Parasitic Mobility for Pervasive Sensor Networks

Mathew Laibowitz and Joseph A. Paradiso

MIT Media Laboratory
20 Ames Street, E15-351
Cambridge MA, 02139
{mat, joep}@media.mit.edu

Abstract. Distributed sensor networks offer many new capabilities for contextually monitoring environments. By making such systems mobile, we increase the application-space for the distributed network mainly by providing dynamic context-dependent deployment, continual relocatability, automatic node recovery, and a larger area of coverage. In existing models, the addition of actuation to the nodes has exacerbated three of the main problems with distributed systems: power usage, node size, and node complexity. In this paper we propose a solution to these problems in the form of parasitically actuated nodes that harvest their mobility and local navigational intelligence by selectively engaging and disengaging from mobile hosts in their environment. We analyze the performance of parasitically mobile distributed networks through software simulations and design, implement, and demonstrate hardware prototypes.

1 Introduction

1.1 Basic Principle

We are at a point in time where advances in technology have enabled production of extremely small, inexpensive, and wirelessly networked sensor clusters. We can thus scatter large quantities of sensors into an environment, creating a distributed sensor network. Each individual node in the network can monitor its local region and communicate with other nodes to collaboratively produce a high-level representation of the overall environment. By using distributed sensor networks, we can sculpt the sensor density to cluster around areas of interest, cover large areas, and work more efficiently by filtering local data at the node level before it is transmitted or relayed peer-to-peer. [1]

Systems of many small sensors can be deployed to cover large areas of any geometry. By adding autonomous mobility to the nodes, the system becomes more able to dynamically localize around areas of interest and adapt to changes in the sensing landscape. [2] Mobile sensor networks are accordingly well suited to sampling dynamic or poorly modeled phenomena. The addition of locomotion further provides the ability to deploy the sensor network at a distance away from the area of interest, useful in hostile environments. Cooperative micro-robots can reach places and perform tasks that their larger cousins cannot. [3] Mobility also allows the design of a system where nodes can seek out power sources, request the dispatch of other nodes to perform tasks that require more sensing capability, seek out repair, and locate data portals from which to report data. [4]

But the creation of mobile nodes is not without a price. Locomotion is costly in terms of node size and power consumption. In dense sensor systems, due to the large quantity of nodes and distributed coverage, it is difficult to manually replace batteries or maintain all nodes. Some researchers [5] have explored using robots to maintain distributed networks, but this is difficult to implement over large, unrestricted environments. Additionally, the added intelligence and processing power required for a node to successfully navigate in an arbitrary environment further increases the power and size

requirements of each node. Large nodes, in physical size, complexity, cost, and power consumption, prevent the sensor network from being implemented in most environments. [6],[7]

This research is concerned with exploring a novel type of mobile distributed sensor network that achieves the benefits of mobility without the usual costs of size, power, and complexity. The innovation that allows this to happen is the design of nodes that harvest their actuation and local navigational intelligence from the environment. The node will be equipped with the ability to selectively attach to or embed itself within an external mobile host. Examples of such hosts include vehicles, fluids, forces (eg. selectively rolling down a hill), people, and animals. These hosts provide a source of translational energy, and in the animate cases, they know how to navigate within their environment, allowing the node to simply decide if the host will take it closer to a point of interest. If so, the node will remain attached; when the host begins to take the node farther away from a point of interest, the node will disengage and wait for a new host.

This area of research addresses the intersection between mobile agents, dense distributed networks, and energy harvesting. This paper presents the design and development of hardware and software systems to address the combination of these interests as "parasitic mobility".

Sensor networks are related to and/or encompassed by many larger fields including pervasive computing. The research and project described herein use sensor networks as an example space for developing and understanding the more general idea of parasitic mobility. These ideas are applicable to any field or space that is concerned with the physical distribution and mobility of computational nodes.

1.2 Related Work

Although this research has no direct precedent, it is inspired by systems in nature and human society (discussed in Chapter 2) and it builds upon current work in the encompassed fields of distributed sensor networks and mobile systems.

Wireless sensor networks have become a large area of research, with many universities and institutes contributing. Strategic seed programs begun in the 1990s, such as DARPA's SENSIT initiative [8], have grown into an international research movement.

The Smart Dust Project at UC Berkeley [9] has set a theoretical goal for extremely small nodes in dense embedded sensor networks. While the project itself did not put an actual hardware platform into production, it spun-off into the Mote [10] and more recently the Spec [11]. The Mote is currently the most popular platform for experimenting with compact wireless sensing. It has also served as a building block for many mobile sensor agent projects, all of which essentially involved putting a Mote onto some sort of robot [9]. The Spec is the current result of a project intended to shrink down the Mote to the theoretical goal of the Smart Dust project. While not yet that small, the Spec is around 4mm x 4mm (not including the battery or antenna) and will open the door for many dense sensor array experiments. Similar work is also proceeding at other institutions (e.g. The National Microelectronic Research Center in Cork, Ireland [12]); the research community is congealing around the goal of producing millimeter-sized multimodal wireless sensor nodes. Parasitic Mobility is intended as a means to add mobility to systems built to meet the specifications of these projects with regards to size, power, and node complexity; as the nodes grow smaller, parasitic mobility becomes increasingly feasible and desirable. As the power source remains a problem, current research in energy scavenging [13] and adaptive sensing [14] is very relevant to this initiative. Adaptive sensing is the technique by which sensing capabilities (active sensors, sampling rate, power consumption, bit-depth, transmission, processing) are increased and decreased according to the sensor data itself, never decreasing below a level capable enough to determine when more sensing power is necessary. Such approaches are currently being implemented using the Stack Sensor Platform [15] at the MIT Media Lab. The Networked InfoMechanical Systems research area at the Center for Embedded

Networked Sensing at UCLA conducts research and builds systems to investigate adaptive sensing [14] and mobility for distributed sensor networks [16].

And finally, while not distributed sensor networks, there are several mobile sensor devices built by attaching large sensor packages to floating platforms that drift about in ambient flows while collecting data. Some examples include Sonobuoys [17] that acoustically hunt for submarines, drifting instrumentation packages to monitor ocean temperature [18], and balloon-borne modules for surveillance and proposed planetary exploration [19].

2 Examples of Parasitic Mobility

2.1 Parasitic Mobility in Nature

The natural world provides us with many examples of parasitic mobility, including organisms that rely entirely on larger organisms to carry them to habitable locations. Parasitic relationships of this sort are called phoretic relationships from the word phoresis, which literally means transmission [20]. In the context of this paper, these examples are separated into three categories: active parasitic mobility consisting of organisms that attach and detach at will from hosts with their own actuation, passive parasitic mobility consisting of passive nodes that are picked up and dropped off, knowingly or unknowingly, by hosts, and value-added parasitic mobility which consists of either passive or active parasitic organisms that provide symbiotic value to the host in exchange for transportation.

Active Parasitic Mobility. The first such example that comes to mind when discussing parasites in nature is the tick. The tick actively attaches to hosts by falling from trees or by crawling directly onto the host. It remains attached by using an actuated gripping mechanism which it can release whenever it decides to seek food elsewhere. Although the tick is transported to new locations by the host, its primary reason for attachment is to use the host as a source of food. It is therefore not normally considered a phoretic organism. It is still relevant to the topic as the main example of an active attachment mechanism.

Several species of nematodes, a.k.a. round worms, exhibit phoretic behaviors. The Pelodera Coarctata is a nematode that is commonly found living in cow dung. When the conditions in the dung deteriorate and become inhospitable for the nematode, it attaches itself to a dung beetle which will carry it to a new fresh dung pat. [21] Another such nematode is the Onchocerca Volvulus which is infamous as the cause of “River Blindness.” This worm attaches itself to Blackflies that in turn bite humans allowing the worm to travel through the skin and infect the host human. These Blackflies themselves are also an example of parasitic mobility. Their larvae require an aquatic stage for growth, so they often attach themselves to freshwater crabs to bring them into the water and protect them. [21]

Marine life is ripe with examples of active parasitic mobility. One example is that of the Remora or Suckerfish. These fish have developed a sucker-like organ that they use to attach to larger creatures such as sharks or manta rays. By attaching to these larger, faster animals the remora covers area faster giving it more access to food. [22]

Passive Parasitic Mobility. Plants often employ parasitic mobility as a means of distributing seeds. A common example of this is the dandelion. The dandelion seeds have a tiny parachute that carries the seed with the wind. This allows the seeds to travel some distance in hopes of landing in an area that provides the requirements of growth. It is completely passive and at the whim of the wind. It is not expected that all the seeds will land in arable areas. This is overcome by the sheer quantity of seeds released into the air.

