system, because carrying anything larger than a pager on

people would only add to their burdens. To improve memory usage, we recycle the available memory by maintaining only a certain number of records per node. This keeps the power consumption low as the size of the databases being transferred between nodes remains small. (While ZebraNet is designed for a big group of sensors moving together in the same direction with same speed, SenSearch is designed to be used in the scenario where sensors move in different directions at different speeds.)

Electronic Shepherd [14] is a low-power, low-bandwidth application for tracking the movement of animals with flock-behavior. Here, a subset of sheep in a flock were equipped with UHF radio tags. They communicate with a mobile access point, basically, a sheep with a specialized collar that has GPS, GPRS and storage equipment. It stores information from the associated radio tags and communicates this along with location information from GPS to the base station via the GPRS modems. This system along with ZebraNet relies on the group-based movement of the tracked entities. Electronic Shepherd widely varies from SenSearch, which tracks the locations of individual entities and uses a peer-to-peer medium to communicate information to the base station. Also, SenSearch is applicable for tracking in locations with no GPRS/GSM or 802.11 coverage like a hiking scenario in National Parks.

SenSearch is based on CenWits [15], a Connection-less Sensor-Based Tracking System Using Witnesses. CenWits is comprised of mobile sensors that are worn by entities (people, wild animals, or in-animate objects), each of which receives its location information periodically from location points and passes it to other nodes during subsequent encounters. This information is then dumped at access-points distributed at various locations. Since, it is not practical to deploy location and access points for information storage and recovery in hostile environments, each node has to maintain its own database and location information. This makes it imperative to have a scheme for efficient utilization of memory and power consumption. SenSearch utilizes a GPS module attached to the sensor to get up-to-date location information about a node. To improve power consumption, the GPS and the radio are power-cycled. Memory usage is reduced by storing only the last n recent entries for any particular node in a node's database.

The Walking GPS [16] is a localization solution for manual deployments of wireless sensor networks. It is carried out in two phases: During the first phase when the deployment of the sensor nodes takes place, the carrier with a GPS-enabled mote beacons its location information from which the sensor nodes infer their position. In the second phase, a sensor node that does not have a location, asks its neighbors for their location information and triangulates its position. The results showed that 100% of the deployed motes localized with average localization errors within 1 to 2 meters, due mainly to the limitations of the GPS devices. Thus, GPS information is one of the most accurate means for localization of nodes in a sensor network. SenSeach leverages this feature for localization of missing nodes from the information collected from the databases of its peers.

The FireBug [17] system is a network of GPS-enabled, wireless thermal sensors. It is a self-configuring system of static nodes which uses GPS for localization. The network is comprised of "hub" motes, which act as intermediate base-station for receiving sample data from nodes and sending command to them. Routing is accomplished using the mh6 protocol. It is used for collecting real time data (temperature, pressure and humidity) from forest wildfires. FireBug differs from SenSearch as it is a manual deployment of static nodes that use GPS only for localization, tracking the path of the wildfires is done from the information collected from these nodes.

Finally, routing can be a big problem in Delay Tolerant Networks with node mobility. Lack of knowledge about mobility patterns and the intermittent-connection nature of the network could lead to incorrect routing decisions. Replicated data transmission is one way of ensuring that the collected data is retrieved but comes at the cost of long delays and transmission overhead. Erasure coding has been proposed to reduce the transmission overhead by compressing long messages [18]. However, the mechanism to spread the erasure-coded blocks is quite complex [19] in the presence of unpredictable node mobility. SenSearch involves simple replication and memory recycling along with the best effort model of forwarding messages as in IP. It is based on the notion of "shared communities" [19] for forwarding data which is applicable for scenarios for which SenSearch was envisioned.

In [20], a generic architecture for enabling network connectivity in rural areas was proposed where the cost of adding traditional infrastructure like Very Small Aperture terminals (VSAT) or long-distance wireless links. It involves the use of vehicles (bus, car) as "mechanical backhaul" devices to carry data to and from remote locations (kiosks) and internet gateways. The vehicles opportunistically communicate with the kiosks and internet gateways in its path over WiFi, transferring data when in range. The Opportunistic Communication Management Protocol (OCMP) stack and Delay Tolerant Network (DTN) router components enable routing, security and addressing. The system supports delay-tolerant applications like email, FTP, HTTP-get, blog updates and e-governance applications like mutual fund registration, birth and marriage certificates, bill payments, etc.

SenSearch, on the other hand, is a delay tolerant localization-centric application. It falls under Pocket Switched Networks (PSN) in which opportunistic data transfers between devices in close proximity are used to route packets. Also, SenSearch differs vastly from [20] due to its resource- constrained hardware and custom (MantisOS) software architecture.

In [21], an analysis of the mobility/contact patterns of students in a campus environment using information from class schedules, is presented. This is in contrast to traditional model-based or measurement-based approaches. They use DTN characteristics like inter-arrival time and distance between students and infer their impact on data dissemination across the entire student population. They say that the data from almost the

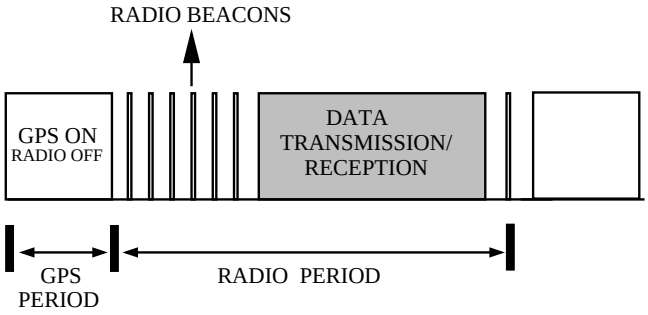


Fig. 1. Nodes alternate between the GPS and Radio periods. When in the GPS periods, nodes have their radios off and they acquire GPS coordinates. In the Radio period, nodes send beacons trying to contact other entities and may exchange their databases if they encounter another node.

entire student population can be aggregated in a small number of nodes when students meet at hubs, which are places like classes, library, and dining halls. This result is relevant to SenSearch scenarios wherein different entities congregate at common spots while hiking/camping/trail-running. They can exchange their location data which will be routed to the base station and can be used to track them in the event that one of the entities is lost in a hostile environment. From the above results, only a small subset of the nodes are required to transfer data to the base station for the system to localize missing nodes.

III. System Description

SenSearch uses Berkeley MICAz motes equipped with an RF transmitter and a GPS receiver to track locations of entities (example: people, animals, etc.) in natural or urban environments in the absence of a communication infrastructure. The key idea here is when two entities come within radio range of each other, they become each others witnesses. Later on, if an entity wearing one of these sensors is lost, the other entity can convey information regarding the history and last known (witnessed) location of the lost entity.

Each sensor unit has a unique ID and keeps record of its locations by periodically using the GPS receiver and saving the information in an internal database. It also emits periodic radio beacons to detect the presence of other units. After the sensor unit is turned on, it tries to acquire the GPS coordinates. The time for the GPS to acquire a set of coordinates can vary from 30 seconds to 5 minutes depending on the number of satellites, the obstructions in the environment and the mobility of the entity. The number of satellites in contact with the GPS receiver will be higher when the entity is moving very slowly, which in turn will result in the sensor unit acquiring GPS coordinates in a shorter duration of time.

