Supporting Situation-Aware services with Virtual Macro Sensors

Nicola Bicocchi
Università di Modena e Reggio Emilia
Via Amendola 2, Reggio Emilia, Italy
nicola.bicocchi@unimore.it

Marco Mamei
Università di Modena e Reggio Emilia
Via Amendola 2, Reggio Emilia, Italy
marco.mamei@unimore.it

Franco Zambonelli
Università di Modena e Reggio Emilia
Via Amendola 2, Reggio Emilia, Italy
franco.zambonelli@unimore.it

ABSTRACT
Next-generation communication services will be required to adapt their behavior to the specific characteristics of the physical and social environment in which they will be invoked. The technology to acquire contextual information will be increasingly available, e.g., in the form of highly-pervasive sensor networks infrastructure. Indeed, such infrastructure can lead to the production of overwhelming amounts of information, difficult to be managed and interpreted by services. This calls for proper solutions to enable services to extract meaningful general-purpose data from distributed sensors in a compact way. The approach presented in this paper relies on a simple algorithm to let a sensor network self-organize a virtual partitioning in correspondence of spatial regions characterized by similar sensing patterns, and to let distributed aggregation of sensorial data take place on a per-region basis. This makes it possible for services to gather information about the surrounding world as if it was generated by a limited number of virtual macro sensors, independently of the actual structure and density of the underlying sensing infrastructure.

Categories and Subject Descriptors
C.2.4 [**Computer-Communication Networks**]: Distributed Systems – *distributed applications.*

General Terms
Algorithms, Reliability, Experimentation, Theory.

Keywords
Self-organization, pattern recognition, mobile services, gossip based aggregation.

AUTONOMICS 2007, 28-30 October 2007, Rome, Italy

DOI 10.4108/ICST.AUTONOMICS2007.2305

1. INTRODUCTION
In the next few years, we will assist to an increasing deployment of sensor network systems [9, 5]. Most likely, such mass deployment will induce a radical change in their *raison d'être*. Rather than being closed special-purpose systems devote to monitor specific phenomena [15, 23], as they are today, they will form the basis of truly pervasive and dense shared infrastructure, publicly available for general-purpose sensing activities by a variety of users and services. Just to make some examples: cars in a city can access sensors around to obtain on-the-fly updated traffic information; tourists can exploit sensors around to discover urban information and activities; in the case of a disaster, sensors can support robots in performing rescuing operations. More in general, software services can exploit the information obtained by local sensors to contextualize their behavior and improve users' satisfaction.

The change in the very nature of sensor networks will also radically change the patterns by which such systems are accessed and exploited (see Figure 1). As of today, most sensor network systems are conceived to report to some base station (sink) at a fixed location specific data related to some specific phenomena of interest [22, 2], possibly after some limited in-network processing/compression of such data [17, 10]. Clearly, such fixed sinks will be present also in future sensor network scenarios. However:

- sinks will have to collect general purpose data for remote users that want to discover about various events/phenomena happening somewhere in the world [1, 3];
- sensors will additionally have to report data on-demand to multiple and mobile services (e.g. running on users' PDA) that can exploit short-range wireless connections to access local sensors around and retrieve user-specific data [19, 7, 16].

This novel perspective of usage introduces peculiar challenging requirements. First, it is expected that the sensor network, despite being intensively used in unpredictable ways, will be able to control its energy consumption. In other words, the energy costs should be bounded and balanced over the network, so as to ensure a minimal guaranteed lifetime or (for self-rechargeable devices [14]) that it will never require more power than it can self-produce. Second, it is expected that the network will be able to provide services (whether remote or local) with expressive and

compact information related to the phenomena under sensing rather than raw individual sensor data. In the presence of dense (virtually continuous) sensor networks generating huge (virtually infinite) amounts of data, dealing with the transfer and the ex-post analysis of individual sensors data can be become simply unmanageable. Third, the network should quickly answer services' requests. Since services can be highly mobile (e.g., running on a moving car) a late answer to a query can either fail to reach it or reach it at a location where the answered information could be useless.

To tackle the above issues, the idea underlying our proposal is that of delegating to the sensor network the execution of distributed gossip-based algorithms [12] that – by continuously running in the network as a sort of background noise with bounded energy costs – can enforce:

- the adaptable self-partitioning of the network into spatial regions characterized by similar patterns for sensed data, via the self-organization of an overlay network.
- the distributed aggregation of whatever sensorial data on a per-region basis.

The result of this process is that a sensor network can be modeled as made up of macro sensors, each associated to a well-characterized region of the physical environment (i.e., a region exhibiting a uniform pattern for some specific property such as a light, temperature, etc.). Within each region, each physical sensor has the local availability of aggregated data related to its region and can act as an access point to such data.

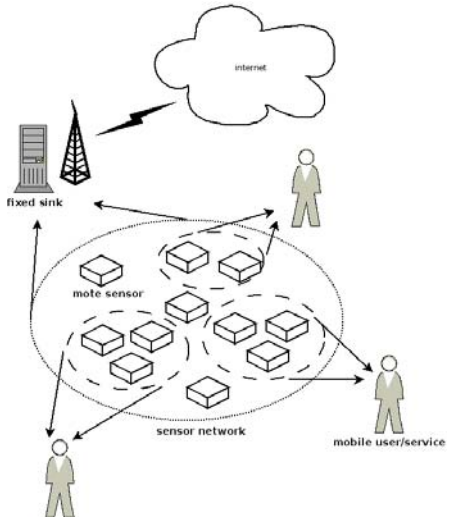


Figure 1: Future sensor network infrastructures.

