Motion Trends Detection in Wireless Sensor Networks

Goce Trajcevski*(1) Besim Avci*(1) Fan Zhou †(2) Roberto Tamassia‡(3)
Peter Scheuermann*(1) Lauren Miller§ Adam Barber§
*Department of Electrical Engineering and Computer Science, Northwestern University
goce,besim,peters@eecs.northwestern.edu
†School of Computer Science and Engineering, University of Electronic Science and Technology
fan.zhou.uestc@gmail.com
‡Department of Computer Science, Brown University
rt@cs.brown.edu
§ Department of Mechanical Engineering, Northwestern University
LaurenMiller,Adam.Barber@u.northwestern.edu

***Abstract*—We address the problem of efficient detection of destination-related motion trends in Wireless Sensor Networks (WSN) where tracking is done in collaborative manner among the sensor nodes participating in location detection. In addition to determining a single location, applications may need to detect whether certain properties are true for the (portion of the) entire trajectories. Transmitting the sequence of (location, time) values to a dedicated sink and relying on the sink to detect the validity of the desired properties is a brute-force approach that generates a lot of communication overhead. We present an in-network distributed algorithm for efficient detecting of the *Continuously Moving Towards* predicate with respect to a given destination that is either a point or a region with polygonal boundary. Our experiments demonstrate that the proposed approaches yield substantial savings when compared to the brute-force one.**

I. Introduction

Wireless Sensor Networks (WSN) consist of hundreds, or even thousands, of tiny devices – nodes – each capable of sensing values of a particular physical phenomena, performing basic calculations and, most importantly, self-organizing in a wireless network to communicate observations from different parts of the network to each other. These features have rendered WSN as a paradigm of choice in a plethora of applications: scientific, traffic management, environmental safety/hazardz, infrastructure, health-care and military purposes [10], [17] – to name but a few. One of the canonical research problems in WSNs is *tracking*, and various aspects of it problems have been addressed by the researchers: from the the accuracy of the tracking process, through trade-off between the tracking accuracy and the energy consumption, to adjusting the routing structures that convey the location-in-time information to a given sink [9]. Typically, the location of a given object is determined either by a a GPS-enabled device or by some form of collaborative trilateration among the tracking sensors. Detecting a sequence of such locations generates the information about the moving object's *trajectory*, which is a sequence of points (L_1, t_1), $(L_2, t_2), \ldots,$ (L_k, t_k) where L_i denotes the (detected) location of the tracked object in some reference coordinate system, at time t_i and $(\forall i, j)$ $(i < j) \Rightarrow (t_i < t_j)$.

In this work, we aim at detecting whether a *trend* can be inferred relating the motion of the tracked object to a given spatial region in the geographic area covered by sensor nodes, which is important for different application domains, e.g.,:

• In habitat monitoring scenarios, one may be interested in detecting that certain type(s) of animals are approaching the region of a pond or a river.

• In security and defense scenarios, one may be interested in detecting when an object is approaching the perimeter of a particular camp or site.

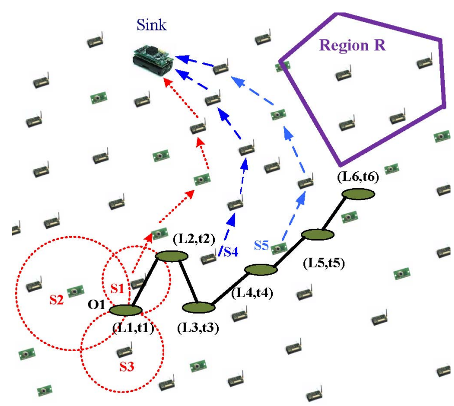

Fig. 1. Motion Trends Predicates

More specifically, we focus on efficient detection of the predicate specifying whether a moving object *moving towards* a given region with a polygonal boundary, as illustrated in Figure 1. Typically, when detecting certain non-local properties about the object's trajectory, the individual (location, time) data is transmitted from one of the sensors performing the collaborative trilateration (the *tracking principal*) to the

(1) Research Supported by the NSF: CNS-0910952
(2) Work performed while visiting the EECS Dept. at Northwestern University, supported by the CSC pre-doctoral fellowship
(3) Research Supported by the NSF:CCF-0830149

