An Overview of Data Aggregation Architecture for Real-Time Tracking with Sensor Networks

Tian He[§], Lin Gu[†], Liqian Luo[‡], Ting Yan[†], John A. Stankovic[†] and Sang H. Son[†]

[†]Department of Computer Science, University of Virginia
[§]Department of Computer Science and Engineering, University of Minnesota
[‡]Department of Computer Science, University of Illinois, Urbana-Champaign

Abstract

Since sensor nodes normally have limited resources in terms of energy, bandwidth and computation capability, efficiency is a key design goal in sensor network research. As one of techniques to achieve efficiency, data aggregation has been extensively investigated in recent literature. Previous research on data aggregation has demonstrated its effectiveness in reducing traffic, easing congestion and decreasing the energy consumption. However few are actually designed for a real-world application and implemented in a running system. This paper describes our design and implementation of a physical tracking system, using an aggressive data aggregation architecture as one of building blocks. This architecture can be generally applied to other sensor systems, where communication efficiency is a paramount concern and networking resources are limited.[1]

1 Introduction

Traditional military surveillance systems, using long-range cameras/radars, are effective in the open terrains where direct line-of-sight is available. While in the urban areas and forests, the effectiveness of these solutions is affected by the obstacles such as tall buildings and trees. To address this issue, the military starts to use wireless sensor networks as an effective surveillance instrument to deal with the non-line-of-sight (NLOS) situations, because that sensor nodes can be deployed anywhere around an environment to provide ubiquitous surveillance. Due to the stealthiness requirement of the military surveillance systems, a tiny form factor is essential. Consequently, sensor nodes have very limited resources, suffering the bandwidth, energy and memory constraints that limit the amount of information that could be transferred. These factors are coupled with unpredictable traffic patterns and dynamic network topologies, making the task of data delivery for such networks difficult. Theoretically, for a given energy budget available in the network, the total amount of bits that can be transmitted is limited. It is desirable to have the capability to deliver more *information* with the same amount of bits over the air. It is often the case that the data represented by these bits is redundant. For example, a series of sensing readings with same values can be concisely described by an average with a zero standard deviation. As one type of data aggregation techniques, this parameterized description of the data distribution can effectively reduce the amount of data transmitted. Since data aggregation can reduce transmissions while still distributing information about the events of interests, it is deemed as a very effective resort to balance the communication needs and energy constraints.

This paper addresses the research challenges related to the data aggregation technique in real-time surveillance applications. We identify the fundamental tradeoffs that can balance the performance within a three-dimensional design space: namely the timeliness, the energy consumption and the information availability. Ideally, we desire to deliver enough information in real-time with minimal energy consumption. Unfortunately, these performance goals are often at odds with each other. For example, a swift delivery prevents a node accumulating sufficient data for energy-efficient data aggregation. It is an interesting research problem to identify a performance surface within this three-dimensional space, so that a system designer can make guided decisions to trade off among the energy, time and data availability, according to the application requirements and system configurations.

To demonstrate our approach to achieve this goal, we employ a typical sensor tracking system, called VigilNet, as a case study. We introduce the data aggregation architecture designed, implemented and integrated in VigilNet and identify the tradeoffs VigilNet provides. Since VigilNet is a typical sensor network system, we believe our studies can render insights for the system designers of similar systems.

The contribution of this work lies in the following aspects: 1) Unlike the previous approaches that mainly focus on the simulation study, we demonstrate how various data aggregation techniques can be designed and implemented practically in a real world application. 2) We reveal the impact of data aggregation on the quality of surveillance, the timeliness and the related overheads. Such an analysis can guide system designers to flex-

[1]This work was supported in part by the DAPRPA IXO offices under the NEST project (grant number F336615-01-C-1905), the MURI award N00014-01-1-0576 from ONR and NSF grant CCR-0098269.

1-4244-0054-6/06/$20.00 ©2006 IEEE

ibly change the system configurations in order to accommodate various kinds of operation scenarios. 3) Compared with the solutions without data aggregation, we demonstrate analytically that our approach significantly reduces the amounts of energy consumed.

2 Related Work

In this paper, we focus on the data aggregation techniques applicable to the wireless sensor tracking systems. We divide our discussion into two categories: data aggregation techniques and sensor tracking systems in general.