This is more opportunistic than parasitic, but still falls within the conceptual boundaries of this research.

Other plants, with behaviors more aptly described as parasitic, distribute their seeds in bur casings. These prickly cases stick to animals that brush up against them or step on them. They are shaken lose or fall off as a result of shedding, usually at a new location.

Value-Added Parasitic Mobility. Fruit-bearing trees distribute their seeds in a value-added method. Animals gather the fruits as a food source and in turn spread the discarded seed-containing cores. This attraction and provision acts as an attachment mechanism for the seeds. The detachment mechanism is the inedibility of the seeds within the fruit, in other words, when the added value has been used up.

Flowers use their scented petals to attract bees and other insects. The flowers also provide nectar. The bees use the nectar to make honey and carry the pollen from flower to flower. This is an extremely well evolved symbiotic system that has very little wasted energy or resources. [23]

The existence of many such well-evolved systems in nature illustrates the validity of this type of mobility, and justifies investigating further how to use this concept in our research.

2.2 Parasitic Mobility in Society

In human society, many of the systems surrounding us exhibit emergent behaviors that exemplify parasitic mobility. It is important to examine these systems, not only as conceptual examples, but also because it may be possible to embed sensor network technology directly into these existing systems and take advantage of their mobility.

Basic examples, such as people being pulled along by a bus on skateboards, exist throughout society. It is often beneficial to attach to something that can travel in ways that a person cannot. This example further illustrates the economies of parasitic mobility; the people are getting a free ride.

A simple example of parasitic mobility is when a lost object, such as a cellular phone, is returned to its owner. This method of actuation is a combination of the device identifying its destination and a desire for the host to bring it there. Keeping this in mind, it may be possible to design devices that could identify some sort of reward for bringing them to a point of interest to the device. Another example of this behavior is that of a consumer survey (a sensor of sorts) that is redeemable as a coupon when returned.

There are many everyday objects that are only useful for brief intervals. One example of this is a writing utensil. A pen is needed to record information when it is presented or invented; afterwards the pen sits dormant awaiting the next burst of usefulness. During this period where the pen is not deemed useful, it is free to be relocated. It is often relocated by a host requiring its use in another location. As a result, pens generally cover large areas over time, and due to their unlikelihood of being returned, people usually have redundant supplies of pens. Equipping pens with a pervasive computing device is a good way to gain coverage of an environment, particularly an office or academic institutional building.

3 Software Simulation

In order to better examine the concept of parasitic mobility, extensive software simulation was performed. Through the process of designing the software simulator, the proposed systems were examined from the ground up, looking at all the factors that influence a potential sensor network of this type. This was a critical first step into research of this topic. Upon its completion, the simulator was an invaluable asset for testing and examining ideas and algorithms, generation of data identifying expected

behaviors, further validating of the overall concept, and providing insights directly used in the design of the hardware system described in Section 4.

3.1 Design Overview

The design of the software simulator can be broken down into three areas: Environmental Simulation, Host Behavior, and Paramor Behavior. “Paramor” is the name given to a parasitically mobile node and is an apt anagram for PARAsitic MObility Research. On top of these areas, the simulator contains all the necessary hooks for interactively changing behaviors and trying out new algorithms, detailed logging of activity data, and unattended running of multiple simulations with a desired timescale.

The simulator is grid based, and has been tested with maps as large as one million cells. The hosts and paramors participating in the simulation move by transitioning from cell to cell.

Environment Setup. The first step in setting up a simulation is to layout the environment using the Parasitic Mobility Simulator Map Editor. This is the interactive graphical application shown below in Figure 1. The user interface for this tool consists of a window displaying a scrollable, tile-based map, and a control panel for editing the parameters of the selected tile.

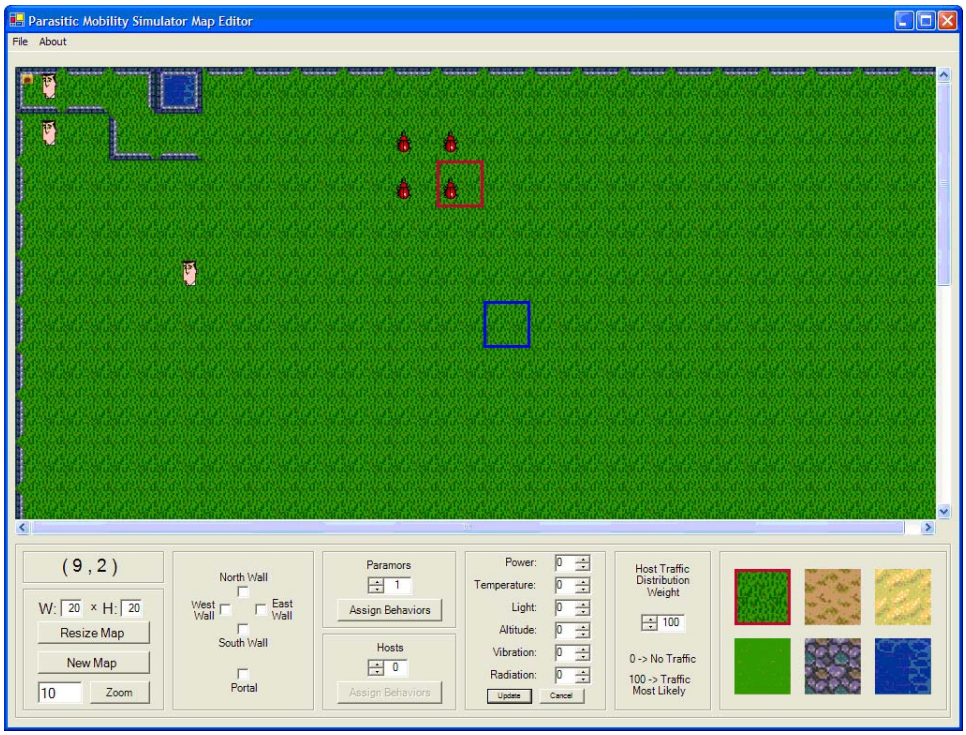

Fig. 1. Screenshot of the Map Editor showing a grid where four parasitic nodes have been deployed, a few hosts have been added, and some walls are being drawn in

Each cell can be assigned levels of environmental properties such as temperature, light, radiation, and the presence and strength of a power source. Basic geographic information can be added in the form of walls and portals.

Each cell in the map can be given a value for the likelihood that a mobile host will pass through it. An example of this would be for an animal-based simulation where locations featuring water and food will have more host traffic than locations without such attractions.

The final action that can be performed in the map editor is populating the map with hosts and choosing deployment locations for the parasitically mobile nodes.

Host Behavior Simulation. In order to accurately simulate parasitic nodes, credible hosts must be designed. It is also necessary to be able to adjust the hosts to simulate different types of real world entities.

A host will essentially stand at a location, list all the possibilities for a new location to travel towards, and randomly select one, with the randomness weighted by the environmental and behavior parameters. The available behavioral parameters in the simulator are stay weight (likelihood that the host will stay in place), stay duration (once a host has decided to stay put this is how long it will stay before reevaluating the situation), covered/uncovered weights (the desire the host has with regards to sticking to places it has already been or exploring new places), portal weight/duration (the likelihood that a host will decide to leave the immediate area and travel through a portal and how long before the host will return), and the speed at which the host travels.

By setting these parameters, hosts can be created that have high levels of randomness or hosts can be created that have no randomness and follow a specific pattern. This implementation allows the simulation of most environments populated with mobile hosts, such as cars, people, and animals. Further enhancements to the host simulation with enable more specifics about the hosts to be simulated, such as the hosts susceptibility to have a parasitic node attach to it.

