Dynamic Localization Control for Mobile Sensor Networks

Sameer Tilak, Vinay Kolar, Nael B. Abu-Ghazaleh and Kyoung-Don Kang

{sameer,vinkolar,nael,kang}@cs.binghamton.edu

Abstract

Localization is a fundamental operation in mobile and self-configuring networks such as sensor networks and mobile ad hoc networks. For example, sensor location is often critical for data interpretation; moreover, network protocols, such as geographic routing and geographic storage require individual sensors to know their coordinates. Existing research focuses on localization mechanisms: algorithms and infrastructure designed to allow the sensors to determine their location. In a mobile environment, a related problem exists: when nodes are mobile, the underlying localization mechanism must be invoked repeatedly to maintain accurate location information. We propose and investigate adaptive and predictive protocols that control the frequency of localization based on sensor mobility behavior to reduce the energy requirements for localization while bounding the localization error. In addition, we evaluate the energy-accuracy tradeoffs that arise: intuitively, higher the frequency of localization, the lower the error introduced because of mobiliy. However, localization is a costly operation since it involves both communication and computation. Since energy is at a premium in wireless devices, it is important to perform localization in an energy efficient fashion. Our results indicate that the proposed protocols reduce the localization energy significantly without sacrificing accuracy.

1. Introduction

Localization is the ability of a sensor to find out its physical coordinates; this is a fundamental ability for embedded networks because interpretating the data collected from the network is possible unless the physical context of the reporting sensors is known. Several higher level services such as aggregation, routing, storage require sensors to know their coordinates. [10]). Existing research has focused on addresing localization problem static sensor networks (sensors once deployed are stationary throughout life-time) [3, 4].

1.1. Background

Localization has received a lot of attention in the context of static sensor networks. We now mention some

of the state-of-the art techniques which can be used for localization for static networks. He et. al [4] have classified existing localization techniques into two categories: *range-based* and *range-free*.In range-based techniques, information such as distances (or angles) of a receiver are computed for a number of references points using one of the following signal strength or timing based techniques and then position of the receiver is computed using some multilateration technique [12]. However, range-free techniques do not depend upon presence of any such information.

Localization techniques typically require some form of communication between reference points (nodes with known coordinates) and the receiver (node that needs to localize). Some examples of communication technologies are RF-based and acoustic based communication. In RADAR system [1], RF-based localization is suggested, where distance is estimated based on received signal strength. Cricket [9] uses concurrent radio and ultrasonic sounds to estimate distance. Some researchers have used Time based techniques such as Time-of-Flight(TOA) [12], Time-Difference-of-Arrival(TDOA) [9, 11] between reference point and the receiver node as a way to estimate distance. Niculescu et. al [7] proposed using angle-of-arrival to estimate position. Recently He et. al [4] proposed range-free techniques for localization.

A straightforward localization approach would make use of Global Positioning System (GPS). Existing research projects such as zebra-net [5] uses a GPS based localization, where mobile sensors find out their location every three minutes. He et. al [4] pointed out, GPS based systems require expensive and energy consuming electronics for precise synchronization with the satellite's clock. GPS uses one-way flight time information whereas other systems such as Local Positioning System (LPS) [12] use round-trip-time to avoid time synchronization.

Bulusu et. al [3] studied signal strength based and connectivity based techniques for localization in outdoor environments.

Perhaps most similar to our work, the pervasive computing community has investigated location and activity monitoring and prediction using wearable sensors [6]. However, the focus is on the accuracy of the estimate and prediction and not on the energy cost. Furthermore, most of these works assume the persence of accelerometers which we do not assume in this paper.

After discussing state-of-the art localization techniques for static sensor networks, we now motivate the problem for mobile sensors in the context of several real-world scenarions.

1.2. Motivation – Mobile Sensor Applications

ZebraNet [5], is a sensor network application for wild-life tracking whose goal is to provide more insight into complex issues. In this application, sensors are attached to zebras. As the zebras move, sensors record various parameters providing insight into mobility and

migration patterns, social structures of these species. In the proposed implementation, sensors perform localization every three minutes using GPS. However, such a fixed sampling period cannot account effectively for different mobility patterns that the animal follows: for example, 3 minute localization period is overly aggressive for an animal that is asleep or grazing, but may be insufficient to localize an animal that is moving at high speed. Clearly, it is better to have self-configuring sensors which will adapt dynamically to the animal behavior to provide an accurate energy-efficient localization. The protocols presented in this paper strive to make the sensor network self-adaptive.

As another motivating application consider, cellular phone companies that are interested in finding out coverage (signal quality) in a customer area to provide better quality service. Future infrastructure deployment decisions (e.g., new base stations) are driven by the collected information. At present, a common way to collect such information is to have a person to comb the area measuring signal strengths at various locations. This method is uneconomical and time-consuming. One can imagine cell phone capable with micro-sensors measuring signal strength. Such sensors need to find out their coordinates to report the measured parameters. All subscribers carrying such cell phone will gather such information as he is moving around.

In this paper we are concerned with the following fundamental energy-quality tradeoff associated with localizaiton in mobile environments. With mobility, nodes must repeatedly invoke localization to maintain an accurate estimate of their location. The more often the localization, the more accurate the location estimate. However, since there is an energy cost involved in localization, we would like to minimize the localization frequency. Thus, the localization must be carried out with a frequency sufficient to capture location within acceptable error tolerance. We call this problem *Location Tracking* (LT). We emphasize that location tracking is orthogonal to localization: we are concerned with the problem of *when to localize* which is largely independent of the underlying localization mechanism. More specifically we assume that the sensors use one of the several existing localization techniques. While we focus on localization control in a mobile sensor network environment, the algorithms and analysis apply for other mobile environments such as mobile ad hoc networks.