The approach based on virtual macro sensors makes it possible for multiple and mobile services to promptly access global information about the surrounding environment by simply querying the closest sensor, at no additional costs for the sensor networks. Also, it makes possible to effectively transfer aggregated data towards a centralized collection point in a more compact and efficient way, yet avoiding loss of information typical of global aggregation algorithms. Moreover this process is independent of the actual density, topology, and dynamics, of the underlying physical sensor network (e.g., making the approach suitable also in the extreme case of extremely dense sensor networks generating huge amounts of data).

Performance studies performed in both a simulation environment and a real testbed confirms the effectiveness of our approach and its potential for being usable in a wide range of applications scenarios. However, the discussion outlines a number of limitations calling for further work.

The paper is organized as follows. Section 2 discusses related work. Section 3 details the proposed virtual macro sensors approach. Section 4 evaluates the approach. Section 5 discusses the applications of the approach and its current limitations. Section 6 concludes.

2. RELATED WORK

Most of the works on data gathering and aggregation in sensor networks assume the presence of fixed sinks (i.e., base stations) to which sensed data should flow. In such situations, the basic approach is that of having sensors build a tree rooted at the sink and supporting the routing of sensed data towards it [22]. To deal with the transfer of possibly large-amounts of data, several forms in-network data aggregation (e.g., averaging or max/min determination) can be performed as data from sensors climb the tree [17, 10, 2] both with the goal of reducing communications between sensors and, thus, the energy costs. However, such aggregation algorithms does not generally account for the data patterns exhibited by phenomena under sensing, and are thus at risk of being either ineffective or very lossy. As far as multiple and mobile services are concerned, tree-based approaches can hardly apply. In fact, the costs of building a tree on demand for many possible services at different and varying locations would be unbearable, both in terms of energy and response time. Although some specific optimizations for mobile services have been proposed [24], these do not eradicate the basic flaws of tree-based approaches.

Several research works in the area of sensor networks start recognizing the need to promote direct access to sensor data by multiple and mobile users/services. These works mostly focuses on defining suitable general-purpose primitives and language constructs to enable users to flexibly query the network and obtain information about individual sensor data and aggregated data related to specific regions. Examples of these approaches include Region Streams [20], TinyLime [7], and Logical Neighbourhood [18]. These kinds of languages could be well complemented by our proposal that could provide a basic algorithmic infrastructure on which to realize the proposed abstractions.

The issue of recognizing regions of a sensor network characterised by similar properties of sensed data is considered very important to improve the reliability and capability of tracking. Indeed, some in-network algorithms for self-organization of region partitioning in sensor networks have been proposed [4, 21], sharing some basic principles with our approach. The key differences with it are that: *(i)* these algorithms require a priori information about the typical patterns exhibited by the environment, while our approach does not and it is fully self-organizing; *(ii)* these algorithms are not conceived for other goals than recognizing regions, while our approach goes further, by

exploiting regions as a basis for aggregation and for building the abstraction of virtual macro sensors.

As far as distributed data aggregation is concerned, diffusive algorithms [6] and gossip-based aggregation algorithms [12, 8] have been proposed as simple yet very effective approaches to compute and make available at each node aggregated values related to some global property of the network. In our approach, we borrow from them by exploiting aggregation algorithms that have a mixed diffusive-based and gossip-based inspiration. However, other than for computing global network values, we use them for computing regional aggregated values (as the basic sensing mechanism of virtual macro sensors). Gossip-based algorithm have been recently exploited as the basis for partitioning a network into clusters of nodes characterized by similar properties [13], a problem similar to the one being addressed in this paper, though with totally different motivations, goals, and scenarios.

3. VIRTUAL MACRO SENSORS

The virtual macro sensors approach considers: *(i)* a self-organized region formation algorithm to define the boundaries of each macro sensor; *(ii)* localized aggregation algorithms to provide macro sensors with regional sensorial capabilities; *(iii)* solutions to self-adapt to transitory and dynamic situations.

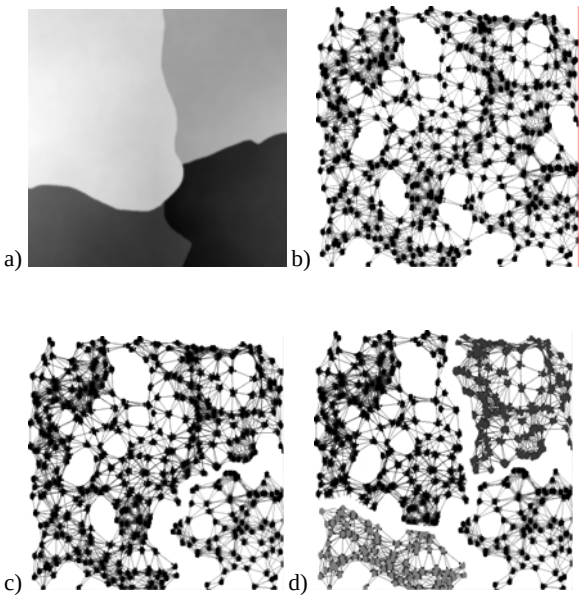


Figure 2. Self-organizing spatial regions: a) a scalar field with 4 regions with different values of a property *v*; b) a 500-nodes sensor network immersed in the above scalar field, with links representing the actual physical layer; c) example of the overlay region organization with *p=0,4* leading to a partitioning into 2 coarse regions (we show only the logical links between the nodes that are logically connected); d) overlay region organization with *p=0,05* leading to a partitioning into 4 small regions.

3.1 Region Formation

We consider a sensor networks deployed in an environment in which the value *v* of some specific environmental property (i.e., a spatial field) can be locally sensed by sensors. The value *v* could represent a temperature, a light level, or whatever property a sensor is capable to infer about its sensed portion of the environment (see Figures 2-a and 2-b). The actual spatial extension and sensor density in the network is irrelevant in our approach, which thus apply also to (virtually) infinite and infinitely dense networks.