Parasitic Node Behavior Simulation. The basics behind a parasitically mobile node's behavior are a set of objectives for the node. When a node is idle and a host body comes in range of it, the node attaches to the host. While attached to the host, the node uses the information it can gain from the environment and host's activities to determine if detaching will help it reach its objectives. The objective and behavior of each node can be described by the following parameters: power rate (the rate at which the node uses power), attachment power (additional power needed to attach and detach from a host), power threshold (the level of power reserves below which the node should seek out power as its main objective), battery life (the power capacity of the node), X/Y Goals (a list of 0 or more specific geographic locations that the node should try to reach), goal time (the amount of time the node should spend at a goal location before moving on), sensor thresholds (e.g. light threshold; the level at which the node considers an environmental condition interesting and part of its objectives), coverage (whether or not the node should try to cover as much area as possible), and hops per locale (number of discarded hosts the node is allowed per location, explained more below).

In general, if the node has a set destination, it will try to reach it. If it comes across an area of sensor interest, it will detach, stay for the sensor duration, and then try to continue towards its destination.

If a node is on a path to a destination or trying to go to where is has not been, it needs to hop off when it is on a host that is taking it farther away from its destination, or back into a covered area. This is a matter of calculating the distance to the destination at two points on the trajectory and testing its change or otherwise estimating the host's direction and comparing it to the node's stored coverage map. It is important to allow the node to travel a distance sufficient to sense the host's direction before hopping off. The "hops per locale" parameter is used to prevent situations with nodes being stuck in a spot surrounded by directions where it does not want to go. In a situation like this, the node would continually hop on and immediately hop off every host thinking that the host would take it in an undesirable direction. Using this parameter, the node will only do this a limited number of times until it will choose to stay on any host just to get to a new location and try again.

3.2 Simulation Results

With appropriate parameters, this package can simulate many environments and help test out algorithms and ascertain the requirements for a particular parasitically mobile sensor network. It can also be used to generate numerical data for predicting behavior, such as how long it will take a node to reach a location according to the properties of hosts in the environment, the power usage compared to standard robotic devices, and how many nodes should be deployed to cover an area in a particular amount of time. This section presents some of these results, collected from thousands of hours of simulation time.

Velocity Data. The data shown in Figure 3 results from simulating an environment with a fixed size and host population. The speed of the hosts is also a constant one grid hop per unit of time. The simulation parameters can be mapped to any units provided everything is scaled appropriately. In the next section, this data is used to calculate real-world energy usage values, hence the simulator distance units are related to meters and the timing units are related to seconds. This would mean the hosts travel with a speed of 1 m/s, similar to the walking pace of a human.

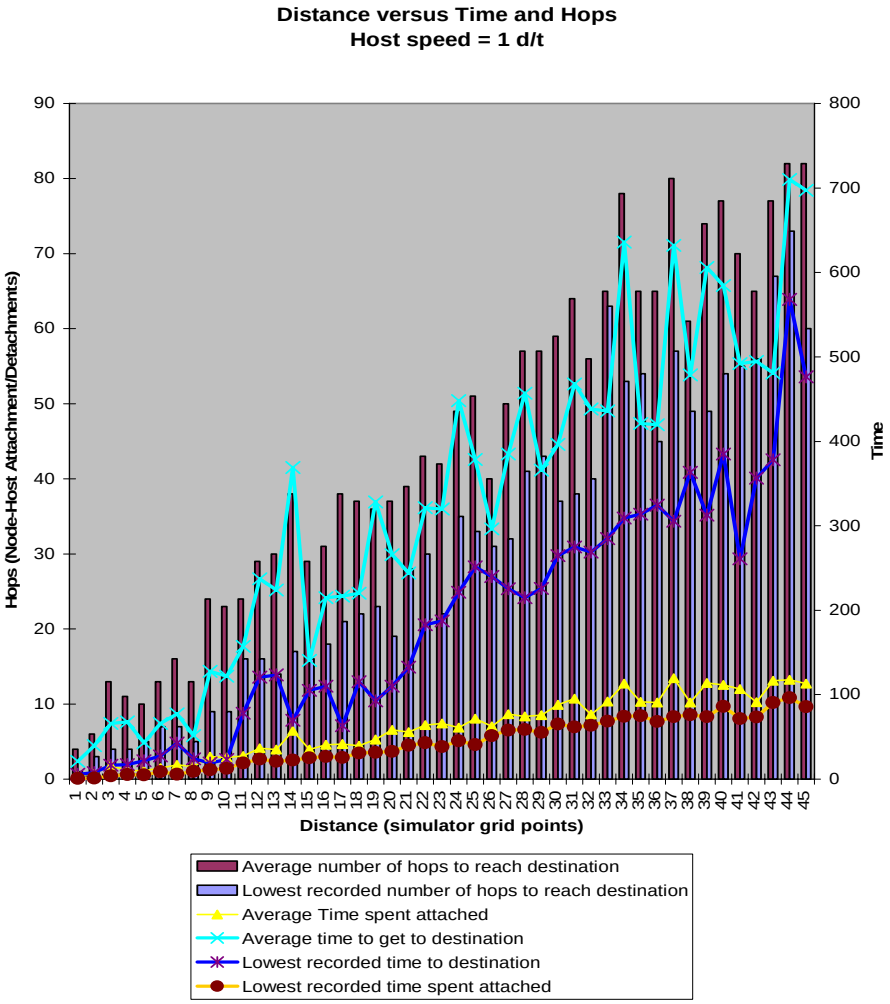

Fig. 3. Graph showing distance versus time data collected and averaged from repeated simulation passes.

The paramor behavior chosen for this test is simply to go to a particular location at a known distance away. The algorithm for attachment and detachment is to attach to every

host that comes by, and then decide whether it is bringing the node closer to or farther away from the destination. For a real device to be able to ascertain this information, it must ride the host for long enough to get a fix on the motion or the new location. Based on the GPS and Bluetooth localization systems described in the next chapter, this was set in the simulator as 1/2 of a time unit before the node knows the new location with a resolution of 1/2 of a distance unit.

This simulation was executed 25 times for each distance and the results were averaged. The linearity of this graph is due to the host's random behavior with respect to the node's destination. In other words, at each point (a point being at the end of the node's minimum cycle required to ascertain the direction that the host is traveling) the host is just as likely to turn away from the node's destination as it is to continue moving towards it. Therefore, the average time it takes for a host to take you from one location to the next location that is closer to the destination is the same regardless of how you arrived at the current location. Since the host reassesses its path at each point, the average time it takes a node to find a host going one step closer to the destination and to ride it to the next point, is constant over the entire travel. Hence, the average time it takes to go N number of steps should be N times the average time it takes to go one step, leading to a linear relationship.

Besides the total time it took to reach a destination, the graph also shows the time spent attached to a host, in other words, the time spent non-idle and actually traveling. This line is quite smooth; especially in comparison to the total time including time spent idly waiting for a host to pick it up. This smoothness shows that the algorithm is working properly as most of the time is spent waiting for a beneficial host, which varies according to the random flows of the hosts. This randomness is eliminated by the paramor's decision process. The reason that the attached time and distance don't exactly scale with the host velocity is because the nodes still need to attach to the host to ascertain its direction, and even hosts that head towards the destination are not guaranteed to go exactly straight, especially considering the coarse location system of the node.

Many other scenarios have been simulated using this software package. The data collected has illustrated that the total time to destination scales fairly linearly if you change basic environmental properties such as number of hosts. Decreasing the density of hosts increases the slope of the total travel time versus distance traveled, but it there is little or no change in the relationship between attached time and distance traveled. Further simulation indicates that increasing the host speed relates in a proportional increase in overall node speed up to a point. At some point, increasing the host speed causes a decrease in overall node speed. This is due to the node's ability to calculate the host's direction. In other words, if the host is traveling quickly, the node will deviate by a large distance before it can figure out that it is going off track and detach. This identifies that a particular node implementation will have a maximum host velocity above which its detachment algorithm becomes useless and the node's behavior becomes mostly random. This will definitely be an issue when designing real mobile systems where in all but very special cases (such as scheduled vehicular systems like trains) the node will have to attach to the host and ride to find out where it is going.

Coverage Data. Of particular interest to mobile sensor networks and the distribution of ubiquitous devices, is the ability to release nodes without a specific destination and have them attempt to scan or deploy over the entire area.