In order to save energy, we duty cycle both the GPS unit and the radio at the expense of decrease accuracy in the localization of missing entities. After GPS coordinates are acquired, the radio starts sending beacons to record encounters with other entities. The decrease in localization accuracy occurs when two entities get within radio range but one or both radios are turned off, and no encounter is recorded. Moreover, even if the radio (active) periods of both entities

Fig. 2. MICAz node with the GPS board and batteries attached. The unit is protected with bubble wrap before closing the enclosure. The blue egg-liked enclosure can be sealed to protect the unit, leaving just a small hole to get the GPS antenna connector inside the unit.

overlap, the information exchanged may correspond to some time and location coordinates obtained in the past, i.e. the previous GPS active period, not the current time and location. We decided to alternate both GPS and radio active periods in order to greatly simplify the software implementation and the resulting code size. The design consists of a simple loop alternating between the active periods of both GPS and radio modules. Fig. 1 shows an illustration of the GPS active and radio active periods, as well as the beacon periods within the radio active period.

When any two sensors are in radio range of one another (encounter), they exchange their databases in response to the radio beacons. When one entity hears a beacon from another entity, it sends its records database to the beaconing entity. The sensor unit of the beaconing entity stores multiple, unique locations per entity while updating its database after an exchange. An encounter can be classified as successful or missed, depending on whether the GPS in each sensor unit is active (and the radio is inactive). Since each entity acquires GPS coordinates periodically, there is possibility of a missed encounter if either or both of the sensors have their GPS active. The possibility of a missed encounter poses additional challenges while choosing times for radio and beaconing periods, where radio period denotes the time between two GPS acquisition and beaconing period is the time between two successive beacons. Shorter radio periods result in greater accuracy in recorded locations of each entity and shorter beaconing periods result in higher probability of exchanging data during an encounter, both at the expense of expending energy (shorter lifetimes).

The information exchanged in an encounter is propagated among multiple sensors in subsequent encounters during database exchanges. The databases from each entity are dumped at the base station. It is possible to estimate the expected position or area of any missing entity from a history of its past locations.

Module	Factor	Value
MICAz Radio	Transmit (0 dBm)	54mW
	Idle	1.278mW
	Tx Rate	250 Kbps
GPS Module	Trickle Mode	65mW

TABLE I

FACTORS AFFECTING THE POWER CONSUMPTION AND DATA TRANSMISSION OF A MICAZ MOTE WITH A GPS MODULE

Parameter	Values for Simulation
Radio period^{*+}	100, 150, 200, 250, 300 (sec)
Beaconing period^{*+}	3, 5, 7, 10, 15 and 20 (sec)
Number of users^{+}	5, 10, 20, 40, 50, 60, 80, 100
DB Memory limits^{+}	200, 400, unlimited (# entries)
Mobility: speed^{*+}	0.5, 1, 2, 3 (m/s)
Mobility: path^{*+}	linear and Y-path

TABLE II

PARAMETERS FOR SIMULATION AND EXPERIMENTAL RESULTS. (NOTE: $^{+}$ DENOTES SIMULATION AND * DENOTES EXPERIMENT)

A. Hardware Architecture

Our choice of hardware was restricted to MICAz and Mica2 motes as they were the only sensors with off the shelf GPS modules at the time of development. Both of these platforms provide the desirable form-factor for our system but we chose the MICAz motes over Mica2 because of the higher transmission bandwidth provided by the MICAz (250Kbps) versus Mica2 (38.4Kbps). The GPS module is connected to the MICAz motes thru an extension board. Fig. 2 shows a picture of the SenSearch node with its enclosure.

The MICAz has an IEEE 802.15.4 compliant RF transceiver with transmission frequency of 2.4 GHz and direct sequence spread spectrum, which is resistant to RF interference and provides inherent data security. It supports the TinyOS and MantisOS programming platforms and offers plug and play support for a wide variety of sensor boards. For location information, we use the MTS420CA environmental sensor monitoring board which has an integrated GPS module (Leadtek 9546) compatible with MICA2 and MICAz. The size of the configuration EEPROM in the MICAz is 4KB. Fitting the operation system (MOS), application and database of the witness records into this 4KB memory, prevented the implementation of algorithms like erasure coding which could have helped reduce errors while operating in noisy, urban environments.

B. Software Architecture

TinyOS (TOS) is widely used for programming sensors modules. However, the concept of interface and configuration is not very intuitive. Providing feedback from the sensor modules to the end-user is complex and involves translating the hexadecimal output from the serial port. MantisOS, on the other hand, offers developer friendly C API's with better ease of programming as compared to TOS, while supporting many of the features provided by TOS.

MOS, however, has its disadvantages. The GPS driver has not been comprehensively tested. This prevented us from dynamically reconfiguring (reprogramming) the motes, forcing us to reset them while in operation, resulting in loss of information (missing entries in database).

In the following sections we cover some of the design decisions and tradeoffs specific to the SenSearch software implementation.

1) Database Entry Format: If we consider acquiring a GPS coordinates event as an incoming packet, the format of the database record is quite intuitive. A source node ID and acquisition time form a unique ID for each packet. While node ID is 2 bytes by default in MOS, we keep one byte each for hour, minutes and seconds. Therefore, our database record consists of a 2 byte Source ID, 4 byte GPS coordinates, 3 byte timestamp and a 1 byte hop count field.

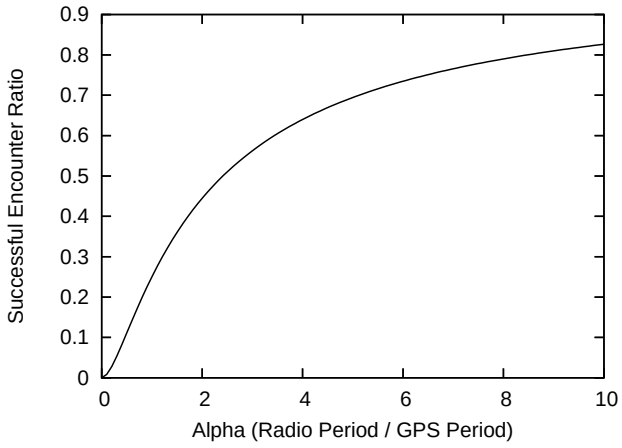
2) Memory Reuse: Each beaconing entity stores a fixed number of unique records per source entity, including itself, in its database. Currently, the SenSearch implementation does not have a way of optimizing the storage for the record database as this would incur additional costs such as higher computational power for updating entries in a compressed database. For this reason, we opted for storing only the last 5 GPS coordinates per entity as opposed to the entire history as a simple strategy for efficient memory utilization.

3) Communication Tradeoffs: Each beaconing entity wants to maximize the chance of exchanging data with other entities. At the same time, it does not want to expend a lot of energy, exchanging redundant information. For example, consider the scenario of hiking in groups and all the packets exchanged among the group members. One way to solve this case is to have a long beacon containing the IDs of all the records in the node database so the receiving node can decide which records it should send to update the database of the beaconing entity. However, in a hiking scenario where people travel in groups more than 90% of the time, making the beacon longer to save energy may result in higher power consumption in the long run. Another way is to have the database in handshake exchanging phase before the real data exchange, while implementing a compression algorithm like erasure coding. However, erasure coding is not foolproof as demonstrated in the case of static nodes planted on Redwood trees running TinyOS with 50% packet loss rates [22] and on Zebra collars that had reports of a single collar containing uncorrupted data [18].

