Intelligent Fluid Infrastructure for Embedded Networks

Aman Kansal, Arun Somasundara, David Jea, Mani Srivastava and Deborah Estrin

***Abstract*—Computer networks have historically considered support for mobile devices as an extra overhead to be borne by the system. Recently however, researchers have proposed methods by which the network can take advantage of mobile components. We exploit mobility to develop a fluid infrastructure: mobile components are deliberately built into the system infrastructure for enabling specific functionality that is very hard to achieve using other methods. Built-in intelligence helps our system adapt to run time dynamics when pursuing pre-defined performance objectives. Our approach yields significant advantages for energy constrained systems, sparsely deployed networks, delay tolerant networks, and in security sensitive situations. We first show why our approach is advantageous in terms of network lifetime and data fidelity. Second, we present adaptive algorithms that are used to control mobility. Third, we design the communication protocol supporting a fluid infrastructure and long sleep durations on energy-constrained devices. Our algorithms are not based on abstract radio range models or idealized unobstructed environments but founded on real world behavior of wireless devices. We implement a prototype system in which infrastructure components move autonomously to carry out important networking tasks. The prototype is used to validate and evaluate our suggested mobility control methods.**

***Index Terms*—mobile router, sensor network, delay tolerant networking, low energy communication, self-deploying infrastructure**

I. INTRODUCTION

MOBILE devices have traditionally been supported in wireless networks for allowing users with portable devices [27]. These methods are geared towards interactive computers where users carry their computing devices with them. Newer research efforts, especially those in the past decade [1,2], have started focusing on proactive computers where most of the components of the system are embedded in the user's environment and work autonomously, such as sensor networks. They may carry out pre-assigned tasks or work by anticipating user needs at run time. The resource constraints and design considerations in such systems differ significantly compared to legacy systems because of the need for a heightened level of autonomy and deeply embedded operation. Mobility in such systems takes on a completely different role. Previous work has emphasized mobility outside the control of the system as far as the communications sub-layer is concerned. Mobility is either treated to be random, following partially predictable patterns or even deterministic but not controlled. This paper follows from previous work such as [4, 5, 6, 7, 8, 29], which made use of external mobility for improving network performance. However, we introduce the new element of deliberate control over mobility which leads to significant performance improvement. We incorporate mobile components into the networking infrastructure, making the infrastructure fluid and demonstrate how this yields practical solutions to some hard networking problems in sensor networks.

II. KEY CONTRIBUTIONS

This paper argues for incorporating mobile components into the networking infrastructure where such inclusion leads to significant performance advantages.

We first show what problems can be solved using mobility in proactive computing and motivate the need to add controllable mobile components into embedded wireless sensor networks. We contrast this deliberate mobility to previous research efforts that have attempted to exploit externally induced random or predictable mobility. While mobility may be used by several applications, the focus in this paper is on network layer functionality.

To illustrate real system parameters that enter the design process, we describe our prototype hardware, which uses mobile infrastructure and battery powered nodes. Our implementation is a sensor network with an autonomous mobile router. To the best of our knowledge this is the first prototype of an embedded network using an autonomous mobile networking device. Using real hardware for our test and evaluation ensures that our methods are directly usable and do not depend on any hidden idealistic assumptions.

Next, we present control primitives that the mobile networking components can use for autonomously managing their motion. These primitives allow the motion to be adjusted to run time dynamics in real time and adaptively improve with growing environmental learning. We outline several design considerations and tradeoffs that exist for mobility control in view of the resource and performance constraints specific to deeply embedded systems. To take a concrete example, we show how the trade-offs are balanced in our working system.
We then discuss the design of our communication protocol which enables support for mobile infrastructure and energy constrained nodes with long sleep durations. The communication protocol and sleep time decisions do not rely on idealized radio models such as the unit disc model, but work on the actual radio connectivity available in the specific deployment scenario. Again, we leave no hidden energy costs or latencies in our design by testing them on the prototype system.

The contributions described above serve as the outline for the paper from section IV to VIII. The next section describes prior work and explicates the novelty of our approach.

III. Related Work

Several research efforts have proved the feasibility of the technology required and illustrated the challenges in realizing the benefit of sensor networks for human productivity, for instance [1, 14, 33, 34].

One of the basic infrastructure service required for these system is that of networking the distributed components in the face of severe energy constraints. Several methods to reduce energy consumption have been considered before in the presence of static devices alone [3, 23, 24, 25, 26]. Our approach is orthogonal to these methods. Further, our approach also gives other advantages mentioned later.

More recently, the use of mobility has been explored for improving the networking facilities in the system. We classify these systems into three categories: random mobility, predictable mobility and controlled mobility. The network layer exploitation of random and predictable mobility has been studied before. Random mobility of all the nodes has been considered for improving data capacity in [30, 4, 6, 7]. However, in such cases the latency of data transfer cannot be bounded, and if the data is cleared from the buffer at the mobile agent, delivery itself is suspect. [5] used a random walk model for mobility of their mule and derived various parameters of interest theoretically. The use of a predictable mobile agent was considered in [8]. In [29], possibility of changing the trajectories of mobile hosts in a disconnected ad hoc network to transmit messages among hosts is explored. While some of the advantages of mobility can be realized in the above approaches, the full potential can be realized much more easily using a controlled mobile element. External mobile elements with appropriate trajectories may not be available in all situations. For current sensor network deployments such as those for ecological research and habitat monitoring [9, 10], no free mobile components are available which the system can use. Control over mobility is also necessary to get control over the reliability of data transfer and providing latency bounds. For a mission critical or performance sensitive system, the design cannot rely on the availability of mobile components from the deployment environment itself. We make the mobile components an integral apart of the infrastructure and give complete control over the mobility capabilities of these components to the infrastructure. In our work we exploit the new opportunities provided by this form of mobility and introduce various design challenges that exist for optimally utilizing it. Our system is thus categorized into the third category of systems that exploit mobility, i.e. systems using controlled mobility for networking, and to the best of our knowledge is the first in its class.

Networks with increased latency have been considered in Delay Tolerant Networking [11, 28] for interplanetary networks and for networks in regions where network penetration is low or unsustainable. Our system can be considered as a new class of delay tolerant networks aimed at embedded devices where battery resources are scarce.

.

IV. Advantages Due to Mobile Components

In this section we consider some of the advantages due to mobility for networking, instead of using multi-hop routing over the static wireless nodes.

The first problem that our approach helps to solve is increasing the system lifetime. In sensor networks, the sustainable lifetime is severely limited by the battery capacity of the embedded static sensor nodes, as replacement or recharging of batteries is not feasible, either because physically reaching the nodes is not possible, or because the cost of carrying out the operation is prohibitive. A mobile agent can help conserve energy at the static embedded nodes. It does this by reducing the number of packets transmitted by the embedded nodes. Consider a sensor network with static nodes only. Data from all nodes to the user location travels over a multi-hop wireless link with intermediate nodes relaying the data towards the destination. The radio is a major energy consumer in embedded sensor nodes [35]. The energy spent in relaying traffic could be saved if data were transmitted over fewer hops. This can be achieved by having a mobile device reach locations close to the sensor nodes such that data can be sent to the mobile device over one or two hops and this device can bring the data to the user. This device has the opportunity to return to base and recharge itself. But the embedded sensor nodes or any static infrastructure provided for these nodes, must operate solely on their battery supply, and saving energy at these nodes, we are able to increase the system lifetime.