978-0-7695-4713-8/12 $26.00 © 2012 IEEE
DOI 10.1109/MDM.2012.38

sink – which can incur a lot of unnecessary communication overhead. For example, assume that, in the scenario shown in Figure 1 we are interested in detecting whether the object O_1 has been *Continuously Moving Towards* the region R for a period of at least 3 consecutive samples. Following the naïve approach, each of the sensor nodes S_1, S_4 and S_5 will initiate a transmission of the detected location (and the time-stamp) data towards the sink. This will cause a lot of intermediate-hops to transmit the data to the sink which – in this specific example turns out to be a waste of communication resources, since the desired predicate is not satisfied by the first three sampled locations.

In WSN settings, the communication, be it a transmission of active listening/reception, drains a lot more energy than sensing and computation [1], hence, avoiding unnecessary communication is a paramount. Our goal in this work is to provide distributed approaches that will enable in-network detection of the predicate, thereby minimizing the communication overhead towards the sink. The main contributions of this work is an efficient distributed algorithm for in-network detection of the predicate pertaining to the trend of a given trajectory with respect to a spatial region: *Continuously Moving Towards (CMT)*. In our earlier work [15] we presented a centralized version of an algorithm for detecting the *CMT* predicate, however, in this paper we present in detail both the distributed algorithm for its in-network processing, as well as the strategy for efficient dissemination of the query by a given sink.

Our experiments demonstrate that our proposed approaches yield substantial savings in terms of communication among the nodes, when compared to the naïve approach(es) of forwarding every location sample and its time-stamp to the sink, and performing the detection of the predicate there.

The rest of this paper is structured as follows. Section 2 gives an overview of the background material needed to present our main results in Section 3. Experimental observations are presented in Section 4. Section 5 compares our results with the related works, summarizes the paper and outlines directions for a future work.

II. PRELIMINARIES

We assume a sensor network consisting of N nodes, $SN = \{sn_1, sn_2, \ldots, sn_N\}$, where each node is capable of detecting an object within its range of sensing, e.g., based on vibration, acoustics or otherwise [8]. Nodes are aware of their locations $sn_k = (x_k, y_k)$ either via a GPS or by using some other techniques e.g., collaborative multilateration [13]. Each node is also assumed to know the locations of all of its one-hop neighbors, and the nodes are assumed to be static. We assume that the network is *dense* enough to ensure coverage for the purpose of detection and localization via trilateration using some standard ranging method, e.g., acoustic/echo-based, RSS(Received Signal Strength) or TDOA (Time Difference of Arrival) and, furthermore, to ensure a selection of a neighbor(s) to whom the task of tracking can be handed-off [9], [11]. We assume that between two consecutive location detections, the tracked objects move along straight line and with a constant speed – hence, the location at any time instant in-between the sampling times can be obtained via linear interpolation.

Throughout this work (and in the corresponding implementation) we did not consider any issues related to the sleeping of the nodes and the epoch-based synchronization and selection of tracking principles [7]. We used a simplifying assumption that the principal is the sensor node from among the ones that can participate in the trilateration which is closest to the sink (in terms of the Euclidian distance). Note that a given sensor node may be a tracking principal for more than one location-sampling process.

Given a set of points $P = \{p_1, p_2 \ldots, p_M\}$, their *Voronoi diagram* [3] is a planar subdivision (faces, edges and vertices) induced by the points from P, with the following basic properties: (1) Each face (Voronoi cell) contains one point $p_i \in P$ in its interior and, for all the other $p_j \neq p_i$, and every point q inside that face, $dist(q, p_j) > dist(q, p_i$, where *dist(.,.)* denotes the Euclidian distance between the two points. (2) Each edge (Voronoi edge) corresponds to a bisector between two points p_k and p_l from P.