2.1 Data Aggregation Approach

Data aggregation techniques have been widely used in wireless sensor networks. In [13], Intanagonwiwat proposes several basic forms of data aggregation methods, including 1) the Center at the Nearest Source method (CNS), where the source nearest to the destination aggregates the data from other nodes; 2) The shortest Path Trees (SPT) method, where data aggregation happens at the intermediate nodes within a shortest path tree rooted at the sink; and 3) the greedy Incremental Trees (GIT) method, where an aggregation tree is constructed by connecting each destination sequentially to the existing tree via a shortest path. GIT assumes a complete knowledge of global topology information; therefore it provides more opportunities for data aggregation. TAG provides a hierarchical data aggregation scheme at a data collection phase. Using an acquisitional query processor for data collection, TinyDB [17] optimizes the query process to aggregate data with a low energy overhead. Directed Diffusion [14], as a popular data-centric architecture for data acquisition, can be augmented by aggregating data along the reinforced paths from the sources to the sinks. Another type of data aggregation techniques focuses on the data placement of caching services. Bhattacharya et al. [1] investigate the optimal placement of caches between multiple sensor sources and sinks. These caches aggregate the updates from the source nodes and distribute data to leaf sink nodes with minimal requested rates. Since all aforementioned approaches are designed for the systems with no stringent time requirement, the designers of these systems naturally treat the timeliness as a less important issue related to the data aggregation. The closest research related to this work is the AIDA protocol [8]. AIDA takes the timely delivery of messages as well as the protocol overhead into account to adaptively adjust aggregation strategies in accordance with assessed traffic conditions and expected sensor network requirements. Through simulations, it demonstrates the feasibility to reduce the energy consumption and the end-to-end communication delay simultaneously. Our work presented here differs from aforementioned approaches in several aspects: First, this work deals with the real-time issue along with the data aggregation. Second, this work introduces not only the usage of multiple aggregation techniques, but also how these techniques can be intergraded within a tiered architecture. Third, our work is not a simulation study. Instead it is designed for a realistic tracking application with a running implementation.

2.2 Research on Sensor-based Tracking

Traditional surveillance systems are widely used for decades. Due to space constraints, we only name a few directly related ones. The ASDE system [3] and Secure Perimeter Area Network (SPAN) [4], normally use long-range cameras/radars with a 360-degree view of an area as the instruments for the detection. While these infrastructure protection systems are effective, they are subject to several limitations: First, they can not be emplaced swiftly without infrastructure, which makes such system not suitable for spontaneous military deployment in the remote areas. Second, the large form factor of these systems makes them easily detectable and evadable. Third, the number of surveillance points is small, which makes systems vulnerable to attack. To overcome these limitations, sensor networks is pursued recently by [2, 10, 11, 20, 19] as a more efficient mechanism to accomplish the remote unmanned surveillance missions. Feng et al. [20] design a surveillance and tracking system using a distributed Bayesian estimation technique. Brook et al. [2] propose a distributed surveillance system based on extended Kalman filter techniques. These solutions provide very nice features to improve the surveillance performance in one aspect or another, however ignoring other performance goals. For example, some systems provide high performance at cost of the system lifetime - a critical performance metric for long-term surveillance. The difference between our proposed work and aforementioned approaches is that we adopt our solution in a multiple-dimensional design space, where we consider not only the tracking performance, communication efficiency through data aggregation, but also the timeliness in delivery. This requires a balanced and flexible system design. Another highlight of our approach is the system implementation, which addresses many practical issues hard to capture in the simulated tracking scenarios.

3 Data Aggregation Requirements in VigilNet

The VigilNet is an online surveillance system, which consists of hundreds of tiny sensor nodes. These nodes detect, track and classify incoming targets in a timely and energy efficient manner. The final results are reported to a remote back-end via a long-haul bandwidth-constrained satellite links. In the current hardware platform, each node is equipped with three types of sensors. Magnetic sensors can detect the changes in the magnetic field caused by the movement of ferrous objects. Motion sensors can detect the changes in the infrared radiation caused by warm objects such as personnel. The acoustic sensors detect sound waves. Target signatures can be identified in VigilNet by sampling aforementioned three sensors at the particular rates (e.g. 10bits data at 1000HZ from acoustic sensors). Suppose 100 sensors participate in a tracking process, VigilNet could generate 1Mbit data in one second by using the acoustic sensor alone. The total amount of data to be transmitted multiplies with the number of hops over the network. Given the fact that the current long-haul satellite link only provides a 1200bps data rate, VigilNet can only send approximately 1-bit aggregated data out of every 1,000 bits of raw sensor readings generated from the

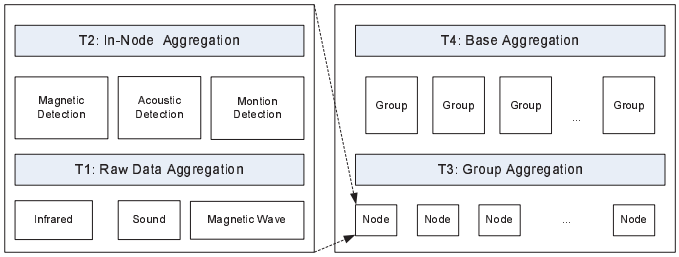

Figure 1. Four Tier Data Aggregation Architecture

network. This requires us to design an aggressive data aggregation architecture in VigilNet.