The proposed region formation algorithm has the goal of letting sensors self-organize into disjoint sets of spatial regions each characterized by "similar" measures of the property *v* (see Figures 2-c and 2-d). Organization in regions occurs via a process of building an overlay of virtual weighted links between neighbor nodes, such that nodes belonging to the same region have strong links, while neighbor nodes belonging to different regions have weak (or null) links. As examples: measuring the light level could be used for a sensor network in a building to self-partition on a "per room" basis (different rooms being characterized by different light level, while the light level inside a room is always quite homogeneous); measuring the vibration level on a mountain slope could lead to self-organizing a sensor network into regions associated to surfaces with different geological properties. More in general, the region organization can reflect some actual property of the physical space and can lead to a "logical" organization of sensors, thus making a region to be elected as the extension of a virtual macro sensor.

Coming to the details of the algorithms, let s_i and s_j be two neighbor sensors, i.e., two sensors whose distance is smaller than their wireless radio range *r*. Let $v(s_i)$ and $v(s_j)$ be the values of a property sensed by s_i and s_j, respectively. Let us assume that a distance function *D* can be defined for couples of *v* values. Region formation is then based on iteratively computing the value of a logical link $l(s_i,s_j)$ for each and every node of the system, as in the following "*Update_link*" procedure:

Update_link:

if (D(v(s_i), v(s_j)) < T) {

$l(s_i,s_j) = min(l(s_i,s_j) + delta, 1)$

} else {

$l(s_i,s_j) = max(l(s_i,s_j) - delta, 0)$

}

Where: *T* is a threshold that determines whether the measured values are close enough for $l(s_i,s_j)$ to be re-enforced or, otherwise, weakened; and *delta* is a value affecting the reactivity of the algorithm in updating link (more details on the threshold *T* follow later on).

Based on the above algorithm, it is rather clear that if $D(v(s_i), v(s_j))$ is lower than threshold *T*, $l(s_i,s_j)$ will rapidly converge to either *1* or *0*. In the simplest case, one could consider two nodes s_i and s_j to be in the same region when $l(s_i,s_j)$ is over a threshold *T*. Transitively, two nodes s_h and s_k are defined in the same region if

and only if there is a chain of nodes such that each pair of neighbors in the chain are in the same region. For the actual execution of the algorithm, each node stores a vector describing, for each of the neighbors, the current value of the link towards it and a flag signaling the status of the link (connected or not). To improve stability in the presence of noise, the connection status of a link relies on a hysteretic cycle with two thresholds T_l and T_h with T_h-T_l>> *noise*.

The distributed execution of the algorithm is based on a sort of diffusive gossip scheme [6, 12] which act as a sort of continuous background activity in the sensor network: each node periodically wakes up, randomly selects a specific number or a specific percentage of its neighbors (for real-world broadcast –based wireless channels, this implies inviting a limited number of neighbors in participating in the protocol), exchanges with them the needed data (i.e., the *v* values, plus other data that will be detailed in the following), and then executes the "*Update_link*" procedure for each selected neighbors. Schematically:

Do_forerever:

Wait(t);

Foreach(neigh[]=Select_neighbor(num_neigh))

Data = Exchange_data();

Update_link(data);

Done

From the above description it is clear that our algorithm tends to impose a pre-defined, tunable, and uniform load, to the system. Each node executes the same amount of operations. The interval *t* determines the frequency of such operations and the number of neighbors *num_neigh* selected at each round determines the communication cost of these operations. Shorter *t* or higher *num_neigh* tend to speed up the convergence of the algorithm, but increase the energy consumed by sensor per time unit (as quantified in the performance evaluation section). Therefore, one can select the "degree of noise" of our algorithm and, so, the energy consumed over time.

Let us now go into more details about the other parameters of our algorithm.

Concerning the parameter *delta*, it determines how fast the link weight *l* changes its value. The choice of this parameter is not crucial, provided that it is chosen small enough to require several cycles of the "*Update_link*" procedure to actually modify the status of link (in other words, it should be notably smaller than the T_l-T_h hysteretic interval). This avoids that random or temporary fluctuations of the measured value at a node continuously cause changes in the established regions.

Concerning *T*, an apparently challenging issue in our approach consists in tackling the difference between the strictly local nature of "*Update_link*" interactions and the inherently global meaning of the threshold *T*. How can two nodes evaluate the right threshold if they don't know anything about the rest of the network? Fortunately, in the vast majority of the cases, a domain expert can provide suitable and relevant thresholds to highlight the phenomena of interest and to drive the self-partitioning accordingly. For example, a difference of 5°C can be considered of relevance for a biologist to distinguish different types of landscape, and (s)he could rely on a region-partitioning based on such a threshold. Alternatively, fire guards may be interested in much higher thresholds (e.g., 40°C) to detect anomalies. In any case, it is worth emphasizing that our approach does not prescribe the existence of a single region partitioning. Depending on application needs, the same background algorithm can be exploited, at no additional costs, to build any number of different overlay partitions, each based on different thresholds. Simply, each node can host and compute an array of virtual link values *l* for each neighbor, each corresponding to a different threshold value.

In the absence of any a priori known domain data, and for networks of finite size, it is still possible to define *T* by exploiting dynamically collected global values of the property *v*. For instance, in our tests, we defined *T* as a portion of the whole range of values seen over the network. Using scalar values, we defined *T* as:

$$T = (globalMax - globalMin) * p$$

where *p* is a real number between *0* an *1*. In this way, one can parameterize the sensibility of the algorithm by using a relative value *p* rather than some absolute value requiring a priori knowledge on the range of *v* values. If one wants to obtain very large regions to organize the network based on macroscopic difference one can select *p* close to *1* (as in Figure 2-c). If one is interested in more fine-grained region organizations one can select *p* close to *0* (as in Figure 2-d).