Voronoi diagrams are one of the most extensively studied structures in Computational Geometry and algorithms for their construction in WSNs have also been proposed [4]. Among the extensions from the original definition (pertaining to a discrete set of points) are the variants for non-discrete sets of points (e.g., line-segments and polygons) [3] – and we will use the concept of a Voronoi diagram for (the exterior of) convex polygons. As illustrated in Figure 2, the edges of the Voronoi diagram of a given region R bounded by a convex polygon are defined by the rays originating at the vertices of the polygon and emanating perpendicularly to the edges. There are two basic types of Voronoi cells:

• Edge cells, which is, the set of points in the plane for which the closest points on the boundary of R are along a given edge (bounded by parallel half-lines); and

• Vertex cells, which is, the set of points in the plane for which the closest point on the boundary of R is one of its vertices (i.e., points within a wedge originating at a vertex);

For a given edge $\overline{A_{i-1}A_i}$ from R's boundary, let *VCell*($\overline{A_{i-1}A_i}$) denote its Voronoi cell with respect to R. Also, let *Edge*(A_i, *VCell*($\overline{A_{i-1}A_i}$)) denote the edge (i.e., the perpendicular half-line to $\overline{A_{i-1}A_i}$) of the *VCell*($\overline{A_{i-1}A_i}$) originating at A_i (similarly for the one originating at A_{i-1}). The Voronoi cell belonging to a given vertex A_i is denoted by *VCell*(A_i) and its boundary edges will coincide with the ones corresponding to the boundary edges of the Voronoi cells belonging to the two adjacent edges to A_i (i.e., $\overline{A_{i-1}A_i}$ and $\overline{A_iA_{i+1}}$.

III. TRENDS DETECTION ALGORITHMS

We now present our techniques for the efficient in-network detection of the occurrence of the motion-trend predicate — *Continuously Moving Towards* in WSN settings. Before we provide the algorithmic details, we address two relevant issues:

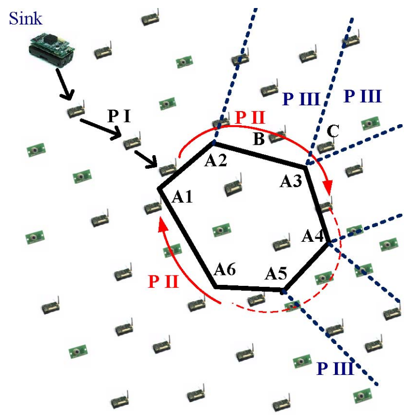

Fig. 2. Disseminating a request

(1) the propagation of the request from the sink node to the rest of the nodes of the network and the creation of the Voronoi diagram of the region of interest; and (2) the consumption policies regarding the past locations detected along the tracking process.

A. Disseminating the Request and Events Consumption

The dedicated sink node, sn_k, needs to detect the occurrence of the desired predicate as a consequence of the tracking process, on behalf of a particular application of interest. Hence, node sn_k has to inform the rest of the nodes in the WSN about all the details of a particular request: (1) Its own location and node_ID; (2) Region R, e.g., specified via the sequence of its vertices in counter-clockwise order; (3) The begin-time and the end-time during which the detection of the predicates is important (e.g., t_b and t_e), along with the duration of the time interval Δ during which it is important that the object is moving continuously towards R.

The sink will need to send a message containing quintuple ($Sink$, R, t_b, t_e, Δ, P_R) throughout the network. One obvious way to do it is via flooding [1], where every node, upon receiving the message will: (1) Forward it to its neighbors that it has not heard from yet (at the time of receiving the message); (2) Proceed with detecting which Voronoi cell of R it belongs to. Towards this, the node needs to find the point on R (along the edges or in a given vertex) which is geographically closest to its location. This naïve approach of disseminating the request(s) may incur a significant overhead in terms of energy consumption, which can be avoided. Namely, we observe that for the purpose of detecting the corresponding Voronoi cell to which a given sensor node, sn_j, belongs to, it need not be aware of all the vertices of R. Hence, we propose the following three-phase dissemination protocol, illustrated in Figure 2.

Phase I (P I): In this phase, instead of starting the flooding process, the sink simply sends the packet containing the quintuple ($Sink$, R, t_b, t_e , Δ, P_R) to the sensor node on the boundary of R that is closest to it. This phase is illustrated in Figure 2, where the sink sends the packet to the sensor node near edge $\overline{A_1A_2}$.

Phase II (P II): In the second phase, the node that received the request from the sink will forward the request to its neighbors along the boundary of R, each of which will recursively propagate it in the chosen reference-direction.