4 Four-Tier Data Aggregation Architecture

Normally, data aggregation ratio for a given system is simply the size of the original data divided by the size of the aggregated data. Based on our experience, a single data aggregation strategy is neither sufficient nor flexible to achieve an aggressive a 1000:1 data aggregation ratio. If data aggregation is only done at the node level, information could be lost too early to be useful. If data aggregation is only done at a central site, a sensor network spends too much energy in transmitting the data and possibly suffers severe network congestion and message losses. To balance energy efficiency and data availability, we design and implement a four-tier data aggregation architecture in VigilNet as shown in figure 1.

1. The first layer (T1) is the raw data aggregation layer, which takes the sensor readings form the individual sensors and converts them to the detection confidences, values between zero and one, indicating how confident a detection algorithm of the individual sensor is about the existence of the target. The data aggregation ratio at this layer is largely determined by the slowest raw sampling rate and the frequency in generating the confidence values. VigilNet can achieve approximately a $50 \sim 100 : 1$ ratio at this layer.

2. At the second layer (T2), a node takes the detection confidence values from different sensors to form a single classification vector, which indicates the target type and the corresponding confidence values. With three sensors, this layer achieves a 3:1 aggregation ratio.

3. At the third layer (T3), all nodes that detect the same target join the same logic group to track the target. Each group is represented by a leader to maintain the status of the target by aggregating all the reports from the member nodes. The leader node aggregates not only the location of reporting nodes as well as their confidence vectors together. Periodically, the leader node sends a report, consisting of a time stamp, an aggregation location and a confidence vector, to the base. The sensing density determines the aggregation ratio. In the VigilNet case, we expect an aggregation ratio between 3:1 and 10:1.

4. The fourth layer of aggregation (T4) happens at the base. A base aggregates the individual reports form the same logic group (a group that tracks the same target) together to generate a final report which contains the target type, bearing, speed and detection time-stamp. The aggregation ratio in this layer is determined by the tracking history length. Normally an aggregation ratio between 2:1 and 10:1 can be achieved.

In the next several sections, we give more detail on each layer, and provide the analysis that reflects the tradeoffs between energy efficiency and other properties of the system such as the timeliness.

5 T1: Sensor-Level Data Aggregation

The sensor data is the raw input to the network's computation work flow. It provides the foundation of the information processing for tracking events in the network. Data aggregation at this layer should meet following requirements.

- Meet real-time constraints: Because the system deals with transient events, such as fast-moving targets, in the network, the sensor data needs to be processed in a timely manner. If the processing latency is too long, a target would move out of the sensing range before the detection finishes

- Deal with a large volume of inputs: Endeavoring to accomplish reliable, timely, and quality sensing and tracking, a surveillance application often uses multiple sensors and samples them at very high rates. As a result, the combined sampling rate is high, especially, when the acoustic sensing is employed, and the volume of the sensor data input is large.

In the rest of this section, we first introduce different types of sensors and sensor data in the VigilNet, then discuss possible aggregation methods, and finally evaluate the aggregation method used in this layer.

5.1 Sensors and Sensor data

VigilNet uses the ExScal motes as sensor nodes. The major difference between the ExScal mote and the Berkeley Mica2 mote is that the former integrates a magnetometer (Honeywell HMC1052[12]), a microphone, and 4 PIR sensors on the same

circuit board as the processor's. Compared to sensors used in other applications, such as the temperature sensors (e.g., the Panasonic ERTJ1VR103J thermistor used on Mica sensor board) and light sensors (e.g., the Clairex CL9P4L photo sensor), the PIR sensors and microphone on ExScal motes track target signals with a relatively high frequency. For example, the microphone can potentially detect a waveform of 16KHz. This leads to a high degree of data availability at the sensor layer. More specifically, we list the sensor and the sensor data used in the following table.

Table 1. System parameters

Sensor	Type	Sampling rate	Note
Magnetometer	DC	32	8-bit POT 2-axis
PIR	AC	50	
Microphone	AC	1000	

5.2 Aggregation Methods

We call the sensor reading at a specific time on a specific sensor on a specific node a *sample point*. When a sensor network starts operation, each sensor on each node in the network produces a sequence of sample points. All the sample points produced by the network form a set and we call it the *global sample set*.