In this paper, we propose two new classes of localization approaches: (1) Adaptive; and (2) Predictive. Adaptive localization dynamically adjusts the localization period based on the recent observed motion of the sensor, obtained from examining previous locations. This approach allows the sensor to reduce its localization frequency when the sensor is slow, or increase it when it is fast. In the second approach, the sensors estimate the motion pattern of and project its motion in the future. If the prediction is accurate, which occurs when nodes are moving predictably, estimates of location may be generated without performing actual localization, allowing us to further reduce the localization frequency thereby sav-

ing the energy.

We propose algorithms that fit the two classes above and compare them to static, fixed-period, localization both using simulation and analysis. We show that dynamic localization can significantly improve the energy efficiency of localization without sacrificing accuracy in the location estimation (in fact, improving accuracy in most situations).

The remainder of this paper is organized as follows. In Section 2 we define the dynamic localization problem and present candidate protocols for addressing it in Section 3. Section 4 presents some analysis of the performance of the protocol under various conditions. In Section 5 we carry out an evaluation study of the protocols. In Section 6 we give more insight into behavior of predictive protocols with unexpected changes in mobility. Finally, in Section 7 we present some concluding remarks.

2. Problem Definition: Localization Control

Figure 1 shows a sensor node in motion. At every *localization point*, the node invokes its localization mechanism (e.g., using GPS, triangulation based localization, or otherwise) to discover its current location (x_i, y_i). The *localization point vector* is the sequence of localization points collected by a sensor is denoted S_i. We assume that the localization mechanism estimates the current position with a reasonable tolerance. In the figure, the uncertainty introduced by the localization mechanism is represented by the shaded circles in the Figure 1.

In the time duration between two consecutive localization points, the error in the estimate of the location increases as the node moves (on average) increasingly further from its last location estimate. In order to control this error, localiztion must be repeated with enough frequency to ensure that the location estimate meets some application-level error requirements (e.g., the estimate remains within a prespecified threshold from the actual location). However, carrying out localization with high frequency drains the node's energy. Solutions to this problem must balance the need to bound error with the cost of carrying out localization. Exploring protocols that effectively estimate location while minimizing the localization operations is the problem we consider in this paper.

We keep our analysis independent of the specific localization mechanism used. Note that dynamic control of localization is needed whether localization is carried out on demand (i.e,, the node queries neighbors or fixed localization nodes for localization information) or proactively (e.g., by having localization nodes periodically transmit localization beacons, or using GPS). If localization is on-demand , the localization mechanism can be invoked when needed. Alternatively, if the localization is done periodically without control of the sensor node, the node can still control its localization frequency by deciding when to start listening to the bea-

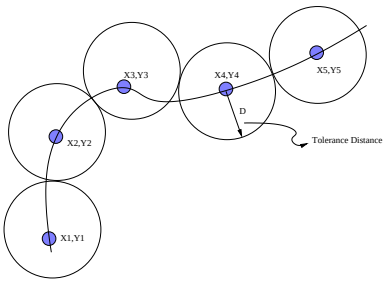


Figure 1. Mobile Sensor with Localization Points.

cons. Since receiving packets or GPS signals consumes significant energy, controlling the localization frequency also applies for such schemes. Also, an underlying assumption in this paper is that an accurate estimate of location is needed continuously. Such a situation would occur, for example, if sensors are continuously collecting data.

2.1. Performance Metrics

The primary tradeoff is between the observed localization error and the energy consumed. The localization error stands for diveregence of reported location from actual location. At any given time, we measure divergence in terms of euclidean distance between actual and reported coordinates – we term this the *instantenous error*. We also consider a threshold based error metric where we compare the absolute error to an appplication defined tolerance distance ($dist_{tolerance}$) (shown in figure 1); a localization error lower than tolerance distance is acceptable to the application. We measure the percentage of the time that the localization estimate is within the application defined threshold.

Consider the example of using cell phone to find out signal quality within a campus and periodically signal quality readings are sent to the base station. Tolerance distance can be specified as 5 meters (say). Intuttion behind is that, for cell companies to plan infrastructure deployment in future it is not required to get exact locations where the signal is low but they are interested in general to find out areas of weak signal. So the granularity is more coarse in that case. To capture that we consider threshold based error (E_{Thresh}).

For certain class of applications such as emergency services, instantenous error (divergence from actual coordinates) might be important. $Error_{absolute_t}$ captures that effect.

In the following equations, (lx_t, ly_t) (lx_{t-1}, ly_{t-1}) be the x and y coordinates of the node at time t and (t-1) repsectively. Also, let (x_{est_t}, y_{est_t}) and $(x_{actual_t}, y_{actual_t})$ denote the estimated and actual coordinates of a sensor at time t.

Let $dist_{trav}$ stands for

$$dist_{trav} = \sqrt{(lx_t - lx_{t-1})^2 + (ly_t - ly_{t-1})^2} \quad (1)$$

$$Error_{absolute_t} = \sqrt{(x_{est_t} - x_{actual_t})^2 + (y_{est_t} - y_{actual_t})^2}$$

$$E_{Thres_t} = \begin{cases} |dist_{trav} - dist_{threshold}| & \text{if } dist_{trav} > dist_{thresh} \\ 0 & \text{otherwise.} \end{cases} \quad (2)$$

3. Dynamic Localization Protocols

In this section, we introduce the proposed protocols for dynamic localization. We evaluate the following three approaches for localization: (1) Static localization: the localization period is static; (2) Adaptive localization: the localization period is adjusted adaptively, perhaps as a function of the observed velocity which can be approximated using the last two localization points; and (3) Predictive localization: in this approach, we use dead reckoning to project the expected motion pattern of the sensor based on the recent history of its motion.