To get a quantitative estimate, we consider a sample topology, shown in Fig 1. On the left is the topology with no mobile infrastructure. The user node where all data is received is labeled 'Base'. Paths from various nodes to the sink have hop counts between one and five. Now consider a fluid infrastructure along the dashed line as shown in Fig 1 on the right. The dashed line shows the path of the mobile device traversing the network. Now, all nodes can reach this infrastructure over one or two hop paths.

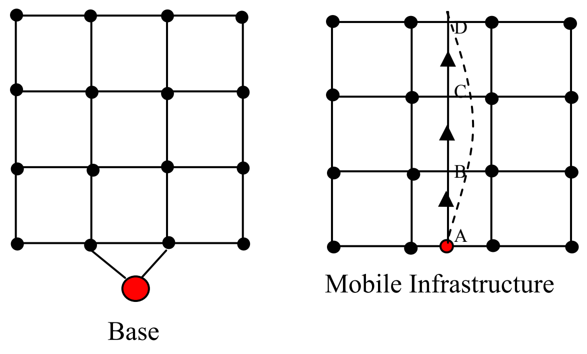

Figure 1. Sample network topology considered to simulate the advantage due to mobile infrastructure.

For getting an exact number of packets transmitted for communication from all the nodes to the sink, we assume a specific data gathering protocol to be running on the network: directed diffusion [12]. In this protocol an 'Interest' message is broadcast by the sink and propagated by the nodes which hear it. All nodes respond with data to this interest message via the nodes from which they heard the interest. Diffusion specific parameters were: interest transmission period = 60s, interest expiry timer = 75s, sample generation period = 5s, samples per data packet = 4.

We simulated the above topologies in TOSSIM [13] with diffusion. To have mobility for the topology with fluid infrastructure, the mobile device was simulated to be present at four locations, marked A to D in the figure, one after the other. Only one of them was active at any given time. It would stay at a location for 60sec. After spending time at D, the mobile device starts from A. The exact communication protocol changes with the presence of the mobile device and we discuss that in a later section. Figure 2 presents the total number of packets transmitted by the nodes at a particular hop from the base in the multihop case. This is compared with the packets transmitted by the corresponding nodes (nodes at the same location as in multihop) in our case.

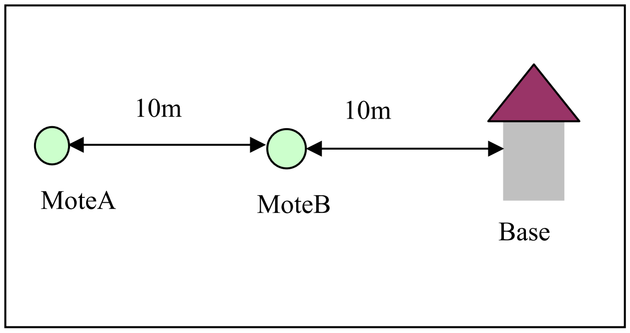

Figure 3. Example topology to compare transmission times for multihop routing and mobile router.

.

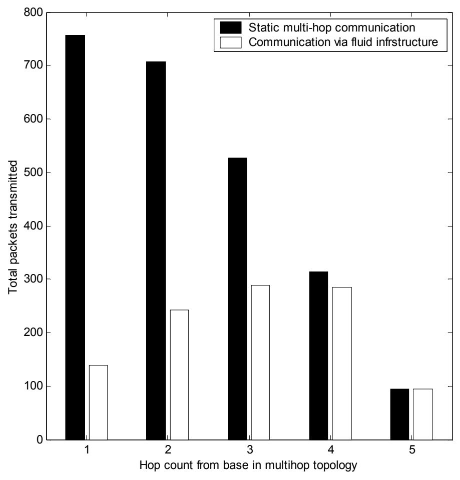

Figure 2. Comparing number of packets transmitted in multi-hop routing and when using mobile router.

The simulation was conducted for 16 minutes. Clearly, the number of packets transmitted when the mobile device is used is significantly lower. As the energy consumed by a node increases with the number of packets transmitted, the resultant gains in system lifetime is substantial.

A second advantage comes in the form of data fidelity. It is well known that the probability of error increases with increasing number of hops that a data packet has to travel. If we reduce the number of hops, this immediately reduces the probability of error. This not only increases the quality of data
received but also further reduces the energy spent at the static nodes by reducing the retransmissions required due to errors.

A third advantage due to the mobile router is that data rate can be increased, or latency can be reduced. The reduction in latency may seem counter intuitive as one expects data to travel faster over a wireless link than over a physical mobile device. However, the addition of new resources increases the capacity. The following example illustrates this.

Consider a typical sensor node, the mica2 mote [14], which we also use in our prototype. Suppose we have a two hop mote network as shown in Figure 3 with two motes, labeled A and B. The mote radio range is taken to be 10m for this example; actual range varies from 1m to as high as 20m with environment. The mote has 512KB of memory. Suppose we have stored sensor data worth 500KB on each mote, say from sensor readings. The mote radio has a baud rate of 38.4Kbps according to the datasheets [14]. Let us try to give advantage to the multi-hop routing case and assume that an ideal MAC protocol is available. So no bandwidth is wasted in collisions. Also assume that the channel does not have any errors; this gives further advantage to the multi-hop case. Now, the minimum time taken to transfer data from both the motes to the base station will be = time to transfer A's data from A to B and then from B to base + time taken to transfer B's data to base. With the mote radio bandwidth mentioned above, this comes to 312.5 seconds. Now consider that a mobile device is used. To ensure that we are not using any unrealistic assumptions in this case, we will use only the characteristics of our implemented mobile router, described in section V. Our prototype mobile router has a maximum speed of 388cm/second. Assume we use it at 200cm/sec. It has an 802.11 wireless connection with the base. We will assume that this connection is used only when the router is physically present at the base and not from a distance, which could have given further benefits to the mobile router. Suppose the mobile router is stationed at the base initially. Then the time taken to collect all the data from A to B is = time to move to A + time to transfer data from A over the mote radio + time to move to B + time to transfer data from B + time to move back to base + time to transfer data to user over 802.11 radio. The total time, using the mote radio parameters and the characteristics of our mobile router, comes to 228.8 seconds. Note that this is 83.7 seconds less (26.7% lower) than the case when all the data was sent wirelessly. Thus, the bandwidth of the network has been increased. The data-rate increase would be even more dramatic when the multi-hop network spans several hops and each mote is connected to several neighbors instead of just one. Considering MAC collisions would waste some

.

bandwidth in the multi-hop routing case and the increased probability of errors due to traversing multiple hops would cause more retransmissions than in the mobile router case, the mobile router has a definite win. Experiments show that the manufacturer specified radio data-rate of 38.4Kbps is actually not observed and achieved rates are lower. As the number of radio channel uses is reduced with the mobile router, this gives further advantage to the mobile router. These benefits are a result of the extra data carrying capacity added to the system in the form of physically carrying the data in a mobile node instead of wirelessly. The extra 802.11 radio for communicating with the base is not important for the increase in data capacity. Even if the same mote radio is used for transferring data to the base from the mobile router, the advantage of the mobile component occurs with 4 or more hops. However, we feel having a higher bandwidth connection between the mobile component and the base is likely in most systems as the mobile device is not energy constrained and a higher bandwidth radio can be used.