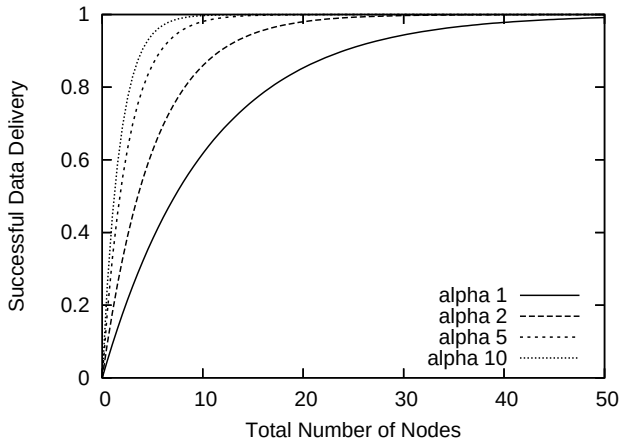
Instead, we implemented the simplest of schemes, the beacon only contains the ID of the beaconing node and the receiver replies it with its entire database. To reduce power consumption in the case of nodes traveling in groups, nodes do not reply to beacons if the timestamp of the last received entry from the beaconing node is close to its own GPS acquisition time.

IV. PERFORMANCE EVALUATION

In this section, we report results from a performance evaluation of SenSearch. A combination of simple mathemat-



(a) Successful Encounter Rate (SER)



(b) Successful Data Delivery (SDD)

Fig. 3. Fig. 3(a) shows the Successful Encounter Rate (SER) as a function of the α (t_{RP}/t_{GPS}) factor. SER tend to increase sub-linearly as α increases. Fig. 3(b) shows the Successful Data Delivery (SDD) as a function of the number of nodes. The larger the total number of nodes in the system, the larger the probability of success to deliver an encounter data to the base station. SDD also increases faster the larger the α.

ical models, simulations and real experiments is used for understanding the behavior and validating the performance evaluation of the system.

A. Goals and Objective Functions

SenSearch, in essence, is an entity tracking system. Tracking systems are characterized by their ability to track/locate a missing entity. If the system is built on a resource-constrained platform, the longevity of the system is a factor affecting the tracking capability. In our case, longevity is affected by the power consumption and limited memory of the sensor mote platform.

Our goals while evaluating the performance of SenSearch were two-fold. First, we wanted to perform analysis, simulations and real experiments to characterize the behavior of the system based on a specific set of objective functions. Second, we wanted to understand the relationships between different parameters and how they impact the performance of the system.

The objective functions we wanted to evaluate when designing the experiments were:

1) Localization Error: How good can we track the missing entities? What is the localization error based on the information provided by other witnesses?
2) Power Consumption: What is the lifetime of the system? What type of batteries should we use?
3) Memory Usage: How much memory we need to get acceptable performance? How is the performance affected as a function of the memory size?
4) Data Delivery Rate: How much information generated by a local node arrives to the base station by means of other witnesses?

The parameter space explored is given in Table II. Each of these parameters may affect one or more objective functions. We explored a subset of the parameters in the real experiments while exploring the entire range in the simulations. In the experiments we had a limited number of users (< 10) and we limited each node database size based on the limitations of our hardware platform. We explored the variation of these two parameters in simulation only. Our goal is to minimize all the objective functions, but they pose contradictory requirements. Analyzing these trade-offs when proposing a solution is one of the key contributions of this paper.

B. Analysis

To understand the relationship between our objective function and the different parameters, we start with simple mathematical analysis.

Suppose that all nodes are traveling along a linear path at a constant speed on any of the of the two directions. Whenever two nodes encounter each other, they exchange their (witness) databases. Nodes do not exchange databases multiple times with the same beaconing nodes during their radio period. Let n denote the number of nodes, t_{RP} denotes the radio period, t_{BP} denotes the beaconing period, t_{GPS} is the GPS acquisition time and T_t is the total time a node has been active. We assume that t_{GPS} is constant and that the memory of a node is not constrained to any particular size.

Under these assumptions, we formulated the following relationships:

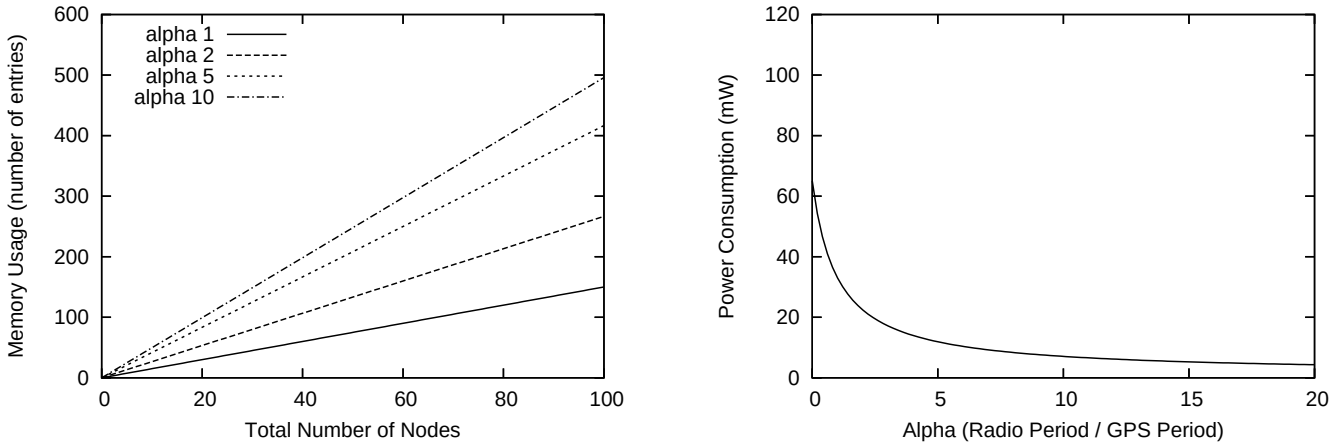
Successful Encounter Ratio (SER) is the probability that both nodes are in the radio period at the same time. It can be expressed as follows:

$$SER = \left(\frac{t_{RP}}{t_{RP} + t_{GPS}}\right)^2 = \left(\frac{\alpha}{\alpha + 1}\right)^2 \quad (1)$$

where α is the ratio of t_{RP} to t_{GPS}. Fig. 3(a) shows the SER as a function of α.

Data Delivery Rate(DDR) is the probability that at least one node in system delivers the encounter information to the base station. It can be expressed as follows:

$$DDR = 1 - (1 - SER)^{(n/3)} \quad (2)$$



(a) Memory Usage (MU) (b) Power Consumption (PC)

Fig. 4. Fig. 4(a) shows the Memory Usage (MU) per node as a function of the number of nodes (n). MU tend to increase linearly as n increases, with a larger slope as a function of α. Fig. 4(b) shows the expected Power Consumption (PC) per node as a function of α. The larger the α, the smaller the PC. Finding the correct t_{RP} settings is critical to control the PC of SenSearch.

assuming conservatively that only 1/3 of the total nodes encounter each other on a particular path. Fig. 3(b) shows the SDD as a function of the number of nodes.

Total Number of Encounters per node (T_e) will be directly proportional to the SER and the number of nodes in the system and will grow linearly with the number of nodes. It is given by:

$$T_e = f_1(n) = \beta \; SER \; n \tag{3}$$

where β depends only on the path followed by a node.