To locally acquire the *globalMax* and *globalMin* value at each node, one can execute a global diffusive aggregation algorithm over the whole network. Simply, as described in [12], each node can exchange with its neighbors the information about the maximum and the minimum he knows so far. Eventually the knowledge about the actual *globalMin* and *globalMax* will reach each node of the network. In details, each node s_i, after having exchanged data with node s_j, can execute the following *"Global_aggregation"* procedure:

Global_aggregation:

$if(globalMin_i > globalMin_j)\ globalMin_i = globalMin_j$

$if(globalMax_i < globalMax_j)\ globalMax_i = globalMax_j$

with $globalMin_i$ and $globalMax_i$ both initialized at v_i. The above aggregation algorithms requires minimal additional effort by nodes. In fact, one can exploit the existing region aggregation noise and its "*Exchange_data*" messages to exchange the *GlobalMin* and *GlobalMax* values, by piggybacking with such messages the additional data needed, and then computing the "*Global_aggregation*" function after the "*Update_link*" procedure inside the main algorithm body. When needed, one can also

decide to exploit the same schema to compute any additional distributed global aggregation algorithms (e.g., computing the average), as well as to compute aggregations over properties different from *v*. But this is not the key point of our approach.

3.2 Per-Region Aggregation

The local availability of aggregated information over the whole sensor network may be of some use independently of regions. However, globally aggregated values give very little details on the status of the network, are prone to obsolescence and high losses and are definitely of little use for users wishing to acquire info about environmental properties around him/her. For this reason, our approach mostly relies on per-region aggregation algorithms. Local aggregation algorithms enable each sensor in a region to act as a sort of access point for aggregated data in that region, and thus realize the concept of virtual macro sensor: from the application viewpoint, services can perceive a region as including a single sensor with sensorial capabilities extended to the whole region.

When regions are already formed (transitory situations will be discussed later on), computing aggregation function in a region reduces to executing a diffusive aggregation algorithm only between those couples of neighbor nodes that are in the same region (i.e., for which the l is over the T_h threshold). Again, computing per-region aggregation function does not introduce significant additional burden to the network. The exchange of data between nodes can occur by piggybacking over the existing messages, and the computation of local aggregation algorithms reduces to add a simple "*Local_aggregation*" function in the main body of our basic scheme, as follows:

```
Do_forerever:
        Wait(t);
        Foreach(neigh[]=Select_neighbor(num_neigh))
                Data = Exchange_data();
                Update_link(data);
                Global_aggregation(); // if needed
                If(connected) Local_aggregation();
Done
```

The "*Local_aggregation*" function can include the identification of the local minimum and the local maximum of any sensed value *w* (other than the *v* property on which regions are based) within the region (computed as in the global case), as well as the calculus of the average *Avg* of any value *w*. In this case, the local aggregation for a node s_i, after having exchanged data with connected node s_j, simply works as follows [12]:

$$Avg_i(w) = (Agv_i(w)+Avg_j(w))/2$$

with $Avg_i(w)$ simply initialized at the local value w_i. The value of regional average is clearly the one that, more than others, gives practical meaning to the concept of macro sensors. Yet, the local maximum and minimum well complement it by adding some hint on the amount of data locally lost in the averaging process.

In our scheme, we also decided to enforce two additional peculiar aggregation functions that are of great use for facilitating the gathering of information by users. A first aggregation function considers that each node at the border of a region (i.e., each node which has at least one virtual link l below the threshold) propagates within the region a "hop counter" initialized at *0*. By having such counter by re-propagated by each node on per-minimum basis, the results is that each node in a region eventually becomes aware of its distance form the closest border. We also plan to experience more sophisticated aggregation function to enable nodes to locally reach a higher-level understanding of the shape and topology of the local region, possibly relying on existing work of distributed topology recognition. What is important to note is that these kinds of topological measures are important to asses, within regions, the sensing coverage of the macro sensors. A second aggregation function exploits a sort of per-region minimum identification towards the election of a region leader. By having each sensors exchange its unique *ID* with its neighbor, the minimum *ID* eventually recognized by each node will define the leader (and the leader itself will recognize itself as that). This is very important to give macro sensor a recognizable unique identity. More details on how such data can be of practical use follows in Section 5.

3.3 Transitory and Dynamic Situations

Let us now analyze the dynamic behavior of the algorithms during region formation and region re-shaping (changes in the values of the property v upon which region formation relies can induce changes in the shape and dimensions of regions, as well as in the aggregated values).

In general, the initial values of the virtual links l between nodes are irrelevant for region formation. Therefore, let us assume an initial situation in which all nodes are disconnected from each other (i.e., each node is a region in itself). As the algorithm will start running, nodes with similar values of v will start connecting with each other, and sets of regions with growing dimensions will start forming and possibly merge each other, until a stable situation will be reached.

Concurrently with the above region formation process, the local aggregation procedure starts executing as soon as two nodes gets virtually connected in the same region, and it proceeds gradually involving more and more sensors, eventually converging when a stable region situation is reached. It can be shown (and it is quite intuitive indeed, due to the cumulative nature of aggregation) that the proposed aggregation algorithms do not experience problems if executed on a growing number of nodes, as in the region formation transitory. This also applies for the identification of the region leader (when two regions merge, one of the two leaders will be eventually overtaken by the other one). Similar considerations apply to the case in which new sensors are dynamically added in the system.