As mentioned before, for this work we want to isolate performance of our protocols from any specific localization algorithm. We assume that the the localization algorithm once executed gives an estimate of its current location with reasonable accuracy. Therefore error introduced because of localization itself if negligible. The focus of this paper is not the localization algorithm but the different policies to determine invocation of the localization algorithm. Excessive invocation of the localization algorithm is not energy efficient while not invocating the algorithm enough will result in unacceptable error. In the remainder of this section, we introduce our proposed protocols for each of these approaches in more detail.

3.1. Static Fixed Rate (SFR)

This is the base protocol where localization is carried out periodically with a fixed time period t. The sensor node reports its co-ordinates as the location captured during the last localization point. For example, let the localization interval be t_{sfr}. Let us assume that the node had localized at time t and calculated its co-ordinates as (x_t, y_t). Then the node is going to report its location as (x_t, y_t) for the time period between t and $t + t_{sfr}$. This protocol is simple and its energy expenditure is independent of mobility; however, its performance varies with the mobility of the sensors. Specifically, if a sensor is moving quickly, the error will be high; if it is moving slowly, the error will be low, but the energy efficiency will be low.

3.2. Dynamic Velocity Monotonic (DVM)

In this adaptive protocol, a sensor adapts its localization as a function of its mobility: the higher the observed velocity, the faster the node should localize to maintain the same level of error. Thus whenever a node localizes, it computes its velocity by dividing the distance it has moved since the last localization point by the time that elapsed since the localization. Based on the velocity, the next localization point is scheduled at the time when a

prespecified distance will be travelled if the node continues with the same velocity. This distance, for example, can be the application specified desired maximum error threshold. Thus, when the node is moving fast, localization will be carried more often; when it moves slowly, localization will be carried out less frequently. Similar to SFR, the location reported by the node between two localization points will be one calculated at the previous localization point.

In this protocol, there is a configurable parameter α that represents the target maximum error. At every localization point, the current estimated velocity is computed. Based on this value we estimate the time required to travel the prespecified tolerance distance ($dist_{tolerance}$). If the node continues with the same velocity – the next localization point is scheduled at that point. Note that this approach assumes that a node is moving with a constant velocity between localization points. This may not be always accurate – for example, if a node was standing still for half the period, then started moving at a velocity v, the estimated velocity will be $\frac{v}{2}$, and we will end up with suboptimal localization (e.g., exceeding the error threshold for some time). Moreover, for very low speeds the localization period may be computed adaptively to be very large (e.g., a period of infinity would be predicted if the node is standstill). Similarly, if the speed is very high, the localization period may become very low, thereby spending a lot of energy. To account for these effects, we place an upper and a lower limit on the localization periods, which we call as *upper and lower query thresholds* respectively.

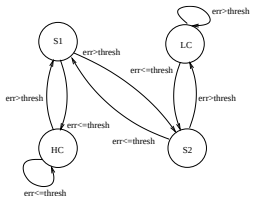


Figure 2. State Diagram for Dead Reckoning

3.3. Mobility Aware Dead Reckoning Driven (MADRD)

This is a predictive protocol that computes the mobility pattern of the sensor and uses it to predict future mobility. To the best of our knowledge, this is the first paper to apply dead reckoning for localization in mobile sensor network.

Using dead reckoning, localization should be triggered when the expected difference between the actual mobility and the predicted mobility reaches the error threshold. This is in contrast to DVM where localization must be carried out when the distance from the last localization point is predicted to exceed the error threshold. Thus, if the node is moving predictably, regardless of its velocity, localization can be carried out at very low frequency; if the predicted mobility pattern is perfect and holds for all future time, no further localization would be necessary.

Account for differences in the predicted model and the actual mobility of the sensor, include errors due to changes in the mobility pattern that occur after or during

dead-reckoning estimation is almost impossible. In the next section, we analyze the effect of this difference for some special cases. We place an upper and a lower limit on the localization periods, which we call as *upper and lower query thresholds* similar to that of DVM. Moreover, we score the performance of our prediction at every localization point by comparing the predicted location to the actual location. If the prediction is erroneous (larger than a prespecified rate of divergence), we move towards a low confidence state and become more aggressive in localization. The intuition is that the mobility pattern is changing, and more localization is needed to capture the new mobility pattern as well as to bound the localization error. However, if the prediction is accurate, our confidence in the predictor increases and we increase the localization period.

A state diagram for MADRD is shown in Figure 2. In this diagram, HC refers to the high confidence state where the predictor is scoring well and localization period is increased. LC refers to the low confidence state where the predictor is not scoring well and the period is decreased. Erroneous predictions move the predictor towards the LC, while correct predictions move it towards HC. States S1 and S2 provide some hysterisis between LC and HC.

4. Error Analysis under Constant Velocity Mobility

Let us assume that the node moves with constant velocity v. Let v_x and v_y be the velocity along X and Y axes. Let the sensor node localize every t_{sfr} seconds using SFR protocol. If DVM protocol is used to localize then the time interval between consecutive localization will be determined by the velocity v. If the node is using MADRD to localize, it will localize at a period as determined by the algorithm highlighed in Figure 2. The MADRD protocol predicts its current co-ordinates based on the velocity and the previously localized co-ordinates. However, DVM and SFR will use a non-predictable model for estimation of current co-ordinates. Hence, the error in DVM is similar to SFR – the difference being that the period is adapted in DVM to try and limit this error. Therefore, the analysis of DVM is not done explictly. The analysis of error under constant velocity motion is broken down into three stages (1) Simple Constant Velocity Mobility; (2) Constant Velocity with pause; and (3) Constant Velocity change of direction.