Memory Usage per node (MU) depends on the size of the database, which it depends mainly on the total number of encounters. The maximum size will depend upon the number of nodes whose records are contained in the database, the number of records per node and the length of each entry (l, constant). It can be expressed as:

$$MU = f_2(n) = \gamma \; T_e \; l = \gamma \; (\beta \; SER \; n + c) \; l \tag{4}$$

where γ would be affected by the path topology and the SER and c is a constant that represents the number of different GPS positions the node recorded for itself.

In the worst case, with no encounters, MU will increase only when a node creates a new entry for itself in the database, after getting new GPS coordinates. In most normal cases, the beaconing node will update its database so that there are no duplicate records for any node. So, memory usage will be mostly independent of beaconing period and vary only with the radio period and the number of nodes. Fig. 4(a) shows MU as a function of the number of nodes.

Total Transferred Data (in bytes) per node (T_D) depends on the number of records transferred during each encounter by a node and T_e. The main problem is estimating the average table size for all the encounters. The MU is a good approximation to the maximum table size after encountering all nodes, so we assume the average table size when encountering a node to be MU/T_e. The final formula can be expressed as:

$$T_D \simeq T_e \; \frac{MU}{T_e} \simeq MU \tag{5}$$

Power Consumption (PC): the GPS module is the dominant factor in the power consumption of the system. In addition, in a system with high levels of radio activity, such as SenSearch, transmissions/receptions/listening idle times account for a majority of the power consumption outside the GPS module. Power consumption would be affected by amount of transmitted data per node (beacon + database), and the total time the radio and the GPS modules are active.

The transmitted data is the sum of the data sent in form of beacons and the transferred data per node. The amount of transmitted beacon data (T_D^B) in turn depends on the number of beacons sent during the lifetime of a node and can be expressed as follows:

$$T_D^B = \frac{t_{RP}}{t_{BP}} \times \frac{T_t}{t_{RP} + t_{GPS}} \times l_B \tag{6}$$

where $\frac{t_{RP}}{t_{BP}} \frac{T_t}{t_{RP}+t_{GPS}}$ is the number of beacons sent and l_B is the length of each beacon in bytes.

For each byte transmitted, there is a reciprocal byte received by a witness node assuming no wireless losses. In the worst case scenario, the total received data is equal to the transmitted data. Moreover, in our case, we assume that the power consumed when receiving a packet is of the same order of magnitude than when transmitting a packet (see Table I for a comparison between Tx and Rx power for the RF transceiver used), so the total transmitted and received data is approximately twice the total transmitted data.

The ratio of time the radio module is active is given as follows:

$$R_{RP} = \frac{t_{RP}}{t_{GPS} + t_{RP}}$$

Conversely, the ratio of time the GPS module is active is given as follows:

$$R_{GPS} = \frac{t_{GPS}}{t_{GPS} + t_{RP}}$$

Therefore, the total power consumption of a node can be expressed as a function of T_D, T_D^B, T_t, R_{RP} and R_{GPS} as follows:

$$PC = 2\ \delta_1\ \frac{(T_D + T_D^B)}{T_t} + \delta_2\ R_{RP} + \delta_3\ R_{GPS} \qquad (7)$$

where δ_1 is a power consumed to transmit one byte of data and δ_2 and δ_3 is the power consumed while having the radio active and acquiring GPS coordinates respectively. In our specific implementation, we calculate the δ values using the information from Table I, being δ_1 = 54mW/(250Kbps/8bit/byte) = 0.0225 mW/byte, δ_2 = 1.278 mW and δ_3 = 65mW. Fig. 4(b) shows the PC as a function of α assuming n = 100 nodes and t_{BP} = 10 seconds.

The **Localization Error (LE)** was not calculated analytically. While the SDD factor gives you an approximate estimate of the amount of data that could be populated in the base station, which specific records arrived are more difficult to infer. Furthermore, the final LE will depend on the location estimation methods used to calculate the estimated node position in time (see Section IV-E). For this reason, we decided to explore the LE only with data collected in simulations and real experiments.

C. Simulation

To observe and analyze the variations in the different parameters, we built a discrete event simulation environment in C. We used an full n-factorial design to explore the entire parameter space for the system. The simulator emulates the behavior of the nodes in the SenSearch system.

When operating in duty cycle mode, for each simulation run, all nodes have the same t_{BR} and t_{RP}. t_{GPS} varies uniformly from 30 to 90 seconds. Every node calculates new GPS coordinates (location along a topology) before the beginning of each beaconing period. Nodes may have the same or different mean value for speed. However, the speed of each node varies around this mean speed, according to a normal distribution at each time step. The nodes move along a path as specified by an input topology file. We assume a constant radio range of 30 meters for all our simulations. The simulation is built on a time gradient of 1 second. Each second, the nodes march one step in the direction specified in the topology file.

At each time step, we check if a node is in GPS acquisition mode or in the radio period. If a node is in GPS acquisition mode, it cannot communicate with any other node. At the end of t_{GPS}, the node creates a new record for itself in its database with the current time and position. If a node is in the radio period, it sends out a beacon every t_{BR} seconds. Any other node which is within radio range and in its radio period, responds to the beaconing node with its database. The beaconing node then updates its database and the process continues. To prevent replicated transmissions, if a node database has not been modified since its last transmission then it does not respond again to beacons from nodes with whom it has exchanged databases. When a node reaches the end of its path, it dumps all its database into the base station. This is equivalent to a hiker who returns his unit before exiting the park.

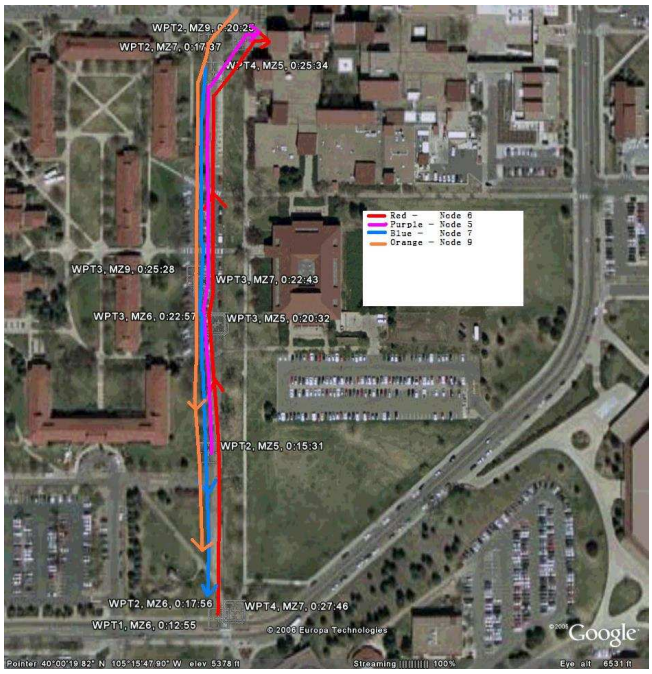


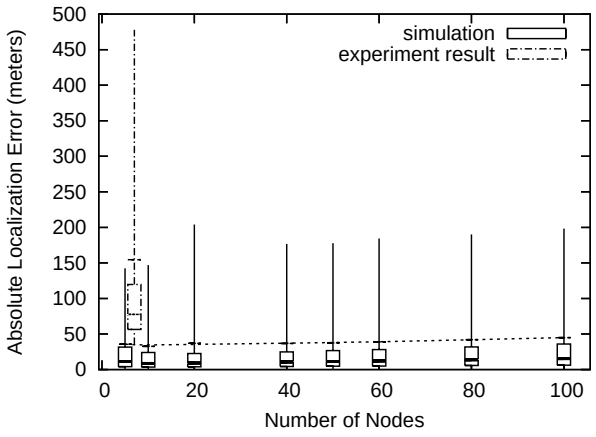
Fig. 5. Straight line paths, 4 nodes walking in opposite directions at University of Colorado, Boulder.

