Recovery Target Tracking in Wildlife

Francine Lalooses
Department of Electrical Engineering
Tufts University
Medford, MA
Email: Francine.Lalooses@tufts.edu

Hengky Susanto
Department of Computer Science
Tufts University
Medford, MA
Email: Hengky.Susanto@eecs.tufts.edu

Chorng Hwa Chang
Department of Electrical Engineering
Tufts University
Medford, MA
Email: hchang@ece.tufts.edu

Abstract— With recent advances in mobile sensor networks, sensor networks are now being used in wildlife environments. Sensor networks are utilized in wildlife to track animals. The discoveries that have been made through such efforts include animal behavior and lifestyles. One of the most significant reasons why wildlife tracking research has escalated is for the protection of humans and animals in the food chain. Another possible reason for wildlife tracking is to protect endangered species.

In this paper, we analyze related work being done and describe how to track mobile sensor nodes. We then present an optimized recovery algorithm to track animals when they cannot be found. Using simulations we show that our proposed algorithm is effective in extending the network lifetime and performing a quick recovery.

Index Terms— **Algorithms, Recovery, Sensor Networks, Tracking**

I. Introduction

SENSOR networks are one of emerging technologies that have many applications. These networks are composed of hundreds, and potentially thousands of tiny sensor nodes, functioning autonomously, and serving different purposes. While the set of challenges range [1] [8] from geography routing, power conservation, data management, these sensors also hold the potential to revolutionize many segments of economy and life, from manufacturing, military, transportation, health-care industry, and environmental monitoring. [9] Sensors are often deployed in constrained environments such as deserts, road, forests, or other physical phenomena for monitoring, observing, or detecting a particular event. One of the applications of sensors network, which will be discussed in this paper, is to monitor the health of the animals or report the condition of wildlife habitat for environmental conservation purposes.

Habitat and environmental monitoring represent a class of sensor network applications that benefit the scientific communities and societies.[3][4][5] This technology allows researchers and biologists to have a closer look on the monitored area by providing [7] local measurement, sampling, and detailed information that are otherwise difficult and expensive to obtain. The traditional approach of obtaining such information requires human presence at the monitored location, which can disrupt normal animal behavior. For example, [10] seabird colonies are extremely sensitive to human presence. In Maine, biologists have discovered that their daily 15 minute visits to the colony increase the egg and chick mortality rate by 20% in a given breeding year. As the disturbance is continuously repeated, the entire colony may abandon the breeding site.

Tracking in sensor networks is also necessary for many other applications, such as computer vision, tactical battlefield surveillance, air traffic control, perimeter security, and emergency response.

Numerous studies have been done for tracking movements and the population of animals in their natural habitat. While there are many tracking movement techniques for single or multiple targets, there are other challenges, which are equally important, such as failures occurring during the tracking process.

We begin our paper analyzing two existing applications that have been deployed in wildlife communities in section II, such as ZebraNet and Great Duck Island project. In Section III, we elaborate on the specifications of tracking mobile sensor nodes in wildlife. Those specifications include hardware used, tracking and prediction algorithm, and recovery approach. Our paper will discuss how to recover and rediscover when failures occur during the habitat and wildlife monitoring operation. We present a comparison of our recovery simulation versus a simulation proposed by another university. The performance of this simulation generates better results for larger sized land animals, such as horse, cow, deer, etc. We conclude by presenting proposals for future ventures.

II. Related work

Tracking mobile nodes in a sensor network deals with approximating the trajectory of one or more moving objects based on information transmitted by the sensors. A common example of animal tracking is the problem of tracking cows in order to determine the origin and isolate the spread of mad cow disease. Tracking animals introduces curiosity of understanding animal behavior, animal interaction between humans and other species, and how animals affect the lives of humans in their participating role in the food chain.

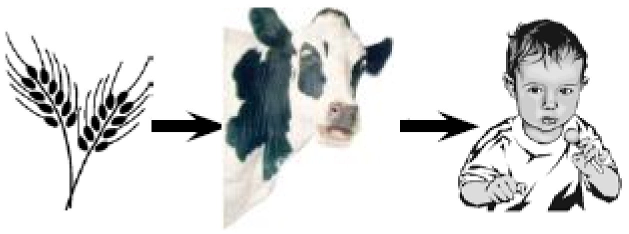


Figure 1: The food chain

Due to the mobile ad hoc nature of these networks, sensor networks face two major problems. First, efficient network traffic protocols and energy-reducing techniques are required. Secondly, the sensors have to communicate with one another or with a "base" to transmit readings or results of local computations. However, several methods for tracking animals have been implemented and studied, such as ZebraNet [6] and Great Duck Island [2].

A. ZebraNet

At Princeton University [6], researchers are investigating advances with wireless sensor networks and applying this technology to support wildlife tracking for biological research. This tracking system, called ZebraNet, can forward data to a mobile base station by using peer-to-peer networking techniques. Their design decisions include custom tracking collars worn by the animal. These collars weigh approximately 2.4 pounds and are enabled with global positioning systems. Since all collars, or nodes, are mobile, ZebraNet can track animals long term and over long distances. The protocols evaluated are flooding (peer-to-peer) and history-based (peer-to-peer) compared against direct (not peer-to-peer). In flooding, animals broadcast to everyone discovered in their range every three minutes. In history-based, animals choose at most one peer to send within range every three minutes; the one with the best past history success rate of delivering data is chosen. Figure 2 depicts the success rate with infinite storage and constrained bandwidth; in short range, flooding proves to be the best protocol, but in long-range, history-based outperforms the other protocols.

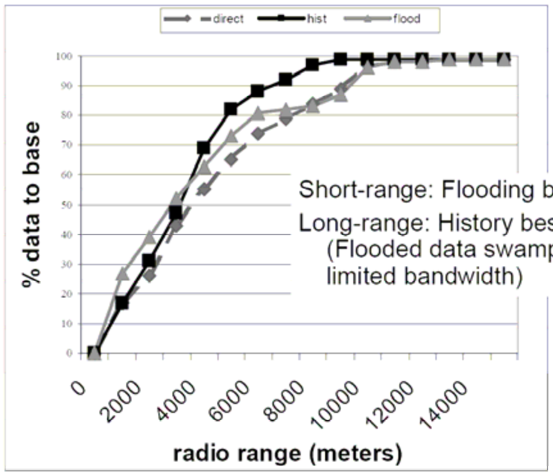


Figure 2: Radio Range Results

When investigating these protocols, it is important to consider energy efficiency. Figure 3 shows the flooding protocol makes sense for small radio range, but not for large ranges. Flooding's energy consumption quickly increases due to the redundant swaps of data already sent to the base, while the history-based protocol grows very slowly from 1.0X at 1km radio range to 1.04X at a radio range of 15km.

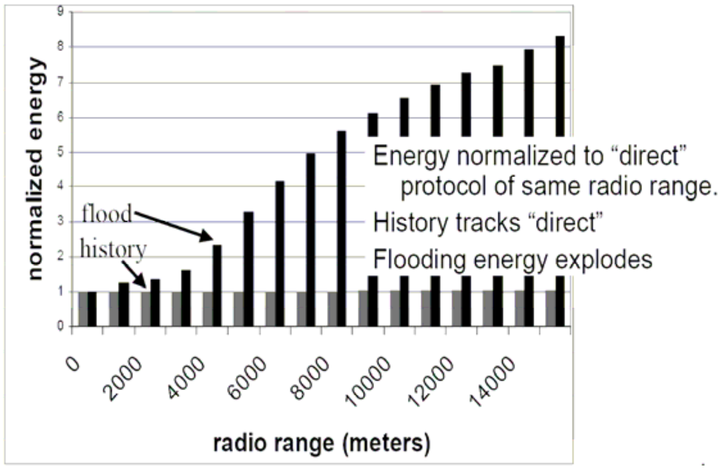


Figure 3: Energy Results

Thus far, ZebraNet has been deployed in Kenya to monitor zebras and has provided us with insight into zebra behaviors, lifestyles, and velocities. Overcoming the challenging terrain, researchers have been able to tag zebras and fly over them with a plane to record information. What they have found includes behaviors like one male typically traveling with many females. Some additional behaviors mostly include herbivores grazing near water and resume ambient motion after drinking, as it is rare for zebras to run towards or away from something.