Phase III (P III): The third phase of the dissemination protocol can actually be pipelined with the second phase. In this phase, the moment a particular node along the outer-boundary of R receives the request, it determines the edge (or vertex) of R closest to it—implying the Voronoi cell that it belongs to. Subsequently, that node will selectively notify its neighbors in the exterior of R about the edge defining the boundaries of its *Vcell*. In the example of Figure 2, sensor node B will send the message ($Sink$, $\overline{A_2A_3}$, t_b, t_e, Δ, P_R) to its neighbors. Sensor nodes that are closest to a given vertex of R (e.g., node $C \in$ *VCell*(A_3) in Figure 2) will transmit the two edges incident to the vertex to its neighbors in R's exterior. Thus, node C will send the message: ($Sink$, ($\overline{A_2A_3}$, $\overline{A_3A_4}$), t_b, t_e, Δ, P_R) to its neighbors. The reason for this approach is to help the subsequent nodes in the WSN – in particular, *VCell*(A_3) for the node C – which may receive more than one such message, disambiguate which *VCell* they belong to and, of course, which *VCell* does a particular location of the tracked object belong to. As our experiments demonstrate, the proposed three-phase dissemination protocol yields savings in terms of communication cost when compared to the naïve flooding.

There is one more aspect that needs to be determined before the detection of the predicates can begin – the *consumption policy* of the individual location-samples. To illustrate this aspect, consider a scenario where the CMT predicate has been detected within five consecutive samples. Clearly, this should initiate a notification sent to the sink. Now the question becomes: if the 6^{th} location sample indicates that the object is *still* moving towards (with respect to the 5^{th} sample), should another notification be sent to the sink or not? Clearly, this is something that will need to be decided by the application which needs the detection of the predicates. A detailed discussion of consumption policies for the primitive constituent events upon a detection of a desired composite event/predicate is beyond the scope of this work (see [5]). However, we note that the algorithms presented in the rest of this section will work correctly for both *chronicle* based consumption (i.e., only the oldest location-sample participating in the detection is discarded, the rest are still considered) and *cumulative* based consumption (i.e., the moment the composite predicate is detected, all the participating location-samples are ignored and the detection starts anew).

B. CMT Predicate

As previously mentioned, when disseminating a given request, part of the desired predicate must be explicitly specified. Hence, in the case of predicate Continuously Moving Towards, the sink, say sn_k, will start the three-phase protocol with the message ($Sink$, R, t_b, t_e, Δ, *CMT*). Upon completing the preprocessing stage, the nodes in the WSN can begin combining the tracking process with detecting whether the *CMT* predicate

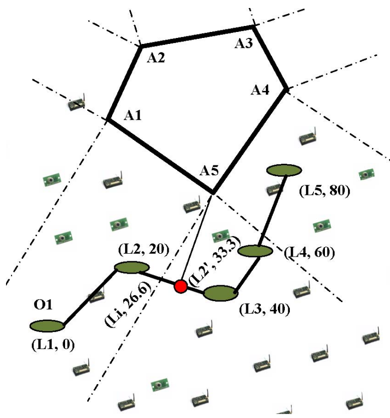

Fig. 3. Detecting Continuously Moving Towards

Algorithm 1 CMT — executed by the tracking principal

Input: Request parameters (*Sink*, t_b, t_e , Δ, *R*, CMT); accumulator structure containing the previously detected location+time (L_p, t_p); accumulated time T_A of continuously moving towards R up to t_p.

Detect the location L_c of the tracked object at the current time t_c
// via trilateration with neighboring nodes
T_{CT} = *TotalTimeTowards*((L_c, t_c), (L_p, t_p), R, T_A)
if $T_{CT} \geq \Delta$ **then**
  notify *Sink*
  Update T_A in accordance with the consumption policy
else
  $L_p \leftarrow L_c$;
  $t_p = t_c$;
  $T_A \leftarrow T_{CT}$;
end if
Send ((L_c, t_c), (L_p, t_p), T_A) to the next tracking principal

has been satisfied upon a current localization. Towards this, upon trilateration, the node elected to be the principal of the tracking process, say, sn_{TP}, will execute Algorithm 1 (*CMT*).