4.1. Constant Velocity Mobility Scenario

The simple case of a node moving in a straight line and not taking any turns is analyzed in this subsection. We first discuss SFR, then MADRD.

4.1.1. SFR protocol Assume that the node had localized at time t and is supposed to localize next at time

Figure 3. Error with no deviation

$(t + t_{sfr})$. Let the location of the node at time t be (x_t, y_t). If the node has to send a data packet at time between t and $(t + t_{sfr})$, then the node will attach its location as (x_t, y_t). Since the node is moving at constant velocity, the actual distance travelled from the time t increases linearly. The error is the measured as the difference between actual location and (x_t, y_t). Hence, the error keeps increasing linearly till $(t + t_{sfr})$ is reached as shown in the Figure 3. The X-axis in Figure 3 shows the distance travelled from the point (x_t, y_t). The slope of this line will proportional to the velocity v.

4.1.2. MADRD protocol In case of MADRD, the node will calculate its location based on the previously localized co-ordinates $(x_{t'}, y_{t'})$ and previously measured velocities (v_x, v_y) and give the calculated location as the current location. For constant velocity straight line motion, v_x and v_y does not change. Hence the location calculated will be accurate as shown in graph in Figure 3.

4.2. Constant Velocity with pause scenario

In this case, the node comes to a standstill after being in motion with velocity v. Let the distance at which the node stops be d meters after the previous localization point (x_t, y_t). In this case, the error in SFR increases linearly until d, when it stops increasing. Conversely, the error in MADRD starts at 0 while the node maintains the speed of v. However, when it stops moving, the error in MADRD starts increasing proprtionately to v since the predictor assumes that the node continues in motion. Interestingly, if the node is standstill but suddenly starts moving with velocity v, SFR and MADRD will behave identically until the next localization point (which may be different for each). The reason is that SFR's uses the implicit prediction that the node remains at the point of the last localization. In this scenario, MADRD uses the same predictor since the node actually was not moving at the last localization point.

4.3. Constant velocity with change of direction scenario

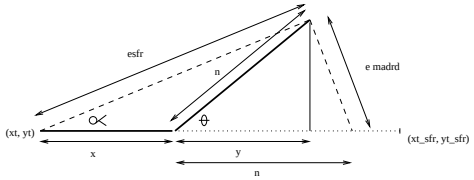


Figure 4. Error for deviation of θ degrees

Now consider the node taking the deviation of θ degrees. Let the distance at which the node takes the devia-

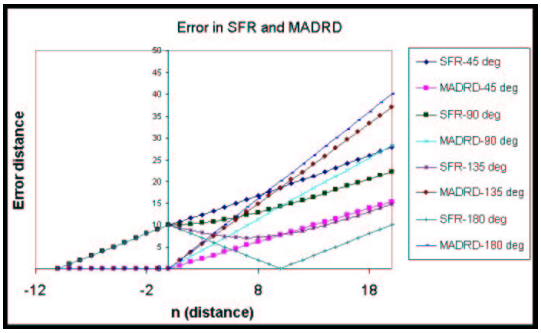


Figure 5. Errors in SFR and MADRD

tion be x meters after the localization point (x_t, y_t). The time at which the deviation occurs is greater than time t and lesser than $t + t_{sfr}$. Figure 4 shows the movement of the node. The distance between points (x_t, y_t) and $(x_{t_{sfr}}, y_{t_{sfr}})$ signifies the distance covered in time t_{sfr} with constant velocity v if there was no deviation along the path.

The error in localization between time t and $t + t_{sfr}$ can be split up into two parts. The first part is error before the deviation occurs (identical to the fixed velocity analysis above) and the second one is after the deviation. Let n be any point on the expected line of motion that the node would have travelled if it had not taken the deviation. If the node would have travelled a distance of n along expected straight line, it will travel the same distance after deflection because of constant velocity. Let $n = 0$ at the point of deviation and increases along the straight line.

4.3.1. SFR protocol Let the node use SFR protocol for localizing. The the error at point n will be the length of line e_{sfr} shown in Figure 4. The equation for e_sfr is given by Equation 4.

$$\tan \alpha = \frac{n \times \sin \theta}{(x + n \times \cos \theta)} \tag{3}$$

$$e_{sfr} = \frac{n \times \sin \theta}{\sin \alpha} \tag{4}$$

Figure 5 shows the graph of error against n . As n increases from 0 to y, the e_{sfr} varies as shown in the graph in Figure 5. We can see that for $n > 0$, the curve is not linear. This can be seen clearly in the case where $\theta = 135\ degrees$.

As the angle of deflection increases from 0 $degrees$ to 90 $degrees$, the error in SFR decreases because the line of motion will be nearer to (x_t, y_t) when θ increases. For angles greater than 90 $degrees$ and lesser than 180 $degrees$. The error decreases as node moves towards (x_t, y_t) and then starts increasing.

At $\theta = 180\ degrees$, the error touches zero after the node has covered x distance and then the error starts increasing linearly. Now the error vector is in other direction than the earlier error vector. Graph in Figure 5 shows the absolute value of the error.