The global sample set is the complete information about what happens in the network. If all the nodes report their sample points to a base, the base can collect the global sample set and perform computation with it. However, as mentioned in Section 1, due to resource and energy limit, we prefer to deliver same information with fewer bits.

Let's first compute the amount of raw bits generated on one sensor. They correspond to all the sample points generated on one sensor. The magnetic sensor has two axes, each sampling at 32Hz, and the ADC output has 10 bits which are represented by two bytes in the software. Since the ADC values are the relative readings to the reference, one additional 8-bit potentiometer value is needed to record the reference value. Therefore, it takes 24 bits to record one sample point. In each second, 64 sample points are generated, and are represented with 1,536 bits, or 192 bytes. Suppose a sensor node keeps awake for one minute when some events of interest happen. The total number of data for this event is 92,160 bits, or 11,520 bytes. Similarly, we can compute the data generated by the PIR sensor has 48,000 bits, or 6,000 bytes, and the acoustic sensor generates 960,000 bits, or 120,000 bytes. In total, one event generates 1,100,160 bits, or 137,520 bytes.

As we can see, even one event generates a relatively large volume of sensor data. One potential method of aggregate the sensor data is to merge identical data and send out a "summary". However, with noise existing, the ADC seldom generates identical data even with a constant input. Hence, such merging is not effective.

Another way is to compute the differences between consecutive sample points, and record the difference, which is usually in a smaller range and can be represented by fewer bits. This leads to, virtually, a compression scheme. To assess the effectiveness of such compression, we use compression tools on PC to compress the sensor data, and examine the compression ratio. Experiments reveal that the compression ratio on DC signals by using gzip 1.3.3 is 100:37, and 100:31 with bzip2 1.0.2. Hence, even for DC signals, which is a simpler case, on a high-power platform with plenty of resources, the performance of the aggregation using such a method can hardly be satisfactory – it reduces approximately 2/3 of the data, but the remaining volume is still too large for sensor networks. Obviously, there are three key limitations for us to employ a traditional compression techniques: First, the transportation of 1/3 of the global sample set consumes an exorbitant amount of energy; Second, sensor nodes do not sufficient memory to accomplish these compression operations. Third, the latency to collecting and compressing these sample points is too long for a system that must react to the events in real-time.

Therefore, to enhance aggregation's performance to the level we desired, we design and implement a semantics based aggregation. By analyzing sensor data to retrieve its semantics, we achieves a highly efficient aggregation, as we will discuss next.

5.3 the Semantic-based Aggregation

The reason for collecting sensor data is to form tracking knowledge. Hence, the semantics of the sensor data is the probability of the existence of specific targets. The VigilNet detects four types of targets. Hence, we use a 4-element vector (BV, SV, PF, PS), called confidence vector, to represent the semantics of the sensor data.

In the confidence vector, the elements BV, SV, PF and PS correspond to the four types of targets – big vehicle, small vehicle, person with ferrous objects, and person. The numeric value of these elements are the relative probability of the existence of a target being of the specific type. By using sensor level sensing and classification algorithms, we transform sensor data into confidence vectors [7].

The tracking report rate is a configurable parameter in the VigilNet. Suppose this rate is 2 reports per second, which is a common setting in some deployed systems. We can evaluate the aggregation performance of such a semantics based approach. Each element of the confidence vector is represented by one byte, hence the confidence vector contains 4 bytes. In each second, at most two (zero in case of no even detection) confidence vectors are needed for two tracking reports. Hence, aggregation ratio for the magnetometer is 25:1, for the PIR sensor is 25:2, and for the acoustic sensor is 250:1. Overall, the aggregation ratio for all the sensors is 100:1. Obviously, the semantics based aggregation is highly efficient.

6 T2: Node-Level Data Aggregation

Each sensor node has four PIR sensors, one magnetometer with two axes, and one microphone. The node performs further aggregation after collecting confidence vectors from the

sensors. It computes the averages of the sensors' confidence vectors and form a single node-level confidence vector.

Because the three types of sensors form their own confidence vectors per type, the input of the node-level sensing and classification module are three confidence vectors. Hence, the aggregation ratio is 3:1. Combined with the 100:1 aggregation ratio accomplished at the sensor level, the overall node-level aggregation ratio is 300:1.

6.1 Analysis w/wo Aggregation

To retrieve semantics from sensor data and aggregate sensor level confidence vectors, the sensing algorithms and the node-level classification module need to buffer sensor data for a period of time and then perform their specific processing. This introduces delay into the network. Hence, we need to examine the length of the latency and verify that they are within a reasonable limit.