B. Great Duck Island

On Maine's Great Duck Island (GDI) [2], biologists have placed sensor devices in duck's underground nests and on four inch stilts placed just outside of duck burrows for a nine month monitoring period. This deployed network of thirty-two nodes continuously streams data onto the web (http://www.greatduckisland.net). Figure 4 shows these underground sensor devices (1), the four inch stilts (2), and the gateway node (3) which transmits all of the information from these devices to a laptop in the research station (4). The data is then sent to a satellite (5) and ultimately to an Intel Research lab at Berkeley California.

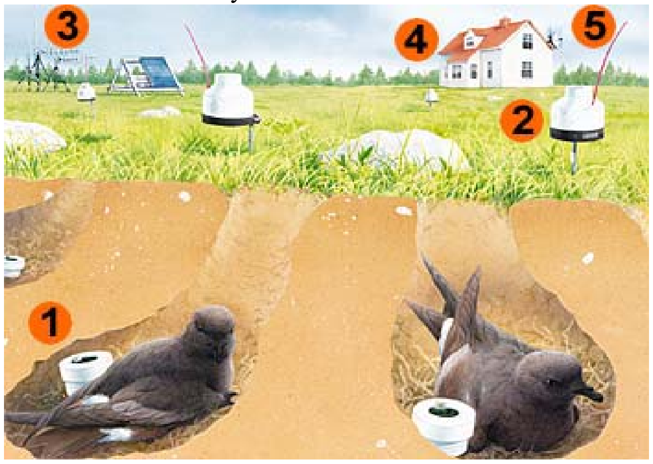


Figure 4: Great Duck Island Sensor Network

The GDI system is a tiered architecture depicted in figure 5. Each layer has storage to protect against data loss during power outages. The lowest level consists of small battery powered sensors nodes collecting data. The sensor nodes chosen are UC Berkeley mica motes. The accuracy of these devices is remarkably within three percent of the actual value. The gateways transmit sensor data from the patch to the transit network. Since these gateways are solar powered, they are always on. The base station provides WAN and data storage. Replicas of the database are necessary for remote users. The base station connects to the database replicas across the Internet and the data is finally displayed through a user interface.

The most efficient routing for low duty cycle sensor network is sending data to the gateway on scheduled periods, synonymous to one-direction communication. The habitat information discovered at GDI consists of temperature, relative humidity, solar radiation, voltage utilization, and live sensor readings, all of which is the data collected from weather motes and burrow motes.

The following tracking methods are applicable to animal tracking, namely Binary Network Model [28] and Distributed Predictive Tracking Algorithm [15].

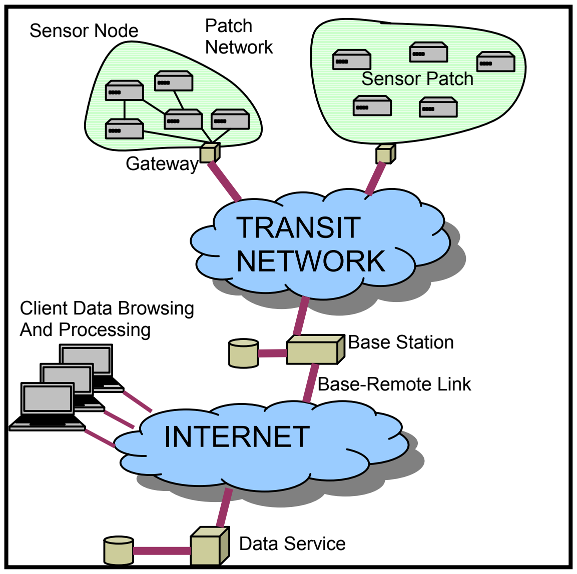


Figure 5: GDI System Architecture

C. Binary Network Model

The Binary Network Model proposed by researchers in Dartmouth College and CSU Los Angeles [28] requires sending one bit of information to a central computer. This bit of information carries the direction of the object, whether it is moving towards (+) or away from (-) the sensor, as shown in figure 6. The star denotes the current location of the animal.

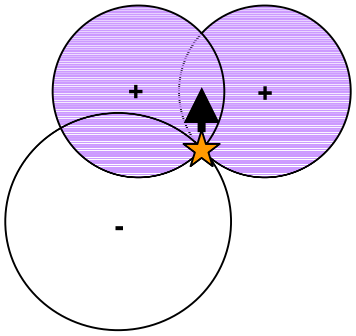


Figure 6: Binary Sensor Network Geometry

Adding another bit will provide the actual location of the object. The advantages to this model are broadcasting single bits over the network is very feasible and the trajectory prediction error is low. The disadvantages to this network are only one animal can be tracked at a time and there are no failure recovery considerations, which is not optimal for our sensor tracking model.

Many additional tracking techniques have been proposed [13] [14] [16] to track moving targets. Researchers have formulated an estimation of signals received from sensors which calculate time dependent measurements to represent the location and characteristics of the target. One approach is to adopt the classic Bayesian formulation which computes the measurement and communication costs to minimize tracking failure.

There is another research study [18] to track and understand the temporal relationship between objects in tracking multiple target environments, such as monitoring the interaction of numerous animals. It is also very useful to investigate predatory activities such as wolves trying to attack his prey of vulnerable, innocent lambs.

D. *Distributed Predictive Tracking Algorithm*

Researchers at RPI developed a distributed protocol for target tracking in sensor networks [15]. This algorithm organizes sensors in clusters and uses sensor triplet triangulation to predict the target's present location, as shown in figure 7. The target's next location is predicted using a linear predictor based on the last two actual locations of the target.

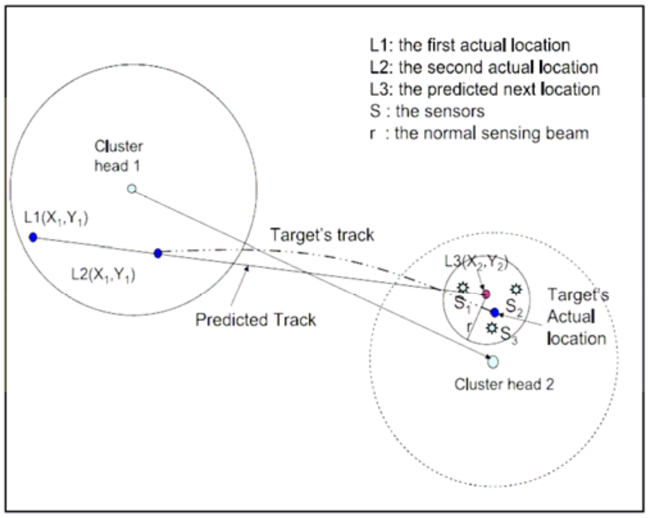


Figure 7: Sensor Triplet Triangulation

Figure 7 also brings up a notion of normal beam and high beam. These sensors are usually set to normal beam if the cluster is active. Most clusters are in hibernation mode when the cluster head does not sense the target in its cluster. High beam is only activated when the target is lost and the cluster radius of the predicted location does not sense the target.