The tracking principal, s_{TP}, executing algorithm *CMT* receives the accumulated time (T_A) of the continuous motion towards the target R from the previous tracking principal up to, and including, the previously-observed location. Subsequently, it calculates the value of variable T_{CT}, which updates T_A in accordance with the object's motion along segment $\overline{L_pL_c}$. If the combined values exceed the desired threshold Δ, node sn_{TP} will initiate a notification to the *Sink* along a shortest-path route. When the notification is sent, the accumulator variable T_A is either set to 0 (cumulative consumption) or decremented by the duration of the time-interval corresponding to the very first localization that initiated the detection of the CMT predicate. When the value of T_A is updated from 0 to some $\varepsilon > 0$, we need to retain a queue (FIFO) that will maintain all the values following ε throughout the rest of the tracking process. To calculate value T_{CT}, node s_{TP} executes procedure *TotalTimeTowards*, which is specified by Algorithm 2. We illustrate this procedure with the scenario of Figure 3, which shows a pentagonal region $R \equiv A_1$, A_2,

Algorithm 2 TotalTimeTowards (TTT) — executed by the tracking principal

Input: Accumulator structure containing the previously detected location+time (L_p, t_p), along with the accumulated time T_A of continuously moving towards R up to t_p, and the currently detected location and time (L_c, t_c)

1: **if** $L_c \in$ *VCell*($\overline{A_{i-1}A_i}$) ($i \in \{1, 2, \ldots, n\}$) **then**
2:   **if** $L_p \in$ *VCell*($\overline{A_{i-1}A_i}$) **then**
3:     **if** dist(L_c, $\overline{A_{i-1}A_i}$) $\geq$ dist(L_p,$\overline{A_{i-1}A_i}$) **then**
4:       // the object is moving away
5:       $T_T \leftarrow 0$;
6:     **else**
7:       $T_T \leftarrow T_A + (t_c - t_p)$;
8:     **end if**
9:   **else**
10:     //L_c and L_p are in different VCells
11:     L_I = *IntersectPoint*($\overline{L_cL_p}$, *Edge*(A_i, *VCell*($\overline{A_{i-1}A_i}$));
12:     t_I = *InterpolateTime*(L_I, $\overline{L_cL_p}$);
13:     T'_A = *TotalTimeTowards*((L_I, t_I), (L_p, t_p), T_A, R);
14:     T_T = *TotalTimeTowards*((L_c, t_c), (L_I, t_I), T'_A, R);
15:   **end if**
16:   return T_T
17: **else**
18:   // L_c is inside a *VCell* of a vertex, say A_i
19:   **if** $L_p \in$ *VCell*(A_i) **then**
20:     **if** The point P_{min} of the minimal distance between A_i and $\overline{L_cL_p}$ is inside $\overline{L_cL_p}$ **then**
21:       $T_T \leftarrow 0$;
22:       // the object switched from MovingTowards
23:       // to MovingAway from R at P_{min}
24:     **else if** dist(L_c, A_i) $\geq$ dist(L_p,A_i) **then**
25:       // the object is still moving away
26:       $T_T \leftarrow 0$;
27:     **else**
28:       // the object is strictly moving towards R
29:       $T_T \leftarrow T_A + (t_c - t_p)$;
30:     **end if**
31:   **else**
32:     //L_c and L_p are in different VCells
33:     L_I = *IntersectPoint*($\overline{L_cL_p}$, *Edge*(A_i, *VCell*($\overline{A_{i-1}A_i}$)));
34:     t_I = *InterpolateTime*(L_I, $\overline{L_cL_p}$);
35:     T'_A = *TotalTimeTowards*((L_I, t_I), (L_p, t_p), T_A, R);
36:     T_T = *TotalTimeTowards*((L_c, t_c), (L_I, t_I), T'_A, R);
37:   **end if**
38:   return T_T
39: **end if**

A_3, A_4, A_5 along with its Voronoi cells. Assume that the application is interested in detecting whether a given object has been continuously moving towards R for at least 35 time-units and five location-samples with the respective time-values.

First, note that Algorithm 2 distinguishes between two main cases: (1) The tracked object is inside the Voronoi cell of an edge (handled by lines 1—16); (2) The tracked object is inside the Voronoi cell of a vertex (handled by lines 17—39)

The rationale is that if the object's location falls inside a VCell of a given edge *and* the previously sampled location is inside the same VCell – then the object can either move completely towards or completely away from the region R, throughout the entire interval of its motion inside that VCell. This is illustrated with locations $(L_1, 0)$ and $(L_2, 20)$ in

Figure 3. Since *dist*$(L_2, \overline{A_5A_1}) <$ *dist*$(L_1, \overline{A_5A_1})$ and both L_1 and L_2 are in the same *VCell*$(\overline{A_5A_1})$, the T_{CT} value is updated to 20.