As a control case, we also run simulations with no duty cycle, i.e. we let the GPS unit and the radio turned on the full time. We would expect this case to give us the maximum accuracy in terms of localization of entities at the expense of the largest power consumption.

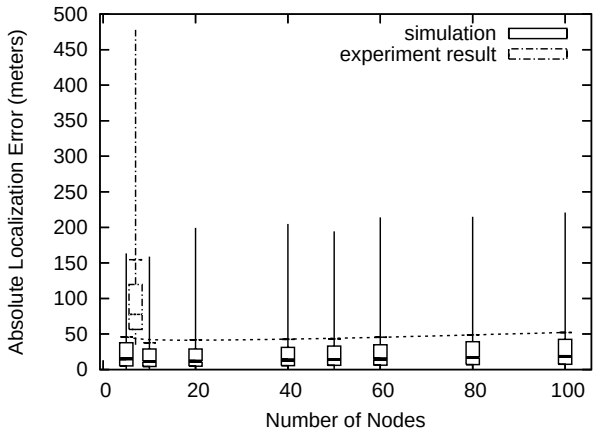
At the end of each simulation run, we compute the statistics for the different objective functions from the information dumped at the base station.

D. Experiment

We conducted several field tests using the Crossbow MICAz motes and MTS420CA weather board with the Leadtek 9546 GPS module. The system was tested and deployed in different weather conditions, at the University of California-Merced (July-August'2006) and at the University of Colorado-Boulder (November-December'2006). Out of the 20 odd experiments, some were conducted in summer when temperatures were between 80-90 degree-F and some in the late fall when it was 20 30 degree-F. In the initial experiments at UC-Merced, we tested the GPS acquisition times for mobile nodes. Using simple experiments, we studied the propagation of a node's data (GPS records) via other nodes to the base station. For example: with large initial separation between adjoining nodes and multiple groups of nodes start walking in opposite directions. We traced the propagation of a node's GPS records to the base station via other nodes using 5 additional fields in



(a) Absolute Localization Error (ON)



(b) Absolute Localization Error (DC)

Fig. 6. Figs. 6(a) and 6(b) show the absolute Localization Error (LE) as a function of the number of nodes. In both cases we see that the LE increases only very slightly as a function of the number of nodes. Moreover, there are no significant differences between the two cases except of a *very* small increase in the mean when doing duty cycle.

the records database to store the IDs of nodes through which the entry is propagated. This way we could trace the path of an entry up to 5 hops from its origin.

In the latter part of the system development, we ran field experiments to test the working of the system. The mobility patterns of nodes in the field experiments were as follows:

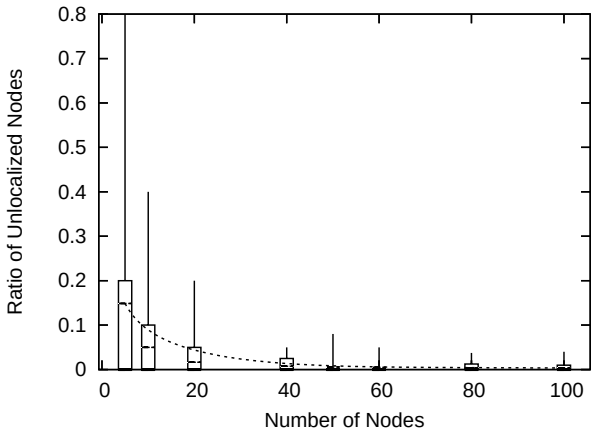
- The ”StraightLine” experiment was used to mimic encounters between entities when traveling along a straight path in opposite directions. Here, we had 5 nodes walk on a straight line toward each other, 2 on one side and 3 on the other. We designed the experiment such that the nodes acquired at least two GPS coordinates before their first encounter with nodes traveling in the opposite direction. The nodes had a beaconing period of 3 seconds, radio period 300 seconds and maximum GPS acquisition time 100 seconds. We coordinated with walkie-talkies and stated walking after every node got its first GPS coordinates. The nodes were constantly moving, i.e. they didn't slow down or stop when encountering each other and continued walking for 5-8 minutes after the last encounter. At this point, the experiment was terminated, we took a reference GPS waypoint using a handheld GPS unit. The localization error is the difference between the handheld GPS coordinate and expected location of a node, computed using its records from the databases of other nodes. Figure 5 shows the path followed during one of the field experiments.
- In the second representative experiment ”Lunch”, three groups of nodes were moving together in opposite directions. A group of 9 people went out to a common lunch place, each carried a node with GPS module. The whole group was divided into 3 groups - group 1 has node 1, 2, 3. Group 2 has node 4, 5. Group 3 has 6, 7, 8, 9. Node 1, and 8 failed to acquire any GPS coordinates or exchange data using the radio. Nodes 2 and 4 failed sometime during the run and were unable to transfer their databases to the base station. Each group has 15 minutes overlap at lunch place and group 1 and 3 ran into each other on the way. In this scenario, we were simulating a hike where people go in groups on a hike, take a rest at the turn around point, where they encounter other group of nodes and finally, head back to the start of the trail.

Temperature was a limiting factor in our experiments, as observed from the hardware failures during the field tests. Without any special packaging for the sensor motes, several of the nodes failed totally in our winter experiment i.e. never acquired any GPS coordinates. Any real-life deployment of the system would need to have a better way of protecting the nodes in hostile weather conditions.

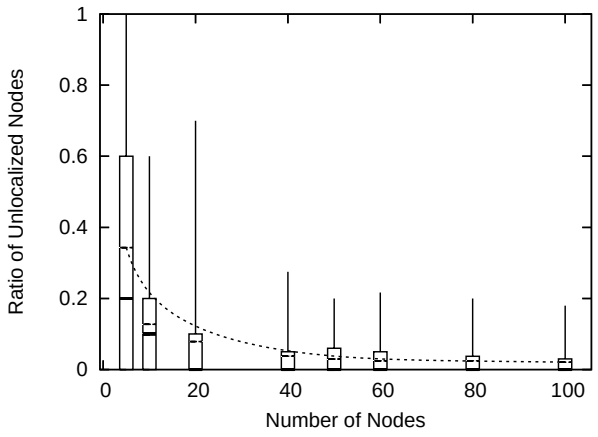
E. Localization Evaluation

Once data arrive to the base station from other nodes, a back-end process goes through all the records available and attempts to localize the nodes in the system. In our implementation, we used simple linear curve fitting to estimate a position as a function of all the previous records in the database. In order for this simple method to work, we need at the very least two known positions and times so we can infer direction and speed of the entity being tracked. The more points available, the better job we can do as long as past movement behavior of the entity is correlated with future one. In addition, having contextual knowledge of the terrain and possible paths could significantly increase the quality of the estimation. More complex methods could be used in this case, but further investigation is necessary and we left this for future work.