4.3.2. MADRD protocol

$$e_{madrd} = 2 \times n \times \sin \frac{\theta}{2} \tag{5}$$

The length of the line e_{madrd} in Figure 4 shows the the error in MADRD protocol. It increases linearly as the n increases. This is given by the equation 5. Graph in Figure 5 shows the comparison of MADRD protocol with SFR for different angles. We observe for acute an-

Simulation area	$300 \times 300\ m^2$
Transmission range	$100\ m$
Initial Energy	$10000\ J$
MAC Protocol	802.11
Transmit Power	$0.660\ W$
Receive Power	$0.395\ W$
Idle Power	$0.035\ W$
Number of Mobile Nodes	24
Number of Beacon Nodes	36

Table 1. Simulation parameters.

gles, MADRD protocol performs better than the SFR. However, if θ is between 90 $degrees$ and 270 $degrees$, SFR starts performing better. This is because the node is moving away from the predicted motion line and e_{sfr} is smaller than the e_{madrd}.

5. Experimental Results

In order to analyze the protocols, we used ns-2 [8], a discrete event simulator. Table 1 summarizes the relevant parameters used during our simulations. We use a query based localization mechanism: a node that is interested in localization broadcasts a request – beacon nodes that receive the request reply with their own coordinates. The node upon receiving the coordinates from the beacon nodes, uses them to infer its location (e.g., it can use aformentioned triangulation technique with the beacon coordinates as reference points). In our simulation, the beacon nodes are placed such that at least three, and sometimes four, beacons are able to answer a query. Note that our results are not dependent on this localization model, regardless of how the localization is carried out.

First, we consider the random waypoint model, widely used in the mobile ad hoc network community. In this model, a node picks a random location in the simulated area and starts moving to it with a controllable average velocity. When the node reaches the destination, it pauses for some fixed pause time and then picks another destination randomly and starts moving towards it. The model is predictable while the node is moving, or for the duration of the pause but not during the period where it pauses or when it starts moving. Since both speed and pause times are important parameters of random mobility model, we conducted simulations to study effect of mobility and pause time on error and energy. In general, if the pause times are short, the node has more unpredictable behavior. We then study performance of MADRD protocol for various upper query threshold values. After studying the performance of these protocols for various important parmeters of a given mobility model, finally, we present some limited results with Gaussian Markovian mobility pattern which does not lend itself well to prediction using a constant velocity model as we do in MADRD. We used BonnMotion tool [2] to generate the various scenarios.

5.1. Fundamental Energy-Error Tradeoff

Figure 6 shows the absolute error for random waypoint mobility model with speed uniformly distributed

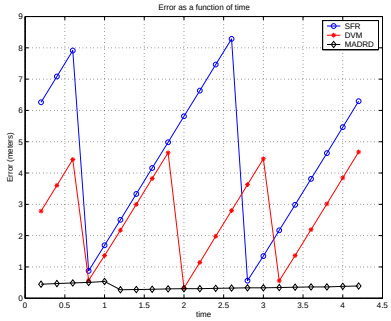


Figure 6. Absolute Error: Speed (4-5 m/s).

between 4-5 m/sec. The SFR period in this case was chosen to be 2 seconds – the node performs localization once every two seconds. Recall that, in the case of SFR and DVM the node assumes that the last measured localization point is its current location. Therefore $Error_{absolute_t}$ continues to grow between two successive localization points as the node moves away from its last localization point. Figure 6 shows the absolute error for SFR, DVM and MADRD protocols. In the case of SFR, sensor 0 localizes approximately at times 0.6, 2.6. As one can see upon localization the error lies within the localization mechanism error range (which we picked to be uniformly distributed between 0 to 0.5 meters). In between the two localization points, the error increases linearly up to 8 meters. In the case of DVM, a similar trend is seen again, howerver due to adaptive localization intervals, the magnitude of the error is lower than that of SFR; DVM was able to discover that it needs to localize more often than once every 2 seconds. In the case of MADRD protocol, the ability to predict the current location gives rise to very low error since the node actually follows the prediction. This graph clearly shows the strength of dead-reckoning procotols due to their prediction capability.

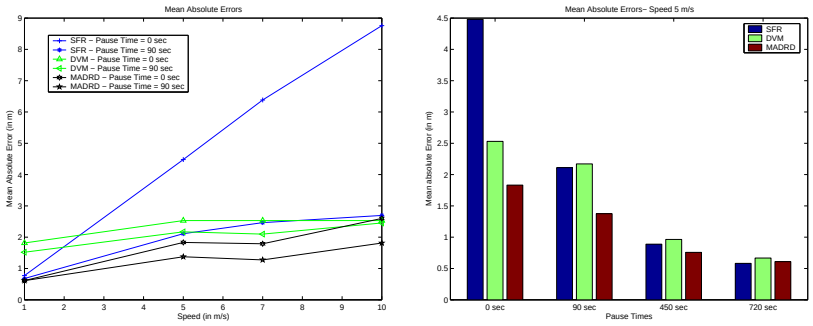
We now systematically vary mobility from very low to high speeds, more specifically we consider four cases: 1 m/sec, 5 m/sec, 7 m/sec, and 10 m/sec.

Figure 7(a) shows the absolute error as a function of mobility for the four protocols for two different pause time values. The primary observation here is that the error for SFR grows linearly with the average velocity while both DVM and MADRD manage to adapt their localization and maintain an error that does not grow significantly with the velocity. Under high mobility, this requires more localization operations than SFR. [1]

Figure 7(b) shows the effect of pause time for one specific velocity. For this given speed, we consider 4 pause times, namely, pause time of 0%, 10%, 50% 90% of the total simulation time. Pause 0% represents continuously moving subjects, while in the case of 90% pause time,the subject pauses for 90% of the simulation time. Intutively, these 4 pause times together cover a very broad range of mobile objects. Since pauses affect the prediction of DVM and MADRD, their advantage in terms of error relative to SFR is highest with zero pause time. At very high pause times, all three protocols perform well.