The first component of the algorithm is to equip the nodes with the ability to record where they have been. When a host takes them back to a location they have already covered, they detach. This proved to be inadequate as the nodes quickly found themselves surrounded by places they had already covered up to the radius of their ability to sense where the host was taking them. Depending on the processing ability of the node, it may be possible to analyze the entire map of coverage and determine general desired directions even if it the node first must travel through an already covered area.

After experimenting with several behaviors, it appears that the key to coverage is to keep moving, even if you might be heading in a direction that has an area in the

immediate vicinity that the node has already visited. The chosen algorithm for this simulation is to limit discarded hosts (hosts that are heading to a location already visited) to one per location. In other words, when a host comes by an idle node, the node will attach, and decide if the host will take it to an unvisited location. If not, the node detaches, and waits for the next host. This time it will take the new host without question and ride it until it finds an uncovered location, or at least arrives at a location that has nearby unvisited locations so it can detach and encounter a high likelihood of a host coming by that will go in that direction. It is also important not to attach to a host that already has a paramor attached to it. If two nodes are on the same path, both looking to cover the environment, they will most likely remain together by making the same decisions. Simple broadcast commands sent from individual nodes can aid with the dispersal of mobile sensor nodes. These commands can tell other nodes which areas have been covered and areas of high host traffic.

Energy Usage Calculations and Comparison. By examining the hops, attached/traveling times, and wait times from the simulator, we can calculate predicted values for the energy consumption rates of a parasitically mobile sensor node. In this section, we introduce the two kinds of nodes designed as the hardware components of this research and compare their predicted power usage statistics to that of two standard mobile robotic sensor devices.

The first device designed for this experiment is a 1 cubic inch node mimicking the passive attachment mechanism of a bur with the ability to actively detach by shaking itself loose. This device is further detailed in Section 4.

Since the attachment is passive and it sticks to every nearby host (bur-like attachment), it requires no additional power, actuation, or sensing during the host discovery and attachment process. When it is idle and waiting for a host, it can remain in a low-power mode and wake up on motion caused by being picked up by a host. The low power mode runs using a 32 KHz clock and keeps alive a comparator on accelerometer data. This low power mode draws around 35 uA. Additionally, the node will wake up once per second and check for a wireless message. This check lasts around 10 ms and uses 16 mA for that duration. However, the power for the communication system will not be included in these calculations because, at this point, we are just looking at the power needed for mobility to compare to standard techniques of moving sensors.

While attached, the semi-passive node can enter a different low-power mode and periodically wake up to check its location, progress, and sense the new surroundings. Assuming that the location system has a resolution of one meter, in the simulation of the environment with the hosts that move at 1m/s, the device will have to wake up and sense once per second of attached time. Depending on the type of location system and sensors, the node could use up to 60 mA for up to 60 ms for gathering data about the location and conditions surrounding it. Two location systems are described in Chapter 4; both systems require less power than this estimate.

And finally, when the node decides that it is time to detach, it needs to activate a detachment mechanism, which in the case of these sticky nodes is a pager motor with a draw of 15mA activated for 500ms to shake loose.

In addition to the semi-passive nodes, an active node was design that adds an actuated method for attachment as well as detachment.

The active node power calculations are fairly similar to those of the semi-passive node. The active node requires 30 seconds, drawing 10mA to wind its spring and execute an attachment or detachment. This node hops at a height of around 6 cm and weighs around 40 grams.

When the active node is not attached it requires more power than the semi-passive node because it needs to identify and locate a potential host to attach to. This requires 30mA of constant power draw to run an IR proximity detection circuit with a fairly high sampling rate. This sampling rate can be reduced, but worst-case ratings are being used for these calculations. Other techniques (e.g. active and passive acoustic sensing, simple

vision with a mini-camera) can be used as well, requiring different amounts of power, but with clever adaptive sensor processing [Ari], we assume that 30mA is still in the ballpark.

The first robot chosen for power comparison with parasitic mobility is NASA's Urban Reconnaissance Robot. [24] This robot is equipped with an enormous array of sensors, actuators, and processing power. It is designed to navigate through very tricky environments. Parasitically mobile nodes gain this ability from the hosts they attach to, and these hosts have evolved to navigate their environment in the best way possible. The NASA robot is an ideal comparison as it is a prime example of the power needed to build a device that navigates in a way that parasitically mobile nodes potentially get for free.

This robot draws 145 Watts while moving, sensing, and navigating on flat ground at a rate of 80cm/second. It can also climb stairs using 245 Watts of power. For this comparison, we will assume it is on flat ground. Using this energy consumption rate and speed of travel we can create power versus distance data and compare it to that of the parasitic node.

On the other end of the spectrum from the NASA robot is University of Southern California's RoboMOTE [25]. It is a small wheeled robot measuring less than 6 cubic centimeters in volume. While it does not have the navigation or actuation abilities of the NASA robot, it makes up for it in size, cost, and power consumption. The RoboMOTE exhibits many of the desirable attributes of mobile sensor networks, such as small cheap nodes that can work together. When all the features required for navigation are active, the RoboMOTE uses 1.5 Watts and can travel at a speed of 0.27 km/h.

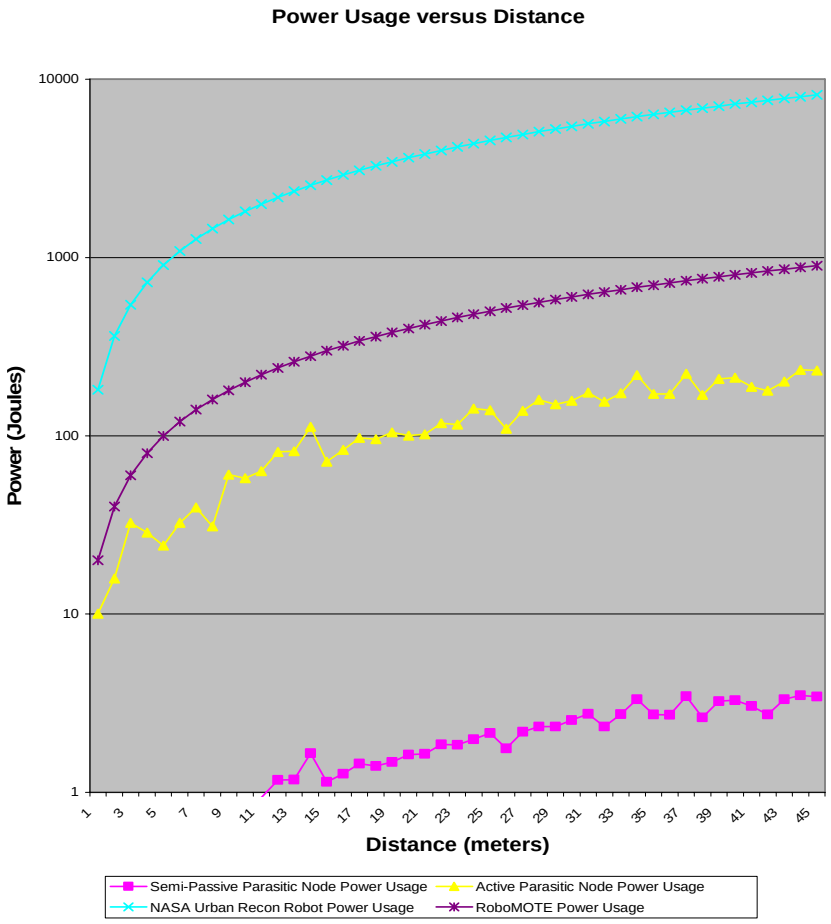

Fig. 4. Energy consumed versus Distance traveled of parasitic and non-parasitic mobile devices. This graph clearly shows the energy usage of parasitically mobile nodes to be an order of magnitude less than that of even the most energy efficient standardly actuated robots.

Parasitic Mobility is an attempt to bring the navigational power of the NASA robot into a device the size and cost of the RoboMOTE. That is why these two projects were chosen as points of comparison for the power consumption of these new types of networks. Figure 4, shown above plots the calculated power consumption of the RoboMOTE and the NASA robot along with values observed from the software simulator based on the power rates from the prototype hardware further detailed in Chapter 4. The graph (Figure 4) shows that the power usage of the active node is an order of magnitude less than the RoboMOTE, an energy-efficient robot that uses standard actuation. The power usage of the semi-passive node is an additional order of magnitude less.