In our experiments, there were three cases when nodes could not be localized. First, if no data points are received from a specific node, this node is simply not localizable. Second, if only one data point is obtained, we cannot estimate a movement vector with a specific speed and direction. However, in this last case we could at least infer the “last seen point” of this specific node. Search and Rescue teams usually use this information to perform searches in expanding rings centered



(a) Ratio of Unlocalized Nodes (ON)



(b) Ratio of Unlocalized Nodes (DC)

Fig. 7. Figs. 7(a) and 7(b) show the Ratio of unlocalized nodes as a function of the total number of nodes for both the GPS and radio always active case (ON) and the duty cycle case (DC) respectively. In both cases we see that the ratio decreases significantly as a function of the number of nodes. In general there is a small increase in mean value when doing duty cycle.

around this point. Finally, because of lack of topographic information in our system, we can only infer a position estimation of a node if we get at least two points in the last direction of travel. This was significant in some of our multiple paths experiments and simulations (the "Y" topology), when the last hiker in a path changes directions of travel at the center of the topology and no other entity encounters him in the new direction of travel. In all these cases, we marked the node as unlocalized due to lack of information.

F. Results

In this section we discussed the performance results of our simulation and experiments based on the different objective functions and goals.

The first aspect we address is the evaluation of the SenSearch performance whether we use a duty cycle strategy (DC case) or we leave the GPS and radio active all the time (ON case).

Fig. 6 shows the Localization Error LE as a function of the number of nodes in the system. The graphs show the box-plots of the min, 25%, median, 75% and 95% distributions as well as the mean for the full n-factorial design, with different combinations of nodes speeds, mobility patterns, beaconing rates, and radio periods. We observe from Figs. 6(a) and 6(b) that the LE tends to increase only slightly when we use SenSearch with a duty cycling (DC) strategy. Furthermore, the simulation and the experimental results for cases with similar number of nodes tend to be consistent, with an increase in errors in our experiments for both cases. Moreover, the only trend that can be perceived is a very small increase in the average LE as we increase the number of nodes, but this increase is not statistically significant. Our results from simulations and experiments show that the LE does not change in a statistically significant way as a function of the radio period t_{RP} or the beaconing period t_{BP} (graphs not shown).

As we explained in Section IV-E, the LE only includes

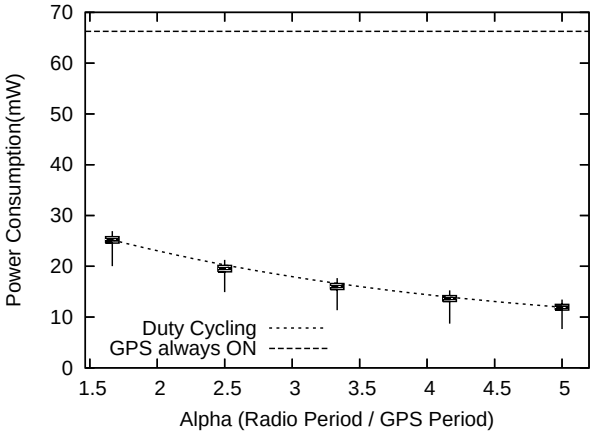
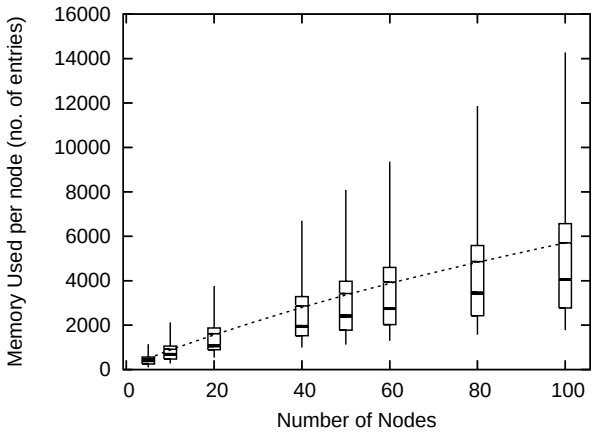


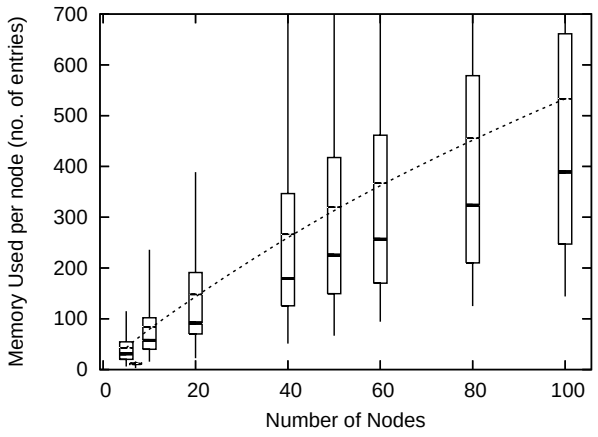
Fig. 8. Power Consumption as a function of the ratio between the radio period t_{RP} and the GPS period t_{GPS}, or the α factor for the always active ON and the duty cycle case DC. For the ON usage is dominated by the GPS board. When duty cycle is in place, the total power consumption gets reduces as we increase α, i.e. we increase the radio period with respect to the GPS acquisition time.

nodes that had enough information collected at the base station, such that we could attempt to localize them. Fig. 7 shows the percentage of nodes that could not be localized (UL) as a function of the number of nodes in the system. The first observation is that the percentage of unlocalized nodes with respect to the total number of nodes UL gets drastically reduced as we increase the total nodes in the system. This is due to the fact that the chances of a node not encountering any witness get reduced as well, maximizing the chances of getting multiple database entries for each node. When comparing both both the DC and ON graphs, we see a small but clear increase in the total number of nodes that cannot be localized. This difference is most significant when the total number of nodes in the system is small.

Being confident that SenSearch does not significantly underperform when duty cycling both the GPS and the radio, we focus our attention on the power consumption. Fig. 8 shows the



(a) Memory Usage (ON)



(b) Memory Usage (DC)

Fig. 9. Figs. 9(a) and 9(b) show the database memory usage as a function of the total number of nodes for both the ON and DC respectively. In both cases we see that there is a linear increase as a function of the number of nodes in the system. However, we see a very significant difference in the total number of entries populated in each node's database. This is explained by the much larger number GPS readings acquired per node.

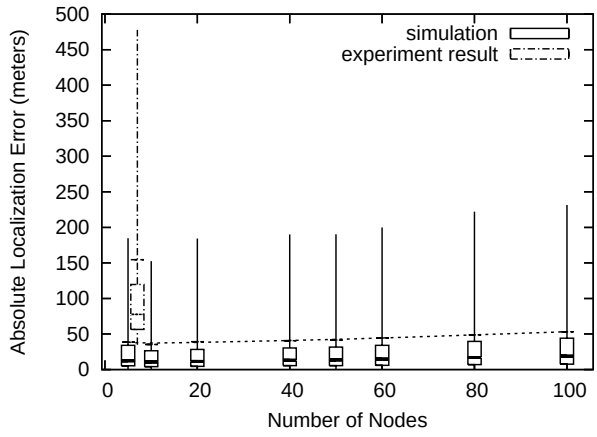
Power Consumption PC as a function of α. In our experiments and simulations, the PC was estimated using the total amount of data (including beacons) sent and received, the total up time of the system, and the time the GPS and the radio module were active, using Eq. 7. The first aspect to notice is that the PC tends to decrease with the increase in α validating some of the analytical results explored in Section IV-B. Similarly to the previous graphs, there are no significant differences between our experimental data and our simulations. However, when comparing both the ON and DC case, we see a significant decrease in PC for the DC case in comparison with the ON case as we increase the value of α. There are many reasons for this result. First of all, it is clear that the GPS board is the dominant factor in power consumption in the system. Having the GPS board always active, even when using the 3.5 times more efficient "trickle mode" (as compared with GPS "continuous mode") is a burden and it considerably reduces the lifetime of the system. Second, the fact that the radio is always active with no duty cycle also contributes to the total power consumption, although to a lesser extent than the GPS board. Finally, due to the increase in the node's database size (see below), the total data transfers increase a 10-fold average, also contributing to larger power consumption but with lesser impact than the previous two factors.