As mentioned before, an alternative measure of localization effectiveness is to monitor the fraction of the simulation time where the localization estimate was

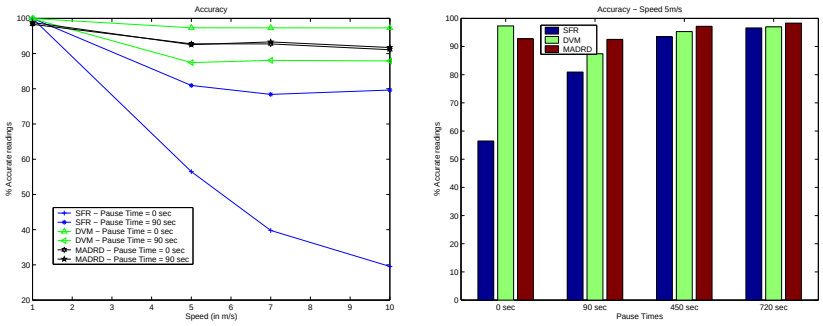
1 For readability,we do not consider all the four pause times for these speedes.



(a) Error Comparison

(b) Error vs. Pause Time

Figure 7. Absolute Error as a function of Mobility and Pause Time



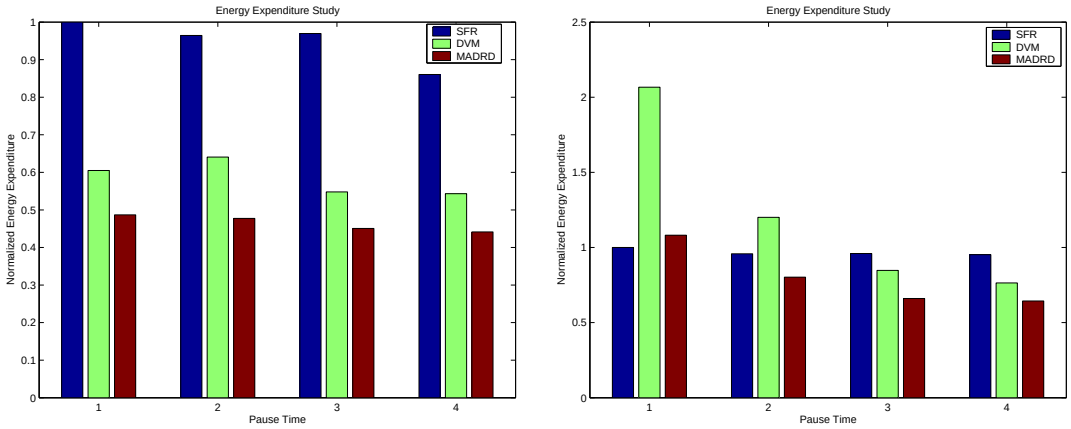
(a) Accuracy Comparison

(b) Accuracy vs. Pause Time

Figure 8. Percentage accuracy study a function of mobility and pause time.

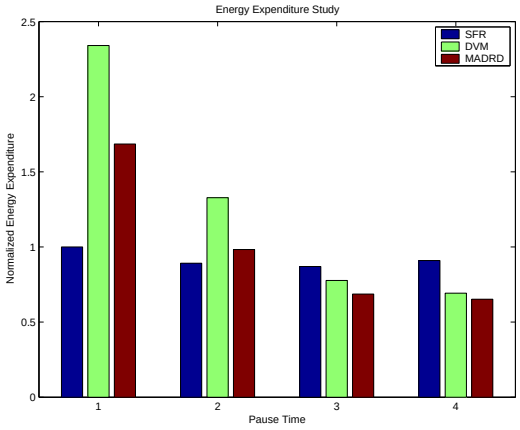
within an application specified threshold; ($dist_{tolerance}$ in this case 5 meters). It represents the E_{Thresh} error defined previously in equation 2. Figure 8(a) shows the accuracy as a function of mobility for two pause times. Again, we observered a similar trend here – DVM and MADRD perform much better than SFR, especially as higher velocities. As the speed increases, the performance of SFR degrades dramatically. However, DVM and MADRD are almost resillient to change in speed. Notably, for lower speeds (1 m/sec), all the three protocols have almost indentical error value, but as shown below, DVM and MADRD achieve significant energy savings than that of SFR.

Figure 8(b) shows the accuracy for one average velocity as the pause time is varied. For lower pause times, DVM and MADRD perform significantly better than SFR. As the pause time increases, all the three protocols perform well. Importantly, as shown later, the energy characteristics of DVM and MADRD are better than those of SFR.



(a) Speed (0.5-1 m/s))

(b) Speed (4-5 m/s))



(c) Speed (8-10 m/s)

Figure 9. Energy Expenditure Study as a function of mobility and pause time.