Of course, the data-rate advantage is easily obtainable only when the motion of the networking device is controlled by the system and no such advantage may be observed when external mobile entities, moving randomly or predictably, are used.

A fourth advantage due to the mobile routing device is that sparse and disconnected networks can be handled. This is useful in several situations. One is that the static devices can reduce their transmission ranges to the lowest value required to reach the mobile infrastructure, even if it fragments the static network topology. This will save transmit energy at the sensor nodes. Another possibility occurs when the sensor network is sensing a phenomenon for which the sensors need to be deployed only at a low density which is not enough to guarantee connectivity. Relationships between sensing range and communication range derived in [24], show that a communication range double that of the sensing range is required for ensuring connectivity. Thus in situations where sensing range is larger than the radio range several relay nodes may be needed just for connectivity. If a mobile component is available, such relay nodes can be saved and communication energy spent by the embedded portion of the network can be drastically reduced. A third case arises when the environment in which the network is deployed has dense occlusions to wireless communication and a reliable multihop topology cannot be established. Here, a mobile device can position itself to reach within communication range of the embedded devices.

There are other advantages of a mobile component apart from communications. It was shown in [36] that time synchronization error increases with increasing number of hops between two nodes. Since in our design, the number of hops between the base and the embedded sensors is reduced, much finer time synchronization is possible than in a multi-hop case. Security can be enhanced as the data does not traverse multiple hops across potentially compromised nodes. Another use of controlled mobility very recently proposed for calibrating a localization system is [22]. Mobile components can also support other system activities such as delivering required resources [15, 32]. We consider only networking activities in this paper.

With the above intuition to motivate the benefits of a fluid infrastructure, we design a working system with an autonomous mobile data gathering device. Since the infrastructure must operate autonomously, we need methods to control the motion of the mobile routing device. Further, power control, topology management and communication strategies are significantly impacted by the fluid nature of the infrastructure. Before presenting our design of the exact algorithms for handling these issues, we present our prototype system on which we experimentally illustrate and verify our proposals.

V. System Architecture

The system consists of two classes of devices:

1. The static embedded sensor nodes, called motes, which are Berkeley motes [14] in our prototype, and
2. A mobile router, which is our fluid infrastructure connecting the motes to the base. The base could be an autonomous controller or a human user based system.

A picture of the prototype system is shown in Figure 4, with the various hardware components marked. The motes are an off the shelf sensor node. We describe our construction of the mobile router here. The mobile router is composed of a traction platform, a processor board and a mote. The three components are described below:

A. *Traction Platform and Control Interface*

We use a rugged terrain robot, the Packbot [16] as our mobile platform, with our own control interface. Packbot is a robot developed by iRobot [17] as an unmanned ground vehicle (UGV).

.

It is suitable for our project because of its belt based traction system allows it to easily maneuver over rocks and rubble, climb up stairs, and navigate narrow, twisting passages. The traction system is geared for our final objective of running it in natural environments such as forests and ecosystem preserves.

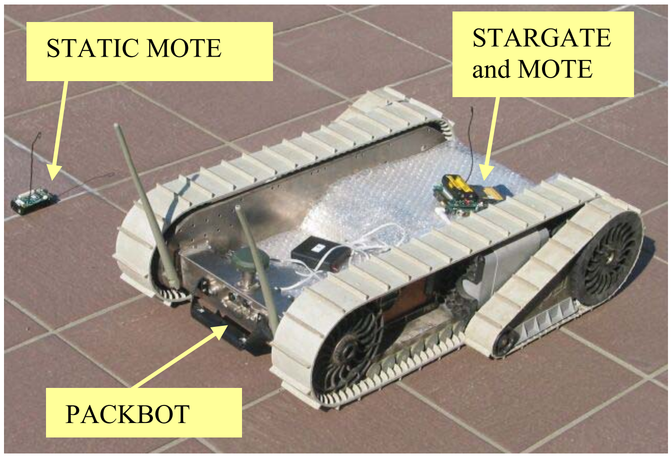

Figure 4. Prototyped system seen in action. The mobile router moves to the static devices for data collection.

The Packbot is commanded by sending it appropriate command packets on its 802.11 interface. The driver for sending the robot specific messages has been integrated into a generic development environment, which we call the Simple Interface for Robots (SIR), to assist controlling the robot movement. The control commands are sent by the Stargate controller, described later.

SIR is still an ongoing application interface library. The core design principle is "Simple to use and Easy to integrate". Unlike the complicated proprietary interfaces that usually come with robots, SIR focuses mainly on controlling the mobility and reduces the API to two functions, robotAdd() and robotAct(). It hides the low level details but still ensures enough functionality. Currently SIR runs on two platforms, Linux and TinyOS [19]. It also supports two different robots which include Packbot and Amigobot [18].

To operate the Packbot using SIR, specific instructions determined from the robot capabilities are provided: moving forward or backward, rotating clockwise or counterclockwise, and flipping. The instructions are specified with parameters required for it. For example, the SIR command robotAct(id, 'f', 30) asks robot earlier added with identification ‘id’ to "move forward" with parameter "30" which represents the specified speed in cm/sec. Combining the above actions with timer, the moving distance and turn angle can be decided, allowing complete 2-dimensional navigation of the robot. For user convenience, advanced abstracted functions wrapped as robotAct(id, 'm', 30, 1.0, 1.0) are also provided. This command will cause the robot to move to coordinate (1m, 1m) in 2D space.

B. Processing Platform

The processing platform added to the robot is a Stargate [20] node which is a xScale processor based computing device, running linux. An 802.11 card is added to it for communicating with the base and for sending control packets to the robot’s wireless interface. A mote is attached to the Stargate for using the mote radio to communicate with the static motes in the network. A block diagram of the software executed on the Stargate is given in Figure 5. The complete localization and navigation component for arbitrary environments is not yet implemented and we use a simple navigation for the current experiments, as described with our mobility control algorithms, in Section VI. The firmware implemented on the mote makes it act as a mote radio interface for the Stargate. This processing platform controls the robot, collects data from the mote and executes our intelligent motion control algorithms to help improve the data collection process.

.

C. *Mote*

The mote is used as network interface to the static nodes which communicate over the mote radio. It relays any data received from its radio to the Stargate over the serial port connection, and transmits any packets received from the Stargate to its radio. The mote can also be used as a data transfer interface to the base, but since radio energy is not a significant portion of the Packbot's battery consumption, a higher rate radio has been used (Power consumption of the Packbot is 60W on an average, while the Stargate processor board with the 802.11 card and mote together take only 1.7W).