The predictive mechanism works without always having to consume more energy by having the sensors operate with high beam. There is a relay message sent from cluster head to cluster head which makes this mechanism successful. Each cluster head activates the appropriate sensors before the target arrives. As the target is leaving the original cluster head's group, that cluster head alerts the next cluster by handing over a target descriptor to the next cluster head. This target descriptor consists of the target's identity, the present and predicted locations, and the time stamp of when the target entered the first cluster. By having this information relayed, the clusters are able to obtain facts about the animal before it even arrives in its cluster.

III. TRACKING MOBILE SENSOR NODES

A. *Hardware*

Both ZebraNet and GDI systems utilize the well-known Berkeley Mica motes, shown in figure 8. The size of the mote is 2.25 x 1.25 by 0.25 inches. Due to their size, these rectangular devices only run on two AA batteries.

Figure 8: Berkeley Mica Mote

The MICA mote uses an Atmel ATmega 128L processor running at 4 megahertz, which is an 8-bit microcontroller that has 128 kilobytes of flash memory to store the program. The ATmega only consumes 8 milliamps when it is running, and 15 micro amps in sleep mode. This mote is noted as being one of the most commonly used and energy efficient devices.

B. *Tracking Algorithm*

When dealing with mobile sensor nodes, the sensor range and average speed of target become very important factors for tracking performance. There are four tracking algorithms that we researched which apply to habitat monitoring. These four strategies include naïve activation, randomized activation, selective activation based on prediction, and low duty cycle operation. The latter two algorithms will be described in the next section due to their predictive behavior.

In naïve activation, all nodes are in tracking mode all of the time. Although this has intensive energy consumption, this strategy yields the best possible quality of tracking.

Randomized activation has, on average, a fraction of p nodes active and in tracking mode. Since each node has a probability p of being on, energy consumption improves slightly, but the quality of tracking decreases.

C. *Prediction Algorithm*

Selective activation based on prediction is when a small subset of nodes is in tracking mode at any given point in time. All other nodes remain in communication mode and may be alerted by signals from tracking nodes. The nodes in tracking mode can also predict the next position of the target which will warn the next subset of nodes that the animal is moving into their area. This notion of predicting the next position is depicted in figure 9: X_a is the actual position of the animal at a certain point in time; X_b is the animal's believed location; X_p is the predicted location. By using prior history of X_b, X_p is determined. The selective activation based on prediction algorithm utilizes the convex hull of both the actual and predicted circles to predict the position of the animal at time t+1. Using a good prediction algorithm has shown that this algorithm has orders of magnitude energy savings and negligible difference in tracking quality.

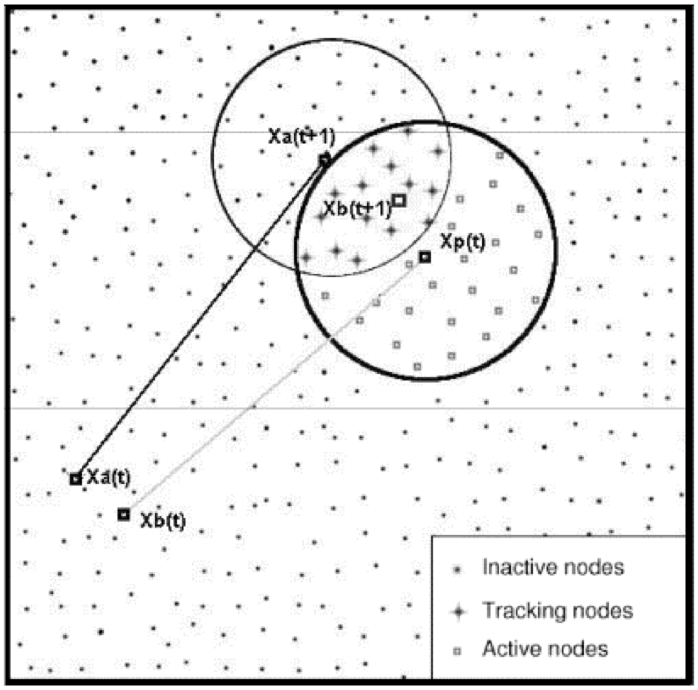


Figure 9: Selective Activation Based on Prediction

Another energy efficient algorithm is low duty cycle operation. This algorithm has the entire sensor network pulsating on and off, introducing a low power operating mode with wakeup. This power saving mode requires less power and is much different than being off since the nodes must wakeup to external stimuli. One application of this algorithm is the Frisbee model presented by researchers in UCLA and USC. The Frisbee model has a "wakeup wavefront" that wakes up nodes in the predicted path of the animal. Figure 10 shows Bubba the Lobster traversing along its yellow path. The blue circular regions are the active sensor nodes, sensing and tracking Bubba's movement. The radius of these blue circles is proportional to the speed of the animal. All of the nodes outside these regions are in power saving mode. Once Bubba moves from one sensor cluster to another, the previous sensor cluster reverts back to power saving mode.

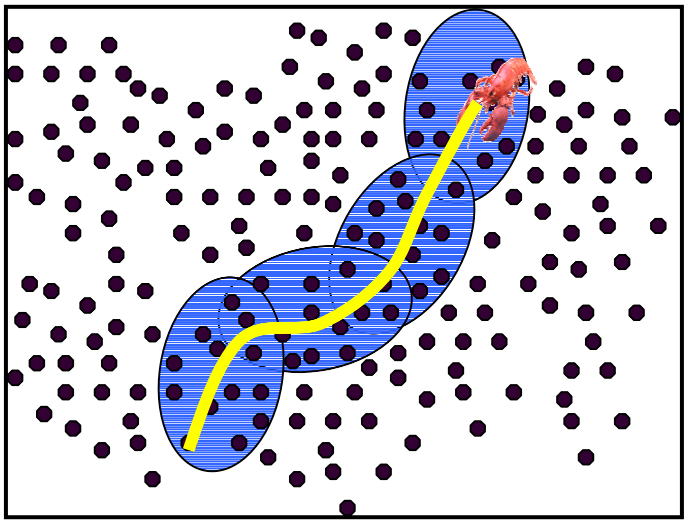


Figure 10: Frisbee Model

When dealing with tracking algorithms, time synchronization is critical. As energy efficiency increases, the quality of tracking decreases. However, selective activation based on prediction and low duty cycle operation algorithms have shown that the energy conserved is considerably higher than other algorithms, such that the quality of tracking becomes negligible.

IV. RECOVERY

In this section, we discuss our proposed failure recovery scheme in tracking a moving target. This failure can occur due to issues such as network failure, prediction error, or node failure. The basic protocol for tracking a moving target is to select and wake up a group of nodes based on the movement of the target. The current group of nodes hands off the current monitoring task to the next group of nodes. However, the next group might never wake up and see the target enter their monitored area due to network failure or the target selects to go towards the opposite area of the selected group. For example, the current group predicts the target is heading north, but the target may make a drastic direction change and head northeast. Hence, prediction error occurs. Failure is detected when the group leader, who is currently monitoring the target about to enter another area, does not receive an acknowledgement from the next selected group. The newly selected group is where the target was predicted to go, so this results in a lost confirmation.

The possibility of not being able to find a target could be a direct result of the following conditions:

1. Network failure occurs when packets from the cluster head, which is currently monitoring the target, fails to reach the next cluster head. This results in a lost acknowledgement.
2. Prediction failure of the next location where the animal is heading. For example, the animal may suddenly change its path resulting in an inaccurate predicted location.