The case in which some existing regions shrink, (either because a confining region has expanded or because some sensor nodes

have died) is a bit more complex to handle. In fact, two problems may arise: (i) the values computed by the local aggregation functions may no longer be valid (e.g., the former maximum may have left the region and/or the average may have changed) and the cumulative nature of aggregation does not enable them to be properly updated; (ii) the region leader may have exited the region.

To overcome the former problem, we decided to enforce a sort of "evaporation" of the values computed by the local aggregation algorithms (except for the leader election algorithm). In other words, the local aggregated values at a node are slowly (compared to the convergence time of the aggregation algorithms) moved towards the initial values, e.g., the local values of the node. In this way, the weight of those data cumulated by the algorithm will gradually diminish, unless properly re-enforced. As an example, consider the case of the maximum of a region, and assume that each node in a region has already locally available the value of such maximum. Now, have each node slightly "evaporate" such value by making it diminish to approach the local value. If the node holding such maximum is still in the region, a node will receive again the maximum value undoing the evaporation effects. Instead, if the node holding the maximum left the region, evaporation will enable to stabilize the new maximum at each node, after proper evaporation. Similar considerations apply, e.g., to the calculus of the average. This solution also enables dealing with situations in which, even if regions do not change their shape, the computed aggregate values change because of local changes in the sensed values.

The second problem cannot be tackled by evaporation (the leader ID is not a value that can be tolerate approximation). Accordingly, each node keeps track of the "oldness" of the value of the leader ID (accounting for the number of cycles of the algorithm since the last time it received from some node such ID). Whenever such oldness becomes excessive, the current leader ID is considered obsolete and a new leader (i.e., the new node with the minimal ID) is identified and elected.

Overall, the above two solutions make our virtual macro sensor approach fully self-organizing and self-adaptable.

4. EVALUATION

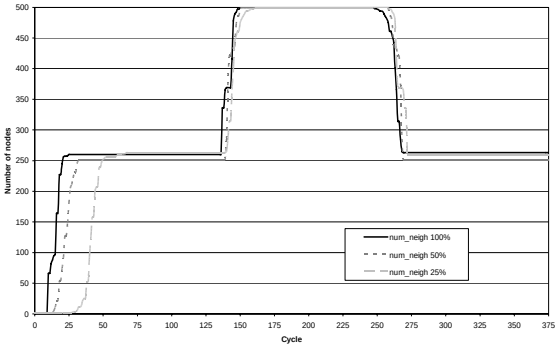
To test the effectiveness of the approach, we have experimented it both in a simulation environment (to verify the convergence and accuracy level of our approach in large-scale scenarios, and the effect on them of the *num_neigh* and *t* parameters) and in a small sensor network testbed (to evaluate its functioning in practice and its actual energy consumption levels).

4.1 Simulations

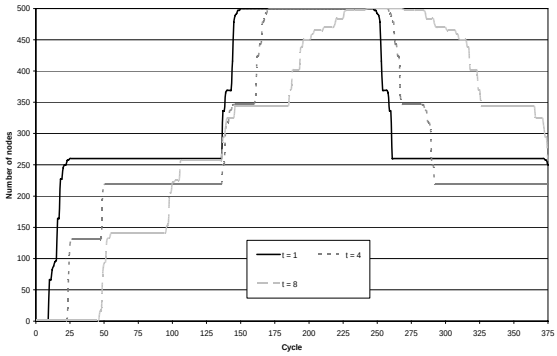
Simulations have been built over the Repast simulation framework [http://repast.sourceforge.net/]. We have conducted several experiments with sensor networks of different sizes and densities, immersed in different types of scalar fields, always obtaining similar qualitative and quantitative results. The results reported here refer to: a scalar field with 4 recognizable spatial regions of similar sizes; a 500-nodes sensor network with a density such that the average number of neighbor of each node is 15 (i.e., similar to the sensor network of Figure 2-b).

Let us firstly analyze the behavior in region formation. From a static viewpoint, simulations confirm that, as described in Subsection 2.1 and as shown in Figure 2, the region formation algorithm converges and variations on the parameter *p* actually induce the network in self-partition into regions of different sizes.

From the dynamic viewpoint, we have studied how variations of the gossip percent *num_neigh* and the sleep cycle *t* affect the speed of convergence and the accuracy of the region detection algorithm. To this end, we traced the evolution of the system by imposing a change on the *p* parameter determining the threshold and, thus, forcing a change in region sizes. At startup, nodes are connected with any neighbor. Within cycles from *0* to *128 p* was set to *0.4*. During this interval the network converges to a partitioning into 2 large-regions of equal size. At cycle *129*, we changed *p* from *0.4* to *1.0*, making the network re-compact into a single large-scale region. At cycle 256 we changed *p* back to *0.4*, making the single region disaggregate again into two regions.



a)

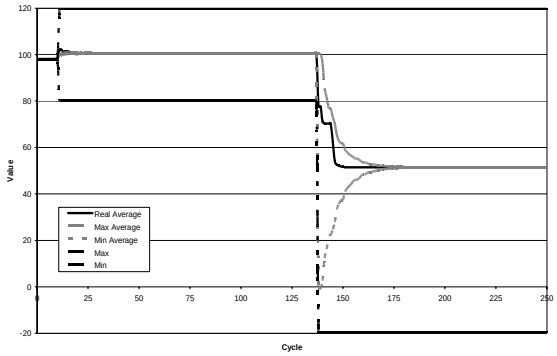


b)

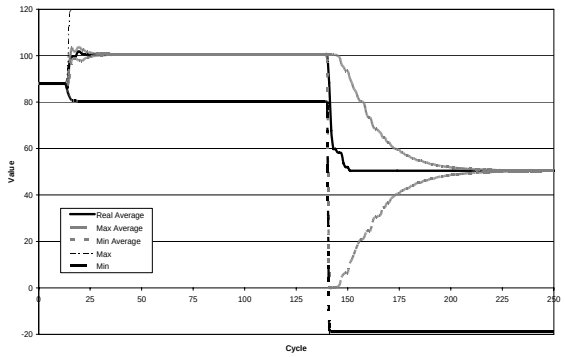
Figure 3. Evolution of region detection when: a) varying the *num_neigh* parameter; b) varying the *t* parameter.