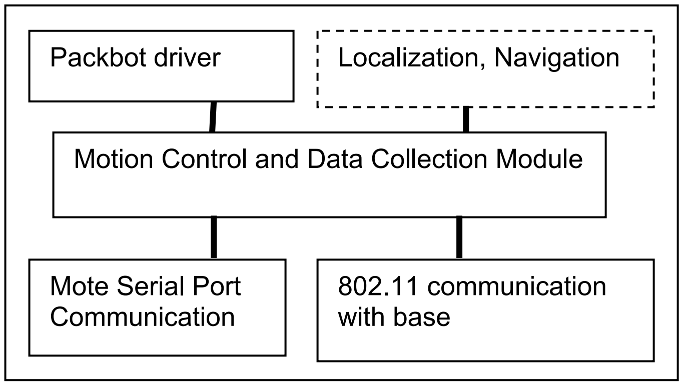
Packbot driver
Localization, Navigation
Motion Control and Data Collection Module
Mote Serial Port Communication
802.11 communication with base

Figure 5. Software architecture of prototyped mobile router.

VI. Adaptive Motion Control

The motion control algorithms used by the mobile components will depend on two factors – the constraints on the mobility imposed by the terrain and the data collection performance parameters which are important for the higher layer application or user collecting the data.

Consider first the mobility constraints placed by the terrain. The terrain may be such that the mobile device can move to any node directly. The mobile device may chose to visit each node or may visit a small subset of the nodes and the remaining nodes transfer their data to this visited subset using multi-hop routing. Various trade-offs in the energy used by static devices and the time taken by the mobile device to complete its patrol of the network exist here. In other situations, the terrain may not allow free mobility to arbitrary locations. Our first application for the fluid infrastructure is a data gathering system for measuring physiological, environmental and ecological variables from sensors installed in a natural ecosystem. Installation of a fixed wired infrastructure is not feasible in this ecosystem, only compact wireless sensors can be embedded. In such terrain, both due to the dense vegetation restricting the mobility of the mobile device and to minimize disturbance to the natural habitat, we decided to move only along the trail passing through the ecosystem; this trail is also used by human users to collect data manually. Thus, we design our motion control algorithms given that the path traversed by the mobile agent is fixed.

The second issue requires various performance trade-offs to be tuned. The algorithm can be designed to achieve several objectives depending on the application priorities. First, the lifetime of the system may have to be maximized. In this case the mobile router should visit each node individually and come within the closest possible distance to it. The embedded energy constrained device should have to use the least amount of energy in transmitting its data. Second, the total amount of data collected could be an important parameter for certain applications. This happens when the static devices are generating data at a rapid rate and the mobile router must transfer the maximum possible data from the static devices to the base. The motion control algorithm would then try to discover the topology consisting of some multihop transfer and some transfer over the mobile infrastructure at an optimal speed where data-rate is maximized. In other applications latency may be a hard constraint. So a third performance metric to optimize for could be data transfer delay and the system would wish to maximize the amount of data collected within the given latency constraint. Fourth, the static devices may also have a finite buffer constraint and the mobile device would then need to collect the data before buffers overflow. This may mean that certain data has to be transmitted via multi-hop routing, or may require the mobile device to use higher speeds and more frequent traverses through the network than required just form latency constraints alone.

The latency and data-rate will also depend on the energy supply of the mobile router itself. The mobile router needs some time at the base to charge its battery. This time may not be a factor in latency if the data is not collected continuously. In this case, there would be some intervals of no activity, when the mobile router may recharge its full battery and operate unhindered when data is actively being collected. If however, data is continuously collected, the recharge time adds to the data latency.

.

We now address some of the design issues in planning the motion of the mobile device and determine speed control primitives for studying one of the trade-offs, that of latency sensitive data collection, in two scenarios – networks with potential for MAC level collisions among the static nodes and in disconnected sparse networks.

A. *Influence of Speed on Data Collection*

Before designing the motion control algorithms, we need to study the influence of the speed of mobile device on data collection. Is there an ideal speed for the mobile router? Intuitively, if at speed **s**, the mobile router can cover the trail in time **t**, and get **n** packets, then at speed **2s**, it can cover the trail twice in the same time **t**, getting **n/2** packets in each round, effectively getting **n** packets overall. We wished to verify this experimentally because if the system behaves in this manner, any speed can be used, depending on the other performance trade-offs such as time for one traversal of the network, data success rate, or energy recharge-time of the robot. The experiment, variations of which also served for detailed debugging and verification of the developed prototype software, was performed as follows. To test the influence of speed alone and to suppress the effect of other parameters which may affect the radio communication, we placed two motes such that at any point the mobile router will be in range with at most one node. The radio range of the motes was reduced for this to happen within a smaller open area. Thus packet loss will occur only due to the mobile router going out of range of the nodes, and we did not want to have any collisions between the transmissions from two nodes. Figure 6 shows the arrangement of the static nodes.

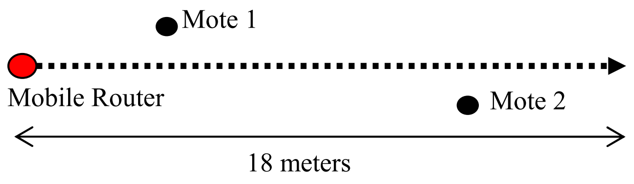

Figure 6. Experimental Topology for studying influence of speed.

The two nodes were continuously transmitting packets, so that the amount of data collected depends essentially on the time for which the mobile router is in range. The mobile router counts the number of received packets. We ran this experiment for seven different speeds (30 cm/s – 150 cm/s). The results at each speed are averaged over three rounds of the trail. The trail length was 18m, and a round is defined as the mobile router going 18m from the base and coming back the same distance. There was no packet reception in the reverse path (because eventually we expect to have a closed path as a trail). The results are plotted in Figure 7. As can be seen, the number of packets received per unit time at each speed is almost same. Hence we do not account for any influence of speed in collision free radio channel for our motion control algorithms.

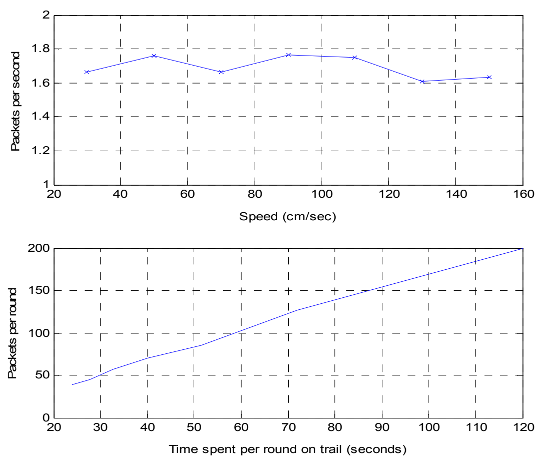

Figure 7. Data collection at different speeds. Data rate, (packets per unit time) does not strongly depend on speed.

.

B. Latency Sensitive Data Collection

Suppose a constraint on the maximum latency of the time for reporting sensor readings is specified. The maximum time that the mobile router can spend in completing one round across the network is then equal to this latency. The objective becomes maximizing the data collected in this round. Let the latency constraint specified be **T**. Given this acceptable round time, and knowing the length of the trail, the speed at which mobile router has to move can be decided. A naïve approach would be for the mobile router to move at this speed. But we propose methods to achieve better performance than this.