3. Failure to recognize multiple targets at once. For example, a cluster might be monitoring two moving horses, which are standing next to each other. The cluster nodes might be mistakenly recognizing the two horses as one entity. When the horses are about to leave the monitored area and enter another cluster group, the horses decide to split up. This split results in two distinct directions and different cluster groups. The current sensor nodes realize there is more than one target, but they do not notify the second cluster head. Hence, one of the two targets is not monitored.
4. The sensor node hardware may malfunction or the battery is weakening. Someone must be available to perform routine maintenance checks on the deployed hardware to ensure the integrity of the system.

We propose a recovery algorithm to take advantage of hierarchical clustering. The algorithm establishes a search region and wakes up the relevant nodes. Simultaneously waking up the necessary nodes at one particular region will reduce the network traffic and avoid the unnecessary extra energy cost.

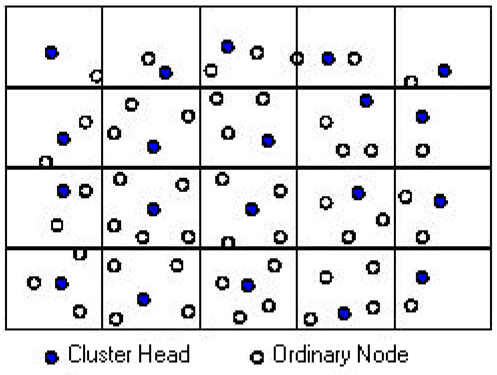


Figure 11: Each grid visually represents a group of clusters

Establishing a search area is much simpler if the nodes positions are known. GPS might be required to determine the precise position of each node. However, our proposed algorithm does not require prior knowledge of the node positions or GPS. Additionally, targets are not exactly tagged as in ZebraNet [6] because the tag also functions as a node that wakes up every sensor node on its path. The tag sends a signal to wake up sensor nodes, which are currently surrounding the target. Therefore, tagging the target actually removes the risk of prediction failure of where the target's next location and thus, the recovery procedure for lost targets will not be necessary. This approach is less opportune because it is not practical to tag every animal in wildlife and potentially disturb the life of the animals. Moreover, tagging animals defeats the purpose of the habitat monitoring.

Figure 11 illustrates the idea of having one cluster head per cluster, or grid. Each cluster consists of a randomly selected number of nodes, one of which is the cluster head. The cluster head is shown in blue, while all other nodes are white.

A. Clustering

Our proposed algorithm takes advantage of hierarchical clustering structures for more efficient utilization of resources, simpler routing protocol, and management [19]. Clustering also allows some nodes to play watchdog or managerial roles over other nodes. This manager node is called the **cluster head CH.** Each cluster head joins another cluster and selects another cluster head. This process continues until it forms a tree with one root, as shown in figure 12. This approach is known as **hierarchical clustering technique**.

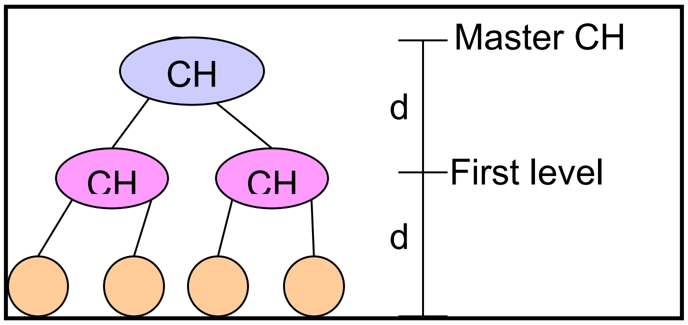


Figure 12: Hierarchical Clustering Technique

After the initial deployment, sensor nodes often are compelled to organize themselves into a group of clusters and select a leader for every cluster. Several protocols exist on how to form a cluster. The initial step in forming a cluster is to select a cluster head. Once a cluster head is selected, each node can join the cluster by contacting their closest cluster head. The distance from the sensor node to the cluster head is the number of hops between them. For instance, a sensor node might prefer the cluster head that can be reached in three hops rather than another cluster head ten hops away. The cluster head selection protocol is based on unique node identifiers. This identifier can initially be assigned randomly to the nodes [29] [30]. In selecting a cluster head, a node may declare itself as the cluster head if it possesses the highest ID among other nodes that have not been claimed by another existing cluster head. Several cluster heads closest to the base station may become the gateway. Currently there is no theoretical proof that guarantees the quality of clustering. Clustering is just a constant bound to measure the effectiveness of reaching and discovering all nodes in the network.

Some may argue that the disadvantage of becoming a cluster head is a bit more energy consuming because the cluster head has more responsibility than other ordinary nodes. The author describes in [20] that every node should take turns being a cluster head during every periodic cycle. The hierarchical clustering approach could be implemented without significant extra energy costs if a node would not repeatedly be selected as the cluster head. Hence, they recommended the formula below where each node in the cluster takes turns becoming the cluster head with probability P_1 and $c = 3.06a\sqrt{\lambda}$ with parameter a.

$$p_1 = \left(\frac{1}{3c} + \frac{\sqrt[3]{2}}{3c(2+27c^2+3\sqrt{3c}\sqrt{27c^2+4})^{1/3}} + \frac{(2+27c^2+3\sqrt{3c}\sqrt{27c^2+4})^{1/3}}{3c} \bullet \frac{1}{\sqrt[3]{2}} \right)^2$$

[22] [23] [24] A tree-based clustering approach is also used to improve energy consumption and reduce traffic simply by having only a cluster of nodes monitor the activities of a moving target. The subordinates only report to their leader, the cluster head. Data is routed among the cluster heads to the base station or a designated destination. Kung also proposed a model that characterized the nodes into geographical group and weighted the edges to represent the movement of the target. Zhang and Cao also proposed that the trees are actively modified, pruned, and expanded to accommodate the network traffic, density of the network, and the routing path. However, our recovery algorithm would work best with less frequent reconstruction and modification.

There are also [24] [25] [26] a number of proposed algorithms to determine the location and radius of the monitored region by a cluster of nodes. We modify the algorithm such that after the initial deployment process, every cluster head reports the radius of its monitored region to its immediate parent. The parent computes the radius of its region based on the information from its subordinates and it propagates its newly computed region radius until it reaches the root. For example, all city governors report their respected city size to their immediate superior, the county governor. All county governors compute the size of their county based on information from their subordinates and report the radius of their area to a higher superior, the state governor. The process is repeated until it reaches the president, the root of the hierarchy tree.

The missing target can be recovered quickly by performing **Space Decomposition Search.** The basic concept of this algorithm is to take advantage of hierarchy clustering and wake up all of the nodes in the area at once and perform a single simultaneous search. The root will wake up its subordinates and the order will be propagated to leaf level. Then those leaf nodes perform an instantaneous search and will propagate the finding result to their superior until it reaches root level. The running time of this search algorithm is O (log N).

B. Computing Velocity

The radius of the search region and the number of nodes involved in the search event are determined by how far the target would travel. One of the challenges is determining the speed of the target. In [15], the author proposes a method of computing the target's speed based on the detection time between sensors and the position of nodes. Another approach described in [16] shows how sensors should determine the underlying target state of its position and velocity based on the sensor measurements up to time t. A prediction-based tracking algorithm is described in [17] where the authors use the velocity estimation of the target to select which sensors to query. Other approaches have been developed using Kalman Filters, which assume Gaussian observation models and linear state dynamics, and Bayesian Filtering techniques.