After studying effect of various parameters on error values, we now consider the energy efficiencies of these protocols. In Figures 9(a), 9(b) and 9(c) show the energy expenditure for various protocols as a function of mobility and pause time. In each of these graphs, the Y-axis represents the energy expenditure of various protocols normalized with respect to SFR. In the case of low mobility 9(a), DVM and MADRD localize less often than SFR. Therefore, both DVM and MADRD result in significant energy savings as compared to that of SFR. However, as the speed increases, the energy expeniture of DVM and MADRD grow more than that of SFR. Note that since these protocols are adaptive, even for high speeds they adapt well with the increase in pause time thereby spending less energy than SFR when pause time is high. For example, when the speed is between 4-5 me-

ters/sec, figure 9(b) indicates that with pause time set to 0 seconds and 90 seconds, DVM localizes more often than SFR and spends more energy. But when the pause time is increased to 450 seconds, DVM being adaptive, spends less energy than SFR.

5.2. Performance stufy of MADRD with Upper Query Thresholds Variations

Recall that to protect against inaccuracies in the prediction model or unexpected changes in the mobility model MADRD must limit the maximum period between localizations (we call this upper query threshold). Figure 10 shows the effect of this tradeoff – we vary the upper query threshold and observe the effect on the accuracy, error and localization energy. If the threshold is raised, this allows MADRD to aggressively predict location without forcing localization operations to ensure that the predictions are accurate. Thus, at high thresholds, higher energy savings are possible 10(c), but the expected error grows 10(a), 10(b). A good value for the upper threshold must balance these two effects.

5.3. Effect of Mobility Model Variation

This study presents our observations regarding performance of these protocols under gaussian mobility model. The results indicate that, energy expenditure of both SFR and MADRD is almost same, while the DVM has higher energy expenditure. Importantly, MADRD has significantly lower error than both SFR and DVM. Eventhough our results with gaussian model are preliminary, they provide a good starting point for our future study. More specifically, in future, we would like to evaluate the performance of these protocols under various other mobility models including group mobility models.

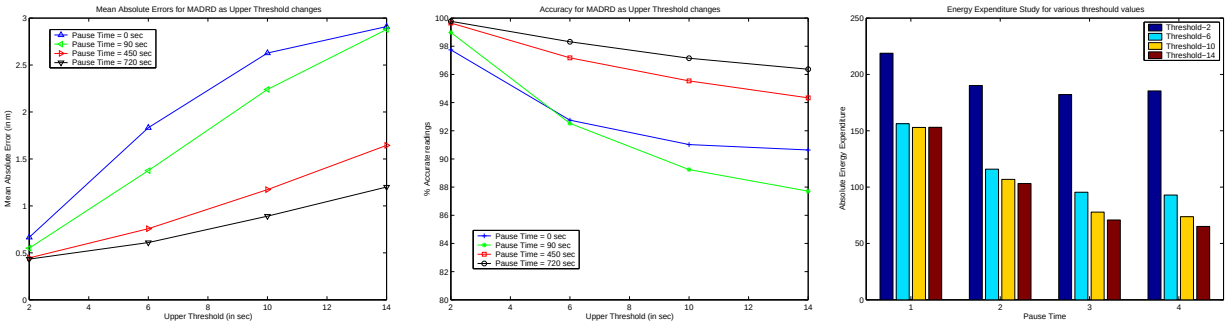
6. Effect of change in mobility pattern

In this section, we analyze behavior of the protocols under unexpected change in mobility pattern.

6.1. Effect of change in direction MADRD:

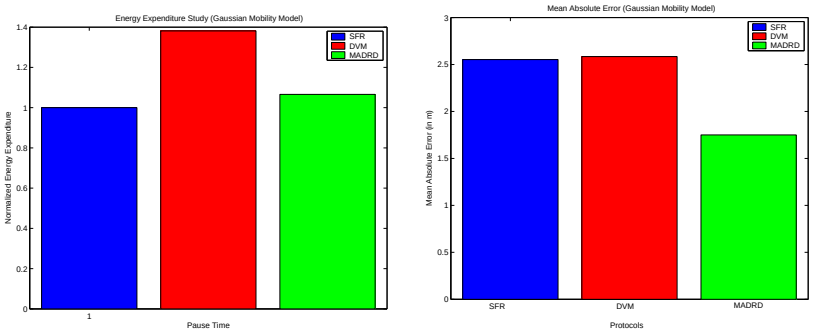
Unexpected changes in mobility pattern of a node such as pause and change in direction alter the accuracy of the protocols. The performance of adaptive protocols (DVM and MADRD) is more prone to variance in mobility pattern of the node than the non-adaptive protocols. DVM uses the observed mobility pattern to alter the localization interval. If the change is not captured appropriately, this leads to inaccurate decisions by the protocol. The effect is more pronounced in MADRD which not only is an adaptive protocol but also a predictive protocol. In that case, the error between the actual and the predicted location of the node is more pronounced because of the change in node movement pattern.

Figure 12(a) shows the behavior of MADRD when a turn occurs. In this case, the MADRD estimate continues predicting motion in the original direction. Moreover, even when localization occurs, the average velocity computed as a predictor for the next period will be off



(a) MADRD speed (4-5 m/s) (b) MADRD speed (4-5 m/s) (c) MADRD speed (4-5 m/s)

Figure 10. Performance of MADRD as a function of Upper Query Threshold for a fixed mobility



(a) Energy: Gaussian Mobility (0.5-1 m/s)(Upper Threshold 6 sec)

(b) Absoulte Error: Gaussian Mobility (0.5-1 m/s)(Upper Threshold 6 sec)

Figure 11. Characterizing protocol behavior for Gaussian Mobility Model.