Figure 3-a shows the evolution in the average number of nodes in one region as time passes, by varying the gossip percentage, while Figure 3-b shows the evolution by varying the sleep period *t* of sensor nodes (*t=1* being an abstract simulation time unit). Both graphs show that the number of nodes of the region start from *0*, grow to *250* during the first phase *[0 – 128 cycles]*, reaches *500* during the second phase *[129 – 255 cycles]*, and then diminish again to *250*. Not surprisingly, reducing the gossip percentage or increasing the sleep period *t* makes the network slower in the region detection process. From Figure 3, it also emerges that the

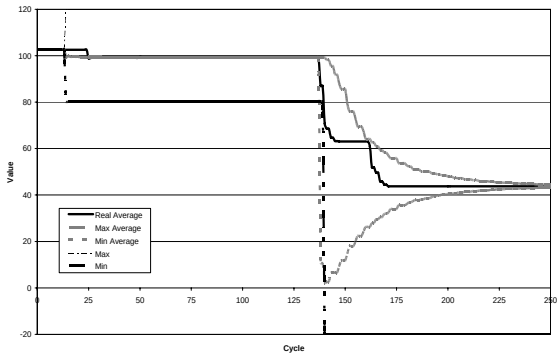
speed of convergence in region formation appears less influenced by variations of *num_neigh* than by variations of *t*.



a)



b)



c)

Figure 4. Per region aggregated values. Minimum estimate of the maximum, maximum estimate of the minimum, minimum and maximum estimates of the average and real value of the average. a) *num_neigh=1.0, t=1*; b) *num_neigh=0.5, t=1*; c) *num_neigh=1.0, t=4*.

The strange "stairs-like" trend of data lines obtained by setting *t=4* and *t=8* (Figure 3-b) clearly show the non-linear nature of the algorithm. These are mostly due to the fact that, when a region is forming, lots of sub regions are growing within, thus connecting the most similar neighbors. Only when the new actual minimum ID of the new region reaches a node, such node recognizes that is becoming part of a new region.

Let us know focus on the behavior of the approach in evaluating aggregated values.

From the static viewpoint, all local aggregation algorithms correctly converge towards the correct (real) values.

From the dynamic viewpoint, Figure 4 shows the trend of several values aggregated on a per region basis, in the first 250 cycles of the simulation scenario already discussed for Figure 3. Curves in each graph represent the minimum (worst case) estimate of the region maximum, the maximum (worst case) estimate of the region minimum, the minimum and the maximum (the two worst cases) estimates of the average, and the real actual value of the average computed over all nodes within the growing region.

Although Figure 4 show results obtained for different values of *t* and *num_neigh*, all the graphs show the same qualitative trend. When regions start forming, after a few cycles, a fast convergence of the local maximum and minimum to their new correct values of *120* and *80*, respectively, is clearly visible. Average related values have a relatively small transitory and eventually reach the correct value of *100* as expected. At cycle 128, *p* is changed to $p = 1.0$ and the region starts growing another time. The local maximum does not have to change its value. The local minimum reaches quickly its new value (*-20*) in a few iterations. Average values instead have a longer transitory but eventually slowly converge to the expected value of *50*. Observing Figure 4 is clear that different aggregate values behave differently varying *num_neigh* and *t*. In particular accuracy of average related values are really more sensible to variations of *num_neigh* and *t* than the local minimum and maximum have. Comparing the effects of *num_neigh* and *t* in convergence (Figure 4-a vs. 4-b and 4-c, respectively) shows that, unlike in region formation, local aggregation is comparably affected by *t* and *num_neigh*.

4.2 Real Implementation Test bed

We have implemented and deployed our algorithms in a testbed of 16 Micaz Crossbow motes, with motes distributed across two confining rooms and the facing corridor in our department. Light levels have been used as the basis for region identification, whereas both light and temperature have been aggregated on a per-region basis. Also, we have also implemented a simple querying system to enable a user with a laptop and a Crossbow interface board to query a sensor and retrieve from it aggregated data about the current region.

The deployed algorithms worked as expected from the functional viewpoint. First, the different lights levels exhibited by the three rooms led the sensor network self-partition in three different regions (i.e., three macro sensors), each associated to a different room. Second, within each region, the sensor correctly computed aggregate light and temperature information. Third, users were able to access such aggregated information by querying any single sensor in a region as if it were a virtual macro sensor representative of its region.

From the viewpoint of energy consumption, Table 1 summarizes the mean values of the energy consumption by sensors. This is expressed in terms of the average decay in the voltage exhibited after two hours of execution of the algorithms by initially fully-charged batteries. The different results for different values of the parameters *t* and *num_neigh* confirm that both these two parameters are effective in tuning the energy consumption. In

addition, the fact that these data remains approximately the same (*±0,05v*) over different experiments and with different configurations of the sensor network confirms that our approach is able to guarantee a well defined bound on energy consumption.

Comparing Table 1 with the simulated results (comparison with simulated results is needed since convergence times in the small test bed provides not significant information), one can summarize that, with our approach, it is possible to tune the desired energy levels by acting both on *t* and *num_neigh*, though one must be ready to pay in any case a proportional slowing down in the convergence of region formation and data aggregation.