Once the velocity of the target is acquired, travel distance of the target can be computed with **distance d = velocity v * time t**. The question is how to decide the condition of the velocity. For instance, it is very difficult to determine if the target is running, walking slowly, or moving in regular speed. It also depends on the radius and the type of animal. A horse runs much faster than a cow. Velocity changes are unpredictable. Determining the condition of the target's speed will be easier if the target carries a wireless collar or tag like in ZebraNet [6]. Hence, more helpful target information can be downloaded from the tag to the local sensor and velocity computations become more precise. Target information also includes the maximum speed and behavior of the target. However, tagging the target with some type of wireless device can be very costly and difficult.

Without proper information it is difficult to determine the type of target's speed, whether the target is running or moving in its regular speed. We make an assumption that there is no secondary party or adversary who might alter the speed of the target, like running away from its predator. In addition, the current cluster head, which is the leader of the current cluster monitoring the target, might get more target description information from the previous cluster head. Even then, such information is not enough to determine the speed condition and it can vary between target types, whether the target is a horse, cow, deer, etc.

Hence we need to compute the velocity $V_{running}$ if the target decides to run. A CH has information of the current velocity $V_{current}$ from its subordinates and $V_{previous}$ from the previous CH. Therefore, the current CH will use the highest velocity value.

$$V_{current} = \max(V_{current}, V_{previous});$$

$V_{current}$ is bounded by at most 2 * $V_{current}$, where $V_{current} < 2 * V_{current}$. We assume that a target increases its speed by at most twice the current speed as long as the target does not feel threatened and $V_{current}$ is the average velocity when the target is in the current cluster monitored area. In the nonthreatening situation, the target is more likely moving in constant speed. Therefore, the probability $P_{running}$ is likely to be less than ½. We can compute the expectation of $V_{running}$ using Bernoulli random variable.

$$E[V_{running}] = (1 - P_{running})V_{current} + 2 P_{running} V_{current}$$

for $P_{running} < ½$

Next we compute distance d using $V_{current}$.

$$d = V_{current} \bullet t$$

The target may accelerate over 2 * $V_{current}$ and maximum velocity V_{max} > 2 * $V_{current}$. Naturally, no living being could run forever without stopping, so at some point, it will eventually stop. The longer or the faster they run, the more likely they will stop. Thus, if a target runs with maximum speed for a long period of time, then the probability of the target stopping increases. We apply the finite Markov Chain to compute the probability the likelihood of a target will stop moving. Suppose $\mathbf{P_v}$ is the probability when the target reaches the maximum speed before it stops and **v** is the initial speed as the target enters the monitored region by a designated cluster of nodes. $P_{accelerate}$ is the probability of increasing the velocity and (1 - $P_{accelerate}$) is the probability of decreasing the velocity, where $P_{accelerate} < ½$.

*Let **I** be an event when the target increase its velocity at time t = 1.*

*Let **R** be an event when the target runs with full speed for $V_{max} \geq 2 * V_{current.}$*

Let $\mathbf{V_i}$ be the velocity at time t.

$$P_v = \Pr(R \mid V_0 = v)$$
$$P_v = \Pr(R \cap I \mid V_0 = v) + \Pr(R \cap \neg I \mid V_0 = v)$$
$$P_v = \Pr(I \mid V_0 = v)\Pr(R \mid I \cap V_i = v) + \Pr(\neg I \mid V_0 = v)\Pr(R \mid \neg I \cap V_0 = v)$$
$$P_v = P_{accelerate}\Pr(R \mid V_0 = v) + (1 - P_{accelerate})\Pr(R \mid \neg I \cap V_0 = v)$$
$$P_v = P_{accelerate}\Pr(R \mid V_0 = v+1) + (1 - P_{accelerate})\Pr(R \mid V_1 = v-1)$$

Since ($I \cap V_0 = v$), ($R \mid V_1 = v + 1$), and ($R \mid V_0 = v - 1$) then

$$P_v = P_{accelerate} \Pr(R \mid V_0 = v + 1) + (1 - P_{accelerate})(R \mid V_0 = v - 1)$$
$$P_v = P_{accelerate} P_{v+1} + (1 - P_{accelerate}) P_{v-1}$$

Then,

$$P_v = P_{accelerate} P_{v+1} + P_v + 1 - P_{accelerate} P_{v-1} = 0$$

where $P_0 = 0$ and $P_{\max} = 1$

Applying the linear recurrence will uncover the characteristic equation regardless of the indexing scheme. The characteristic equation of this recurrence is:

$$r^2 P_{accelerate} - r + (1 - P_{accelerate}) = 0$$
$$r = \left(\frac{1 - P_{accelerate}}{P_{accelerate}}\right)$$

Providing the characteristic of the equation (1 - $P_{accelerate}$) / $P_{accelerate}$, we compute the probability of the target completely stopping at some point in time t. Let **a** be the acceleration where **S** maximum speed = a + current velocity V_i.

$$\left(\frac{1 - P_{accelerate}}{P_{accelerate}}\right)^a = \left(\frac{1 - P_{accelerate}}{P_{accelerate}}\right)^0 P(V_0 = 0) + \left(\frac{1 - P_{accelerate}}{P_{accelerate}}\right)^S (1 - P(V_0 = 0))$$
$$\left(\frac{1 - P_{accelerate}}{P_{accelerate}}\right)^a = P(V_0 = 0)1 + \left(\frac{1 - P_{accelerate}}{P_{accelerate}}\right)^S (1 - P(V_0 = 0))$$
$$P(V_0 = 0) = \frac{\left(\frac{1 - P_{accelerate}}{P_{accelerate}}\right)^a - \left(\frac{1 - P_{accelerate}}{P_{accelerate}}\right)^S}{1 - \left(\frac{1 - P_{accelerate}}{P_{accelerate}}\right)^S}$$

From the formulas, the higher S gets, the more likely the target will eventually stop.

The most important aspect of determining the radius search area is the time t and distance d that a target will travel. Time t is a parameter and a deciding factor of the distance length. Parameter t depends on many factors, such as the node's computing power, network traffic, and the furthest distance a target can travel from the last seen position. Naturally, any living being cannot continue running without stopping, but the question is how far the animal will travel before it decides to stop. Hence P(V_0 = 0) can be utilized to determine time t. The lower the value of P(V_0 = 0), the more likely the target will travel more distance and value t should be increased.

C. Popular Place

Once value d is determined, then d needs to be multiplied by 2 simply because there are two opposing possible directions and the target may select either one of these directions. Next, that 2d value is propagated from the CH, where the target was last seen, to its superior. The superior compares the value of d with the diameter of its region. If the diameter of its region is roughly equal to 2d, then it will wake up all of its subordinates and leaf nodes to perform a search event. Otherwise, the 2d information is propagated to the higher superior until the proper region is found. Figure 13 shows the 2d value with the black circle being the last position the animal was seen, and the orange circles being the next popular places with their hop counts denoted.