We make the following observations about practical network deployments which form the basis for our proposed improvements. The network is assumed to be randomly deployed. This will cause certain nodes to be in the range for longer duration, such as being very close to the path of the mobile router. Also, at certain locations, more than one node may be in range of the router, and communication bandwidth will be divided among them, suffering further due to MAC collisions. Further, the wireless channel may introduce more errors at particular nodes at certain times, causing data-rate to be reduced for these nodes.

The key insight is that the data collected can be increased if the mobile router moves slower when faced with a situation where more time is needed to collect the data and speeds up where high data-rates are achieved. We provide a run time algorithm for the mobile router to adaptively adjust its speeds in response to the run time dynamics of the network and improve the system performance.

To ensure practical applicability, we design an algorithm that is not based on the idealistic radio disc model. This model does not hold in practice as is observed in several experimental studies such as [31]. We also do not use geographic information such as used in [21] about the location of the static nodes as location may not be available or the error in location estimate may well be of the same order of magnitude as the radio range of the static devices; for instance the GPS localization error and the mote radio range are of the same order of magnitude. In fact our algorithm does not even require the router to know the number of static nodes on the path. This number may change based on new node installation, node damage or battery deaths, and will not affect our algorithm. The mobile router estimates the number of nodes based on node IDs in received data.

Suppose from the latency constraint **T** for round traversal time, the naïve speed calculated is **s**. At speed **2s**, the router can complete the round in time **T/2**. This leaves an extra time **T/2** for collecting data in regions where the wireless bandwidth is congested or data transmission is poor due to other reasons. In our scheme, the router adaptively learns where such regions occur, based solely on radio connectivity. We consider two approaches to manage the extra time spent in such locations. One is to stop at locations where static nodes are found waiting with data. The other is to slow in regions where data collection is moderately poor and stop in regions where data loss is severe. The precise methods to carry out this speed adaptation are described below. The communication specific details for both these methods are explained in the next section.

The first one is Stop to Collect Data (SCD), which stops at all the nodes, presented below.

Algorithm 1. ***Stop to collect data*** (SCD)

```
1. Round 1: Move at constant speed s and count the number of distinct node IDs heard in the
   data. Let this number be N. This initialization phase may be omitted if higher layer
   programs have already provided the number of static nodes.
2. Repeat in every round:
     a. Start moving at Speed 2s, transmitting INTEREST packet at periodic intervals.
        Listen for responses from static nodes.
     b. If a response is heard from a static node, stop, start a stop timeout timer set to
        (T/2)(1/N), and continue to receive data from this node.
     c. If timer expires and no response is heard from any other node, start moving at
        speed 2s. If a response is heard before timer expires, add (T/2)(1/N) to timer.
     d. Round will finish in time less than or equal to T. Now transfer collected data to
        base over the 802.11 link.
```

.

As can be noted from the algorithm, no external information about node locations or time synchronization are required. The algorithm does not deadlock regardless of packet loss or channel errors. We will compare the performance of this algorithm to that of moving at constant speed in the section on experimentation.

The second algorithm adaptively decides to slow down or stop depending on the communication characteristics of the deployed network. Hence, it needs to learn the data transfer characteristics to control its motion. To achieve this, the mobile router maintains state information. The processing platform on the router itself is not severely energy constrained and the required processing capability can be provided here. The Stargate processor board used in our prototype is easily capable of performing this computation. Every node includes the remaining number of data samples it wishes to transfer in every data packet it transmits after having received the INTEREST message from the mobile router. Based on the first and last packet heard from a static node, the router can calculate the percentage of total samples that were received. Percentage = (# unique samples got from a node) ÷ (# samples remaining in the 1st packet + (last packet time–1st packet time)÷Sampling period). The second term in the denominator sum corresponds to the samples collected in this round. The router maintains this percentage for every node ID that it hears from. Let the percentage of data samples (which is 4 times the number of packets, as we are sending 4 samples in each packet) successfully received from a node be denoted d_i. Choose two thresholds, ω_1 and ω_2, $(0 < \omega_1 < \omega_2 < 100)$ on the data delivery success percentage. Denote the class of nodes for which $d_i < \omega_1$ by N1 and the class of nodes for which $\omega_1 < d_i < \omega_2$ by N2. The mobile router will stop where it encounters a node of class N1, and will slow down when encounters a node of class N2. An encounter is defined as the event when a packet is first received from a node. The router will move at speed **2s** elsewhere. This method of taking a decision on encountering a node keeps the algorithm free of location information. Suppose the router found n1 nodes in class N1 and n2 nodes in class N2. Calculate:

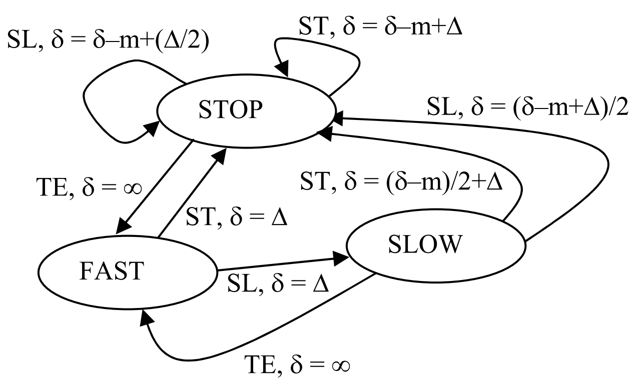

Figure 8. Traction-state diagram of the mobile router, with timer calculation.

$$\Delta = \frac{T/2}{n1 + (n2/2)}$$

As can be seen from the above expression, extra time given to a node of class N1 is double that of N2. This is because if the mobile router moving at speed **2s** stops for Δ, time lost is Δ, whereas if it slows down to speed **s** for Δ, time lost is Δ/2.

The router can be in three states, moving fast, moving slow or stopped. Define the following events:

1. SL: Encounter a node of class N2
2. ST: Encounter a node of class N1
3. TE: Current state timer expired.

Let M denote the timer. Let m denote the time elapsed since the timer was reset. Let δ denote the duration to which a timer is reset, δ can be different at every reset. With these events, the exact traction state of the mobile router itself can be modeled as shown in the state diagram in Figure 8. The exact durations to which the timer is reset corresponding to each event, are also shown. The transitions in the state diagram mean that on happening of an event (SL, ST, TE), the action to be taken is (setting δ and moving to a new state). For instance, when it is in SLOW state, and gets a SL event, it moves to the STOP state with new value of the timer (δ-m+Δ)/2. It is because if it goes to SLOW again (increased duration), it may happen that the node which requested the SL may get out of range by the time it gets its share of SLOW. The total time is divided by 2 as the time lost due to stopping is double that due to slowing. If a TE event is received, it reverts to FAST, and sets the timer δ to ∞ as it has to move at 2s speed till a new encounter.

.

The events that cause the router to transition between these states are based solely on the specific characteristics of network operation and no design time knowledge about the topology is needed. Like the previous algorithm this algorithm too does not fail in the presence of wireless losses. The precise methods used and the exact details of how the latency constraint is managed in the prototype implementation are presented in the algorithm specification below.