If, on the other hand, the tracked object's current and previous locations are in *VCell*(A_i) belonging to vertex A_i of the polygonal boundary of R, then we have an additional case to consider (cf. line 20 of Algorithm 2): namely, if the perpendicular from A_i to the line defined by L_c and L_p falls inside the line-segment $\overline{L_cL_p}$, we know that at the terminus of the perpendicular, the motion plan of the tracked object has changed from MovingTowards (i.e., the distance to R begins to increase). This is illustrated with the portion $\overline{L_iL_3}$ of the segment $\overline{L_2L_3}$ in Figure 3. Both L_2' and L_3 are in *VCell*(A_5) and, initially, the object is moving closer towards R, i.e., its distance to A_5 is decreasing. However, at point L_2', which is the terminus of the perpendicular line from A_5 to $\overline{L_i, L_3}$, the distance has reached its minimum at begins to grow. Hence, at $(L_3, 40)$ the time of moving towards R is set to $T_{CT} = 0$.

Finally, we note that in both main cases – the current location-sample being within a Voronoi cell of an edge or a vertex – Algorithm 2 makes recursive calls when the previous and the current location-samples belong to different Voronoi cells. This scenario applies to both sampling $(L_3, 40)$ and $(L_5, 80)$ in Figure 3. Specifically, when calculating the value of T_{CT} at $(L_3, 40)$, firstly the location of L_i – intersection of $\overline{L_2L_3}$ and *Edge(*A_5*, VCell(*$\overline{A_5A_1}$*))* is found and the expected time at that location (26.6 time-units) is calculated via linear interpolation. Subsequently, we have two recursive calls (lines 13—14 and 35—36 in Algorithm 2), each calculating the respective time spend moving towards R along the segments $\overline{L_2L_i}$ and $\overline{L_iL_3}$.

Since each recursive calls decreases the total length of the segment used, Algorithm 2 is guaranteed to terminate. As for its complexity, note that, in the worst case (e.g., a fast-moving object coupled with a small convex polygon with n vertices), a given line segment can intersect the edges of at most $O(n)$ Voronoi cells. This is the bound on the number of the recursive calls, each of them taking a constant amount of time to complete within a single cell. Hence, the running time of Algorithm 2 is $O(n)$, which is also an upper bound on the number of the messages that the current principal may need to exchange in order to determine all the actual cells that participate in the CMT predicate satisfaction. Assuming that the localization step is performed regularly in intervals of δ_s, the running time of Algorithm 1 is $O(\lceil (t_e - t_b)/\delta_s \rceil n)$.

IV. Experimental Evaluations

The experimental observations were generated using the open-source, SIDnet-SWANS simulator for WSN [6]. We considered a WSN with 750 homogeneous nodes with simulated ranging capabilities that implement the equivalent of an active ultrasonic echo ranging system, running on a standard MAC802.15.4 link layer protocol. To alleviate the lack of spatio-temporal dependency among consecutive motion-segments present in the random way-point model, we used trajectories based on the *Gauss-Markov Mobility Model* (GMMM) [12] which does exhibit spatial and temporal dependency – at each slot the speed and direction are computed based on the ones from the previous time-slot. We used three different categories of speeds of motion, corresponding to walking, riding a bicycle, and driving a car. The first group of experiments that we present illustrates the benefits of our three-phase protocol for disseminating the requests via selective flooding.