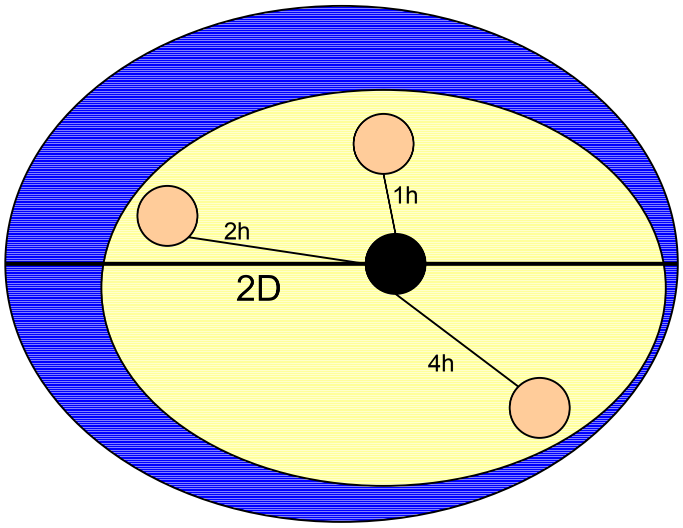


Figure 13: Tracking Algorithm based on popularity

Once the proper region is acquired at a high-rank CH, it sends request to its subordinates to find the popular place. One might ask what characterizes a popular place. Take for instance, tourists visiting Boston, MA. A popular place would be where the tourists tend to go. Touristy places would include Faneuil Hall, Boston Common, New England Aquarium, or the USS Constitution. Now that we have four of the most popular places, we can try to locate a tourist by first asking these four places. By asking the most popular places, you are minimizing the amount of energy in a sensor network by avoiding the energy-consuming operation of turning all sensors on. This is the convention of popular place.

A larger scale example of popular place focuses not only on one city in one state, but in one country with numerous states. Consider tourists visiting the United States of America. First time visitors may want to visit three or four states, including Massachusetts, Florida, California, and Illinois. By using the popular place convention, if a tourist got lost, we would ask these 4 states first. If the tourist had visited one of these states, then we would traverse through the chain of further investigating which county, city, or neighborhood the tourist was last seen. This is how the popular place tree can grow very quickly but at the same time reduces energy of every sensor in all fifty states being activated.

We apply this approach to wildlife animals since they tend to visit places where food and water is available. Other factors which commonly dictate animals' locations are those which provide ideal climate or some degree of safety from predators. This is where animals typically prefer to rest, nest, or reproduce.

D. Broadcasting

Once the popular places are defined, then each CH of popular places send beacon packets and computes the distance, or the number of hops, to the CH where the target was last seen. Each beacon packet contains the descriptions of the targets.

After receiving packet from the CHs of popular place, the CH where the target was last seen broadcast packets with the life span equal to the distance or hops from one of the popular places. The purpose of broadcast is to establish and wake every node in the search region. Packet life span is decreased by one in every forward event. Flooding based mechanisms in the hierarchy tree are described in [11] [12] and the authors claim performing search operations with flooding based techniques is energy efficient. In fact, the flooding technique in [12] is specifically designed to support querying information in sensor networks for bird habitat monitoring.

Rules of broadcasting:

1. A node only forwards packet p if and only if the new packet sequence is greater than the previously received packet sequence.
2. The packet sequence is decreased by one at each node.

$\omega(u, v)$ is the packet sequence from u to v. $s(n)$ is the current sequence value of the packet at node k.

Lemma 1

Let $G = (V, E)$ be an undirected graph with source vertex s, and all edge $(u, v) \in E$. Cycles are avoided when $\omega(u, v) > s(v)$.

Proof:

We prove the lemma by using contradiction. Let us say that there is no cycle between u and v and $\omega(u, v) < s(v)$. If packet p travels between node u and v for $s(u) = n$ and $\omega(u, v) = n-1$, then $\omega(v, u) = \omega(u, v) - 1 = n - 2$. u forwards p to v and $\omega(u, v) = \omega(v, u) - 1 = n-3$. There is clearly a contradiction in our earlier assumption when $\omega(u, v) = n-3$ and $s(u) = n$, so $\omega(u, v) < s(u)$ and cycle is exist. Hence, contradiction establishes the claim.

Here is the second proof by contradiction. Let us say that there is no cycle between u and v and $\omega(u, v) < s(v)$. $\exists$ z such that edge $(z, u) \in E$. Initially, u forwards packet p to v. Packet p travels and reaches z. Later, v receives packet p from z and u forwards packet p to v. Since $s(u) = n$ and $\omega(u, v) = n-1$, then $\omega(z, u)$ is at most n-3. Therefore, $\omega(z, u) < s(u)$ and cycle occurs. Hence, this contradicts the earlier assumption.

Once the broadcast operation is executed, the search region is established and all the leaf level nodes are awake. For some period of time, all leaf nodes perform a simultaneous search operation to find the lost target based on target description. Subsequently, all leaf nodes will report their findings to their cluster head and then this information will be propagated to the cluster head responsible for the area where the target was last seen. If the target cannot be found, this “master” cluster head will report these results to the base station. Otherwise, it will hand off the monitoring task to the new group, which will attempt to find the target.

E. *The Recovery*

In this section, we discuss the recovery procedure when the search region fails to capture the target and the target’s actual position is outside the search region.

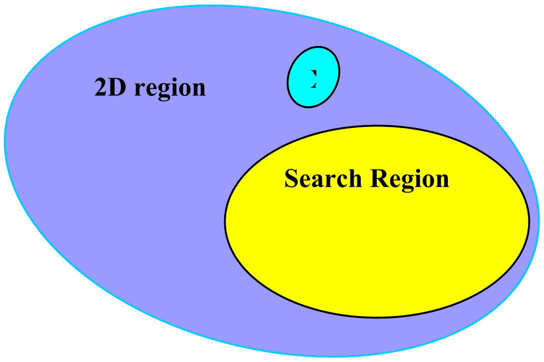


Figure 14: X is the actual position of the target.

These failure occurrences are caused by the lack of proper information in finding popular places, especially when nodes are recently deployed in the new environment. Performing a recovery operation by finding the parameters of where the lost target might go is not effective if the nodes are recently deployed because there is not enough data to capture the places that have been previously visited. For example, during the first minute after the deployment of nodes, a zebra passes the monitored region and most sub-regions are at most visited once. Hence, every place has the potential in becoming the popular stopping place for the zebra and it is difficult to determine the popular place given very limited information.

Our approach is to perform another search using the maximum search area to find the popular place. The 2D search region is $V_{running} * t * 2$. All nodes in the radius 2d are activated and perform the search. The success of the search relies heavily on time t. Thus, finding the proper t is very critical for a successful search. Figure 14 illustrates 2D as the maximum search area, X as the actual location of the animal hiding, and the yellow region as the location the animal was last seen.

V. SIMULATION

A. *Overview*

We simulated and compared Yang’s and Sikdar’s [15] battery utilization algorithm against ours. The purpose of comparing these two algorithms is to demonstrate the improvement of energy efficiency. Yang and Sikdar offered a simple, straightforward solution to find the missing target: in order to locate the missing target, a group of sensors, of radius X meters from where the target was last seen, is activated. There is no clear description whether it is a group clusters or just one single cluster that is selected and activated. Hence, we simulate two circumstances and compare the energy cost when:

1. A single cluster per round is activated.
2. A group cluster per round is activated: 3 nodes per group since the minimum number of required nodes per cluster is 3.

We made several assumptions that both algorithms require equal amounts of time to perform a search, and the same hardware is used. Therefore, both algorithms need the same amount of energy to activate a node and to execute the search.

The simulation is categorized into several scenarios and we compare the performance between both algorithms. The scenarios are:

1. Both algorithms have the equal amount of nodes to perform the search task.
2. When our algorithms miss the target, they retry using the maximum predicted number of nodes needed (all nodes in 2D region).
3. The actual number of required nodes is less than the predicted number. Hence, the search region computed by our algorithm might be too large compared to Yang’s and Sikdar’s. We compare the cost of energy usage when the number of nodes generated by their algorithm is 1/3, 1/6, and 1/10 less than ours.
4. Comparing the worst case scenario in which the entire nodes in the monitored region have to be activated. Our algorithm will perform the default task - 2D task - and space decomposition. The algorithm will continue expanding its search region until the entire nodes are awake.