The delay introduced by the magnetometer is minimal – it buffers only one sample point and updates the confidence vector for each sample point. Hence, there is no delay introduce except for the mandatory sampling interval, which is about 16 milliseconds. Similarly, the PIR sensor introduces little delay because it buffers only a very small set of sample points. The delay introduced is still at the millisecond level. The acoustic sensor, however, buffers sensing data for about 1.2 seconds for processing. Hence, it introduces a latency of 1.2 seconds plus the sampling interval which is negligible in this case. Overall, the latency for accomplishing the best sensing and detection result is the maximum of the three sensors' latencies, i.e., 1.2 seconds.

The benefit, on the other hand, is obvious – with an aggregation ratio of 300:1, we reduces 99.67% of the communication payload. Though the semantics retrieval consumes CPU time and energy, it is a known fact that communication imposes orders of magnitude more energy overhead than computation in a sensor device. Hence, our estimation of energy saving due to data aggregation is still close to 99%.

7 Group-Level Data aggregation

After forming node-level detection results, VigilNet starts to estimate the current position of the tracked target as well as uniquely and identically represent the target in a logical space. Estimation of target positions is usually done by calculating the weighted average of the locations of nodes reporting detections, using their individual detection confidence values as weights. However, there is a design decision to be made on when and where to conduct such calculations. Representation of targets, as a traditional topic in target tracking, has been widely addressed by either centralized or distributed temporal and spatial correlation algorithms. Our system borrows the distributed group management algorithm from EnviroSuite [15], an programming middleware for tracking and monitoring applications in sensor networks. In the following subsections, we describe in more detail the group aggregation algorithm used in the system and provide a theoretical analysis of how it trades off among information quality, delay and energy consumption.

7.1 Group Based Data Aggregation

As stated above, tracking of a target consists of two main parts: position estimation and target representation. A simple way to tackle these problems would be to send the detection results and locations of individual nodes that detect targets to a centralized base station. Based on these received node positions [9, 18] , the base is able to estimate current positions of targets, and assign and maintain unique and consistent identities for targets by running temporal and spatial correlation algorithms. However, such a centralized scheme is inefficient both in energy and latency. It incurs excessive power consumption due to communicating multiple reports to a centralized base and may unduly increase latency, especially when targets are far away from the base. In addition, this centralized scheme can easily propagate the false alarms occurred in one part of the network to the base, which could overwhelm the base when the false alarm rate is high.

To avoid these limitations, we adopted a lightweight, distributed solution proposed in our previous work EnviroSuite [15]. Different from centralized solutions, EnviroSuite chooses to process data at or near the location where a target is detected, and sends only aggregates to the base for further processing. Specifically, EnviroSuite contains a set of group management algorithms to 1) instantiate globally addressable objects for targets as their logical representatives, 2)maintain a unique mapping between objects and targets, 3) guarantee the consistency of the mapping despite of target movements, and 4)suppress the false alarms locally.

Original EnviroSuite is completely dynamic, where nodes in the vicinity of a target is dynamically organized into one group, in which, a leader node is dynamically elected to host a corresponding object for the target. Though the dynamics of EnviroSuite enhances its robustness to failures including both message loss and node failures, it suffers a prolonged delay due to the long latency during the leader election, which is undesirable. To address this issue, our system uses a more static version of EnviroSuite, called Lightweight EnviroSuite [16], where groups are still dynamically formed while leaders are pre-elected in an initialization phase.

The following part explains Lightweight EnviroSuite in more details through a step-by-step description. Note that to be concise, we review only features relevant to this paper. Concrete descriptions of detailed algorithms on leader election and object maintenance can be found in [16]. In the initialization phase, a subset of nodes are elected to be potential leaders with the guarantee that they provide 100% sensing coverage. Later on in the tracking phase, when a target gets detected by local nodes, these nodes immediately report their locations and detection results to their corresponding potential leaders (A potential leader L is claimed to be one of node's potential leaders if and only if L is within 2 times sensing range from that node). These reports are kept by the potential leaders until one of them have collected

enough reports and is sensing the target. This potential leader then becomes a real leader, estimates the current position of the target and sends an aggregate report (containing position estimation as well as aggregated confidence vectors) to the base. Upon the reception of the aggregate report, other potential leaders drop their collected data and start over from the beginning again.

7.2 Analysis w/wo Aggregation

To aggregate, the leader running EnviroSuite waits for enough reports from members. A configurable parameter *DOA (Degree of Aggregation)* is introduced to measure whether there are enough member reports. All potential leaders withhold their aggregate reports to the base until the number of collected member reports reaches DOA.