Maze Environment. The simulator was also used to simulate many real-world situations and sensor network deployments. Once such set of simulations involved designing a maze-like structure and having the hosts follow particular routes similar to people working in an office environment or vehicles obeying traffic regulations.

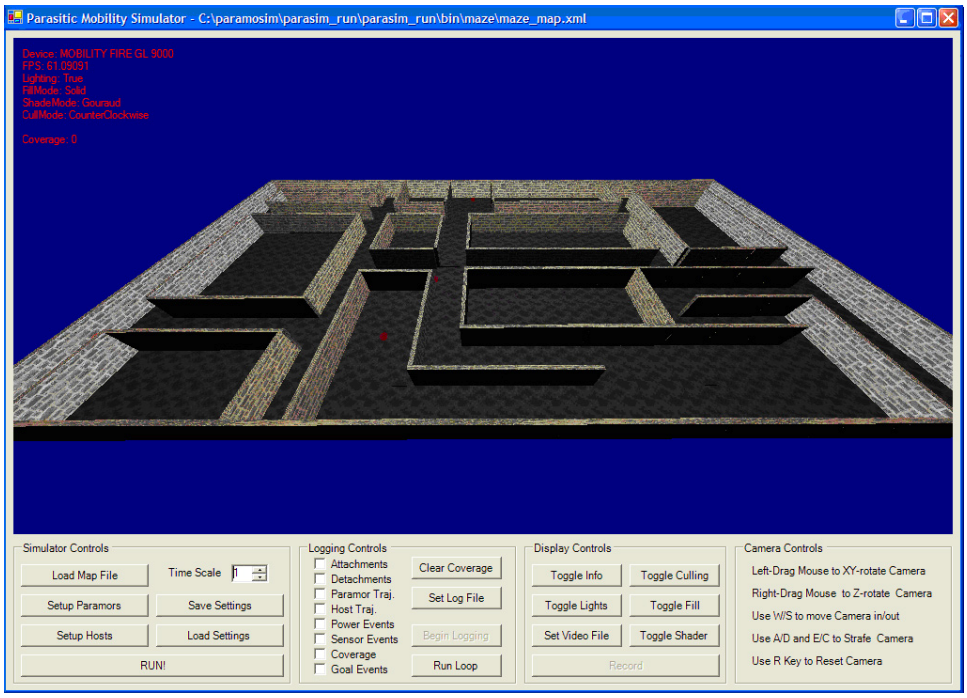

Fig. 5. The maze simulation executing.

Three different decision algorithms were tested for the parasitic node behavior: an omniscient algorithm where the node knows the whole map and can pick the best route to get to its destination, a distance algorithm where the node calculates improvements in distance while attached to a host, and a right-hand rule algorithm, a common map solving strategy, where the node always wants to go right at every intersection.

Table 1. Results of maze simulation comparing maze solving algorithms

	Node Behavior 1 (distance algorithm)	**Node Behavior 2 (right-hand rule)**	**Node Behavior 3 (path omniscient)**
Total Time	230	600	210
Attached Time	100	300	80
Number of Hops	22	30	12

As expected, the omniscient behavior performed the best, because these nodes will always take the same path, and the number of decision points they will pass is fixed, making their trajectory a probabilistic function of the number of decision points and possible directions that a host can turn at each of these points. It is further helped by the fact that hosts in the test do not turn around, cutting down the number of wrongful directions.

The distance algorithm behavior did not perform significantly worse than the omniscient behavior. This is probably due to the fact that the maze is relatively simple, and the distancing algorithm can easily resolve the situation and more-or-less find the same path as the omniscient node.

Of the three behaviors, the right-hand-rule is the only one that had the potential to get far off track. By the nature of the pattern resulting from this type of behavior, it can take a long time to reach the destination, but it is guaranteed and the coverage of the area is procedural and predictable, which may be desirable for some applications.

In the maze environment, the host velocity and other parameters of the host's behavior have less bearing on the time it takes for the node to reach a destination than the decision algorithms of the node and information that the node has which can help identify the best hosts to ride. In many real-world situations, it may be possible to know many details about the paths that the hosts will take and this information can be fed into the nodes at release, or communicated from node to node as the details are discovered.

The maze simulation is a first attempt to model a particular environment from a geographic and host-flow perspective. The software simulator will be further developed to add more environmental-specific parameters such as probability of a successful host attachment, multiple types of hosts, and areas where a detachment causes the node to become lost.

3.3 Software Simulation Conclusion

The software simulator has proven an invaluable tool with which to experiment with ideas and specific algorithms for implementing parasitic mobility. The experiences with the software simulation and the presence of these types of systems in nature have presented a favorable proof of concept for this type of mobility. The following chapter describes the design and implementation of an actual parasitically mobile sensor network, which takes into account the quantitative and qualitative results from the software simulator.

4 Hardware System

4.1 Electronics Design

A hardware system comprised of electronic nodes equipped with all the necessary elements to implement the specific ideas introduced through the software simulation was designed and built. The nodes required processing, communication, data storage, a location system, a suite of sensors, and an onboard rechargeable power source. The electronics should also facilitate experimentation with different types of attachment and detachment mechanisms.

The electronics were designed as small as could be easily built by hand using easy to obtain components. The design, shown below in Figure 6, is based on stackable layers, each roughly 1 square inch in size. When four layers are stacked, they are less than 1 inch high. More details on the mechanical specifications of the node hardware are given following the breakdown of the individual layers.

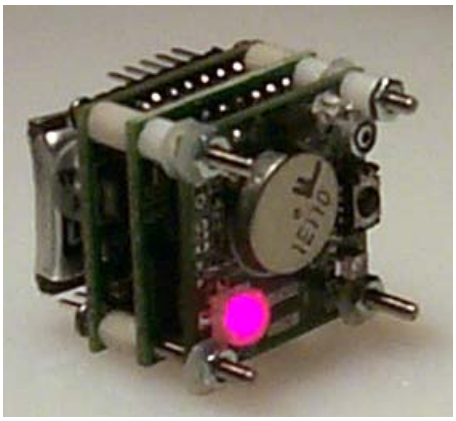

Fig. 6. Node hardware shown with 3 of the 4 available layers in place.

Power Module. The power module is equipped with a small 3.5 gram Lithium Polymer battery, which can provide 145 mAh at a voltage of around 3.7 V and can discharge at a rate of up to 1 amp. The power module circuit board contains a charge controller to control recharging the battery from any available power source, a hot-swap switch to power the circuit from an external power source (if available) while charging, a gas gauge chip to tell the processor the expected time left before needing a recharge, and an efficient voltage regulator with a battery supervisor circuit.
Processing and Communication Module. The processing module is equipped with a Silicon Labs microcontroller that can run up to 25 MHz using its internal oscillator or go into a low power mode and run off an external 32 kHz crystal. The communication is enabled with a Bluetooth module running an embedded Bluetooth stack that supports Scatternet formations for communication with other nodes. Lastly, this layer contains a 16Mb flash memory for storage of sensor data and map information.
Sensor and Control Module. The sensor layer is equipped with a 2-axis, 2G accelerometer, a microphone, an active infrared proximity sensor, a temperature sensor, a light sensor, an RGB LED, a pager motor to shake off of hosts, and a motor controller for the active attachment mechanism.
GPS Layer. The final layer was a GPS layer designed around a recent embedded GPS chipset, the FSOncore from Motorola. Although this design was tested with sample chips and worked well, even indoors with the assisted mode, the quantity of chips necessary to perform the application test was not yet available. An alternate location system was designed using the Bluetooth radios. This system consisted of deploying enough Bluetooth beacons to cover the test area. By simply checking which beacons were in range, the nodes could tell their position with a resolution depending on how close together the beacons were placed. The maximum resolution of this system was shown to be roughly 3-5 meters with the beacons placed around 5 meters apart. For the actual deployment the beacons were placed farther apart to simplify the environment. This worked well as a temporary solution until the GPS chips could be delivered.