B. *Equations*

The first assumption that Yang and Sikdar introduce is expanding the search region by only activating one cluster group at a time. Let **E** be the variable for energy, **n** is the actual number of nodes activated or required to perform a successful search, **m** nodes per cluster, time **t** needed to activate a sensor node, and all sensors consume **Π** of energy for every t. The formula to compute cost energy for Yang’s and Sikdar’s algorithm is:

$$E = \Pi\, t\, \Sigma\, i \text{, where } 0 \leq i \leq n/m,$$

The second assumption is the region expansion by a group of clusters.

$$E = \Pi\, t\, \Sigma\, m^{i} \text{, where } 0 \leq i \leq n/m;$$

Our algorithm activates nodes simultaneously. Hence,

$$E = \Pi\, t\, n$$

In our simulation, for the purpose of simplicity, Π and t are equal to 1.

C. Simulation Environment

The simulation is developed and runs in the UNIX environment, Pentium 4, and is written in Ansi-C.

D. Results

In this section, we compare and contrast how much energy is utilized by using both recovery algorithms. First, we compare the energy savings when Yang's and Sikdar's activate a single cluster per round. Each chart below demonstrates the energy consumption based on the number of nodes involved during the recovery operation. Each line in the graphs represents how much energy is used when X number of nodes is involved[1].

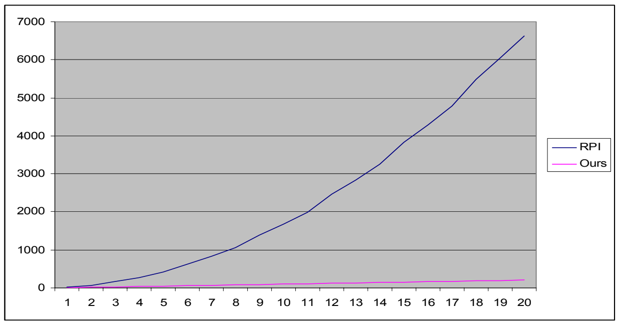
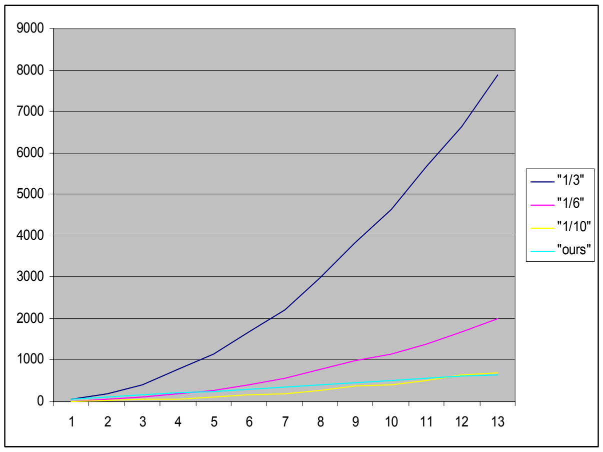


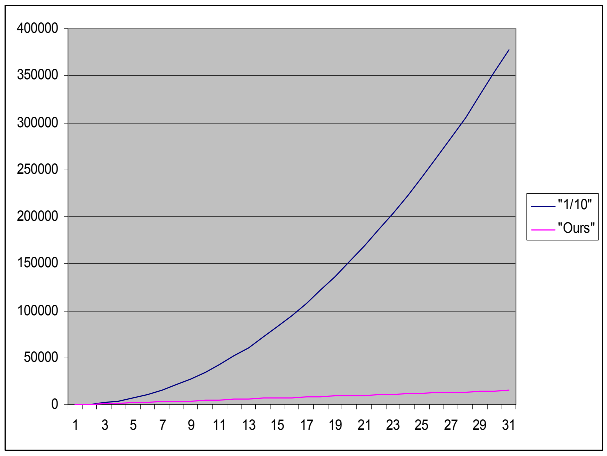
Figure 15: A single cluster per round is activated

We believe that simultaneously activating the entire nodes in the search region will significantly minimize the energy consumption, as it is shown in figure 15. The energy used amplifies as the number of nodes involved increases; however our algorithm consumes much less energy.

[1] In all graphs shown, the x-axis represents the number of experiments performed, while the y-axis represents the units of energy used.



(a)



(b)

Figure 16: Worst case scenario

The worst-case scenario is when a computation error occurs and our search region is 10, 6, and 3 times larger than Yang's and Sikdar's. Figure 16 (b) shows the worst case scenario despite the error calculating search region, where our algorithm still manages to outperform the other approach. Figure 16 (a) also shows that, with more nodes involved, the energy used by our algorithm remains low compared to the search by region, which is 10 times smaller than our algorithm's search region.

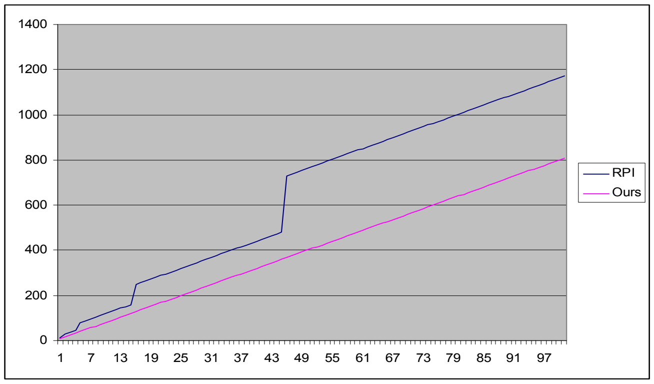


Figure 17: A group cluster per round is activated

Secondly, consider a different scenario in which Yang's and Sikdar's algorithm activates a group of clusters per round. Figure 17 shows that our algorithm consumes less energy.

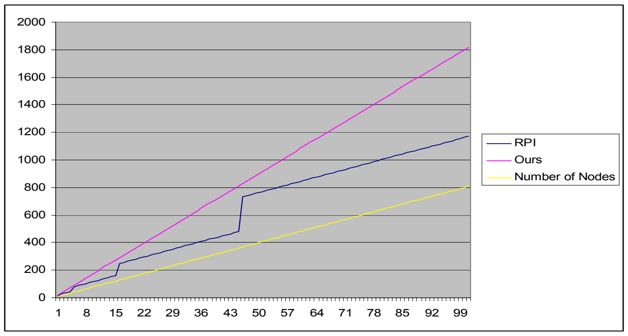


Figure 18: Entire 2D region utilized

However, if our algorithm fails to establish the proper search region and requires 2D region to find the target, then our algorithm will consume more energy than Yang's and Sikdar's algorithm, as shown in figure 18.

E. Observations

One of our biggest challenges is to determine the characteristic and behavior of the targets. Since we focus on larger sized wildlife animals (such as a horse or cow), their velocity change is more predictable than smaller animals. Smaller and lighter animals like rabbits have the ability to make an abrupt halt after running full speed without deceleration. In contrast, heavier and larger animals like horses need some time to decelerate before making a complete stop. Hence, it is very unlikely that our algorithm will make such a significant error computing the search region.

Another obvious observation is that simultaneous node activation significantly reduces energy cost compared to that of increment node activation.

The key to a successful recovery operation determining the appropriate time *t* to compute **distance *d* = time *t* * velocity *v*.**
Parameter *t* depends on many conditions such as the processor power, network traffic, target's behavior, and so on. The target's behavior can vary depending on the condition of the environment. There are many factors that could drive the target to behave differently and it is very difficult to make such a prediction with very limited information. That is why another discipline, such as machine learning, could be quite valuable in learning the pattern behavior of a particular target.