Fig. 9 shows the Database Memory Usage MU as a function of the number of nodes. Note that in the simulations we did not put any bound on the memory to understand the dynamics of the system at very large scales. Recalling our analytical expression for memory usage (see Eq. 4), we said that as the number of nodes in the system increases, the size of databases transferred during encounters would grow as a function of the number of nodes and the number of GPS acquisitions. The number of GPS acquisitions depends on t_{BP} and would affect the number of records a node creates for itself in the database. This number would be small compared to the number of records from other nodes a node has in its database on an average. So, memory usage is dominated by the number of nodes in the system as compared to the beaconing period and is independent of the beaconing rate. Thus, the analysis is validated by the simulation results as seen from the figures.

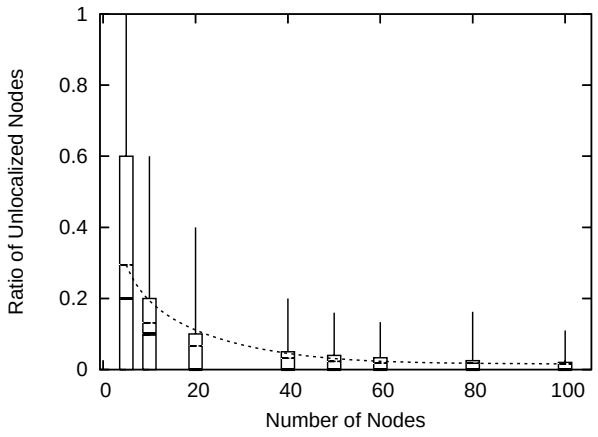
When comparing both Figs. 9(b) and 9(a) we see almost an order of magnitude increase in MU for the ON case. When GPS is constantly ON, the number of GPS locations acquired per node is very large. So, the size of the records database exchanged during an encounter with other nodes grows quite rapidly. Both, memory usage and transferred data are impacted by this behavior. When we power cycle the GPS and radio (DC), we get less GPS locations per node. Hence, the size of the records database exchanged during an encounter with other nodes remains reasonably small, leading to much lower memory usage and transferred data. For example: if we assume that when GPS is constantly ON, it acquires one reading every 30 seconds, during a 3 hour experiment each node will acquire 360 locations. On the other hand, if the GPS and radio are duty cycled (GPS ON time = 30 seconds and RADIO ON time = 270 seconds), the node will acquire 36 GPS readings in a 3 hour run. Now, if these nodes were to exchange records database between similarly configured nodes, with GPS constantly ON we would get almost 10 times higher memory usage and transferred data per node.

The second aspect we explore is how memory limitations in our hardware platform can affect SenSearch, in particular when running the system with larger number of nodes. Fig. 10(a) shows the Localization Error LE as a function of the number of nodes in the system when the database size is limited to 200 entries and performing duty cycling. When making a comparison with Fig. 6(b) we see that there is only a small increase in the average LE for any number of nodes in the system. Perhaps more importantly, the percentage of nodes that cannot be localized UL, does not change in a meaningful way as it can be seen when comparing Fig. 10(b) with Fig. 7(b).

Finally, the last aspect we investigate is Data Delivery Rate (DDR). For our experiments and simulations we define

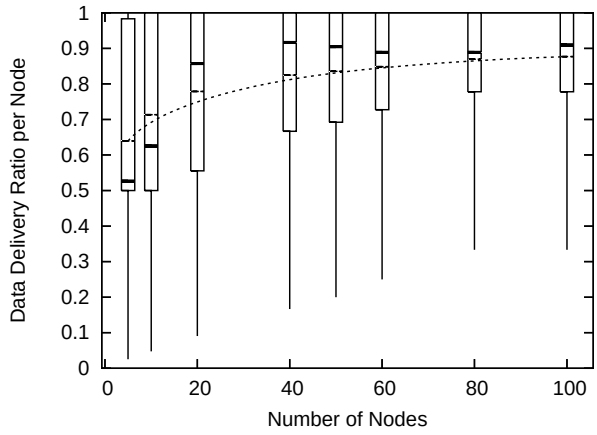


(a) LE with limited memory (DC)

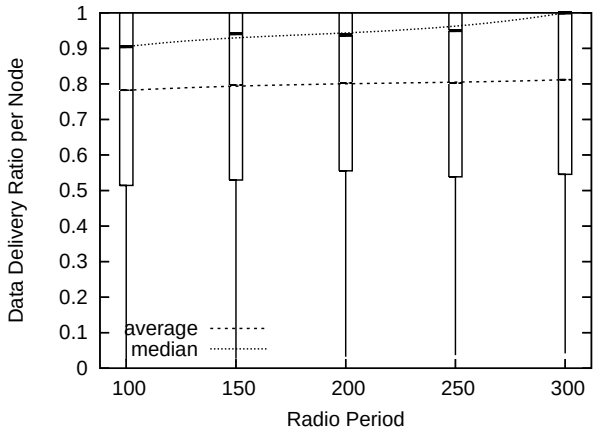


(b) UL with limited memory (DC)

Fig. 10. Fig. 10(a) shows the absolute LE as a function of the total number of nodes in the system when the system duty cycles and has a limit of 200 entries in the DB. We do not see a statistically significant increase in the LE for any number of nodes in the system with respect to Fig. 6(b). Fig. 10(b) the percentage of unlocalized nodes UL as a function of the total number of nodes. Again, we do not detect any significant statistical difference when comparing with the case of unlimited memory in Fig. 7(b).



(a) DDR vs Nodes



(b) DDR vs Radio Period

Fig. 11. Fig. 11(a) depicts the Data Delivery Rate DDR as a function of the total number of nodes in the system when the system duty cycles. As expected, the DDR tends to improve as we increase the total number of nodes in the system. Fig. 11(b) shows DDR as a function of the radio period t_{RP}. We see a small increase in DDR as we increase t_{RP}.

the DDR as the ratio between the unique GPS coordinates from any node/source that arrive to the base station by means of some other witness node versus the total number of coordinates generated by the local node/source. Recalling our analytical expression for memory usage (see Eq. 2), we see that the DDR should asymptotically move closer to 1 as we increase the number of nodes. Fig. 11(a) shows the DDR as a function of the number of nodes in the system for the duty cycle case. We see that the DDR increases asymptotically as a function of the number of nodes, but it never reaches the optimal 1, even for very large total number of nodes in the system. This is mainly due to the way we defined DDR for our experiments. A node that does not encounter any witnesses in some part of the path before reaching the base station for some period of time, will continue generating GPS entries in the local database that never get a chance to be delivered by any other node. Furthermore, in our simulations, after the node dumps a database to the base station, continue operating, generating additional GPS entries that may not be witness by any other node. The combinations of these factors implies that the total number of these GPS entries can be very large for some nodes, so the worst cases DDR may be even less than 10%. This why we see the average dropping due to some extreme DDR minimum cases.