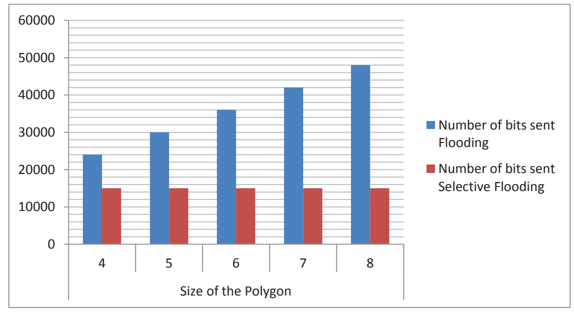

Fig. 4. Bits transmitted for Request Dissemination

As shown in Figure 4, the total number of bits transmitted between the pairs of nodes in the network grows linearly with the size (number of vertices) of the polygon bounding the query region R. On the other hand, using the proposed selective flooding – which guarantees that the nodes in the WSN will be able to correctly process the request for detecting the *CMT* predicate – the total number of bits transmitted is almost a constant. Both observations are, in a sense, "in concert" with the intuitive expectations.

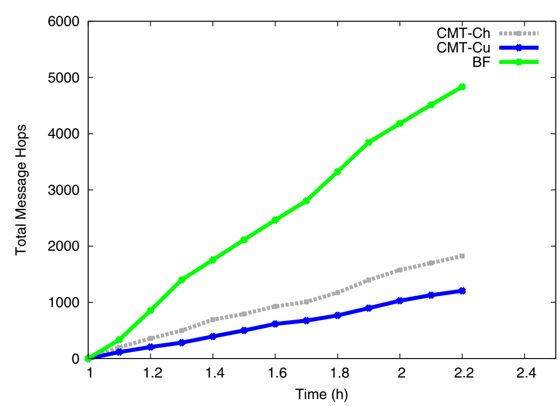

Fig. 5. Detecting *Continuously Moving Towards*

Our next set of simulations aimed at providing some quantitative observations regarding the savings obtained when using our Algorithm 1 for processing the *CMT* predicate, when compared to the naïve, or Brute-Force approach of transmitting every single location to the sink. The results are illustrated in Figure 5, which shows averaged observations for sampling intervals of 5 seconds, 10 seconds and 30 seconds; and the criteria of $\Delta = 3 \cdot$ *SamplingInterval* and $\Delta = 5 \cdot$ *SamplingInterval* needed to satisfy the *CMT* predicate. As can be seen, the

Brute-Force approach (denoted ”BF” in Figure 5) generates much higher volume of messages than the *CMT* approach. Note that there are two curves, one for each of the consumption policies (Ch – Chronical, and Cu – Cumulative) corresponding to the in-network *CMT* processing. As expected, the Ch–consumption will generate more messages towards the sink, since it ”recycles” all the former (location, time) pairs, except for the oldest one; whereas the Cu-consumption completely eliminates the history upon detection of the predicate.

V. Related Work and Conclusions

Localization and tracking can be viewed as canonical problems in WSN settings and the results abound, ranging from efficient maintenance of tree-structures that route data towards a dedicated sink and energy-efficiency [9], [13], through incorporating unreliability of the tracking nodes and quality-guarantees [18], to considering fast-moving objects and low-frequency snapshots [2]. In this work, we did not attempt to present any new tracking or localization methodologies – our goal was to use the existing results and, in some sense, augment their use for the purpose of developing efficient distributed algorithms for in-network detection of the two motion trend predicates.

Using geometric concepts in WSN settings has been part of many research efforts. Several results have been reported for distributed computation of various structures – e.g., Voronoi diagrams [4], convex hulls [14] – and their use for efficient solutions to problems related to boundaries/holes detection, routing, etc. In this work, we also relied on a geometric concept – the Voronoi diagram [3] for polygonal boundary of a region of interest. We presented an efficient protocol for disseminating a request whose processing depends on the knowledge of the distribution of the sensor nodes among the Voronoi cells of the region, along with efficient detection satisfiability of the CMT predicate in WSN settings, where the detection of the location of the moving object in a given time instant is done by collaborative trilateration of tracking sensors.

We also discussed the policies for consuming the primitive *(location, time)* events upon detection of the composite event of detecting the occurrence(s) of the *CMT* and *PMT* predicates. Our experiments demonstrated that the proposed algorithms indeed bring substantial savings in terms of reducing the number of messages that need to be communicated throughout the network, when compared to the naïve approach which transmits every detected location (along with the time-stamp) to the sink.