VI. CONCLUSION

Tracking wildlife has been a hot topic for the past few years. There has even been talk [27] from lawmakers to force the government to track livestock nationwide. An identification system has been introduced to implant an RFID chip – VeriChip – into the animals. The first implanted chip on a human happened soon after 9-11-2001. The purpose of tracking livestock is to track infected livestock. This syringe-injected chip is the size of a grain of rice.

In this paper, we proposed another approach in tagging animals not only for tracking diseases, but also to find out more about the animal and its behavior. We analyzed the tracking methods for mobile sensor networks and proposed a recovery method to search the system when the target is lost.

VII. FUTURE WORK

We are currently working on testing our algorithm and making any improvements as we see fit. We would like to discover whether there is a softer tradeoff between energy consumed and prediction accuracy. In addition, we are also interested in collaborating our study with other disciplines such as Machine learning or Artificial Intelligence to compute a more precise search region. These disciplines offer different approaches to solve problems by analyzing a data set collected from particular events and learning the pattern of certain behaviors. Our biggest goal is minimizing energy consumption when searching for the target. By having most sensor nodes in the system in hibernation, the risk of losing the target may increase. Another possible area of work is testing this on the field and seeing how multiple animal tags interact with the sensor network we have described. This could be a truly fun and active task.

ACKNOWLEDGMENT

We would like to thank all those who participated in our wildlife tracking efforts. Special thanks to Edwin for his continuous assistance in this project.

REFERENCES

[1] D. Ganesan, A. Cerpa, W. Ye, Y. Yu, J. Zhao, and D. Estrin. *Networking Issues in Wireless Sensor Networks.* UCLA, 2003.

[2] A. Mainwaring, J. Polastre, R. Szewcyk, D Culler, and J. Anderson. *Wireless Sensor Networks for Habitat Monitoring.* Berkeley, 2002.

[3] A. Cerpa, J Elson, M. Hamilton, and J. Zhao. *Habitat Monitoring: Application Driver for Wireless Communications Technology.* UCLA, 2000.

[4] R. Szewczyk, J Polastre, A. Mainwaring, and D. Culler. *Lesson From A Sensor Networks Expedition.* University of California at Berkeley, 2004.

[5] H. Wang, D, Estrin, and L. Girod. *Preprocessing in a Tired Sensor Network for Habitat Monitoring.* UCLA, 2002.

[6] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Peh, D. Rubenstein. *Energy-Efficient Computing for Wildlife Tracking: Design Tradeoffs and Early Experiences with ZebraNet.* Princeton University, 2002.

[7] J. R. Polastre. *Design and Implementation of Wireless Sensor Networks for Habitat Monitoring.* University of California at Berkeley, 2003.

[8] J. Elson and D. Estrin. *Sensor Networks: A Bridge to the Physical World.* UCLA, 2004.

[9] D.Estrin, D. Culler, K. Pister, and G. Sukhatme. *Connecting the physical world with pervasive networks.* IEEE Pervasive Computing, Pages 59-69, January 2002.

[10] J. Anderson. *Pilot survey of mid-coast Maine seabird colonies: An evaluation of techniques*. In Report to the State of Maine Department of Inland Fisheries and Wildlife. Bangor, ME, 1995.

[11] Y. Baryshnikov, E. Coffman, P. Jelenkovic, P. Momcilovic, and D. Rubenstein. *Flood Search Under the California Split Rule*. Operations Research Letters, Volume 32, Number 3, May 2004.

[12] N. Sadagopan, B Krishnamachari, and A. Helmy. *The ACQUIRE Mechanism for Efficient Querying in Sensor Networks.* In Proc. IEEE International Workshop on Sensor Network Protocols and Applications (SPNA), [ages 149-155, 2003.

[13] J.J. Liu, J. Liu, J. Reich, P. Cheung, and F. Zhao. *Distributed Group Management for Track Initiation and Maintenance in Target Localization Applications*. Proceedings of 2nd International Workshop on Information Processing in Sensor Networks (IPSN'03), April, 2003.

[14] J. Shin, L. Guibas, F. Zhao, *A Distributed Algorithm for Managing Multi-Target Identities in Wireless Ad-Hoc Sensor Networks.* Proceedings of 2nd International Workshop on Information Processing in Sensor Networks (IPSN'03), April, 2003.

[15] H. Yang and B. Sikdar. *A Protocol for Tracking Mobile Targets using Sensor Networks*. RPI, 2003.

[16] F. Zhao, J. Liu, J. Liu, L. Guibas, and J. Reich. *Collaborative Signal and Information Processing: An Information Directed Approach*. Proceedings of the IEEE, 91(8):1199-1209, 2003.

[17] R.R, C. Grin, and D. S. Friedlander. *Self-organized distributed sensor network entity tracking.* International J. of High Performance Computing Application, 16(3) 2002.

[18] L.Guibas, *Sensing, tracking, and reasoning with relations*. IEEE Signal Processing Mag., vol. 19, pp. 73-85. Stanford University, March 2002.

[19] S. Marti, T. J. Giuli, K. Lai, and M. Baker. *Mitigating Routing Misbehaviour in Mobile Ad Hoc Networks.* In Proc. 6th Annual International Conference on Mobile Computing and Networking (MobiCom 2000), pages 255-265, Boston, AM, ACM Press, August 2000.

[20] S. Bandyopadhyay and E. Coyle. *Minimizing Communication Cost in Hierarchically Clustered Network of Wireless Sensors.* Purdue University, 2004.

[21] W. Zhang and G. Cao. *Optimizing Tree Reconfiguration for Mobile Target Tracking in Sensor Networks*. University of California, 2004.

[22] W. Zhang and G. Cao. *Dynamic Convoy Tree-Based Collaboration for Target Tracking in Sensor Networks*. IEEE Volume 3, Number 5, 2004.

[23] H. T. Kung and D Vlah. Efficient *Location Tracking Using Sensor Networks*. IEEE wireless Communication and Networking Conference (WCNC 2003), March 2003.

[24] C. Savarese, J. M. Rabaey, and J. Beutel, *Location in Distributed Ad-hoc Wireless Sensor Networks.* Acoustics, Speech, and Signal Processing, 2001. Proceedings. (ICASSP '01). 2001 IEEE International Conference on Volume 4, 7-11 May 2001 Page(s):2037 - 2040 vol.4.

[25] W. Chen, J. C. Hou, and L. Sha, *Dynamic Clustering for Acoustic Target Tracking in Wireless Sensor Networks.* Network Protocols, 2003. Proceedings. 11th IEEE International Conference on 4-7 Nov. 2003 Page(s): 284 – 294.

[26] R. Iyengar and B. Sikdar. *Scalable and Distributed GPS Free Positioning for Sensor Networks.* Communications, 2003. ICC '03. IEEE International Conference on Volume 1, 11-15 May 2003 Page(s):338 - 342 Volume 1.

[27] Sherrie Gossett. *'Spy chips' for nation's livestock?* February 28, 2004. © 2004 WorldNetDaily.com

[28] J. Aslam, Z. Butler, F. Constantin, V. Crespi, G. Cybenko, D. Rus. *Tracking a Moving Object with a Binary Sensor Network.* Dartmouth College, CSU Los Angeles.

[29] S. Basagni. *Distributed clustering for ad hoc networks.* In Proc. '99 International Symposium on Parallel Architectures, Algorithm, and Network (ISPAN'99). Northeastern University.

[30] J. Gao, L Guibas, J. Hershberger, and L. Zhang. *Discrete mobile centers*. Discrete and Computational Geometry, 2003. Stanford University.