Intuitively it is expected that setting DOA to a very small value (e.g., 1) would minimize delays, which, however, is proved incorrect in a realistic, noisy environment the system operates on. Field experiments reveal that in such an environment, false positives of individual nodes are so frequent that, without filtering, excessive traffics between these nodes and the base tend to congest the network and impose large latency to other traffics that contain meaningful data. Therefore, it is critical to use a big DOA to filter out the false alarms of individual nodes, which not only improves the quality of information but also has *positive* affects on latency. Besides, energy consumption is also reduced due to less traffic towards the base. On the other hand, setting DOA to a high value (e.g., 5) is not desirable either, since a high DOA inevitably introduces longer delay. A DOA value higher than the number of nodes within a sensing range should be avoid, otherwise no report will be generated by the leader node. These tradeoffs between information quality, energy consumption and latency require that DOA has to be carefully chosen to satisfy these various aspects of design requirements. Analytical models on energy consumption and group aggregation delay as functions of DOA are built below to provide guidelines for system designers.

7.2.1 Energy Gain with Data Aggregation

It is extremely challenging to analyze realistic systems complicated by various factors, e.g., sensing range, target motion model, and node density. However, a rough approximation can be derived by making several simplified assumptions, including circular sensing range (R_s), straight target trajectory with velocity V_t, and uniformly distributed nodes with density d. This approximation gives us some insight on the system performance in general.

Assuming that a target enters the monitored field and moves for time duration T_t as shown in Figure 2, we compares energy consumption with/without aggregation as below. Since all the node that can detect the target are located within the gray rectangle or semi-circle as shown in Figure 2, the total number of reporting nodes is:

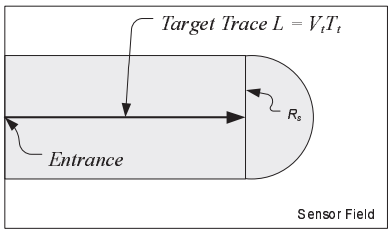

Figure 2. The Detection Area

$$d(\pi R_s^2/2 + 2R_s V_t T_t) \tag{1}$$

Without group-level aggregation, each node directly sends its detection result to the base, where the expected energy consumption without aggregation(E_{wo_aggr}) is:

$$E_{wo\_aggr} = d(\pi R_s^2/2 + 2R_s V_t T_t) E\bar{n} \tag{2}$$

where E represents the energy consumed to send a one-hop message and $\bar{n}$ represents the average hop count between these detection nodes and the base.

With aggregation, each node sends its detection result only locally and, for every DOA detection nodes, there is an aggregate report sent to the base. Therefore, the expected energy consumption with aggregation (E_{w_aggr}) becomes:

$$\begin{aligned} E_{w\_aggr} &= d(\pi R_s^2/2 + 2R_s V_t T_t) E + \\ &\quad \frac{d(\pi R_s^2/2 + 2R_s V_t T_t)}{DOA} E(\bar{n} - 1) \end{aligned} \tag{3}$$

Based on these equations, the percentage of conserved energy (P) due to group aggregation can be calculated as:

$$P = 1 - \frac{E_{w\_aggr}}{E_{wo\_aggr}} = 1 - \frac{1}{\bar{n}} - \frac{1}{DOA} \tag{4}$$

when $\bar{n} \geq 2$ and $DOA \geq 2$. These equations reveal the relation between energy consumption and DOA. As an example, when DOA is set to be 3 and the base is 3 hops away on average, group-level aggregation consumes approximately 33% less energy compared with the no-aggregation scheme.

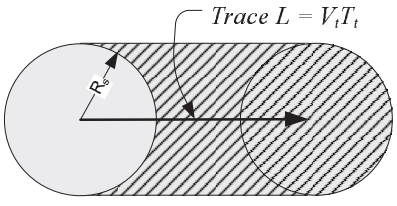

Figure 3. Movement of a target during time period T

7.2.2 Delay Introduced by Aggregation

This section analyzes how tracking report latency is affected by DOA settings. Figure 3 depicts the movement of a target during time period T_t. The white circular and grey circular represent, respectively, the detection area of the target before and after the movement. Nodes located in the diagonally lined area are new detectors of the target and send out member reports for aggregation during this time period, assuming that they have a common potential leader. Let's assume that an aggregate report has been sent out just before the depicted movement, which means that a new aggregate report is not to be sent until the number of reports sent by these new detectors reaches DOA. Therefore, we have the expected delay caused by group-level aggregation T_g:

$$T_g = \frac{DOA}{2R_s V_t d} \tag{5}$$

This equation shows quantitatively the tradeoffs between delay and DOA (indication of information quality). To be more concrete, Figure 4 illustrates different group aggregation delays for varied target velocities and DOA values (setting sensing range to be 10 m and node density to be 1 per 100 m^2).