Algorithm 2. **Adaptive Speed Control** (ASC)

```
1. Round 1: Move at constant speed. Determine the sets N1 and N2 for this round.
2. Repeat in every round: If n1+n2=0, move at speed s, else:
      a. Set δ = ∞, and start moving at speed 2s, transmitting INTEREST packet at periodic
         intervals. Listen for responses from static nodes.
      b. If any of the events ST, SL or TE occurs change speed and timer value according to
         state diagram of Figure 8.
      c. Round will finish in time less than or equal to T. Now transfer collected data to
         base over the 802.11 link.
```

C. Latency Sensitive Communication in Sparse Networks

Another speed control problem arises in sparse networks. A sparse network is a network where the devices are too far apart to ensure connectedness of the induced topology graph and the graph is split up into multiple fragments, potentially as many fragments as the total number of nodes. The static devices are energy constrained and hence want to use a small radio range, which means that each device is in communication range of the mobile router only for a limited portion of the router's path. There exist transacts in the path where there are no nodes in range for the mobile router. Clearly, the data collected in a given latency constraint can be increased if the router could stop or move slower when connected to a static node and move faster where there is no node in range. We propose using the ASC algorithm mentioned earlier for this scenario too.

These motion control algorithms can be used with the vanilla directed diffusion [12] based communication protocol. However, there are several issues which arise due to the presence of the fluid infrastructure and further performance gains can be achieved, both in terms of data quality and lifetime of the network, if the communication protocols are optimized for the mobile router. We discuss these issues next.

VII. Communication Protocols for the Fluid Infrastructure

Directed diffusion [12] is a communication protocol designed for collecting data from a large number of sensor nodes. It is summarized here, since our method builds on this successfully tested scheme. The network is composed of sensor nodes, which are data sources and one or more data collection devices, called sinks. A sink expresses interest in particular sensor data satisfying some constraint. The interest packets are broadcast from sink periodically, containing the constraint on data and a time to live (TTL) field. The TTL field denotes the number of hops the interest will flow. The nodes which hear the interest store it in their interest cache, and rebroadcast it after decreasing the TTL field. As the interest flows through the nodes, it establishes reverse paths towards the sink (called Gradients), along which the data in response to the interest will be returned to the sink. Whenever a node has data, it will check its interest cache to see if this data matches any of the constraints received in an interest. If so, it sends the data to the node from which it heard the interest. This node will in turn see its interest cache, and the packet is relayed subsequently to the sink. To handle network dynamics and changing topologies, there is an expiration time for the interests. We have made certain changes to this basic scheme for adapting operation with the fluid infrastructure. In our system, the static motes are the sources, and the mobile router is the single sink.

The first observation is that the static nodes should send their data directly to the mobile router when possible, otherwise use multi-hop communication to reach the nearest node which can communicate with the mobile router. There is no design time knowledge of the radio propagation characteristics of the deployment environment and the mobile router cannot know a-priori what the maximum hop distance is between the router's trail and a node. Consider the scenario shown in Figure 9.

.

The solid lines represent which nodes are in communication range; the node locations shown are arbitrary. Node A will broadcast the interest heard from the mobile router, which will be heard at node B. This will trigger the node B to send data to node A, which will in turn relay it to the mobile router. It is possible that node B will have a direct connection to the mobile router after some time. If such is the case, we need to save the energy that node A spent in forwarding B's data. However, node A has no way of knowing if there is another node which can only reach the router through it. Our first modification to diffusion takes care of this. The first round of the mobile router is used to learn the connectivity of the mobile router. For every interest received, each node notes if the interest came from the mobile router or another static node. If a node hears an interest from the mobile router, it will not respond to or rebroadcast any subsequent interests from another static node. Such a node will only rebroadcast interests directly received from the router. Other nodes will respond to and rebroadcast every unique interest message received. A node which does not receive interest from the router, may receive interest messages from two different nodes on the router's trail. Diffusion takes care of this.

The second observation is that the mobile router will be in range of a source only for small time duration. There may be a duration for which the interest has not expired but the mobile router has moved out of range. The data transmitted in this duration will be lost. It may be important for the static node to know what data was successfully received. We use an acknowledgement-retransmit scheme to provide this. The mobile router sends acknowledgement to the nodes it receives data from. The node will send the next packet only after it gets acknowledgement from the mobile router. There is a retransmit timer at the motes, after expiration of which the mote will send the same data again. This also takes care of lost packets due to channel errors, collisions etc. Eventually, the interest message on the motes will expire, leaving the interest cache empty, and the mote will stop sending out packets. When the mote again hears an interest, it may either resume sending data where it left or send other data according to the new interest. The modified behavior of the static devices compared to the raw diffusion protocol is summarized in the algorithm below in view of the above two modifications.

Algorithm 3. Communication protocol at static node

```
1. FLAG=NOT_ON_TRAIL
2. If an INTEREST message is heard:
     a. If FLAG == ON_TRAIL and INTEREST is not from router, break
     b. Start INTEREST_EXPIRY timer
     c. If the INTEREST message is from the mobile router set FLAG=ON_TRAIL
     d. Decrement TTL, If TTL > 0 rebroadcast INTEREST message
3. If there is a valid INTEREST in the interest-cache
     a. Decrement INTEREST_EXPIRY
     b. Transmit data in buffer to node from which interest message was heard, and start
        retransmit timer
     c. If timer expires and no ACK is heard, retransmit data and reset timer
     d. If ack is heard, decrement number of remaining samples and goto set 3
4. Collect sample every sample period. If buffer overflows, discard oldest data.
```

Another observation is that since some nodes which have direct communication with the router have to forward data from some other nodes as well, which are not directly queried by the router and since connection with the router exists for a limited time, the forwarding nodes can pre-fetch data from nodes for which they forward data. This can be achieved as follows. From the second round onwards, if a node hears a response to an interest transmitted by it, it knows that there are other nodes which forward their data through it. This node can then initiate a local diffusion. It sends a local interest message with TTL=1. Nodes which receive this message may themselves have deeper nodes in the tree for which they transfer data. Such nodes would have initiated a local diffusion with TTL =1 as well. These nodes will then respond to an interest when their local diffusion is complete.

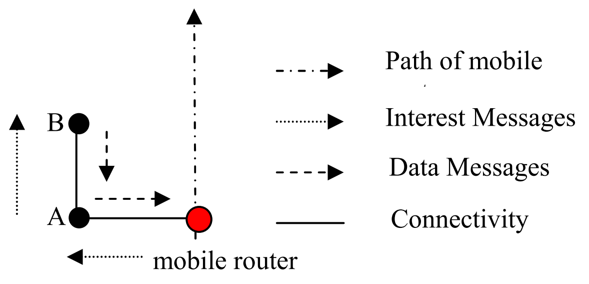

Figure 9. The communication protocol needs to restrict multi-hop communication but support nodes which are not in direct range of the router's trail.

.

This way, the nodes need not have explicit knowledge about the network topology and any length of path to the mobile router can be pre-cached. When the mobile router visits, the data from all nodes in the region can be quickly transferred.