The electronics were then mounted onto one of several different mechanical structures. The first was a hopping mechanism that created an active node which can hop on and off of a nearby host, and attach with a hook. [26]

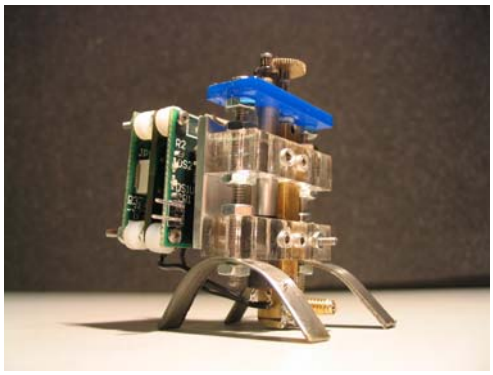

Fig. 7. Active Node, nicknamed the ParaHop and its hopping actuator.

The next two mechanical structures were created by encasing the electronics in a plastic sphere. To make a semi-active or passive node out of this structure, the sphere is then covered with sticky silicone that creates a bur-like device that sticks to anything that comes in contact with it. The semi-passive version can shake itself off once stuck whereas the passive version waits for the bond to wear out.

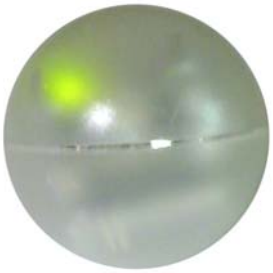

Fig. 8. Electronics encased in 1 inch diameter plastic sphere.

The sphere can be further enhanced by adding a sticker or other method of instruction for a host to pick it up. This symbolizes the value-added or attraction method of parasitic mobility as described in Section 2. Although they conferred no direct benefit to their hosts, people were amused and intrigued enough to pick up these artifacts.

4.2 Test Run

Ten such value-added nodes were released into a high traffic hallway of the MIT Media Lab building. All ten were spherical nodes that were labeled with instructions to pick them up and to put them down if they shook. The LED also blinked various colors whether it wanted to be picked up, was carried and mobile, or preferred to be stationary. Using these simple methods for attachment and detachment, the nodes were lifted up, carried, and placed down many times throughout the test allowing the basic concept of parasitic mobility to be examined.

The ten nodes were given specific goals to try and achieve. Six of the nodes were programmed to reach different geographic destinations. The remaining four nodes were given specific environmental conditions to look for. Although all ten nodes collected sensor data throughout their tasks, these nodes were programmed with conditions that will cause them to detach upon detecting the desired phenomena. When this happens, the node will try and stay at this location for a pre-determined amount of time or until the sensor condition disappears. If the node is picked up from this state or from a state where it has reached its geographic goal, it will immediately attempt to detach until its allotted time for the goal or sensor point of interest has elapsed.

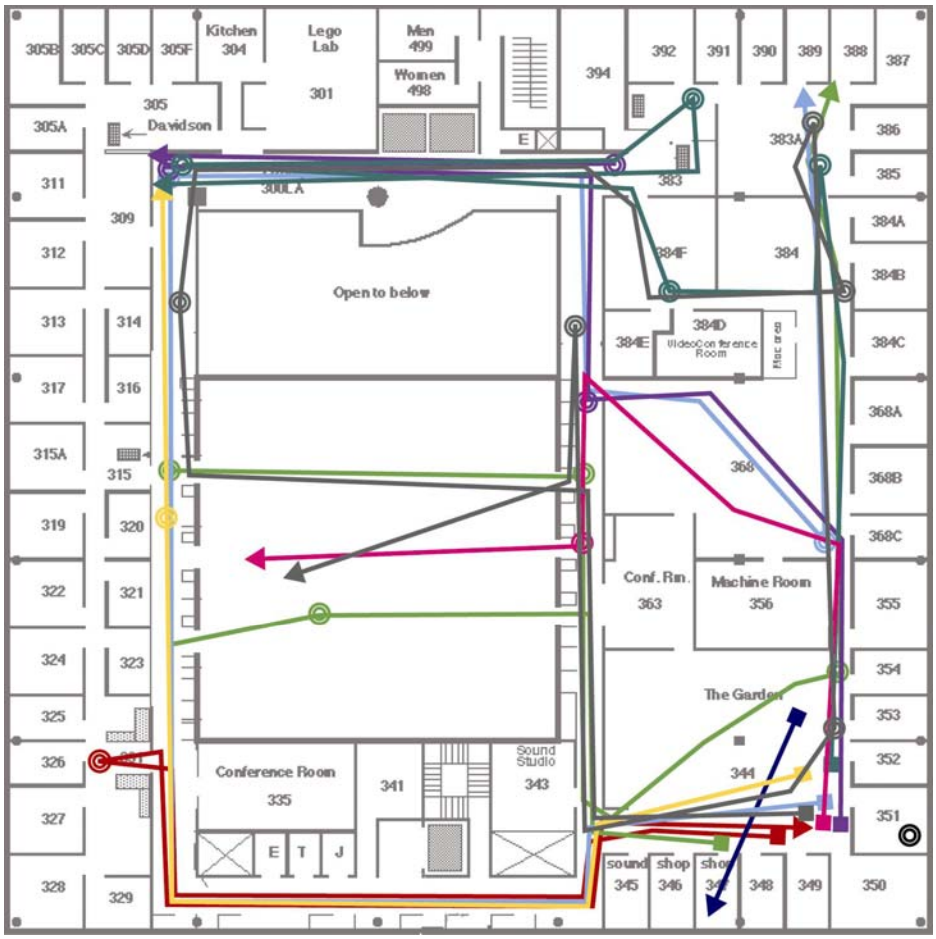

Fig. 9. The trajectories of all 10 value-added style nodes across the test area (third floor of The Media Lab). The squares indicate the deployment point, circles indicate a detachment and reattachment, and triangles indicate their final location. This illustrates that the nodes covered all the public spaces in the test area, indicating that parasitic mobility (with the necessary node redundancy) can be used to deploy nodes to many locations and cover wide regions.

As an example, we analyze the trajectory (Figure 10) of Node #7 in more depth. Node 7 found a location in the top left corner that it considered interesting, which in this case contained a bright light source, as the node was instructed to seek bright illumination. The data recovered from the node showed that this was indeed the case, but that the lighting condition eventually dipped and became uninteresting. Especially how the node winds up being oriented becomes important, since they are spherical, for the light sensor. According to the data from the node, it was picked up but then noted a lighting condition that was below the threshold for it to consider it interesting. In fact, it was far below the level that it had seen before it was picked up. This indicated that the person who picked it up must have covered the light sensor. The node therefore thought that the location was undesirable and did not ask to be put back down. When it returned the second time, it was better oriented and noticed the lighting condition, detached, and remained there for the rest of the test. This took only 26 minutes to happen, meaning that the node was able to fend off hosts picking it up for the remaining hour and a half of high traffic time. The data collected shows 4 attempted pick ups, all successfully ending in the node being placed back in the same location.

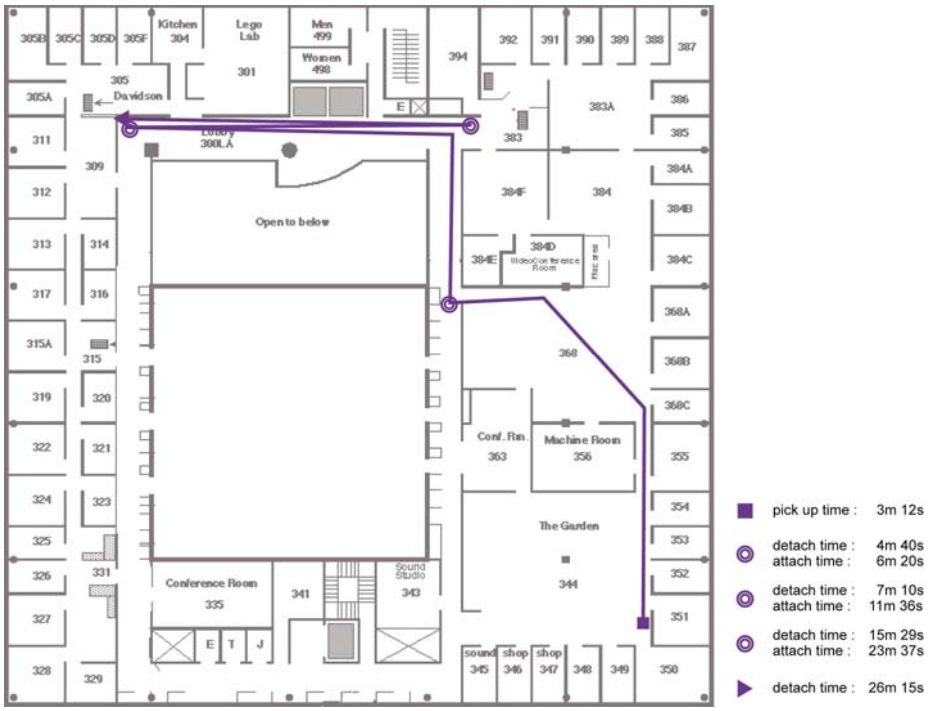

Fig. 10. Specific Trajectory data for Node #7.