In Fig. 11(b) we see how DDR changes as a function of the radio period t_{RP}. Recall that the α factor was directly proportional to the radio period for any fix GPS period t_{GPS}. We observe a small increase in DDR as we increase the t_{RP} in the median values. This is consistent with our analytical finding of increased DDR as we increase the value of α.

V. Design Considerations and Discussion

From a conceptual point of view, the first important observation is that duty cycle the GPS board is critical for this class of systems. It is possible to drastically increase the lifetime of the systems without incurring in significant penalties for the

localization and tracking of nodes as can be seen from Figs. 6 and 8. Another, possible more impacting ramification is that a large number of local GPS coordinates actually decrease the overall performance. This due to the large increase in database memory usage and the consequential increase in total data transferred, as we observed in Fig. 8.

One of the main problems first encountered when debugging SenSearch in our initial experiments is the issue of multiple nodes moving together in groups. This scenario in combination with our very simple request/response protocol for database transfers ends up with significant increases in memory usage by all the nodes in the group first, and then by the rest of the nodes in the system once other witness nodes are encountered along the paths. We further explored this in simulations by forcing groups of nodes moving together and by running different nodes with a plethora of slightly different time-varying speeds. After experimenting and simulating with different scenarios, we implemented a simple optimization that consisted of limiting each pair of node data exchanges to only once during a particular radio period. This simple optimization helped us significantly reducing the memory usage.

Another lesson learned was that our policy of storing only the last n recent entries for any particular node in a node's database also help the system memory usage to be bound, without a significant decrease in localization accuracy as shown in Fig. 10.

VI. Future Work and Conclusion

In the future we would like to further explore more complex methods for localization and tracking than the simple linear fitting methods used in this paper. We would also like to get further experimental data with larger number of nodes to verify some of the simulation findings at larger node scales. Finally, we would like to investigate the use of adaptive techniques to determine the duty-cycle periods as a function of past history and/or as estimation of the total nodes in the system.

In summary, we introduced a balanced architecture for a personnel tracking systems equipped with GPS modules to get accurate the location information. By using witnesses and a right set of parameters, Sensearch is capable of tracking down stranded people with acceptable accuracy. The system scales efficiently with the number of nodes while consuming bounded energy and memory resources.

References

[1] C. T. Inc., "http://www.xbow.com/."

[2] M. Corporation, "http://www.moteiv.com/."

[3] TinyOS, "http://www.tinyos.net/."

[4] nesC: A Programming Language for Deeply Networked Systems, "http://nescc.sourceforge.net/."

[5] M. M. N. of In-situ Sensors, "http://mantis.cs.colorado.edu/."

[6] E. S. for Wireless Sensor Networks, "http://cvs.cens.ucla.edu/emstar/."

[7] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson, "Wireless sensor networks for habitat monitoring," in *WSNA '02: Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*. New York, NY, USA: ACM Press, 2002, pp. 88–97.

[8] T. He, S. Krishnamurthy, J. A. Stankovic, T. Abdelzaher, L. Luo, R. Stoleru, T. Yan, L. Gu, J. Hui, and B. Krogh, "Energy-efficient surveillance system using wireless sensor networks," in *MobiSys '04: Proceedings of the 2nd International Conference on Mobile Systems, Applications, and Services*. New York, NY, USA: ACM Press, 2004, pp. 270–283.

[9] N. Bulusu, J. Heidemann, and D. Estrin, "GPS-less low-cost outdoor localization for very small devices," in *Personal Communications, IEEE [see also IEEE Wireless Communications]*, 2000, pp. 28–34.

[10] R. Gupta and S. R. Das, "Tracking moving targets in a smart sensor network," in *VTC '03: Proceedings of 57th IEEE Vehicular Technology Conference*, 2003, pp. 3035–3039.

[11] L. Gu, D. Jia, P. Vicaire, T. Yan, L. Luo, A. Tirumala, Q. Cao, T. He, J. A. Stankovic, T. Abdelzaher, and B. H. Krogh, "Lightweight detection and classification for wireless sensor networks in realistic environments," in *SenSys '05: Proceedings of the 3rd international conference on Embedded networked sensor systems*. New York, NY, USA: ACM Press, 2005, pp. 205–217.

[12] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. S. Peh, and D. Rubenstein, "Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with zebranet," in *ASPLOS-X: Proceedings of the 10th international conference on Architectural support for programming languages and operating systems*. New York, NY, USA: ACM Press, 2002, pp. 96–107.

[13] P. Zhang, C. M. Sadler, S. A. Lyon, and M. Martonosi, "Hardware design experiences in zebranet," in *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*. New York, NY, USA: ACM Press, 2004, pp. 227–238.

[14] B. Thorstensen, T. Syversen, T.-A. Bjornvold, and T. Walseth, "Electronic shepherd: a low-cost, low-bandwidth, wireless network system," in *MobiSys '04: Proceedings of the 2nd International Conference on Mobile Systems, Applications, and Services*. New York, NY, USA: ACM Press, 2004, pp. 245–255.

[15] J.-H. Huang, S. Amjad, and S. Mishra, "Cenwits: a sensor-based loosely coupled search and rescue system using witnesses," in *SenSys '05: Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems*. New York, NY, USA: ACM Press, 2005, pp. 180–191.

[16] R. T. H. S. J. A. Stoleru, "Walking GPS: a practical solution for localization in manually deployed wireless sensor networks," in *29th Annual IEEE International Conference on Local Computer Networks*, 2004, pp. 480–489.

[17] D. M. Doolin and N. Sitar, "Wireless sensors for wildfire monitoring," in *Proceedings of SPIE Symposium on Smart Structures and Materials/ NDE 2005*, 2005.

[18] Y. Wang, S. Jain, M. Martonosi, and K. Fall, "Erasure-coding based routing for opportunistic networks," in *WDTN '05: Proceeding of the 2005 ACM SIGCOMM Workshop on Delay-tolerant Networking*. New York, NY, USA: ACM Press, 2005, pp. 229–236.

[19] P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot, "Pocket switched networks and human mobility in conference environments," in *WDTN '05: Proceeding of the 2005 ACM SIGCOMM Workshop on Delay-tolerant Networking*. New York, NY, USA: ACM Press, 2005, pp. 244–251.

[20] A. Seth, D. Kroeker, M. Zaharia, S. Guo, and S. Keshav, "Low-cost communication for rural internet kiosks using mechanical backhaul," in *MobiCom '06: Proceedings of the 12th annual international conference on Mobile computing and networking*. New York, NY, USA: ACM Press, 2006, pp. 334–345.

[21] V. Srinivasan, M. Motani, and W. T. Ooi, "Analysis and implications of student contact patterns derived from campus schedules," in *MobiCom '06: Proceedings of the 12th annual international conference on Mobile computing and networking*. New York, NY, USA: ACM Press, 2006, pp. 86–97.

[22] G. Tolle, J. Polastre, R. Szewczyk, D. Culler, N. Turner, K. Tu, S. Burgess, T. Dawson, P. Buonadonna, D. Gay, and W. Hong, "A macroscope in the redwoods," in *SenSys '05: Proceedings of the 3rd international conference on Embedded networked sensor systems*. New York, NY, USA: ACM Press, 2005, pp. 51–63.