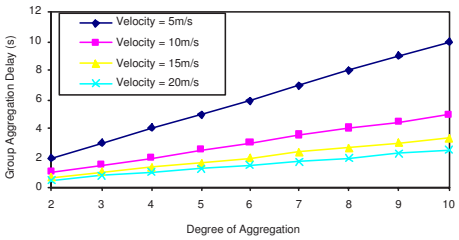

Figure 4. Group aggregation delays for varied DOA values and target velocities

We note here that as one part of the tier-architecture, the group-level aggregation is not simply another method to further reduce energy consumption. With the inputs from multiple nodes, the group-level aggregation can eliminate the false alarms due to the faulty nodes and improve data quality, which can not be achieved by the node-level aggregation.

8 T4: Base-Level Data Aggregation

After receiving the reports from individual leader nodes, the base needs to aggregate the information further, an operation to serve three main objectives:

- Flow control: The long-haul link to the remote back-end could be the bottle neck of the system. A base is required to address the bandwidth mismatch between the sensor network output and the long-haul link capability. This is more likely to happen when the system tracks multiple objects simultaneously.
- Filtering: A base needs to prevent sending the duplicate reports as well as the false alarms to the back-end. To filter out the false alarms, a system needs to correlate the spatiotemporal properties of consecutive reports. Since the base is the only place in the network that has the complete global knowledge of a tracked target, it is at a better position than in-network nodes to filter out the system-wide false alarms.
- Consolidated View: End users is more interested in a consolidated view of the tracked targets instead of the individual sensing reports from the sensor network. Although the group-level aggregation does provide some persistent information such as object ID, it is not effective to keep a long trace history of a target through a sequence of handover operations among the leader nodes. To improve the efficiency, a base should be used to create a consolidated view instead.

8.1 Base-Level Data Aggregation

The base bridges the system back-end and the sensor network. Its aggregation functionalities can be summarized as follows. First, the base takes the input from the in-network group leaders, and creates logical targets according to the spatiotemporal correlations of the input reports. Second, according to the information of the logic targets, the base filters out duplicate reports, messages with long delays, and false alarms, and provides flow control to match the bandwidth of the upstream link. Third, the base makes use of the incoming messages to provide extra information for the logic targets, such as target velocity and target classification information.

More specifically, when the base received a detection report from a certain location, it tries to associate the report with the closest logical target created recently. If the distance from the location of the report to that of the logical target exceeds a predefined threshold, a new logic target is created for the report. If the distance is below the threshold, the report is added into the history of the logical target and the location of the logical target is updated according to the report. An exception is that if the location of the report is the same as one of the last few locations in the history, the report is dropped as a duplicate report. A logical target expires after a certain amount of time if there has been no new report that is added into its history. By this approach the reports are categorized into logical targets based on their spatiotemporal correlations. When the base creates a new logical target, it needs to differentiate valid target reports from false alarms. This is done by accumulating spatiotemporal correlated reports up to a certain length before confirming the detection.

Another value that the base-level aggregation adds into the system is velocity calculation for the logical target. Since each message from the network includes a location and a time-stamp, we can use the locations and time-stamps in the logical target's history to infer the velocity of the target. We use a standard linear regression procedure to calculate the velocity. The formula

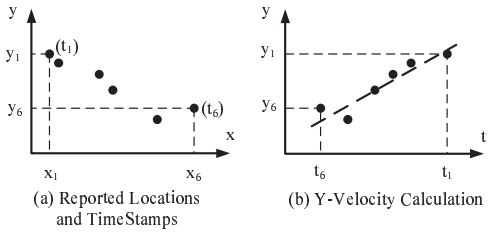

Figure 5. Velocity Calculation from Reported Locations

for calculation of the x-axis component of the velocity is

$$v_x = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(t_i - \bar{t})}{\sum_{i=1}^{N}(t_i - \bar{t})^2} \quad (6)$$

in which x_i is the x component of the ith location, and t_i is the ith time-stamp. The x component of the latest location is denoted by x_1, and that of the one right before it is x_2, etc. N denotes the number of reports and time-stamps used for the calculation. The v_y calculation is similar to the Equation 6. As shown in Figure 5, v_y is the slope of the fitting straight line (the thick dashed line shown in Figure 5(b)).

To deal with the bandwidth limit to the system back-end, a flow rate parameter is set during the initial phase of the network operation. The flow rate parameter can be used to calculate the minimum interval between two consecutive reports of a logical target from the base: $D_{base} = 1/Nr$, in which N is the maximum allowable logical targets in the system and r is the flow rate. Obviously, the worst case delay introduced by the base-level aggregation is D_{base}. Supposing the reporting rate of each leader node is R, the aggregation ratio at the base is NR/r. We note that a naive solution is to do flow control at the group-level by setting the reporting rate of each leader node to r/N and the base relays every report to the back-end. In this naive design, a system could save more energy, however less data is available to create a consolidated target view. For example, the velocity estimation becomes less accurate and less fresh with fewer reports.