Node #7 found a light source and remained there for 4 attempted attachments. This can be seen by looking at the graphs in Figures 11, 12, and 13. The light sensor shows two areas where the light value dropped very low, indicating a bright area. These two areas roughly match the times that the node was in zone 16. During the second stay in zone 16, there were several level shifts in the accelerometer data. These indicate a change in orientation, and the activity in between these levels is due to the node being picked up and vibrating to be put back down because the node liked where it was.

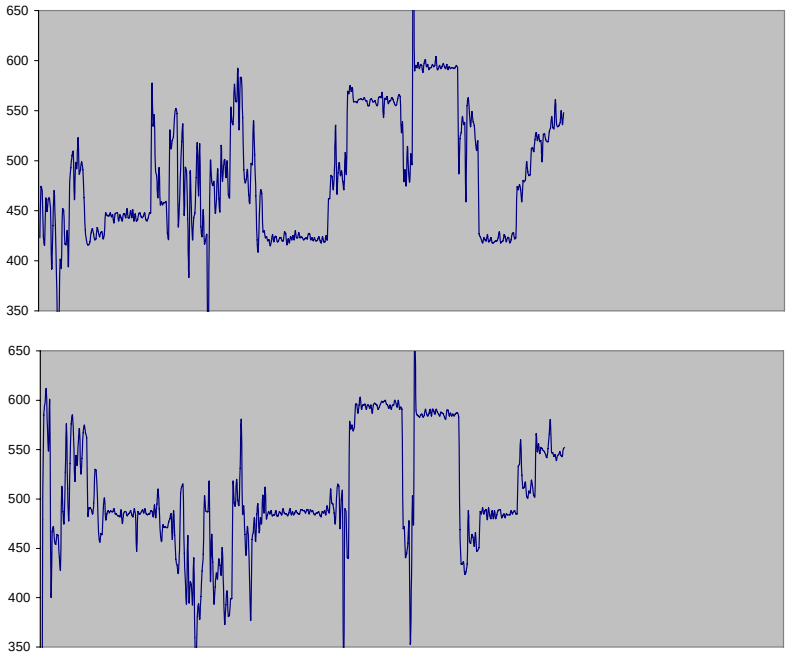

Fig. 11. Accelerometer (X-Axis top, Y-Axis Bottom) data collected from Node #7

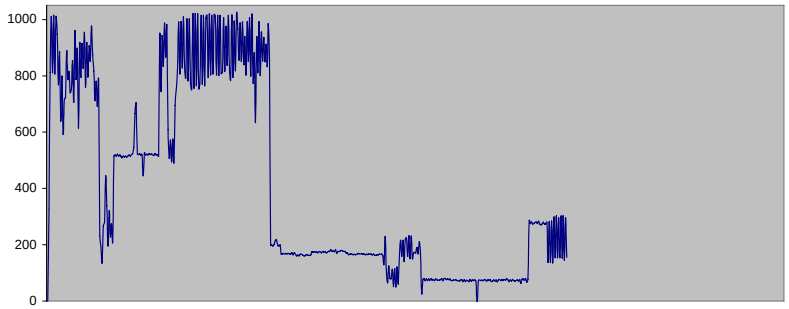

Fig. 12. Light Sensor data collected from Node #7

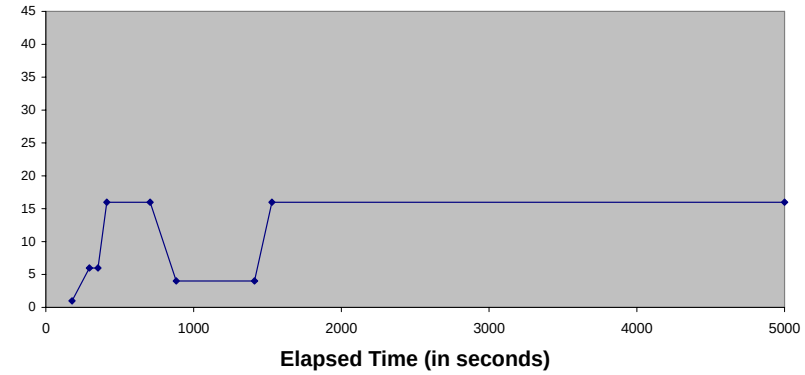

Fig. 13. Location data collected from Node #7

4.3 Hardware Wrap-up

The hardware design proved adequate for experimentation with the concept of value-added/attraction parasitic mobility. The specific prototype used for this test was mechanically robust and none of the nodes stopped working due to mechanical stress. The firmware also performed well and there were no unexpected firmware crashes or missed or mangled data.

One issue was that the sampling rate of the sensors was too slow. It was kept low while testing to simplify development, testing, and debugging, as well as to allow the flash memory to last for the entire test. As the development progresses, the sampling rate will most likely increase and become adaptive. The sampling rate at the time of this test was too slow to pull features out of the audio other than the net amplitude.

One of the major successes of the hardware design was the power supply and power management systems that each node incorporates. The battery lasted as long, if not longer, than expected considering the additional data logging and communication requirements of running such a test. Further development on a clever wakeup system and adaptive sensing [15] can improve this considerably.

The size of the nodes is currently too big to attach to a human or an animal without its knowing. The current size is more than adequate, however, for applications where the hosts are vehicles. This study explored simple proof-of-concept active attachment mechanisms; actual deployed systems would require more work on the host detection, self-righting enclosures, and the attachment and launching mechanism.

The location system, while adequate for this test, was not accurate enough for fine-grained indoor tracking. It would also need to refresh faster and more consistently. Many indoor location systems are under active development [27], hence this situation is rapidly evolving.

5 Conclusion

Through the work described in this paper, a new field of research has emerged. This field sits at the crossroads of distributed sensor networks, mobile systems, pervasive computing, and power harvesting.

Through simulation and initial tests, we have illustrated that it is possible to develop a parasitically mobile sensor network. Our results indicate that they can, in some ways, perform as well as standard robotic mobile sensor networks, but with huge potential savings with regards to power consumption, node complexity, and general robustness due to their relative simplicity. This indicates that it's possible to use parasitic mobility to distribute components of many types of pervasive computing systems throughout a physical space.

More detail is available, including all the collected data, higher resolution images, source code, schematics, and videos in the associated graduate thesis [26] and on the Parasitic Mobility website at http://www.media.mit.edu/resenv/paramor.

5.2 Possible Applications of Parasitic Mobility

In certain environments, parasitic mobility can be used as a replacement for standard mobility for dense, distributed sensor systems. Systems of this sort include applications to sense toxic or hostile areas requiring sensor deployment at a safe distance, dynamically reconfigurable systems such as weather monitoring sensors that need to follow the relative phenomena, and systems where the accuracy of node deployment is minimal, such as for nodes being released in water or from a vehicle.

Going further, parasitic mobility can possibly lead to applications that can only be done (or are better done) with parasitic mobility than standard mobility. Any example where the host behavior is part of what is desired to be monitored would fit this category. In these systems, parasitically mobile nodes would attach to their subjects and would always be at the points of interest.

One application that would be interesting to explore is the idea of a rating system based on breadcrumb trails. Essentially, the parasitic nodes would attach to hosts and pool up in spots of high traffic. These points can propagate through the system and provide information on the popularity of certain pathways and locations. Another similar application would be as a test bed for the spread of entities such as viruses. With a system like this you get data from the viewpoint of the contagion.

Parasitic Mobility can also be used to provide site-specific pervasive technologies. An example would be a system for urban safety where the devices know of areas where crime is abundant and wait at the edges of these locations. When someone enters this area they attach and ride through the dangerous area. The devices can sense an attack to its host and can call for help. If nothing happens and the host leaves the area, the device will detach and wait for the next host.