Apart from reducing energy by limiting the number of transmissions, further improvements can be obtained in lifetime by utilizing the sleep mode. Sleep mode utilization is is generally referred to as topology management and several schemes such as [23, 24, 25, 26] have been suggested for this purpose with static nodes. We present a topology management method designed for the embedded devices communicating through a mobile router. The radio is a major power consuming component, and the motes must turn it on only when the mobile router will be in their range. Suppose the period of the mobile router to complete one round is **T**. This number can be made known to the mote at design time, piggybacked on the interest message, or can be learnt at run time. If it is learnt at run time, the mote can start a timer when it first hears a packet from the mobile router. Then it will keep hearing from the mobile router for the time it is in its range (the acknowledgement packets mentioned before). It records the value of the timer for each packet received, overwriting the time of the previous packet. After some time it will stop hearing from the mobile router. The timer value of the last packet received gives the time duration **t** for which the mobile router was in range. The minimum time the router takes to return to a mote is **T**/2, it could be more depending on how the speed varies in a particular run. The estimate of **t** can have some variation due to speed control. Thus the mote can switch off the radio for an amount of time equal to $\mathbf{(T/2) - t - \tau}$, where τ is an extra margin for error in estimate of **t**.

As mentioned earlier, the time synchronization between the base and sensor nodes can be significantly improved since the number of hops is much lower. Thus, it would be beneficial for the system to utilize the data packets exchanged to execute a time synchronization protocol. In our prototype, the nodes in addition to the data, send the number of samples remaining to be transmitted in their buffers as specified in the speed control algorithm. The router notes the time at which the sample is received. From the number of samples remaining at the mote, it can calculate how many sampling periods have elapsed since the sample was collected. Using this and the sampling rate it can back calculate the time of the current sample.

VIII. EXPERIMENTAL RESULTS

In this section we present the experimental results of the performance evaluation of our proposed speed control methods on our prototype system. For our present experiments, we use a straight trail (with no packet reception in reverse path) as the deployment scenario because we do not yet have the navigation equipment for traversing a complex closed trail currently. The experiments were done outdoors. The speed control methods will be the same on a more complex trail as well.

Table 1. Data Collection Parameters

Parameter	**Value**
Implementation platform	TinyOS
Interest Repetition Interval	2s
Interest Expiry Time	10s
Sampling Period	1s
Retransmit timer	1s
Buffer size	150Bytes
MAC random backoff	0 to 500ms

The first experiment is carried out on a dense network topology where the static nodes do form a connected network. The topology is shown below in Figure 10. The communication protocol specific parameters are listed in Table1.

Figure 11 shows the performance in terms of the amount of data collected within the equal latency constraint for the naïve approach of constant speed, and the two speed control algorithms proposed above. The latency constraint used is 72 seconds, which yields a naïve speed of 50cm/sec. For the ASC and SCD algorithms a fast speed of 100cm/s is used and ASC uses a slow speed of 50cm/sec. For ASC ω_1 was 25% and ω_2 was 75%. All results plotted are averaged over seven rounds. While both speed control algorithms perform better than the naïve approach, the relative performance among the two does not follow a specific trend.

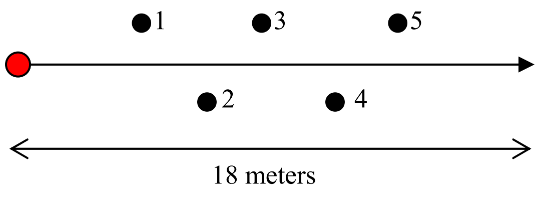

Figure 10. Topology for experiment on a connected network.

.

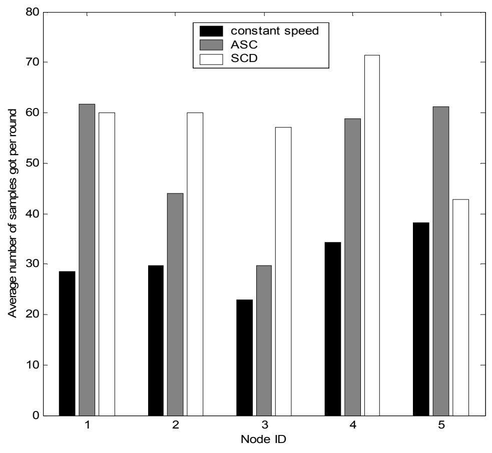

Figure 11. Performance of three speed control methods on connected topology.

The second experiment is performed on a topology with two groups of nodes, where nodes within a group are within wireless collision range of each other, leading to wastage of bandwidth due to MAC level contention. The two groups are separated. The topology is shown in Figure 12. The data collection parameters are as before and the results are again averaged over seven rounds.

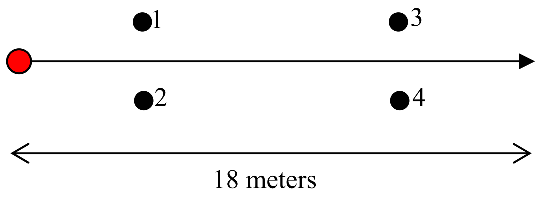

Figure 12. Topology for experiment on a disconnected network with two colliding groups.

Performance is plotted in Figure 13. The figure shows that the both the adaptive speed control methods perform better than the naïve approach to move at constant speed. The performance gains vary at different nodes. In this case SCD performs better than ASC as moving even at a slow speed may take the router outside the range of the colliding nodes.

A third experiment was performed on the sparse network case as shown in Figure 6. Two nodes were placed well outside the range of each other. Thus, there existed a segment in the trail when the mobile router was not in range of either node. The SCD algorithm was employed. As expected, controlling the speed helped improve the amount of data collected within the latency constraint. The performance is shown in figure 14 for both the nodes.

.

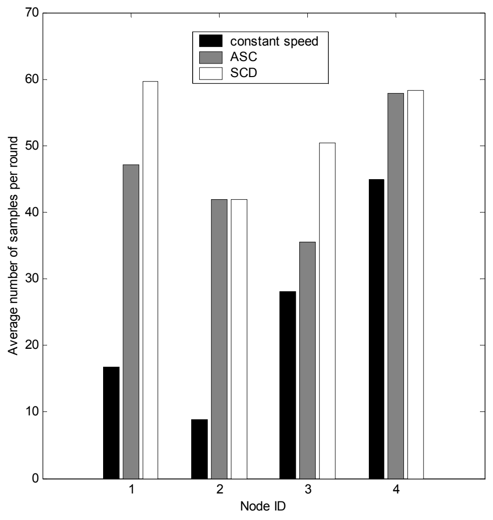

Figure 13. Performance comparison on the topology of Figure 12.

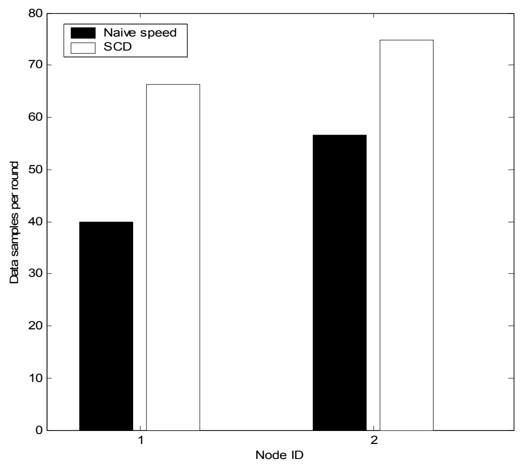

Figure 14. Performance comparison on sparse network topology.