Table 1. Energy consumption in the testbed.

	t = 5sec	*t = 10sec*	*t =20sec*
num_neigh = 1.0	*0.225v*	*0.118v*	*0.072v*
num_neigh = 0.5	*0.145v*		
num_neigh = 0.25	*0.122v*		

5. APPLICATION SCENARIOS AND LIMITATIONS

The virtual macro sensor approach can be fruitfully exploited in a number of applications scenarios, although it still exhibits a number of limitations calling for further research work and extensions.

5.1 Querying by Multiple and Mobile Users

The most direct way of exploiting the virtual macro sensor approach is for supporting queries by multiple and mobile users (or by services on users' PDA). A user/service that wants to retrieve information about the surrounding will typically access the nearest sensor and query it about some local patterns of sensed data. For example, "give me the maximum temperature within 500 meters" or, by referring to some more logical environmental concept, "give me the average temperature in this room". At this point, in most of the cases, the queried sensor can immediately answer to the user without further burdening the network, independently of the number of mobile users.

The answer to a user/service can be immediate whenever: *(i)* the query relates to some functions of a property *w* which have been already aggregated as part of the background aggregation noise; and *(ii)* the query concerns information within a single region, i.e., a single macro sensor. With this regard, we recall (Subsection 3.2) that each sensor in a region knows its distance to the closest border of the region. Therefore it can recognize whether a query can be answered within the region because involving a single macro sensor (e.g., "give me the data in this room" or "give the data within a distance that is less then you distance to the border") or not. Still, the fact that some classes of queries cannot be directly answered by macro sensors is indeed a limitation.

To make virtual macro sensors able to answer to very general queries related to any function of any property *w* sensed within a region, our approach should be extended to make macro sensors programmable. This implies the possibility of dynamically "injecting" into a sensor network the specification of additional local aggregation functions, and let these be integrated into the existing background aggregation noise. This can make it possible – within the sensing capabilities of individual sensors – to have macro sensors able to answer to both general-purpose and application-specific queries, as proposed by e.g. Region Streams [20] and Logical Neighborhoods [18]. Clearly, such way of injecting new behaviors into the network could also be used to dynamically enforce a partitioning of regions based on different properties that a single one *v*, and based on different distance functions (the possibility of building multiple region partitions based on different threshold values for a single property *v* has already been discussed in Section 2.1).

To make virtual macro sensors able to answer also to inter-region queries (e.g., to let a macro sensor answer about "the average temperature within 500 meters" even if the query is performed at a distance of 300 meters from the confining regions) without falling back to a raw tree-based approach, we are currently verifying the possibility of implementing efficient inter-region aggregation algorithms based on gossiping. This will make any node in a region able to provide users with some aggregate information related to neighboring regions other than to its own region.

5.2 Centralized Data Collection

As already anticipated, the expected dramatic increase in the number and density of sensor networks deployed in our world, will soon reach a point in which the overall amount of data generated by such network will make it impossible to transfer these data to some centralized location in a raw way. Collection of aggregated data will become the only solution to extract useful information from them. The virtual macro sensors approach already solves this problem. In fact, it is possible to route aggregated macro sensorial data periodically to a fixed sink at very limited communication costs, simply by having the leader of each region take care of this alone. This can take place with the additional advantages that collected data fully abstracts from the structure and dynamics of the sensor network, and that local aggregation limits the loss of information which is instead associated with global aggregation algorithms that do not take into account data patterns.

However, to make the aggregated data collected at a centralized point really meaningful, it should be also possible to associate a specific dimension and shape to each region, and it should be possible to put regions in proper spatial relations with each other, so as to actually perceive the "network of macro sensors". More in general, the virtual macro sensors approach could be seen as a way to effectively extract and collect high-level semantic knowledge about the structure and characteristics of an unknown environment [1]. To make an example, one could think at randomly deploying sensors over some labyrinthine environment, have the sensor network self-organize into macro sensors based on light and/or sound patterns and then, if each region could recognize and report about neighboring regions, dynamically reconstruct a map of the area and of its salient environmental characteristics.

Unfortunately, so far, we are able to compute only the distance of a sensor from the region border. Yet, we expect similar diffusion

algorithms (possibly exploiting the already mentioned inter-region aggregation algorithms) can be defined to compute more advanced topological properties, and to extract high-level knowledge from a network. These, together with the mentioned possibility of enforcing multiple partitioning based on different properties and thresholds, could enable to produce, within the same network, multiple knowledge views for the use of diverse users/services.

5.3 Situation Recognition

The possibility of identifying regions characterized by specific patterns of sensed data, and the possibility of computing aggregated data within the network can also be effectively used to improve the capability of the sensor network to self-recognize unusual patterns of sensing and, in case, to automatically generate alarms.

To make a practical example, we are cooperating with the geological department of the Reggio Emilia Apennine to exploit our approach for effective landslide detection. Inertial sensors deployed on a mountain slope generally sense a random background noise, without any recognizable patterns. However, when a slip surface starts detaching, all the sensors on such surface will exhibit peculiar acceleration patterns. In this case, our algorithm can be able – despite noisy data – to dynamically self-partition the network into two distinct regions, one of which associated the slip surface, and of alerting the geological department by reporting macro sensorial information about the slip surface and its behavior. This can avoid the costly process of continuously reporting data back to the department for off-line analysis. Other examples include detecting anomalies in buildings, streets, or parks.

Also in this case, to make situation recognition fully practical and general purpose, both algorithms for dynamic injection of aggregation functions and algorithms for the recognition of advanced topological properties may be required.