5.3 Future Work

Our hardware can be further perfected, e.g., a first step would be to increase the performance of the system by increasing the sample rate of the sensors, the onboard processing power, and the resolution and refresh rate of the location system. By increasing the node's capabilities, it will be possible to give the nodes more information about the environment such as onboard databases of map information.

Hooking this research up with actual power harvesting could be a natural fit, allowing self-maintaining, perpetual systems to be developed. These systems can harvest the power from their environment (taking inspiration from the tick, which harvests chemical energy from its host) or from forces acted upon the nodes, such as when they are in the

attached state. Smarter power management can be developed, as well as power adaptive sensing, to improve battery conservation.

Also, adding more distributed, node-node communication to the test system would open up some new venues for research. By collaboration, the sensor nodes could optimize their mobility and detachment and attachment algorithms, or maximize network coverage. Such communication could allow nodes to tell other nodes about host-rich locations and other information about the environment.

More experimentation with new types of attachment and detachment mechanisms could lead to new applications of parasitic mobility, e.g. attaching to vehicles. Also, embedding sensor nodes into everyday objects that migrate through a population is an interesting prospect for practical value-added symbiotic mobility. These directions can benefit from smaller nodes. Adding sensors (e.g. camera, motion sensor, and magnetic sensor) can also allow detection and attachment to a larger variety of hosts, as well as a wider range of sensing applications.

The biological inspirations for this research are all from natural systems that have developed over millions of years in synchronicity with their environment and are very niche-dependent. Further conceptual analysis is required to investigate how to map parasitic mobility to particular niches. This analysis would include looking at population density and traffic patterns of various hosts in real world environments. There would also need to be analysis of how effective the attachment mechanisms are with respect to certain hosts and environments. Such investigations would identify where and when parasitic mobility would be advantageous and identify many new applications.

5.4 Acknowledgements

This research was made possible thanks to the support of Motorola, Things That Think Consortium, and other sponsors of the MIT Media Lab. The authors would like to acknowledge and thank William Kaiser of UCLA and David Reed of the MIT Media Lab for their helpful advice and involvement, Talia Dorsey for help with node fabrication, and all the “hosts” who happened to be in the Media Lab building during the test deployments.

References

1. Meguerdichian, S., S. Slijepcevic, V. Karayan, and M. Potkonjak. Localized algorithms in wireless ad-hoc networks: location discovery and sensor exposure. In MOBIHOC 2001. Proceedings of the 2001 ACM International Symposium on Mobile Ad Hoc Networking and Computing. 2001. Long Beach, CA, USA: ACM. p. 106.
2. Estrin, Deborah and Cerpa, Alberto. ASCENT: Adaptive Self-Configuring sEnsor Networks Topologies. In IEEE Transactions on Mobilke Comuting Secial Issue on Mission-Oriented Sensor Networks, Vol. 3, No. 3, July-September 2004.
3. Grabowski, R., L. E. Navarro-Serment, and P. K. Khosla. Small is beautiful: an army of small robots. Scientific American (International Edition), 2003. 289(5): p. 42.
4. Howard, A., M. J. Mataric, and G. S. Sukhatme. An incremental self-deployment algorithm for mobile sensor networks. Autonomous Robots, 2002. 13(2): p. 113.
5. LaMarca, A., W. Brunette, D. Koizumi, M. Lease, S. B. Sigurdsson, K. Sikorski, D. Fox, and G. Borriello. Making sensor networks practical with robots. In Pervasive Computing. First International Conference, Pervasive 2002. Proceedings (Lecture Notes in Computer Science Vol.2414). 2002. Zurich, Switzerland: Springer-Verlag. p. 152.
6. Sinha, A. and A. Chandrakasan. Dynamic power management in wireless sensor networks. IEEE Design & Test of Computers, 2001. 18(2): p. 62.
7. Rahimi, M., H. Shah, G. S. Sukhatme, J. Heideman, and D. Estrin. Studying the feasibility of energy harvesting in a mobile sensor network. In 2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422). 2003. Taipei, Taiwan: IEEE. p. 19.
8. Chee-Yee, Chong and S. P. Kumar. Sensor networks: evolution, opportunities, and challenges. Proceedings of the IEEE, 2003. 91(8): p. 1247.

9. Kahn, J. M., R. H. Katz, and K. S. Pister. Next century challenges: mobile networking for "Smart Dust". In MobiCom'99. Proceedings of Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking. 1999. Seattle, WA, USA: ACM. p. 271.
10. Warneke, B., B. Atwood, and K. S. J. Pister. Smart dust mote forerunners. In Technical Digest. MEMS 2001. 14th IEEE International Conference on Micro Electro Mechanical Systems (Cat. No.01CH37090). 2001. Interlaken, Switzerland: IEEE. p. 357.
11. Hill, Jason. Spec takes the next step toward the vision of true smart dust. 2003. http://www.jlhlabs.com/jhill_cs/
12. Brendan O'Flynn, et al, "The development of a Modular Miniaturised Platform for Wireless Sensor Networks," to appear in the Proc. of the Fourth International Conference on Information Processing in Sensor Networks (IPSN 05), Los Angeles, CA, April 25-27, 2005.
13. Paradiso, J.A. and Starner, T., "Energy Scavenging for Mobile and Wireless Electronics," IEEE Pervasive Computing, Vol. 4, No. 1, February 2005, pp. 18-27.
14. Rahimi, Mohammad, Richard Pon, William J. Kaiser, Gaurav S. Sukhatme, Deborah Estrin, and Mani Srivastava. Adaptive Sampling for Environmental Robots. In Proceedings of the 2004 IEEE International Conference on Robotics and Automation. New Orleans, LA. April 2004. pp. 3537-2544.
15. Benbasat, A.Y. and Paradiso, J.A., Design of a Real-Time Adaptive Power Optimal Sensor System. in the Proc. of the 2004 IEEE Sensors Conference, Vienna, Austria, October 24-27, 2004.
16. Pon, R., et al, "Networked Infomechanical Systems: A Mobile Wireless Sensor Network Platform," to appear in the Proc. of the Fourth International Conference on Information Processing in Sensor Networks (IPSN 05), Los Angeles, CA, April 25-27, 2005.
17. Houston, Kenneth M. and Kent R. Engebretson. The Intelligent Sonobuoy System - A Concept for Mapping of Target Fields. In the Proceedings of the Autonomous Vehicles and Mine Counter-Measures Symposium, April 4-7, 1995, Naval Post-Graduate School, Monterey, CA. pp 6-129.
18. Irish, James D., Walter Paul, J. N. Shaumeyer, Carl C. Gaither III, and John M. Borden. The Next-Generation Ocean Observing Buoy in Support of NASA's Earth Science Enterprise. Sea Technology, May 1999. (40): p. 37-43.
19. Kerzhanovich, Viktor V., James A. Cutts, and Jeffery L. Hall. Low-cost balloon missions to mars and venus. In European Space Agency, (Special Publication) ESA SP. 2003. p. 285.
20. Bush, Albert O., Jacqueline C. Fernandez, Gerald W. Esch, and J. Richard Seed. Parasitism: The Diversity and Ecology of Animal Parasites. 2001, Cambridge University Press: Cambridge, UK. p. 391-399.
21. Ibid., p. 160-196.
22. Ibid., p. 306-310.
23. Ibid., p. 6-9.
24. Matthies, L., Y. Xiong, R. Hogg, D. Zhu, A. Rankin, B. Kennedy, M. Hebert, R. Maclachlan, C. Won, T. Frost, G. Sukhatme, M. McHenry, and S. Goldberg. A portable, autonomous, urban reconnaissance robot. Robotics and Autonomous Systems, 2002. 40(2-3): p. 163.
25. Sibley, G. T., M. H. Rahimi, and G. S. Sukhatme. Robomote: a tiny mobile robot platform for large-scale ad-hoc sensor networks. In Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292). 2002. Washington, DC, USA: IEEE. p. 1143.
26. Laibowitz, Mathew. Parasitic Mobility for Sensate Media. Thesis, MIT Media Lab. August 2004.
27. Hightower, J. and G. Borriello. Location systems for ubiquitous computing. IEEE Computer, 2001. 34(8): p. 57.