Currently, we are adapting our approaches towards capturing motion trends for larger groups of objects, similar to the concepts of a *flock* of trajectories (cf. [16]) – but in WSN settings. There are few other challenges that we would like to address in the future as extensions of this work. Firstly, we will need to modify the existing algorithms so that the epoch-based synchronized operation of the nodes can be taken into account, along with the corresponding policies for selecting tracking principals. Secondly, we would like to investigate the impact of having heterogeneous networks' settings where, in addition to the static nodes, there are also mobile nodes, to the processing of motion trends related predicates.

Acknowledgements: We thank Herve Bronninman for his constructive suggestions.

References

[1] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38(4), 2002.

[2] Aysegul Alaybeyogly, Kayhan Erciyes, Aylin Kantarci, and Orhan Dagdeviren. Tracking fast moving targets in wireless sensor networks. *IETE Technical Review*, 27(1), 2010.

[3] Franz Aurenhammer. Voronoi diagrams - a survey of a fundamental geometric data structure. *ACM Comput. Surv.*, 23(3), 1991.

[4] Boulat A. Bash and Peter Desnoyers. Exact distributed voronoi cell computation in sensor networks. In *IPSN*, pages 236–243, 2007.

[5] S. Chakravarthy, V. Krishnaprasad, E. Anwar, and S.K. Kim. Composite events for active databases: Semantics, contexts and detection. In *20th VLDB Conference*, 1994.

[6] Oliviu Ghica, Goce Trajcevski, Peter Scheuermann, Zachary Bischoff, and Nikolay Valtchanov. Sidnet-swans: A simulator and integrated development platform for sensor networks applications. In *SenSys*, pages 385–386, 2008.

[7] Oliviu Ghica, Goce Trajcevski, Fan Zhou, Roberto Tamassia, and Peter Scheuermann. Selecting tracking principals with epoch-awareness. In *Proceedings of the 18th ACM SIGSPATIAL GIS Conference*, pages 222–231, 2010.

[8] Tian He, Pascal Vicaire, Ting Yan, Liqian Luo, Lin Gu, Gang Zhou, Radu Stoleru, Qing Cao, John A. Stankovic, and Tarek F. Abdelzaher. Achieving real-time target tracking using wireless sensor networks. In *IEEE Real Time Technology and Applications Symposium*, 2006.

[9] Jaehoon Jeong, Taehyun Hwang, Tian He, and David Hung-Chang Du. Mcta: Target tracking algorithm based on minimal contour in wireless sensor networks. In *INFOCOM*, 2007.

[10] Sukun Kim, Shamim Pakzad, David E. Culler, James Demmel, Gregory Fenves, Steve Glaser, and Martin Turon. Health monitoring of civil infrastructures using wireless sensor networks. In *IPSN*, pages 254–263, 2007.

[11] Loukas Lazos, Radha Poovendran, and James A. Ritcey. Analytic evaluation of target detection in heterogeneous wireless sensor networks. *TOSN*, 5(2), 2009.

[12] Ben Liang and Zygmunt J. Haas. Predictive distance-based mobility management for multidimensional pcs networks. *IEEE/ACM Trans. Netw.*, 11(5):718–732, 2003.

[13] G. Mao and B. Fidan. *Localization Algorithms and Strategies for Wireless Sensor Networks*. IGI Global – Invormation Science Publishing, 2009.

[14] Ivan Stojmenovic, Anand Prakash Ruhil, and D. K. Lobiyal. Voronoi diagram and convex hull based geocasting and routing in wireless networks. *Wireless Communications and Mobile Computing*, 6:247–258, 2006.

[15] Goce Trajcevski, Peter Scheuermann, H. Brönnimann, and Agnes Voisard. Dynamic topological predicates and notifications in moving objects databases. In *Mobile Data Management (MDM)*, 2005.

[16] Marcos R. Vieira, Petko Bakalov, and Vassilis J. Tsotras. On-line discovery of flock patterns in spatio-temporal data. In *GIS*, pages 286–295, 2009.

[17] Geoffrey Werner-Allen, Konrad Lorincz, Matt Welsh, Omar Marcillo, Jeff Johnson, Mario Ruiz, and Jonathan Lees. Deploying a wireless sensor network on an active volcano. *IEEE Internet Computing*, 10(2), 2006.

[18] Ziguo Zhong, Ting Zhu, Dan Wang, and Tian He. Tracking with unreliable node sequences. In *INFOCOM*, pages 1215–1223, 2009.