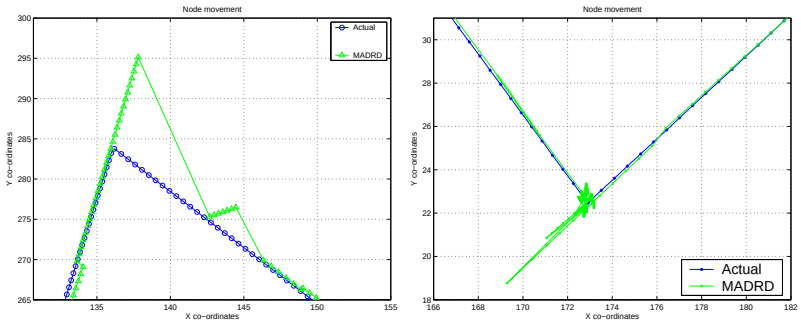
as well (it represents the weighted average of the original as well as the new velocities).

6.2. Effect of Pause time in MADRD

Consider Figure 12(b) where a node moving with a constant velocity comes to a pause. In this case, MADRD estimate overshoots the node along the old trajectory when it pauses. Because of the nature of MADRD protocol, the predicted co-ordinates oscillates around the actual location. The dampening of these oscillations can be observed in the Figure 12(b) as the time progresses.

6.3. Effect of mobility pattern on the protocols

The assumed mobility model has significant implications on the performance of the localization protocols. For SFR, the primary property is the average velocity in the model – this determines the expected lo-



(a) Change in direction

(b) Effect of pause

Figure 12. MADRD Behavior with Unexpected Change in Velocity

calization error given a certain localization frequency. For the adaptive approach, the computed velocity affects the localization frequency. Thus, if there are frequent changes in velocity (magnitude or direction), this will affect the estimate for the velocity for the next period, possibly increasing the error. For MADRD, the effect of the mobility pattern is even more pronounced. Specifically, changes in velocity will cause the algorithm to continue predicting the node motion along the previous trajectory of its motion. Further, changes in motion will cause the velocity at the next point to be predicted erroneously – for example, if the node was moving for half the period then came to a stop, the velocity will be predicted as half the original velocity when the node is actually at a standstill. Thus, for MADRD to operate best, the mobility pattern should be predictable.

7. Concluding Remarks

In this paper, we explored approaches and tradeoffs to the problem of localization in mobile sensor networks. A basic localization scheme; SFR, localizes periodically, with a fixed period. However, this approach may be insuffient to localize accurately if the period is large relative to the speed of the node. Moreover, if the localization period is small, localization will be carried out more frequently than would ideally be needed to localize causing a proportionate loss in energy needed to carry out localization.

We explored two algorithms for dynamic localization: (1) DVM: an adaptive algorithm that matches the localization period to the observed velocity of the node; and (2) MADRD: a predictive algorithm that uses dead reckoning to estimate the location of a node assuming it is following its recently tracked tracjectory. We characterized the performacne of these algorithms for two mobility patterns under different velocities and pause times. Both the proposed approaches significantly outperform static localization both from an energy and accuracy perspectives. In particular, MADRD performance was excellent in almost all situations that were studied; how-

ever, it is best suited to mobility patterns that are predictable and this result may not generalize to other mobility scenarios as discussed before.

In the future we would like to implement these protocols on existing sensor prototypes (eg. Motes) and study their performance. The Zebranet project has developed a simulator for studying systems tradeoffs in wild-life tracking environment in a realistic setting. We would like to port our protocols from ns-2 to ZNetSim [5], a simulator for Zebranet, and study the performance for an existing application. At present our work is limited to individual mobility models; but in the future we will also explore group mobility models. For military scenarios for example we can imagine a group of soldiers moving together to achieve certain goal. We would like to evaluate the protocols proposed in this paper for such scenarios and suggest some improvements.

References

[1] P. Bahl and V. N. Padmanabhan. RADAR: An in-building RF-based user location and tracking system. In *INFOCOM (2)*, pages 775–784, 2000.

[2] BonnMotion. http://web.informatik.uni-bonn.de/IV/Mitarbeiter/dewaal/BonnMotion/.

[3] N. Bulusu, J. Heidemann, and D. Estrin. Gps-less low cost outdoor localization for very small devices. *IEEE Personal Communications Magazine*, 7(5), October 2000.

[4] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. Abdelzaher. Range-free localization schemes for large scale sensor networks. In *Proceedings of the 9th annual international conference on Mobile computing and networking*, pages 81–95. ACM Press, 2003.

[5] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. S. Peh, and D. Rubenstein. Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with zebranet. In *Tenth international conference on architectural support for programming languages and operating systems on Proceedings of the 10th international conference on architectural support for programming languages and operating systems (ASPLOS-X)*, pages 96–107. ACM Press, 2002.

[6] S.-W. Lee and K. Mase. Activity and location recognition using wearable sensors. In *IEEE Pervasive Computing*, 2002.

[7] D. Niculescu and B. Nath. Ad hoc positioning system (aps) using aoa. In *In Proceedings of INFOCOM*, 2003.

[8] Network Simulator. http://isi.edu/nsnam/ns.

[9] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The cricket location-support system. In *Mobile Computing and Networking*, 2000.

[10] S. Ratnasamy, D. Estrin, R. Govindan, B. Karp, S. Shenker, Yin, and F. Yu. Data-centric storage in sensornets. In *Proceedings of the First ACM SIGCOMM Workshop on Hot Topics in Networks*, Oct. 2002.

[11] A. Savvides, C.-C. Han, and M. B. Strivastava. Dynamic fine-grained localization in ad-hoc networks of sensors. In *Proceedings of the 7th annual international conference on Mobile computing and networking*, pages 166–179. ACM Press, 2001.

[12] A. Ward, A. Jones, and A. Hopper. A new location technique for the active office. In *IEEE Personnel Communications*, 1997.