6. CONCLUSIONS AND FUTURE WORK

The proposed virtual macro sensor approach makes a sensor network self-organize into regions characterized by similar sensing patterns, so as promote aggregation of data on a per-region basis, as if each region were monitored by a single macro sensor. Such an approach can be very effective in supporting multiple and mobile services, in facilitating data collection in very dense and large-scale sensor networks, and in enforcing advance situation recognition activities within the network.

Despite the encouraging results obtained so far, we are aware of a number of limitations of our work, subjects of our current research work. These include: generalizing the approach to support multiple overlays and general-purpose queries; exploring inter-region algorithms to support more global queries; defining algorithms to promote the building of high-level knowledge about the global structure and properties of the of virtual macro sensors network.

7. ACKNOWLEDGMENTS

Work supported by the project CASCADAS (IST-027807) funded by the FET Program of the European Commission.

8. REFERENCES

[1] F. Zambonelli, N. Bicocchi, M. Baumgarten, M. Mulvenna, R. Kusber, “Self-organizing Knowledge Networks for Pervasive Situation-aware Services”, IEEE SMC International Conference, 2007.

[2] A. Boulis, S. Ganerival, M. Srivastava, “Aggregation in Sensor Network: An Energy-Accuracy Trade-off”, Workshop on Sensor and Actuator Network Protocols and Applications, Anchorage (AK), 2003.

[3] G. Castelli, A. Rosi, M. Mamei, F. Zambonelli, “A Simple Model and Infrastructure for Context-aware Browsing of the World”, International Conference on Pervasive Computing and Communication, New York (NY), 2007.

[4] E. Catterall, K. Laerhoven, M. Strohbach, “Self-Organization in Ad-Hoc Sensor Networks: An Empirical Study”, International Conference on Simulation and Synthesis of Living Systems, Sydney (AU), 2002.

[5] C. Chong, S. Kumar, “Sensor networks: Evolution, Opportunities, and Challenges”, Proceedings of the IEEE, 91(8):1247-1256, Aug. 2003.

[6] A. Corradi, L. Leonardi, F. Zambonelli “Diffusive Load Balancing Policies for Dynamic Applications”, IEEE Concurrency, 7(11):22-31, 1999.

[7] C. Curino, M. Giani, M. Giorgetta, A. Giusti, A. Murphy, G. Picco, “Mobile Data Collection in Sensor Networks: The TinyLime Middleware”, Journal of Pervasive and Mobile Computing, (4)1:446-469, 2005.

[8] A. Dimakis, A. Sarwate, M. Wainwright, “Geographic Gossip: Efficient Aggregation for Sensor Networks”, International Conference on Information Processing in Sensor Networks, Nashville (TN), 2006.

[9] D. Estrin, D. Culler, K. Pister, G. Sukjatme, “Connecting the Physical World with Pervasive Networks”, IEEE Pervasive Computing, 1(1):59-69, 2002.

[10] J. Gehrke, S. Madden, “Query Processing In Sensor Networks”, IEEE Pervasive Computer, 3(1):46-65, 2004.

[11] B. Harrington, Y. Huang, “In-Network Surface Simplification for Sensor Fields”, Conference on Geographic Information Systems, Bremen (D), 2005.

[12] M. Jelasity, A. Montresor, O. Babaoglu, “Gossip-based Aggregation in Large Dynamic Networks”, ACM Transactions on Computer Systems, 23(3):219-252, 2005.

[13] M. Jelasity, A. Kermarrec, “Ordered Slicing of Very Large-Scale Overlay Networks”, International Conference on Peer-to-Peer Computing, Cambridge, UK, 2006.

[14] X. Jiang, J. Polastre, D. Culler, “Perpetual Environmentally Powered Sensor Networks”, International Symposium on Information Processing in Sensor Networks, Los Angeles (CA), 2005.

[15] L. Chen, Z. Chen, S. Tu, “A Realtime Dynamic Traffic Control System Based on Wireless Sensor Network”, International Conference on Parallel Processing Workshops, Poznan (P) 2005.

[16] C. Lu, G. Xing, O. Chipara, C. Fok, S. Bhattacharya, “A Spatiotemporal Query Service for Mobile Users in Sensor

Networks", International Conference on Distributed Computing Systems, Columbus (OH), 2005.

[17] S. Madden, J. Hellerstein, "Distributing Queries over Low-Power Wireless Sensor Networks", Conference on Management of Data, McLean (VA), 2002.

[18] G. Mottola, G. P. Picco, "Logical Neighborhoods: A Programming Abstraction for Wireless Sensor Networks", Conference on Distributed Computing in Sensor Systems, San Francisco (CA), USA, 2006.

[19] R. Müller, G. Alonso, "Shared Queries in Sensor Networks for Multi-User Support", Technical Report 508, ETH Zürich (CH), 2005.

[20] R. Newton, M. Welsh, "Region Streams: Functional Macroprogramming for Sensor Networks", International Workshop on Data Management for Sensor Networks, Toronto (CA), 2004.

[21] A. Panangadan. G. S. Sukhatme, "Data Segmentation for Region Detection in a Sensor Network", Center for Robotics and Embedded Systems, University of Southern California, Technical Report 05-005, 2005.

[22] J. Polastre, R. Szewcyk, A. Mainwaring, D. Culler, J. Anderson, "Analysis of Wireless Sensor Networks for Habitat Monitoring", in Wireless Sensor Networks, Kluwer Academic Publisher, pp. 399-423, 2004,.

[23] N. Ramanathan et al., "Designing Wireless Sensor Networks as a Shared Resource for Sustainable Development", International Conference on Information and Communication Technologies and Development, Berkeley (CA), 2006.

[24] T. Schoellhammer, B. Greenstein, D. Estrin, "Hyper: A Routing Protocol To Support Mobile Users of Sensor Networks", Symposium on Mobile Ad Hoc Networking and Computing, Florence (I), 2006.