In summary, the base-level aggregation is essential and can not be replaced or eliminated by the aggregation at other layers, because of its ability to correlate the system-wide reports and resolve the flow rate mismatch between a sensor network and the link to back-end system.

9 Conclusion

In this paper, we describe a multi-tier data aggregation architecture for target tracking applications. Due to space constraints, we omit the evaluation on this architecture. Details on its performance can be found at [10]. Since sensing data has different semantics at different layers, the pure in-network aggregation leads to low data availability for the high-level aggregation, while the pure centralized aggregation leads to excessive energy consumption. In contrast, the architecture proposed in the work can flexibly achieve the balance between energy, timeliness and data availability.

References

[1] S. Bhattacharya, H. Kim, S. Prabh, and T. Abdelzaher. Energy-Conserving Data Placement and Asynchronous Multicast in Wireless Sensor Networks. In *MobiSys'03*, May 2003.

[2] R. R. Brooks, P. Ramanathan, and A. Sayeed. Distributed Target Tracking and Classsification in Sensor Networks. *Proceedings of the IEEE*, 2002.

[3] S. Corporation. Asde-x brochure. http://www.sensis.com/docs/194//.

[4] T. S. Corporation. Surveillance for airport perimeter security. http://spacecom.grc.nasa.gov/icnsconf/docs/2002/11/Session_E2-5_Barry.pdf.

[5] CrossBow. *Mica2 data sheet*, 2003. at `http://www.xbow.com`.

[6] P. Dutta, M. Grimmer, A. Arora, S. Biby, and D. Culler. Design of a Wireless Sensor Network Platform for Detecting Rare, Random, and Ephemeral Events. In *IPSN'05*, 2005.

[7] L. Gu, D. Jia, P. Vicaire, T. Yan, L. Luo, A.Tirumala, Q. Cao, J. A. S. T. He, T.Abdelzaher, and B. Krogh. Lightweight Detection and Classification for Wireless Sensor Networks in Realistic Environments. In *SenSys*, 2005.

[8] T. He, B. M. Blum, J. A. Stankovic, and T. F. Abdelzaher. AIDA: Adaptive Application Independent Data Aggregation in Wireless Sensor Networks. *ACM Transactions on Embedded Computing Systems, Special issue on Dynamically Adaptable Embedded Systems*, 2004.

[9] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. Abdelzaher. Range-Free Localization Schemes in Large-Scale Sensor Networks. In *MOBICOM'03*, September 2003.

[10] T. He, S. Krishnamurthy, J. A. Stankovic, and T. Abdelzaher. An Energy-Efficient Surveillance System Using Wireless Sensor Networks. In *MobiSys'04*, June 2004.

[11] T. He, P. Vicaire, T. Yan, Q. Cao, G. Zhou, L. Gu, L. Luo, R. Stoleru, J. A. Stankovic, and T. Abdelzaher. Achieving Long-Term Surveillance in VigilNet. In *InfoCOM*, 2005.

[12] Honeywell. Hmc1052 magnetometers. http://www.ssec.honeywell.com/magnetic/.

[13] C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann. Impact of network density on data aggregation in wireless sensor networks. In *ICDCS*, 2002.

[14] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks. In *MobiCom'00*, 2000.

[15] L. Luo, T. Abdelzaher, T. He, and J. A. Stankovic. Envirosuite: An environmentally immersive programming framework for sensor networks. In *ACM Transactions on Embedded Computing Systems, to appear*.

[16] L. Luo, T. He, T. Abdelzaher, and J. Stankovic. Design and comparison of lightweight group management strategies in envirosuite. In *DCOSS '05: International Conference on Distributed Computing in Sensor Networks*, June 2005.

[17] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. The Design of an Acquisitional Query Processor for Sensor Networks. In *SIGMOD*, June 2003.

[18] R. Stoleru, T. He, J. A. Stankovic, and D. Luebke. A High-Accuarcy, Low-Cost Localization System for Wireless Sensor Networks. In *SenSys'05*, November 2005.

[19] T. Yan, T. He, and J. Stankovic. Differentiated Surveillance Service for Sensor Networks. In *First ACM Conference on Embedded Networked Sensor Systems (SenSys 2003)*, November 2003.

[20] F. Zhao, J. Shin, and J. Reich. Information-Driven Dynamic Sensor Collaboration for Tracking Applications. *IEEE Signal Processing Magazine*, March 2002.