IX. Conclusions And Future Directions

We discussed several advantages of having a controlled mobile element for the networking infrastructure. Our device uses energy at the base user end for saving energy in the embedded sensor nodes. The prototype system has clearly established the feasibility of using a mobile networking device for data gathering and some of the advantages of doing so. The test system constructed is now being used for research on several interesting issues in the use of fluid infrastructure. We elaborated on several design issues in the development of such an infrastructure, focusing on motion control and communication protocol design. The proposed motion control methods yield significant advantage in terms of data quality.

Our results are very promising and provide sensor network designers with a new method to solve their problems of network lifetime, data fidelity, time synchronization and bandwidth utilization. One of the components which we are in the process of adding to our fluid infrastructure is navigational support. Interesting design issues exist in the case when the time to complete a full round of the trail is more than the latency constraint and a trade-off must be made in the extent to which data travels wirelessly and over the mobile device. We also mentioned that when the time taken by the robotic platform for recharging is an issue in latency, new trade-offs in network lifetime and data latency would come into play. Future work also includes considering scenarios where the terrain allows the mobile router to move on any path or when multiple mobile components are available.

.

References

[1] G. J. Pottie and W. J. Kaiser. Wireless integrated network sensors. In *Communications of the ACM*, May 2000.

[2] David Tennenhouse. Embedding the Internet: Proactive computing. In *Communications of the ACM*, May 2000.

[3] Ya Xu and John Heidemann and Deborah Estrin. Geography-informed energy conservation for Ad Hoc routing. In *Proc. ACM Mobicom* July 2001.

[4] Henri Dubois-Ferriere, Matthias Grossglauser, Martin Vetterli. Age Matters: Efficient Route Discovery in Mobile Ad Hoc Networks Using Encounter Ages. In *ACM Mobihoc* June 2003.

[5] Rahul C Shah, Sumit Roy, Sushant Jain and Waylon Brunette. DataMULEs: Modelling a Three Tiered Architecture for Sparse Sensor Networks. In *First IEEE International Workshop on Sensor Network Protocols and Applications (SNPA)*, May 2003.

[6] Tara Small and Zygmunt J. Haas. The shared wireless infostation model: a new ad hoc networking paradigm (or where there is a whale, there is a way). In *ACM Mobihoc*, June 2003.

[7] Philo Juang, Hidekazu Oki, Yong Wang, Margaret Martonosi, Li Shiuan Peh and Daniel Rubenstein. Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with ZebraNet. In *Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, October 2002.

[8] A Chakrabarty, A Sabharwal and B Aazhang. Using Predictable Observer Mobility for Power Efficient Design of a Sensor Network. In *Second International Workshop on Information Processing in Sensor Networks (IPSN)*, April 2003.

[9] Alberto Cerpa, Jeremy Elson, Deborah Estrin, Lewis Girod, Michael Hamilton and Jerry Zhao. Habitat monitoring: Application driver for wireless communications technology. In *ACM SIGCOMM Workshop on Data Communications in Latin America and the Caribbean*, 2001.

[10] Alan Mainwaring, Joseph Polastre, Robert Szewczyk, David Culler and John Anderson. Wireless Sensor Networks for Habitat Monitoring. In *First ACM Workshop on Wireless Sensor Networks and Applications (SNPA)*, September 2002.

[11] Delay Tolerant Networking Research Group. www.dtnrg.org.

[12] Chalermek Intanagonwiwat, Ramesh Govindan and Deborah Estrin. Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks. In *Mobicom*, August 2000.

[13] Philip Levis, Nelson Lee, Matt Welsh, and David Culler. TOSSIM: Accurate and Scalable Simulation of Entire TinyOS Applications. In *ACM SenSys*, November 2003.

[14] Mica2 motes. Product Datasheet. http://www.xbow.com/Products/Wireless_Sensor_Networks.htm.

[15] A. LaMarca, W. Brunette, D. Koizumi, M. Lease, S. B. Sigurdsson, K. Sikorski, D. Fox, and G. Borriello. PlantCare: An Investigation in Practical Ubiquitous Systems. In *Ubicomp*, September 2002.

[16] Packbot, The Next Step in Unmanned Tactical Mobile Robots. www.packbot.com.

[17] iRobot. www.irobot.com.

[18] ActivMedia Robotics. www.amigobot.com.

[19] TinyOS: a Component-based OS for the networked sensor regime. http://webs.cs.berkeley.edu/tos/.

[20] X-Scale Single Board Computer and Wireless Networking Platform.
http://www.xbow.com/Products/XScale.htm.

[21] Dragos Niculescu and Badri Nath. Trajectory based forwarding and its applications. In *Mobicom*, September 2003.

[22] James Scott and Mike Hazas. User-Friendly Surveying Techniques for Location-Aware Systems. In *Ubicomp*, October 2003.

[23] Curt Schurgers, Vlasios Tsiatsis, Saurabh Ganeriwal, Mani Srivastava. Optimizing Sensor Networks in the Energy-Latency-Density Design Space. In *IEEE Transactions on Mobile Computing*, January-March 2002.

[24] Xiaorui Wang, Guoliang Xing, Yuanfang Zhang, Chenyang Lu, Robert Pless, and Christopher Gill. Integrated Coverage and Connectivity Configuration in Wireless Sensor Networks. In *ACM SenSys*, November 2003.

[25] Benjie Chen, Kyle Jamieson, Hari Balakrishnan, Robert Morris. Span: An Energy-Efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks. In *ACM Mobicom*, July 2001.

[26] Alberto Cerpa and Deborah Estrin. ASCENT: Adaptive Self-Configuring sEnsor Networks Topologies. In *Infocom*, June 2002.

[27] A.K. Salkintzis. A Survey of Mobile Data Networks. In *IEEE Communication Surveys*, 3rd Quarter 1999.

.

[28] Kevin Fall. A Delay-Tolerant Network Architecture for Challenged Internets. In *Sigcomm*, August 2003.
[29] Qun Li and Daniela Rus. Sending messages to mobile users in disconnected ad-hoc wireless networks. In *ACM Mobicom*, August 2000.
[30] Matthias Grossglauser and David Tse. Mobility Increases the Capacity of Ad-hoc Wireless Networks. In *Infocom*, April 2001.
[31] Jerry Zhao and Ramesh Govindan. Understanding Packet Delivery Performance in Dense Wireless Sensor Networks. In *ACM Sensys*, November 2003.
[32] Mohammed Rahimi, Hardik Shah, Gaurav S. Sukhatme, John Heidemann and D. Estrin. Studying the Feasibility of Energy Harvesting in a Mobile Sensor Network. In *IEEE Int'l Conference on Robotics and Automation*, May 2003.
[33] Deborah Estrin, Ramesh Govindan and John Heidemann. Embedding the Internet: introduction. In *Communications of the ACM*, May 2000.
[34] DARPA Energy Harvesting Projects. http://www.darpa.mil/dso/trans/energy/projects.html.
[35] V. Raghunathan, C. Schurgers, S. Park and M. Srivastava. Energy aware wireless microsensor networks. In *IEEE Signal Processing Magazine*, March 2002.
[36] Jeremy Elson, Lewis Girod and Deborah Estrin. Fine-Grained Network Time Synchronization using Reference Broadcasts. In *Proceedings of the Fifth Symposium on Operating Systems Design and Implementation (OSDI)*, February 2002.

.